

**UNIVERSITY OF CYPRUS**  
**DEPARTMENT OF COMPUTER SCIENCE**

**PROSPECTUS**  
**Academic Year 2025 - 2026**

POSTAL ADDRESS

Department of Computer Science  
University of Cyprus  
P.O. Box 20537  
CY 1678 Nicosia  
CYPRUS

Tel.: +357-22892700

Email: [cs@ucy.ac.cy](mailto:cs@ucy.ac.cy)

Web: <http://www.cs.ucy.ac.cy>

twitter/X: [@csdeptucy](https://twitter.com/csdeptucy)

## *Open Letter from the Chairperson of the Department*

### **Dear Students,**

The Study Guide is a concise but comprehensive presentation of the Department of Computer Science of the University of Cyprus. Within the Guide you will find information that you will need during your studies at the Department, such as undergraduate and postgraduate study programmes, course descriptions, rules of attendance, prerequisites for graduation, short CVs of academic staff, etc. announcements, news, etc. are also published through the Department's website at <http://www.cs.ucey.ac.cy> address, our Twitter (@csdeptucey) and Facebook accounts, as well as through e-mail and notice boards near the Secretariat. At the same time, it is important to maintain regular contact with the **academic advisor** assigned to you to discuss any issues that concern you, primarily studies, courses, scientific or professional orientation, postgraduate studies, etc.

Computer Science is internationally recognized as an important science with constant evolution and a strong impact on the development of other sciences and society in general, despite the fact that it only began to be founded in the second half of the 20th century. The increased importance and applicability of Computer Science have contributed to the consolidation of the Department of Computer Science as one of the most dynamic and competitive Departments of the University of Cyprus, with international recognition. The abilities and high level of education of our graduates are recognized by the local labour market and internationally, as our graduates score commendable performances either by being employed in the domestic industry immediately after graduation, or by pursuing postgraduate studies in leading Universities in Europe and America. In recent years, about 80 undergraduate and 30-35 postgraduate students have entered our Department every year. In the current academic year, approximately 450 students are enrolled in our Department, of which 82% are undergraduate students and the rest are Master's or doctoral students. The Department works systematically to attract excellent students, organizing and supporting annually various training seminars and competitions in Computer Science, such as the Computer Science Day for Students and the Logipagnion competition. In fact, during the last eight years (2017-2024) the Department of Computer Science won the first place among all branches of natural and applied sciences, and engineering, in the number of candidates' first preferences in the Pancyprian Examinations.

Both the Undergraduate and Postgraduate Curriculum are designed to respond to modern developments in Computer Science and to offer students a comprehensive curriculum, which equally covers the theoretical foundations, technological knowledge and experimental methodologies of Computer Science. The Department, by monitoring and contributing on a continuous basis to the scientific developments in the field of Computer Science, regularly revises and modernizes the curricula it offers, enriching the list of advanced elective courses, and ensuring that the interdisciplinarity of the studies provided is enhanced. Also, the Department offers a secondary curriculum in Computer Science for students of other Departments of the University of Cyprus.

At the postgraduate level, the Department of Computer Science offers a Doctoral Programme of Study, as well as a number of Master's level Postgraduate Programmes, both research-oriented and professionally oriented programmes. In addition, since the academic year 2017/18, in collaboration with the Open University of Cyprus and the Department of Psychology of the University of Cyprus, the Department offers the interdisciplinary Master's Programme in Cognitive Systems with distance learning in English, while from the academic year 2020/21 the interdisciplinary, interdepartmental Master's level programme in Data Science began its operation. in collaboration with the Departments of Mathematics and Statistics, and Public Administration and Business Administration. And this master's programme is English-speaking. With funding from the European Commission, another specialized master's programme in Artificial Intelligence, which is also English-speaking, was designed and started operating from the academic year 2022-23. From the academic year 2023-24, this

programme evolved into an interdepartmental programme with the Department of Electrical and Computer Engineering.

All the Department's study programmes are harmonized with the provisions of the European Credit Transfer and Accumulation System (ECTS) and each student receives the relevant Diploma Supplement upon completion of their studies. During your studies, you are encouraged to participate in the European Student Exchange Programme ERASMUS+ which gives you the opportunity to carry out part of your studies at other academic institutions in the European Higher Education Area.

The aim of all our curricula is to incubate scientists who are able to help the industry of Cyprus to grow and spread in new sectors, maintaining high competitiveness in relation to the industries of other countries in our region and Europe in general. Our Department is in constant dialogue with industry, with competent government services and with centres of excellence abroad, in order to be able to contribute decisively to the promotion of Cyprus as a centre for the provision of services, know-how and innovation in Information Technology through the development of domestic scientific potential of high competitiveness and know-how. Also, the Department works tirelessly for the international promotion of our students' skills through participation and distinctions in international competitions of programming skills, innovation, entrepreneurship, etc.

The Department's strong research presence also plays an important role in the above efforts. In the more than thirty years since its establishment, the Department has developed a particularly important activity and has contributed to the promotion of science and knowledge at an international level. Particularly important is the participation of the Department in European Union (EU) research programs, as well as in domestic programs of the Foundation for Research and Innovation (RIF, formerly RPF) of Cyprus. In the last decade alone (2014-2024), members of the Department's Academic Staff have attracted more than 160 research projects from the EU and the RIF with a total budget of more than €20 million. Most of these funds have been used to subsidize hundreds of young scientists - researchers, PhD candidates, postdoctoral fellows and postgraduate students from Cyprus and abroad (more than 300), for the development of modern research and teaching infrastructures and for the development of innovative software, hardware and application systems. Research results of members of the Department are presented every year in leading journals and conferences of Computer Science, have received worldwide distinctions, and are utilized by the international scientific community and industry. Finally, the contribution of the Department to the support of government actions on Communications and Computer Science related to the Internet, the network interconnection of Cyprus with abroad, the technological upgrade of Secondary Education and Health Services, the promotion of Cypriot cultural heritage through modern means, etc. is particularly noteworthy.

During your studies, I encourage you to actively participate in the teaching and research activities of the Department in order to gain the maximum knowledge and experience that your university education can offer you. I also invite you to take advantage of the opportunity to attend courses in other disciplines offered by the various Departments of the University in order to enrich your knowledge and acquire a more comprehensive and comprehensive education. This possibility is also offered through the European ERASMUS+ Programme already mentioned.

Dear Students,

The unprecedented situation we have experienced in recent years due to the new Covid-19 virus, in addition to the important lessons it has taught us, showing in practice that with collectivity such crises can be overcome and even turned into opportunities, highlighted the importance of technology and information tools. Consequently, you are in a Department that objectively studies, shapes, implements and produces tools, means and processes and cultivates ways of thinking and methodologies that shape the future and that can improve the present. At the end of this path, you will be a Graduate of the Department, you will have dealt with a wide range of topics in Computer Science and you will be able to contribute in many ways to developments. It is up to you to make the most of all the opportunities offered by the Department and the University with the aim of a better tomorrow for each of you, but also for our world.

On behalf of the Members of the Department, I would like to once again welcome you and wish you all every progress in your studies.

With warm regards,

A handwritten signature in black ink, appearing to read 'Chryssis Georgiou', written in a cursive style.

Professor Chryssis Georgiou

President of the Department of Computer Science

## CONTENTS

<b>STAFF</b>	<b>6</b>
<b>ACADEMIC FACULTY INFORMATION</b>	<b>7</b>
<b>ACADEMIC CALENDAR 2025 - 2026</b>	<b>12</b>
<b>INTRODUCTION</b>	<b>13</b>
<b>THE ECTS SYSTEM</b>	<b>14</b>
<b>GENERAL INFORMATION</b>	<b>15</b>
Class Attendance and Teaching	15
Academic Advisor	15
Student Representation	15
Secretariat	15
Course Schedule	15
Library Use	15
Electronic Announcements / Department Website	16
Email	16
Laboratory Equipment	16
Laboratory Equipment Use Regulations	16
Awards	17
Administrative Duties of Academic Staff	18
<b>UNDERGRADUATE PROGRAMME OF STUDIES</b>	<b>20</b>
Objectives and Prospects	20
Course Areas	20
Curriculum	21
Computer Science Programme	22
Restricted Elective Courses	24
Elective Courses	24
Foreign Language Courses	25
Diploma Project	25
Minor Programme in Computer Science	25
Minor Programme in Biomedical Engineering	26
Short Course Description	30
Courses for other Departments	88
<b>POSTGRADUATE PROGRAMMES OF STUDY</b>	<b>99</b>
Master's Programmes	99
Master in Computer Science	101
Brief Description of Master in Computer Science Courses	103
Master in Artificial Intelligence	134
Master in Data Science	135
Master in Cognitive Systems	136
PhD in Computer Science	137
<b>SHORT BIOGRAPHICAL NOTES OF ACADEMIC STAFF</b>	<b>140</b>
ANNEX A: Rules for Diploma Projects	147
ANNEX B: Rules for Graduate Studies	151
ANNEX C: Specifications for Preparation of the Master Thesis	156
ANNEX D: Specifications for Preparation of the Doctoral Dissertation	158

## STAFF

### **Academic Faculty**

Chryssis Georgiou (**Chairperson**)  
Yiannis Dimopoulos (**Vice-Chairperson**)  
Andreas Aristidou  
Elias Athanasopoulos  
Chris Christodoulou  
Giorgos Chrysanthou  
Eleni Constantinou  
Marios D. Dikaiakos  
Georgia Kapitsaki  
Elpida Keravnou-Papailiou  
Panayiotis Kolios  
Marios Mavronicolas  
Mihalis Nicolaou  
George Pallis  
George Papadopoulos  
Constantinos Pattichis  
Anna Philippou  
Andreas Pieris  
Yiannakis Sazeides  
Vasos Vassiliou  
Haris Volos  
Demetris Zeinalipour

### **Professors Emeritus**

Antonis Kakas  
Andreas Pitsillidis

### **IT Systems Technical Support**

Isabella Christodoulou  
Katerina Christodoulou  
Andri Michaelidou  
Maria Tsiolakki

### **Administrative Support of Research**

#### **Programs**

Marios Vrahimi

### **Special Teaching Scientists**

Michalis Agathocleous  
Christoforos Christoforou  
Marios Hadjiaros  
Argyris Konstantinidis  
Marios Konstantinidis  
Christos Kyriakou  
Kleanthis Malialis  
Dimitris Paschalidis  
Irene Schiza  
Moisis Symeonidis  
Natalie Temene

### **Special Teaching Staff**

Pavlos Antoniou  
Pyrros Bratskas  
George Hadjipollas  
Yiannakis Mylonas  
Petros Panayi  
Christoforos Panayiotou

### **Secretariat**

Dora Georgiou  
Maria Kittira  
Melina Menelaou - Chrysostomou  
Angeliki Pavlou

## Academic Faculty Information

### *Academic Faculty*

**Andreas Aristidou**

Associate Professor  
Office: FST01 B113  
Phone: +357 22892698  
Email: [andarist@ucy.ac.cy](mailto:andarist@ucy.ac.cy)

**Elias Athanasopoulos**

Associate Professor  
Office: FST01 B105  
Phone: +357 22 892754  
Email: [eliasathan@ucy.ac.cy](mailto:eliasathan@ucy.ac.cy)

**Chris Christodoulou**

Professor  
Office: FST01 113  
Phone: +357 22892752  
Email: [cchrist@ucy.ac.cy](mailto:cchrist@ucy.ac.cy)

**George Chrysanthou**

Professor  
Office: FST01 013  
Phone: +357 22892719  
Email: [yorgos@ucy.ac.cy](mailto:yorgos@ucy.ac.cy)

**Eleni Constantinou**

Assistant Professor  
Office: FST01 107  
Phone: +357 22892733  
Email: [constantinou.a.eleni@ucy.ac.cy](mailto:constantinou.a.eleni@ucy.ac.cy)

**Marios Dikaiakos**

Professor  
Office: FST01 012  
Phone: +357 22892720  
Email: [mdd@ucy.ac.cy](mailto:mdd@ucy.ac.cy)

**Yiannis Dimopoulos (Vice Chairperson)**

Professor  
Office: FST01 014  
Phone: +357 22892718  
Email: [yannis@ucy.ac.cy](mailto:yannis@ucy.ac.cy)

**Chryssis Georgiou (Chairperson)**

Professor  
Office: FST01 016  
Phone: +357 22892745  
Email: [chryssis@ucy.ac.cy](mailto:chryssis@ucy.ac.cy)

**Georgia Kapitsaki**

Associate Professor  
Office: FST01 119  
Phone: +357 22892692  
Email: [gkapi@ucy.ac.cy](mailto:gkapi@ucy.ac.cy)

**Elpida Keravnou-Papailiou**

Professor  
Office: FST01 117  
Phone: +357 22892694  
Electrical Phone: [elpida@ucy.ac.cy](mailto:elpida@ucy.ac.cy)

**Panayiotis Kolios**

Assistant Professor  
Office: FST01 116  
Phone: +357 22892695  
Email: [kolios.panayiotis@ucy.ac.cy](mailto:kolios.panayiotis@ucy.ac.cy)

**Marios Mavronicolas**

Professor  
Office: FST01 106  
Phone: +357 22892702  
Email: [mavronic@ucy.ac.cy](mailto:mavronic@ucy.ac.cy)

**Mihalis Nicolaou**

Assistant Professor  
Office: FST01 117  
Phone: +357 22892707  
Email: [nicolaou.mihalis@ucy.ac.cy](mailto:nicolaou.mihalis@ucy.ac.cy)

**Vasos Vassiliou**

Associate Professor  
Office: FST01 B114  
Phone: +357 22892750  
Email: [vasosv@ucy.ac.cy](mailto:vasosv@ucy.ac.cy)

**George Pallis**

Professor  
Office: FST01 B119  
Phone: +357 22892743  
Email: [gpallis@ucy.ac.cy](mailto:gpallis@ucy.ac.cy)

**George Papadopoulos**

Professor  
Office: FST01 118  
Phone: +357 22892693  
Email: [george@ucy.ac.cy](mailto:george@ucy.ac.cy)

**Constantinos Pattichis**  
Professor  
Office: FST01 114  
Phone: +357 22892697  
Email: [pattichi@ucy.ac.cy](mailto:pattichi@ucy.ac.cy)

**Anna Philippou**  
Professor  
Office: FST01 105  
Phone: +357 22892699  
Electrical Phone: [annap@ucy.ac.cy](mailto:annap@ucy.ac.cy)

**Andreas Pieris**  
Assistant Professor  
Office: FST01 111  
Phone: +357 22892705  
Email: [pieris.andreas@ucy.ac.cy](mailto:pieris.andreas@ucy.ac.cy)

**Yiannakis Sazeides**  
Professor  
Office: FST01 109  
Phone: +357 22892704  
Email: [yanos@ucy.ac.cy](mailto:yanos@ucy.ac.cy)

**Haris Volos**  
Associate Professor  
Office: FST01 115  
Phone: +357 22892696  
Email: [volos.haris@ucy.ac.cy](mailto:volos.haris@ucy.ac.cy)

**Demetris Zeinalipour**  
Professor  
Office: FST01 B106  
Phone: +357 22892755  
Email: [dzeina@ucy.ac.cy](mailto:dzeina@ucy.ac.cy)

### *Special Teaching Scientists*

**Michalis Agathocleous**

Office: 0EE 01 203 Lab

Phone: +357 22892682

Email: [agathocleous.michalis@ucy.ac.cy](mailto:agathocleous.michalis@ucy.ac.cy)

**Augoustis Augousti**

Office: FST 01 B111

Phone: MS Teams

Email: TBA

**Christoforos Christoforou**

Office: FST 01 B116

Phone: +357 22 892749

Email: [christoforos@ucy.ac.cy](mailto:christoforos@ucy.ac.cy)

**Marios Hadjjaros**

Office: FST 01 B116

Phone: MS Teams

Email: [hadjjaros.marios@ucy.ac.cy](mailto:hadjjaros.marios@ucy.ac.cy)

**George Ioannou**

Office: FST 01 B111

Phone: MS Teams

Email: TBA

**Artemis Kontou**

Office: FST 01 B120

Phone: +357 22893450

Email: TBA

**Demetris Kouzapas**

Office: FST 01 B111

Phone: MS Teams

Email: TBA

**Marios Konstantinidis**

Office: FST 01 208 Lab

Phone: +357 22892674

Email: TBA

**Maria Konstantinou**

Office: FST 01 203

Phone: +357 22895193

Email: TBA

**Panayiota Nicolaou**

Office: FST 01 B120

Phone: MS Teams

Email: TBA

**Dimitris Paschalidis**

Office: FST 01 217 Lab

Phone: +357 22 892663

Email: TBA

**Moisis Symeonidis**

Office: FST 01 217 Lab

Phone: +357 22 892663

Email: [msyмео03@ucy.ac.cy](mailto:msyмео03@ucy.ac.cy)

**Natalie Temene**

Office: FST 01 B120

Phone: +357 22 892746

Email: [temene.natalie@ucy.ac.cy](mailto:temene.natalie@ucy.ac.cy)

### *Special Teaching Staff*

**Pavlos Antoniou**

Special Teaching Staff  
Office: FST01 B109  
Phone: +357 22893927  
Email: [antoniou.pavlos@ucy.ac.cy](mailto:antoniou.pavlos@ucy.ac.cy)

**Pyrros Bratskas**

Special Teaching Staff  
Office: FST01 B107  
Phone: +357 22893930  
Email: [bratskas@ucy.ac.cy](mailto:bratskas@ucy.ac.cy)

**Giorgos Hadjipollas**

Special Teaching Staff  
Office: FST01 B110  
Phone: +357 22893932  
Email: [hpollas@ucy.ac.cy](mailto:hpollas@ucy.ac.cy)

**Yiannakis Mylonas**

Special Teaching Staff  
Office: FST01 B115  
Phone: +357 22893931  
Email: [mylonasy@ucy.ac.cy](mailto:mylonasy@ucy.ac.cy)

**Petros Panayi**

Special Teaching Staff  
Office: FST01 B118  
Phone: +357 22893926  
Email: [petrosp@ucy.ac.cy](mailto:petrosp@ucy.ac.cy)

**Christoforos Panayiotou**

Special Teaching Staff  
Office: FST01 B104  
Phone: +357 22893928  
Email: [panayiotou.christoforos@ucy.ac.cy](mailto:panayiotou.christoforos@ucy.ac.cy)

### *IT Systems Technical Support*

**Isabella Christodoulou**

IT Assistant  
Office: FST01 001  
Phone: +357 22892711  
Email: [christodoulou.isavella@ucy.ac.cy](mailto:christodoulou.isavella@ucy.ac.cy)

**Andri Michaelidou**

University Officer  
Phone: +357 22892734  
Office: FST01 B117  
Email: [andrim@ucy.ac.cy](mailto:andrim@ucy.ac.cy)

**Katerina Christodoulou**

University Officer  
Office: FST01 001B  
Phone: +357 22893921  
Email: [christodoulou.katerina@ucy.ac.cy](mailto:christodoulou.katerina@ucy.ac.cy)

**Maria Tsiolakki**

Senior IT Assistant  
Office: FST01 001A  
Phone: +357 22892727  
Email: [tmaria@ucy.ac.cy](mailto:tmaria@ucy.ac.cy)

### *Administrative Support of Research Programs*

**Marios Vrahimi**

Special Scientist for Research Support  
Office: FST01 017  
Phone: +357 22892713  
Email: [vrachimi.marios@ucy.ac.cy](mailto:vrachimi.marios@ucy.ac.cy)

### *Secretariat*

**Dora Georgiou**

Secretary  
Office: FST01 019  
Phone: +357 22892722  
Email: [addora@ucy.ac.cy](mailto:addora@ucy.ac.cy)

**Melina Menelaou-Chrysostomou**

Secretary  
Office: FST01 015  
Phone: +357 22 892721  
Email: [melina@ucy.ac.cy](mailto:melina@ucy.ac.cy)

**Maria Kittira**

Secretary  
Office: FST01 007  
Phone: +357 22892700  
Email: [manak@ucy.ac.cy](mailto:manak@ucy.ac.cy)

## ACADEMIC CALENDAR 2025 - 2026

	Fall Semester 2025 – 2026	Spring Semester 2025 – 2026	Summer Semester 2025-2026
Registration week	2 – 4 September	14 - 16 January	June 4
Commencement of classes	September 8	19 January	June 8
Deadline for course selection	September 12	January 23	12 June
Deadline for course selection removal	September 26	6 February	12 June
Deadline for course selection withdrawal	24 October	6 March	-
End of classes	December 5	April 30	July 24
Easter holidays		6 - 19 April	
Final exams	9 - 23 December	11 - 25 May	July 27-31
Public holidays	1 October 28 October	February 23 (Green Monday) March 25 1 April April 12 (Easter) May 1	June 1 (Holy Spirit)

## **Introduction**

Computer Science initially started as a field of limited scope, mainly related to the automation of mathematical calculations. But it quickly evolved into a fascinating scientific amalgamation of theory and technology. Today, Computer Science deals with a variety of topics, such as the expansion of the range of problems that can be solved efficiently with computers, the creation, maintenance and optimization of software and hardware systems for the construction of high-performance computers, the way in which humans formulate, converse and plan their activities, and even the modelling of brain function and the role of language and logic in dealing with its practical problems. Thus, Computer Science is directly linked to all sciences, but also to many other sciences such as, for example, Philosophy, Psychology, Cognition, Linguistics and Business Administration. This multidisciplinary nature of this relatively new science ensures flexibility and a multitude of possibilities for exploring new fields of research.

Today, Computer Science has penetrated all areas of life with significant applications in industry, commerce, economics, education, and medicine. The emergence and spread of computers has decisively improved the quality of life, offering a multitude of services, performing dangerous or complex projects for human measures and contributing, recently, to the dissemination of knowledge and skills through the technology of knowledge base systems. Computer communication networks, distributed processing techniques and mobile computing have become widespread and are an integral part of the modern work environment. The initial impression that the computer could replace humans has been removed, as it has been understood that the computer assists humans and enhances their abilities to carry out their tasks more efficiently.

## The ECTS System

Since the academic year 2005-2006, the University of Cyprus has adopted the European Credit Transfer and Accumulation System.

### *What is the ECTS System?*

The ECTS System, or European Credit Transfer and Accumulation System, was developed by the Commission of the European Communities with the aim of providing common procedures to ensure academic recognition of studies uniformly across Europe. The system provides a way to measure and compare learning achievements and transfer them from one European university to another.

The ECTS system provides transparency through the following means:

- ECTS credits, which are a numerical value, are divided into course credits to describe the workload needed to be successfully completed by students.
- The Information Pack, which provides written information to students and staff about universities, departments/faculties, the organization and structure of studies and course units.
- The Diploma Supplement, which shows students' academic achievements in a comprehensive, commonly understood and easily transferable way from one European university to another.

### *The ECTS Credits*

ECTS Credits are credits that are divided into courses and describe the *workload* required for courses to be successfully completed by students. ECTS Credits reflect the *amount* of work needed for each course in *relation* to the total amount of work needed to complete a full Academic Year of study at the university. The work includes lectures, practical work, seminars, homework and exams or other assessment methods. ECTS credits represent a *relative value*. In the ECTS System, *60 credits* represent the workload of an Academic Year of study. Normally, *30 credits* are allocated for a semester.

ECTS credits are awarded only when the course has been successfully completed.

For the acquisition of a Bachelor's Degree in the Undergraduate Curriculum, (at least) 240 ECTS credits are required, while for the acquisition of a Master's Degree in the Postgraduate Study Programme, (at least) 90 ECTS credits are required.

## **General Information**

### ***Class Attendance and Teaching***

Teaching is conducted in the internationally established way of lectures, tutorials, workshops, etc. Students are expected to participate continuously in all the activities of the courses they attend (e.g. lectures, tutorials, workshops, etc.). The Department reserves the right to prohibit the participation in the examinations of those students who systematically abstain from the activities of the courses they attend.

In most courses, tasks are assigned at regular intervals in order to consolidate the curriculum and develop practical skills. The preparation of these works is carried out individually or in groups of small groups.

Assessment is usually based on assigned assignments, as well as oral and written exams, etc. Particular efforts are being made to make the widest possible use of a system of continuous evaluation.

Students are informed about the individual ways of teaching and evaluation of each course by the instructor of this course. This information is contained in the course Information Brochure which is distributed to students during the first week of courses of the semester. In addition, the instructor of each course is available to students at predetermined office hours per week.

The University's general rules of attendance, the rules governing students' rights and obligations, as well as the procedures to be followed, are included in separate forms. These are available to students from the Studies and Student Welfare Service. Students must be aware of all the rules that apply to them.

### ***Academic Advisor***

Each student is assigned as his Academic Advisor a Member of the Academic Staff of the Department. The Academic Advisor provides help or advice (e.g. course difficulties, appropriate combinations of limited choice courses), but also advice on other issues, such as personal problems.

The role of the Academic Advisor cannot be effective without the cooperation of the student. Therefore, undergraduate and postgraduate students, especially first-year students, are encouraged to have frequent meetings with their Academic Advisors, in order to better organize their study program and solve related problems. It is emphasized that the final responsibility for their choices in matters of study lies with the students. However, they, in turn, must inform their academic advisor of their decisions.

### ***Student Representation***

Six elected student representatives participate in the Department's Council.

### ***Secretariat***

Activities of daily nature are carried out through the Secretariat of the Department. The Secretariat is at the disposal of students to provide general information regarding the Department or the University.

### ***Course Schedule***

The Timetable is not included in this Study Guide but is announced on the website of the Studies and Student Welfare Service at the beginning of each semester. Although every effort is made to satisfy all reasonable combinations of course attendance, there is nevertheless a case that some students may encounter difficulty in combining some of their choices, due to e.g. simultaneous lectures. In this case, these students are advised to inform their Academic Advisor immediately.

### ***Library Use***

The University Library is equipped with a large number of books and scientific journals on Computer Science. These include books that will help the student in consolidating the curriculum and, above all, those proposed by the lecturers for each course. The Regulations for the use of the Library are given to students separately at the beginning of the academic year. A prerequisite for the use of the library by students is the possession of a student ID. More on the library's website: <https://library.ucy.ac.cy/>.

## ***Electronic Announcements / Department Website***

The Department's Announcements are made electronically through the Department's website, which is hosted at <http://www.cs.ucey.ac.cy>. This website also hosts information on all the Department's activities and links to relevant websites and websites. Announcements are also sent by e-mail, to the e-mail accounts created for students. Topics of general interest are also announced through the social networking pages maintained by the Department on the Internet.

## ***Email***

The use of E-mail for communication between academic staff and students, as well as between the latter, is considered mandatory by the Department, and students are encouraged to learn how to use it effectively, as quickly as possible. However, it is emphasized that the use of E-mail is not a right of students, but the provision of a service by the Department. In case of misuse of E-mail, the Department reserves the right to remove its license.

## ***Laboratory Equipment***

A significant part of the Study Programmes is of practical content. The Department has the following laboratories fully equipped for teaching and research.

- \* The Postgraduate Laboratory (201) consists of 31 state-of-the-art computers with Microsoft Windows 10 Education operating system and has software to support specialized courses at the postgraduate level and especially in the Internet Computing and Intelligent Systems programs.
- \* The General Teaching Laboratories of Unix Systems (B103) and (103) consist of 33 workstations each, of the latest generation, with Linux operating system (CentOS 7) and Free NX Server and Remote Desktop for remote access with a graphical environment. They have software for developing a variety of applications.
- \* The Undergraduate Studies Laboratory I (B121) consists of 31 state-of-the-art computers with Microsoft Windows 10 Education operating system and has software for developing a variety of applications.
- \* The Undergraduate Studies Lab II (B123) consists of 31 state-of-the-art computers with Microsoft Windows 10 Education operating system and has a variety of software and application development.
- \* The Digital Design and Microprocessor Laboratory (101) consists of 30 state-of-the-art computers with Linux operating system (CentOS 7). It provides digital logic and microprocessor cards, oscilloscopes, digital multimedia, signal generators, logic analyzers, integrated circuit controllers, and peripheral design devices. This equipment is used in hardware practice and in microcomputer system design and standard development.
- \* The Walk-in Laboratory (B101) in which staff and students can carry their laptop and connect to the Internet and the Department's computer systems.
- \* The Telelearning Room (148) is a lecture hall and includes 20 seats for the audience. It can be used: (i) to disseminate data to remote users, (ii) to broadcast lectures and other presentations live as well as on demand, (iii) to record lectures on a digital versatile disc (DVD).

More information is given through the websites of the Technical Support of the Computer Science Laboratories at: <http://its.cs.ucey.ac.cy/>

## ***Laboratory Equipment Use Regulations***

Students are required to respect the elementary rules of professional conduct regarding the hygiene and safety of public areas, as well as the responsible use of laboratory equipment. In particular, the following are not allowed:

- Access to computer systems with foreign code.
- Misuse of Email.
- Use of computer systems for purposes other than those appropriate (e.g., development of commercial products, user nuisance, etc.).
- Use of software products other than those provided by the Computer Centre or the Department's Laboratories, without the consent of the Centre or the Department, respectively.

- Attempting to access confidential information.
  - Copying software products owned by others, in violation of international copyright rules.
- More specific information regarding the rules of use and opening hours of the Laboratories is given to students at the beginning of the academic year.

### *Awards*

Each academic year, the Department of Computer Science awards academic performance awards to its undergraduate students, from external funders. For the academic year 2022/2023, the following awards were awarded:

- [1] "MetaQuotes Software Award" by MetaQuotes Ltd for undergraduate, postgraduate and undergraduate students with excellent performance (cash prize of €10000)
- [2] "Logicom (Public) Award" by Logicom Public Ltd for the final year undergraduate student with the highest academic performance (cash prize of €1000)
- [3] "JCC Award" in Memory of the late Director General of JCC Takis Fekkos for the senior undergraduate student with the second highest academic performance (cash Award of €500)
- [4] "HF Markets Award" by HF Markets (Europe) Ltd for the final year undergraduate student with the second highest academic performance (cash prize of €500)
- [5] "EY Excellence Award" by Ernst & Young Cyprus Ltd for the senior undergraduate student with the highest academic performance (cash prize of €500)
- [6] "Award of the Guild of Scientific Staff of the Cyprus Electricity Authority (SEPAEK)" by SEPAEK for the final year undergraduate student with the highest academic performance (cash Award of €350)

Also, the Department itself may award the following awards to its undergraduate students:

- [1] Special Award for the senior of the Minor Programme of Study in Computer Science with the highest overall academic performance in the courses of this minor programme (cash prize of €340).
- [2] Special Prize for the senior who demonstrated excellent social contribution (cash prize of €340)
- [3] Award for the senior year who demonstrated an excellent and commendable effort in memory of Polyvios Polyvios (cash prize of €500)
- [4] Award of Excellence in Memory of Professor Giorgos Samaras for the senior year with the highest academic performance combined in the compulsory course of the Databases, as well as with the highest SMO respectively (cash Award of €500)
- [5] Excellence of the Department of Computer Science, for the senior year of the Department of Computer Science with excellent performance (at least 8.5/10) (cash prize of €200/each).

Furthermore, the Department may award the following awards to its postgraduate students, as well as an award from an external funder, as follows:

- [1] "1st Prize of the Master's Programme of the Department of Computer Science", by the Department of Computer Science, for the final year postgraduate student with the highest academic performance (cash prize of €680)
- [2] "2nd Prize of the Master's Programme of the Department of Computer Science", by the Department of Computer Science, for the final year postgraduate student with the second highest academic performance (cash prize of €340)
- [3] "Marios Tsakalakis Award", in memory of the late Marios Tsakalakis (who served as a Systems Architect at the former Laiki Bank), on behalf of his wife Elli Tsakalaki and his family, for the final year postgraduate student with the highest academic performance (cash prize of €500)

Finally, the Department of Computer Science awards Honors to the 1st, 2nd and 3rd year excellent student respectively.

\*\* Please note that the Awards may vary from year to year, depending on the sponsors' suggestions.

## *Administrative Duties of Academic Staff*

### **Committees/Groups**

- [1] **Postgraduate Programme:** G. Sazeides (Coordinator), V. Vassiliou, E. Keravnou, Chr. Christodoulou
- [2] **Undergraduate Curriculum:** G. Pallis (Coordinator), E. Athanasopoulos, E. Konstantinou, A. Pieris
- [3] **Transfer and Correspondence Students:** G. Papadopoulos (Coordinator), G. Chrysanthos, G. Dimopoulos
- [4] **Erasmus+ Programme:** G. Kapitsaki (Coordinator), G. Dimopoulos, A. Philippou
- [5] **Communication and Promotion:** A. Pieris (Publications Coordinator), G. Pallis (Website Coordinator)
- [6] **"Logipaignion" Competition:** Rotation every year
- [7] **Computer Science Workshop:** Rotation every year
- [8] **Strategic Planning:** G. Dimopoulos (Coordinator), Chr. Georgiou, M. Dikaiakos, E. Keravnou, G. Pallis, G. Sazeides
- [9] **Liaison with Industry and Student Internship:** Ch. Volos (Coordinator) E. Athanasopoulos, M. Dikaiakos, P. Kolios
- [10] **Liaison with ERCIM:** A. Aristidou (Contact Point), G. Kapitsaki (Editorial Committee)
- [11] **Summer School:** V. Vassiliou, G. Chrysanthos, P. Kolios
- [12] **Contact Point with Computer Science Club:** G. Sazeides
- [13] **Computer Systems and Infrastructure Committee:** V. Vassiliou (Coordinator), E. Athanasopoulos, C. Volos, D. Zeinalipour
- [14] **Contact Point with Informatics Europe:** E. Keravnou

**Representatives/Coordinators:**

- [1] **Master in Cognitive Systems:** Chr. Christodoulou
- [2] **Master in Data Science:** G. Pallis
- [3] **Master in Artificial Intelligence:** E. Keravnou-Papailiou
- [4] **Library:** E. Constantinou
- [5] **Diploma Thesis:** M. Mavronicolas
- [6] **Study Guide:** A. Pieris
- [7] **Department Seminars:** D. Zeinalipour
- [8] **Teaching Assistants:** A. Pieris, G. Hadjipollas
- [9] **"ACM ICPC Cyprus" Competition:** Chr. Georgiou
- [10] **International Relations:** A. Aristidou
- [11] **YUFE (Young Universities for the Future of Europe):** G. Kapitsaki
- [12] **Contact Point with Secondary Education:** Chr. Georgiou
- [13] **Departmental Academic Support for Students with Disabilities:** G. Dimopoulos

## Undergraduate Programme of Studies

It is reminded that (at least) 240 ECTS credits are required for the acquisition of a Bachelor's Degree in Computer Science in the Undergraduate Curriculum.

### *Objectives and Prospects*

The Undergraduate Programme of Study leads to the acquisition of a Bachelor's Degree in Computer Science. The Department of Computer Science aspires to prepare graduates capable of pursuing careers in positions of high responsibility in the professional or academic field, where they will effectively promote the development and application of new methods and ideas. The Department attaches particular importance to its connection and continuous communication with the domestic industry and hopes to create, through its graduates, a channel with the Cypriot area.

Regardless of its subject, a curriculum should provide the student with an education in the broadest sense of the term but also cultivate his desire to continue learning at higher levels, thus acquiring maturity, independence of intellect and the ability to criticize. This overall objective exists in addition to and beyond the individual objectives of each programme.

Our graduates will be able to consolidate Computer Science both as a scientific field and as an occupation within the wider society. The Undergraduate Curriculum covers practical techniques, as well as the deeper theory on which they are based. Our graduates will be able to be employed by the domestic market as Information Technology scientists, continue their studies for a master's degree, or even specialize in specific areas of research at a research centre. It is also expected that they will be able to teach Computer Science in secondary education after successfully attending a pedagogical training programme. Regardless of the career that our graduates choose to pursue, the foundations they will have acquired regarding the deeper concepts of Computer Science will allow them to keep up with the rapid developments in the scientific and technological field.

The expected completion time of the Undergraduate Study Programme is eight semesters. This period may be extended to twelve semesters.

### *Course Areas*

The courses that make up the Undergraduate Curriculum are classified into the following areas or components: *Theory*, *Computer Systems*, *Problem Solving*, and *Applications*.

The area of *Theory* covers the basics of theory and models of computation, the design and analysis of algorithms and in general, intends to cultivate a typical way of thinking, organizing and processing information. It also introduces Mathematical Logic and the role it plays as Computational Computing. The necessary Discrete Mathematics is taught through the relevant courses. Moreover, students attend courses offered by the Department of Mathematics and Statistics, which enable them to further develop the ability to abstract and formal meditation and acquire other useful mathematical abilities.

The area of *Computer Systems* deals with system hardware and software and develops the concepts of parallel and embedded systems. It includes basic principles of computer organization and architecture, operating systems, design and implementation of programming languages, microprocessor systems, data transfer, networks, distributed systems, parallels and new architectures.

The area of *Problem Solving* aims to develop algorithmic thinking with an emphasis on programming principles and the design of algorithms. The acquisition of competence in the use of various programming languages is, of course, a key objective of this area. In addition, students are taught various programming models (procedural or imperative, object-oriented, logical). The elective courses in this area cover advanced solving techniques problems based on parallelism and simultaneity. This area facilitates an understanding of the techniques required to design, implement, and evaluate solutions to relatively small but important problems. These techniques are used in the context of a broader methodology needed to solve realistic problems. This topic is also covered in the area of Applications through the analysis of systems and design techniques.

The *Applications* area intends to combine the knowledge and skills gained from courses in other areas, in order to develop useful applications to solve realistic problems. Important technological constructs,

such as databases and knowledge bases, graphics and user-machine interface systems, are considered as applications in their own right, but also as tools for the development of higher-level applications. New software technology methodologies are examined that cover all stages of designing, developing, and maintaining high-quality applications. These methodologies are further used in the context of Professional Software Technology Practice. The factors that are important for the successful outcome of a project are also assessed, using current applications as examples. Finally, the main social and ethical issues regarding the relationship between Computer Science applications and society as a whole are raised and discussed.

The Undergraduate Curriculum includes *Compulsory Courses* that form its core, *Restricted Elective Courses*, which are offered by the Department and allow the student to focus on a specific specialization of Computer Science or to acquire knowledge that covers a wider scientific spectrum and *Free Choice Courses*, which are offered by other Departments. More specifically, the Undergraduate Curriculum includes a set of compulsory courses of the Department of Computer Science and compulsory courses from other Departments of the University of Cyprus (*Compulsory Courses*), 2 English language courses, 4 Free Options (*Free Choice Courses*), 5 Limited Options (*Restricted Elective Courses*), and a Diploma Project.

As of the academic year 2016/2017, the Department of Computer Science offers its students the following specializations:

- Computer Networks
- Fundamentals of Computer Science
- Big Data and Internet Computing
- Real World Computation
- Artificial Intelligence
- Software Engineering
- Digital and Embedded Systems

Each specialization is related to a set of courses. These specializations aim to give students the opportunity and motivation to focus, through the limited options they will choose and through their diploma project, on a specific area of Computer Science. In case a student chooses at least 3 limited options from the set of limited options of a specialization and prepares a Diploma Project from that specialization under the supervision of a member of the academic staff of the Department, then this specialization will be reflected in his/her analytical score. Requests for the recognition of specialization will be submitted in a special form to the Undergraduate Studies Committee after the additions of the students' last semester of studies. The Department is not obliged to offer, for the purposes of serving specializations, all Restricted Elective courses, nor will it change the limitations governing the audience sizes of the limited options and/or the subject selection process for the diploma thesis.

Some courses are a prerequisite for successful attendance of other courses. The dependencies between the lessons are shown in Table 1.

### ***Curriculum***

Course codes are of the XYZ format where X represents the level or type of course and Y represents the area to which it belongs. Courses offered for other Departments have level or type 0. The compulsory courses are 1st, 2nd or 3rd level or type, the limited options are 4th level or type, while the Diploma Project is 4th level or type. The areas have code 1 (Theory), 2 (Computer Systems), 3 (Problem Solving) and 4 (Applications). General courses have an area code of 0.

Below we list the programme offered by our Department after the last revision (June 2025), as well as all the courses related to each specialization.

## Computer Science Programme

Semester	Curriculum	ECTS
<b>First Semester</b>	CS 111 Discrete Structures in Computer Science and Computation	7,5
	CS 131 Programming Principles	7,5
	MAS 012 Calculus I	5
	LAN 100 General Advanced English	5
	Elective Course	5
<b>Second Semester</b>	CS 121 Digital Systems	7,5
	CS 132* Object Oriented Programming	7,5
	MAS 013 Calculus II	5
	MAS 029 Linear Algebra	5
	LAN 111 English for Computer Science	5
<b>Third Semester</b>	CS 221 Computer Organization and Assembly Programming	7,5
	CS 231 Data Structures and Algorithms	7,5
	CS 232 Programming Techniques and Tools	7,5
	MAS 055 Introduction to Probability and Statistics	7
<b>Fourth Semester</b>	CS 202 Explorations into Computer Science	3
	CS 211 Theory of Computation	7,5
	CS 222 Operating Systems	7,5
	CS 236 Algorithms and Complexity	7,5
	Elective Course	5
<b>Fifth Semester</b>	CS 324 Communications and Networks	7,5
	CS 342 Database Systems	7,5
	CS 343 Software Engineering	7,5
	Restricted Elective Course	7,5
<b>Sixth Semester</b>	CS 325 Parallel Processing	7,5
	CS 326 Systems Security	7,5
	CS 341 Artificial Intelligence	7,5
	Restricted Elective Course	7,5
<b>Seventh Semester</b>	CS 400 Diploma Project I	5
	Restricted Elective Course	7,5
	Restricted Elective Course	7,5
	BPA 369 Principles Entrepreneurship and Innovation	5
	Elective Course	5
<b>Eighth Semester</b>	CS 401 Diploma Project II	10
	Restricted Elective Course	7,5
	Restricted Elective Course	7,5
	Elective Course	5

---

\* Code changed from CS133 to CS132

### **Computer Network Specialization Elective Courses**

CS 421	Systems Programming
CS 422	Advanced Networks
CS 423	Network and Information Security
CS 425	Internet Technologies
CS 427	Mobile Computer Networks
CS 428	Internet of Things: Programming and Applications
CS 432	Distributed Algorithms
CS 450	Network Virtualisation and Management

### **Computer Science Foundations Specialization Elective Courses**

CS 412	Logic in Computer Sciences
CS 414	Basic Principles of Programming Languages
CS 431	Synthesis of Parallel Algorithms
CS 432	Distributed Algorithms
CS 433	Constraint Programming and Satisfaction

### **Big Data and Online Computing Specialization Elective Courses\***

CS 421	Systems Programming
CS 425	Internet Technologies
CS 446	Advanced Database Systems
CS 448	Data Mining on the Web
CS 450	Network Virtualisation and Management
CS 452	Datacenter Computing
MAS 458	Statistical Data Analysis

### **Real-World Computing Specialization Elective Courses**

CS 426	Computer Graphics
CS 435	Human Computer Interaction
CS 444	Computational Intelligent Systems
CS 445	Digital Image Processing
CS 447	Computer Vision

### **Artificial Intelligence Specialization Elective Courses**

CS 412	Logic in Computer Sciences
CS 433	Constraint Programming and Satisfaction
CS 434	Logic Programming and Artificial Intelligence
CS 442	Machine Learning
CS 444	Computational Intelligent Systems
CS 445	Digital Image Processing
CS 447	Computer Vision
CS 448	Data Mining on the Web

### Software Technology Specialization Elective Courses

CS 421	Systems Programming
CS 425	Internet Technologies
CS 435	Human Computer Interaction
CS 441	Advanced Software Engineering
CS 443	Software Reuse
CS 449	Professional Practice in Software Engineering
CS 484	Software Evolution

### Digital and Embedded Systems Specialization Elective Courses

CS 420	Computer Architecture
CS 421	Systems Programming
CS 424	Digital Signal Processing
CS 428	Internet of Things: Programming and Applications
CS 429	Theory and Practice of Compilers
CS 445	Digital Image Processing
CS 452	Datacenter Computing

\* Students of the *Big Data and Online Computing* Specialization can choose the “MAS458 Statistical Data Analysis” course from the Department of Mathematics and Statistics, which will reduce the credits of the programme by 0.5 credits, since the Restricted Elective courses of the above department are 7 credits while Computer Science Department courses are 7.5 credits. For the smooth implementation of the specialization, students are encouraged to accumulate an additional 0.5 credits in free elective courses.

With regard to the distribution of the free elective and Restricted Elective courses of the 4<sup>th</sup> year in each semester of study, if the student so wishes, he/she can choose the following alternative program:

**7<sup>th</sup> Semester:** Diploma Project I  
3 Restricted Elective Courses  
BPA 369 Principles of Entrepreneurship and Innovation

**8<sup>th</sup> Semester:** Diploma Project II  
1 Restricted Elective Course  
2 Free Elective Courses

### ***Restricted Elective Courses***

Each student, in consultation with his/her Academic Advisor, selects the Restricted Elective Courses based on his/her interests and professional goals. The student can choose to specialize in a specific area by following the proposed specializations or choose a combination of more than one area. The Restricted Elective Courses can, therefore, be selected to satisfy to some significant extent the goals and inclinations of each student. In addition, there is the possibility of recognizing a course offered in the Department's Postgraduate Curriculum as a Limited Choice Course for the student, following relevant approval by the Committee of the Undergraduate Curriculum.

### ***Elective Courses***

Each student, in consultation with his/her Academic Advisor, chooses the Free Choice Courses, based on his/her particular interests and goals, which are offered by other Departments. According to the University's Undergraduate Studies Rule, for Free Choice courses, students must choose at least 20 credits of free choice courses from at least three different Faculties (see the relevant table below). Also, please note that the courses offered by the Language Centre, the Sports Council and the Entrepreneurship Centre will be considered to belong to their own independent schools. .

As far as foreign language courses **are concerned, only a** first-level foreign language course can be counted as a free choice course, **unless** the student has passed the second level of the same language, in which case both levels will be counted as free elective courses.

<b>Faculties</b>	<b>Departments</b>
<b>Faculty of Humanities</b>	<ul style="list-style-type: none"> <li>• Department of English Studies</li> <li>• Department of French and European Studies</li> <li>• Department of Turkish and Middle Eastern Studies</li> </ul>
<b>Medical School</b>	
<b>Faculty of Pure and Applied Sciences</b>	<ul style="list-style-type: none"> <li>• Department of Biological Sciences</li> <li>• Department of Mathematics and Statistics</li> <li>• Department of Computer Science</li> <li>• Department of Physics</li> <li>• Department of Chemistry</li> </ul>
<b>Faculty of Social Sciences and Education</b>	<ul style="list-style-type: none"> <li>• Department of Education</li> <li>• Department of Social and Political Sciences</li> <li>• Department of Law</li> <li>• Department of Psychology</li> </ul>
<b>Faculty of Economics and Management</b>	<ul style="list-style-type: none"> <li>• Department of Business Administration and Public Administration</li> <li>• Department of Accounting and Finance</li> <li>• Department of Finance</li> </ul>
<b>Faculty of Engineering</b>	<ul style="list-style-type: none"> <li>• Department of Architecture</li> <li>• Department of Electrical and Computer Engineering</li> <li>• Department of Mechanical and Manufacturing Engineering</li> <li>• Department of Civil and Environmental Engineering</li> </ul>
<b>Faculty of Letters</b>	<ul style="list-style-type: none"> <li>• Department of Byzantine and Modern Greek Studies</li> <li>• Department of History and Archaeology</li> <li>• Department of Classics and Philosophy</li> </ul>

**Other Entities:**

- Language Centre
- Sports Council and
- Entrepreneurship Centre

***Foreign Language Courses***

Each student must successfully attend two courses in a foreign language. The Department has allocated 10 ECTS credits for these courses and identifies English as the foreign language.

***Diploma Project***

During the last two semesters of study, each student prepares a Diploma Project according to rules approved by the Department Council (Session 19/7/95). These rules have been revised by the Department Council (Session 06/12/2010) and constitute Annex A. Diploma Projects are registered upon completion in the Open Access Digital Library (<http://godigital.cs.ucy.ac.cy>).

***Minor Programme in Computer Science***

The Minor Programme in Computer Science is open to all students of the University outside the Department of Computer Science. It consists of 8 courses with a total workload of at least 60 credits. Given that the study will begin in the Spring Semester, the programme can be completed in four consecutive semesters by attending 2 courses per semester. The organization of the lessons is as follows:

### **1<sup>st</sup> Semester** (considered as Spring Semester)

CS 131 Programming Principles

CS 111 Discrete Structures in Computer Science and Computation

### **2<sup>nd</sup> Semester**

CS 132 Object Oriented Programming

CS 121 Digital Systems

### **3<sup>rd</sup> Semester**

CS 231 Data Structures and Algorithms

One course from the Compulsory Courses or the Restricted Elective Courses of the Computer Science Curriculum

### **4<sup>th</sup> Semester**

Two courses from the Compulsory Courses or the Restricted Elective Courses of the Computer Science Curriculum.

The number of students admitted to the Minor Programme in Computer Science is 10. A necessary criterion for admission is that the weighted average from all previous semesters is at least 6.5.

## ***Minor Programme in Biomedical Engineering***

The Minor Programme in Biomedical Engineering (BM) is open to all students of the University. It consists of 10 courses with a total workload of at least 60 credits. Due to unevenness in the number of credits between participating departments, in special cases an exception will be made for 10 courses of the programme that accumulate 57 or more credits. Students must complete the secondary degree within the 8-12 semesters provided for the completion of a regular degree. The choice of courses can be made between the courses offered in the Winter and Spring semesters:

### **Fall Semester**

BIO 102	Principles of Biology
BIO 230	Introduction to Computational Biology
BIO 442	Internship in Biology
BIO 495, 496, 497, 498, 499	Current Topics in Biology
BIO 491/492 (ECE 623)	Undergraduate Thesis
CS 428	Internet of Things: Programming and Applications
CS 434	Logical Programming and Artificial Intelligence
CS 435	Human-Computer Interaction
CS 442	Machine Learning
CS 445	Digital Image Processing
ECE 421	Intelligent Systems (or ECE 634)
ECE 429	Digital Signal Processing (or ECE 623)
ECE 434	Introduction to Photonics
ECE 473	Instrumentation and Sensors (or ECE 665)
ECE 476	Biomedical Imaging (or ECE 6xx)
MME 420	Robotics
MME 435	Introduction to Biological and Biomedical Engineering
BMT XXX	Supervised Biomedical Engineering Study
BMT XXX	Specialized Topics in Biomedical Technology Engineering

## Spring Semester

BM XXX	Supervised Biomedical Engineering Study
BM XXX	Specialized Topics in Biomedical Engineering Technology

Students can enrol in up to 2 of the following master's courses:

BIO 630	Nucleic Acids
BIO 650	Special Topics in Bioinformatics
BIO 670	Imaging in Biological Sciences
CS 667 / MAI 647	Computational Neuroscience
CS 668	Mechanical Vision (similar to HMY 627)
CS 679	Electronic Health
ECE 623	Digital Signal Processing
ECE 626	Image Processing
ECE 627	Machine Vision (similar to CS 668)
ECE 634	Introduction to Computational Intelligence
ECE 645	Optics and Photonics
ECE 665	Instrumentation and Sensors
ECE 671	Neurophysiology and the Senses
ECE 6XX	Biomedical Imaging
MME 555	Polymers in Medical Applications
MME 531	Continuum Mechanics
MME 532	Biomaterials in Tissue Engineering and Regenerative Medicine

For the acquisition of a secondary degree, students must complete at least 5 additional courses that have not been counted towards the acquisition of their main degree. One of the elective courses of the Minor Programme in Biomedical Engineering may be one of the courses "Supervised Study of Biomedical Engineering" or "Specialized Topics in Biomedical Technology Engineering".

**Table 1: Dependencies among Courses**

<b>Code</b>	<b>Course</b>	<b>Prerequisites / Conditions</b>
CS 111	Discrete Structures in Computer Science and Computation	
CS 121	Digital Systems	
CS 131	Programming Principles	
CS 132	Object Oriented Programming	CS 131 Programming Principles
CS 202	Explorations into Computer Science	
CS 211	Theory of Computation	CS 111 Discrete Structures in Computer Science and Computation MAS 012 Calculus I
CS 221	Computer Organization and Assembly Programming	CS 121 Digital Systems
CS 222	Operating Systems	CS 221 Computer Organization and Assembly Programming CS 232 Programming Techniques and Tools
CS 231	Programming Techniques and Tools	CS 111 Discrete Structures in Computer Science and Computation CS 132 Object Oriented Programming
CS 232	Programming Techniques and Tools	CS 131 Programming Principles
CS 236	Algorithms and Complexity	CS 231 Programming Techniques and Tools
CS 324	Communications and Networks	CS 131 Programming Principles
CS 325	Parallel Processing	CS 221 Computer Organization and Assembly Programming CS 232 Programming Techniques and Tools
CS 326	Systems Security	CS 232 Programming Techniques and Tools
CS 341	Artificial Intelligence	CS 231 Data Structures and Algorithms
CS 342	Database Systems	CS 231 Data Structures and Algorithms
CS 343	Software Engineering	CS 132 Object Oriented Programming
CS 400 –CS 401	Diploma Project	Academic Advisor Approval At least 156 ECTS
CS 412	Logic in Computer Sciences	CS 111 Discrete Structures in Computer Science and Computation
CS 414	Basic Principles of Programming Languages	CS 211 Theory of Computation CS 231 Programming Techniques and Tools
CS 420	Computer Architecture	CS 222 Operating Systems
CS 421	Systems Programming	CS 222 Operating Systems
CS 422	Advanced Networks	CS 324 Communications and Networks
CS 423	Network and Information Security	CS 324 Communications and Networks CS 326 Systems Security
CS 424	Digital Signal Processing	CS 111 Discrete Structures in Computer Science and Computation MAS 029 Linear Algebra MAS 012 Calculus I
CS 425	Internet Technologies	CS 132 Object Oriented Programming CS 324 Communications and Networks
CS 426	Computer Graphics	CS 232 Programming Techniques and Tools
CS 427	Mobile Computer Networks	CS 324 Communications and Networks
CS 428	Internet of Things: Programming and Applications	CS 222 Operating Systems
CS 429	Theory and Practice of Compilers	CS 211 Theory of Computation CS 231 Programming Techniques and Tools
CS 431	Synthesis of Parallel Algorithms	CS 231 Programming Techniques and Tools
CS 432	Distributed Algorithms	CS 211 Theory of Computation CS 231 Programming Techniques and Tools

CS 433	Constraint Programming and Satisfaction	CS 111 Discrete Structures in Computer Science and Computation CS 231 Programming Techniques and Tools
CS 434	Logic Programming and Artificial Intelligence	CS 111 Discrete Structures in Computer Science and Computation
CS 435	Human Computer Interaction	
CS 441	Advanced Software Engineering	CS 343 Software Engineering
CS 442	Machine Learning	CS 231 Programming Techniques and Tools
CS 443	Software Reuse	CS 132 Object Oriented Programming CS 343- Software Engineering
CS 444	Computational Intelligent Systems	
CS 445	Digital Image Processing	CS 231 Programming Techniques and Tools MAS 029 Linear Algebra
CS 446	Advanced Database Systems	CS 342 Database Systems
CS 447	Computer Vision	CS 231 Programming Techniques and Tools MAS 029 Linear Algebra
CS 448	Data Mining on the Web	CS 342 Database Systems
CS 449	Professional Practice in Software Engineering	CS 343 Software Engineering
CS 450	Network Virtualisation and Management	CS 324 Communications and Networks
CS 451	Software Analysis	CS 211 Computer Organization and Assembly Programming CS 232 Programming Techniques and Tools
CS 452	Datacenter Computing	CS 222 Operating Systems
CS 484	Software Evolution	CS 343 Software Engineering
CS 499	Special Issues in Computer Science: Mobile Computing Systems	<i>Depending on the offered subject</i>
CS 500	Industrial Placement	CS 342 Database Systems CS 343 Software Engineering
MAS 013	Calculus II	MAS 012 Calculus I
LAN 111	English for Computer Science	LAN 100 General Advanced English or equivalent

## Short Course Description

Each description shows the name of the instructor offering the course in the academic year 2025/2026 or the instructor who suggested the course for Restricted Elective courses not offered during the academic year 2025/2026. The language of instruction of all courses is Greek, unless otherwise stated in the course description (a small number of courses are offered in English).

Course Title	<b><i>Discrete Structures in Computer Science and Computation</i></b>						
Course Code	<b><i>CS 111</i></b>						
Course Type	Compulsory						
Level	Undergraduate						
Year / Semester	1st year/1st semester						
Teacher's Name	A. Pieris						
ECTS	7,5	Lectures / week	2 x 1.5 hours	Recitation / week	1 x 1 hour	Laboratories / week	1 x 1.5 hours
Course Purpose and Objectives	Introduction to basic concepts of discrete mathematics and how they are applied to Computer Science. Developing a mathematical way of thinking.						
Learning Outcomes	<p>After the completion of the course students will have acquired the following knowledge and skills:</p> <ul style="list-style-type: none"> <li>• Understand and use Propositional and Predicate Logic.</li> <li>• Concepts of logical conclusion and logical proof of conclusion.</li> <li>• Basic knowledge of sets and functions.</li> <li>• Concept of countability of infinite sets and perception of the existence of non-computable problems (via the concept of uncountability of infinite sets).</li> <li>• Understand and use mathematical induction (weak induction, strong induction, structural induction).</li> <li>• Understand and use basic principles of counting, permutations, and combinations.</li> <li>• Set relations and partial orders with applications to Computer Science.</li> <li>• Basic concepts of graph theory.</li> </ul>						
Prerequisites	None		Required		None		
Course Content	Logic: Propositional and Predicate Logic. Mathematical reasoning: inference rules, proof methods. Foundational concepts: sets and functions. Countability and computability. Mathematical induction: weak induction, strong induction, structural induction. Counting principles: sum rule, product rule, inclusion-exclusion principle, pigeonhole principle. Permutations and combinations. Set relations: properties and applications, equivalence relations, partial orders. Basic concepts of graph theory.						
Teaching Methodology	Lectures (3 hours weekly), Recitation (1 hour weekly), Laboratory sessions (1.5 hours weekly).						
Bibliography	1. K. Rosen, <i>Discrete Mathematics and its Applications</i> , 5 <sup>th</sup> Edition, McGraw-Hill, 2003.						
Assessment	Final exam, quizzes and homework.						
Language	Greek						

Course Title	<b>Digital Systems</b>						
Course Code	<b>CS 121</b>						
Course Type	Compulsory						
Level	Undergraduate						
Year / Semester	1st year/2nd semester						
Teacher's Name	C. Pattichis / P. Kolios						
ECTS	7,5	Lectures / week	2 x 1.5 hours	Recitation / week	1 x 1 hour	Laboratories / week	1 x 1.5 hours
Course Purpose and Objectives	Introduction to the basic principles for the design and operation of digital systems, the organization and operation of a simple computer and programming at the assembly level.						
Learning Outcomes	<ul style="list-style-type: none"> <li>• Familiarization with basic trends of computer technology</li> <li>• Understanding methods for representing digital information</li> <li>• Introduction to basic design and analysis of circuits with combinational and sequential logic</li> <li>• Understanding the basic functional units needed to build a computer</li> <li>• Familiarization with basic concepts and methods for organizing a simple computer</li> <li>• Introduction of the concept of Instruction Set Architecture</li> <li>• Introduction and familiarization with assembly level programming</li> </ul>						
Prerequisites	None		Required		None		
Course Content	Principles of design and construction of digital electronic systems and computers. Representation of data with binary sequences. Data storage and processing by electronic digital circuits. Consolidation of theoretical knowledge through practical exercises in the design and construction of digital circuits in the laboratory for Digital Systems Design and Microprocessors.						
Teaching Methodology	Lectures (3 hours weekly), Recitation (1 hour weekly) and Laboratory sessions (1.5 hours weekly).						
Bibliography	<ol style="list-style-type: none"> <li>1. D. A. Patterson, J. L. Hennessy, Computer Organization and Design, 5th Edition: The Hardware/Software Interface (The Morgan Kaufmann Series in Computer Architecture and Design) Paperback, 2013.</li> <li>2. M. Morris Mano, Charles Kime, Tom Martin, Logic &amp; Computer Design Fundamentals, 5th Edition, Pearson, 2015.</li> </ol>						
Assessment	Final exam, midterm exam, quizzes and homework (final project and exercises).						
Language	Greek						

Course Title	<b>Programming Principles</b>						
Course Code	<b>CS 131</b>						
Course Type	Compulsory						
Level	Undergraduate						
Year / Semester	1st year/1st semester						
Teacher's Name	Y. Sazeides / E. Keravnou-Papailiou						
ECTS	7,5	Lectures / week	2 x 1.5 hours	Recitation / week	1 x 1 hour	Laboratories / week	1 x 1.5 hours
Course Purpose and Objectives	Introduction of methods for problem-solving through programming. Development of procedural and object-oriented problem solving skills and algorithmic thinking. Provision of deep understanding of basic programming principles and algorithmic techniques, design, implementation, testing and debugging of modular programs. Understanding the important concepts of program abstraction and data abstraction. Mastering of a high-level programming language (Java).						
Learning Outcomes	<ul style="list-style-type: none"> <li>• Understanding basic algorithmic structures.</li> <li>• Understanding fundamental programming concepts.</li> <li>• Grasping principles of good programming, algorithmic techniques and program structures.</li> <li>• Becoming familiar with the concepts of program abstraction (functions, methods), data abstraction (abstract data types), reusability, modularity and hierarchical structuring.</li> <li>• Developing skills in algorithmic problem solving using structured procedural and object-oriented thinking.</li> <li>• Developing competence in designing, implementing, testing, debugging and documenting computer programs.</li> <li>• Being able to reason about the correctness, behavior and quality of algorithmic solutions.</li> <li>• Being able to evaluate the extensibility potential of programs.</li> <li>• Developing algorithmic thinking, independently of any particular programming language.</li> <li>• Developing competence in using the Java high level programming language.</li> </ul>						
Prerequisites	None		Required		None		
Course Content	Presentation of the software development process and introduction to the basic principles of programming and program design using the Java programming language. Global overview of the Java programming language with emphasis on built-in and abstract data types, control structures, functions, modular programming and reusability. Use of recursion.						
Teaching Methodology	Lectures (3 hours weekly), Recitation (1 hour weekly) and Laboratory sessions (1.5 hours weekly).						
Bibliography	1. R. Sedgewick and K. Wayne, Introduction to Programming in Java: An Interdisciplinary Approach, Addison Wesley, 2008						
Assessment	Final exam, midterm exam, homework (programming assignments) and quizzes.						
Language	Greek						

Course Title	<b>Object Oriented Programming</b>						
Course Code	<b>CS 132</b>						
Course Type	Compulsory						
Level	Undergraduate						
Year / Semester	1st year/2nd semester						
Teacher's Name	Chr. Christoforou						
ECTS	7,5	Lectures / week	2 x 1.5 hours	Recitation / week	1 x 1 hour	Laboratories / week	2 x 1.5 hours
Course Purpose and Objectives	The course aims to teach the fundamentals of object-oriented analysis, design, and programming using the Java programming language. The main goal of the course is to familiarize students with the principles of Object-Oriented Programming (OOP), to help them develop proficiency in using the Object-Oriented Methodology and the Java language to solve computational problems, to apply advanced programming techniques, and to address complex problems. Students will gain experience in large-scale software development and teamwork.						
Learning Outcomes	<p>A student successfully completing the course should:</p> <ul style="list-style-type: none"> <li>• Be able to explain and apply adequately the fundamental principles of object-oriented programming and their implementation in the JAVA programming language.</li> <li>• Be able to understand and explain adequately the Java memory management model and the concepts of stack, heap and garbage collection.</li> <li>• Be able to analyze computational problems and invent object-oriented solutions to computational problems described in natural language using object-oriented design techniques.</li> <li>• Be able to describe the plan to solve a problem in an object-oriented manner, using natural language and UML diagrams.</li> <li>• Be fully familiar with the syntax, semantics and textbooks of the Java API.</li> <li>• Be able to code the solution of a computational problem using the Java language.</li> <li>• Be able to develop, document, test, and debug Java programs efficiently and effectively to implement algorithms in an efficient manner.</li> <li>• Addresses complex computational problems using object-oriented programming.</li> <li>• Use the Integrated Development Environment Eclipse as well as editors like Sublime, Atom or Emacs.</li> <li>• Code JAVA programs with the appropriate programming style.</li> <li>• Be able to collaborate effectively with other developers for problem analysis and program coding, using techniques such as pair programming and tools such as the git publishing control system and the Github and Stackoverflow online services.</li> <li>• Communicate the findings of his work (object-oriented drawings, Java programs) to other developers in an effective manner.</li> <li>• Has a good knowledge of English terminology of object-oriented programming.</li> </ul>						
Prerequisites	CS 131	Required			None		
Course Content	Analysis and Design of Computational Problems. Abstraction. Invariants. Overview of the Java Programming Environment. The Java Virtual Machine (JVM). Compilation and Interpretation. Application Programming Interface (API). Commenting and documenting Java code. Pair programming. Integrated Development Environments, debugging tools, performance evaluation. Version control tools: git, GitHub. Introduction to Object-Oriented Design and Methodologies and Unified Modeling Language (UML). Inheritance. Polymorphism. Java Interfaces and Inner Classes. Exceptions and Exception Handling. Files and Streams. Generic Classes. Linked Structures. Collections and Iterators. Lambda functions.						
Teaching Methodology	Lectures (3 hours weekly), Recitation (1 hour weekly) and Laboratory sessions (3 hours weekly).						
Bibliography	<ol style="list-style-type: none"> <li>1. "Absolute Java" Walter Savitch. 6th Edition (Global Edition). Pearson 2016.</li> <li>2. "Object-oriented Design and Patterns" Cay Horstmann, 2nd edition. Wiley, 2006.</li> <li>3. "Modern Java in Action" R-G. Urma, M. Fusco, A. Mycroft. Manning, 2021.</li> <li>4. "Java Tutorials" Oracle.</li> <li>5. "The Practice of Programming." B. Kernighan and R. Pike, Addison Wesley, 1999.</li> </ol>						
Assessment	Final exam, midterm exam and homework (programming assignments).						
Language	Greek						

Course Title	<i>Explorations into Computer Science</i>						
Course Code	<i>CS 202</i>						
Course Type	Compulsory						
Level	Undergraduate						
Year / Semester	2nd year/2nd semester						
Teacher's Name	CS or Visiting Faculty						
ECTS	3	Lectures / week	1 x 2 hours	Recitation / week	0	Laboratories / week	0
Course Purpose and Objectives	Introduction to topics that compose a global picture of Computer Science. Creation of enthusiasm and interest in Computer Science. Update about current developments in Computer Science. Familiarization with practical applications of Computer Science.						
Learning Outcomes	<ul style="list-style-type: none"> <li>• Understanding the importance and impact of Computer Science in the modern world.</li> <li>• Developing communication skills through digital and networking tools.</li> <li>• Information about the professions and IT specializations.</li> <li>• Professional and scientific orientation of students.</li> <li>• Familiarization with the technological infrastructure, the research process and the research work of the Department of Computer Science.</li> </ul>						
Prerequisites	None		Required		None		
Course Content	Weekly lecturers/seminars that cover a broad spectrum of Computer Science and its basic areas, starting from its birth and reaching its modern evolutions. Revolutionary ideas for the foundation and development of Computer Science.						
Teaching Methodology	Lectures (2 hours weekly).						
Bibliography							
Assessment	Group project and presentation, class participation and attendance.						
Language	Greek						

Course Title	<i>Theory of Computation</i>						
Course Code	<b>CS 211</b>						
Course Type	Compulsory						
Level	Undergraduate						
Year / Semester	2nd year/2nd semester						
Teacher's Name	M. Mavronicolas						
ECTS	7,5	Lectures / week	2 x 1.5 hours	Recitation / week	1 x 1 hour	Laboratories / week	0
Course Purpose and Objectives	Introduction to foundational concepts of the Theory of Computation. Development and cultivation of formal and mathematical-based reasoning. Familiarization with fundamental techniques for proofs and mathematical reasoning. Realization of the limitations on the capabilities of computers.						
Learning Outcomes	<p>Upon successful completion of the course, students should be able to:</p> <ul style="list-style-type: none"> <li>• Appreciate the concept of computation and its capabilities and limitations.</li> <li>• Be familiar with basic language theory concepts on which subjects such as Programming Language Theory and Algorithmics are supported.</li> <li>• Be familiar with (i) finite deterministic and non-deterministic automata as well as with the class of regular languages, (ii) context-free grammars, push-down automata, and the class of context-free languages, and (iii) Turing machines.</li> <li>• Categorize problems and construct appropriate solutions using suitable abstract machines.</li> <li>• Recognize the importance of complexity classes (classes P, NP, NP-completeness) and calculate the complexity of algorithms</li> </ul>						
Prerequisites	CS 111, MAS 012		Required		None		
Course Content	Mathematical concepts of computation, computer, algorithm and universality. Universal models of computation (such as Turing machines, logical circuits, etc.) and their computational equivalence. The Church-Turing Thesis. Time, space and non-determinism as computational resources. Non-universal (or restricted) models of computers (such as finite automata and context-free grammars) and proofs of non-universality. Reductions among computational problems and the concepts of equivalence, hardness, completeness and algorithmic solvability. Unsolvability proofs via reductions. Time complexity. Polynomial-time reductions and polynomial-time equivalence, hardness and completeness. The class NP, NP-hardness and NP-completeness. NP-complete problems and proofs of NP-completeness via polynomial-time reductions. The class P.						
Teaching Methodology	Lectures (3 hours weekly) and Recitation (1 hour weekly).						
Bibliography	<ol style="list-style-type: none"> <li>1. M. Sipser, Εισαγωγή στη Theory of Computation, Πανεπιστημιακές Εκδόσεις Κρήτης, 2007.</li> <li>2. H. R. Lewis and X. Παπαδημητρίου, Στοιχεία Θεωρίας Υπολογισμού, Εκδόσεις Κριτική, Φεβρουάριος 2005.</li> </ol>						
Assessment	Final exam, midterm exam, homework.						
Language	Greek						

Course Title	<b>Computer Organization and Assembly Programming</b>						
Course Code	<b>CS 221</b>						
Course Type	Compulsory						
Level	Undergraduate						
Year / Semester	2nd year/1st semester						
Teacher's Name	Y. Sazeides						
ECTS	7,5	Lectures / week	2 x 1.5 hours	Recitation / week	1 x 1 hour	Laboratories / week	1 x 1.5 hours
Course Purpose and Objectives	Εισαγωγή στις βασικές αρχές οργάνωσης, λειτουργίας, χαμηλού επιπέδου προγραμματισμός, and ανάλυση απόδοσης υπολογιστικών συστημάτων						
Learning Outcomes	<ul style="list-style-type: none"> <li>• Familiarization with the concepts of interrupts and exceptions and introduction to the low-level programming of kernel handlers</li> <li>• Introduction to the basic organization of a pipelined datapath and familiarization with mechanisms for hazard detection and the concept of control flow predictability</li> <li>• Introduction to the basic organization of a hierarchical memory system and virtual memory, and familiarization with the concept of memory locality</li> <li>• Understanding for methods used for performance measurement and analysis of computer systems and introduction to benchmarking</li> <li>• Appreciation for techniques used to improve the performance of cores found in modern central processing units and introduction to multicores</li> </ul>						
Prerequisites	CS 121		Required		None		
Course Content	Introduction to computer organization and architecture. Programming in MIPS assembly language and introduction to x86. Low level programming interfacing of CPU and Input/Output units. Pipelined data path, dependencies and hazards, forwarding and control flow prediction. Memory hierarchy, locality, caching, virtual memory, page tables and translation-look-aside buffers. Benchmarking, performance monitoring and performance analysis. Basics of modern out-of-order superscalar cores and multi-cores						
Teaching Methodology	Lectures (3 hours weekly), Recitation (1 hour weekly) and Lab (1.5 hours weekly).						
Bibliography	<ol style="list-style-type: none"> <li>1. D. A. Patterson, J. L. Hennessy, Computer Organization and Design, 4th Edition: The Hardware/Software Interface (The Morgan Kaufmann Series in Computer Architecture and Design) 2013. (Greek or English edition)</li> <li>2. Computer Organization and Design MIPS Edition: The Hardware/Software Interface (The Morgan Kaufmann Series in Computer Architecture and Design) 6th Edition, 2020</li> </ol>						
Assessment	Final exam, midterm exam, homework						
Language	Greek						

Course Title	<b><i>Operating Systems</i></b>						
Course Code	<b><i>CS 222</i></b>						
Course Type	Compulsory						
Level	Undergraduate						
Year / Semester	2nd year/2nd semester						
Teacher's Name	G. Papadopoulos						
ECTS	7,5	Lectures / week	2 x 1.5 hours	Recitation / week	1 x 1 hour	Laboratories / week	1 x 1.5 hours
Course Purpose and Objectives	Introduction to the basic principles of design and operation of modern operating systems. Familiarization with the various operation levels and mechanisms, case studies involving typical operating systems like Linux and Windows as well as the dual role of the operating system, as manager of the various parts of the computer hardware and supplier of offered services to the user.						
Learning Outcomes	<p>At the end of the course the students are expected to be able:</p> <ul style="list-style-type: none"> <li>• Understand the basic principles governing the operation of an operating system.</li> <li>• Associate the functions of an OS and how applications are running on it.</li> <li>• Be aware of the basic services provided to users by a typical operating system.</li> <li>• Compare similar mechanisms and services provided by different modern operating systems.</li> </ul>						
Prerequisites	CS 221, CS232			Required		None	
Course Content	Introduction, history and evolution of operating systems. General structure, operations and characteristics of an operating system. Concurrency. Process management. CPU scheduling and dispatch. Real and virtual memory management. I/O interface and management. Disk scheduling. File system interface and management. Reliability.						
Teaching Methodology	Lectures (3 hours weekly), Recitation (1 hour weekly), and Laboratory sessions (1.5 hours weekly)						
Bibliography	1. W. Stallings, Operating Systems: Internals and Design Principles, 9th Edition, Prentice Hall, 2018.						
Assessment	Final exam, midterm exam and homework (theoretical and programming assignments).						
Language	Greek						

Course Title	<b>Programming Techniques and Tools</b>						
Course Code	<b>CS 231</b>						
Course Type	Compulsory						
Level	Undergraduate						
Year / Semester	2nd year/1st semester						
Teacher's Name	G. Pallis						
ECTS	7,5	Lectures / week	2 x 1.5 hours	Recitation / week	1 x 1 hour	Laboratories / week	1 x 1.5 hours
Course Purpose and Objectives	Familiarization with data structures and the algorithms manipulating them. Appreciation of the importance of careful organization of information for efficient searching and modification. Acquaintance with techniques for the analysis of algorithm efficiency. Developments of capabilities for designing algorithms to minimize their execution time and space requirements.						
Learning Outcomes	<ul style="list-style-type: none"> <li>• Ability to analyse and compare the algorithmic complexity (<math>O</math>, <math>\Omega</math>, <math>\Theta</math>).</li> <li>• Understanding of fundamental Data Structures including linked-lists, trees, binary search trees, AVL trees, stacks, queues, priority queues, graphs and hash-tables.</li> <li>• Understanding of fundamental abstract data types.</li> <li>• Ability to devise novel solutions to small scale programming challenges involving data structures and recursion.</li> <li>• Ability to sensibly select appropriate data structures and algorithms for problems and to justify that choice.</li> <li>• Understanding the social and ethical implications of algorithm design and implementation and apply these principles to develop socially responsible solutions.</li> <li>• Ability to program data structures and use them in implementations of abstract data types in JAVA.</li> </ul>						
Prerequisites	CS 111, CS 132			Required		None	
Course Content	Algorithm complexity and analysis of average and worst-case scenarios. Data types and abstract data types. Types of lists, stacks, and queues. Non-linear data structures. Trees. Search trees. Balanced trees. Bit-vectors. Hashing techniques. Space vs time trade-offs. Iteration vs recursion. Priority queues. Sorting algorithms and analysis of their efficiency. Graphs and graph processing algorithms.						
Teaching Methodology	Lectures (3 hours weekly), Recitation (1 hour weekly) and Laboratory sessions (1.5 hours weekly).						
Bibliography	<ol style="list-style-type: none"> <li>1. Data Structures and Algorithm Analysis in Java, 3rd edition Published by Pearson (2021) © 2012</li> <li>2. M. Goodrich and R. Tamassia, Data Structures and Algorithms in Java, Wiley, 2011.</li> </ol>						
Assessment	Final exam, midterm exam (or quizzes) and homework (theoretical and programming assignments)						
Language	Greek						

Course Title	<b>Programming Techniques and Tools</b>						
Course Code	<b>CS 232</b>						
Course Type	Compulsory						
Level	Undergraduate						
Year / Semester	2nd year/1st semester						
Teacher's Name	G. Chrysanthou						
ECTS	7,5	Lectures / week	2 x 1.5 hours	Recitation / week	1 x 1 hour	Laboratories / week	1 x 1.5 hours
Course Purpose and Objectives	The course teaches intermediate and advanced programming concepts, techniques and tools through a language that compiles to machine code. The course familiarizes the students with advanced programming constructs utilized for handling memory and files. Advanced topics in compilation, debugging, documentation and optimization of software. Methodological aspects in developing large-scale system software that addresses complex problems. Basic commands for programmers in the UNIX operating system.						
Learning Outcomes	<p>Upon successful completion of the course, the student will be able to:</p> <ul style="list-style-type: none"> <li>• Design, write and test moderately complicated low-level programs using a systems programming language, like C, which compiles to machine code.</li> <li>• Describe how computer and operating system executes and switches programs. Gain an insight into the anatomy of a computer program, understanding its physical limits imposed by the operating system.</li> <li>• Proficiently use a preprocessor to implement code that is portable between different computing platforms.</li> <li>• Use operating system kernel calls from within a programming language to allocate/free virtual memory and exploit the use of pointers to improve efficiency. Implement routines of complex data structures which superimpose arrays, records, and references on unstructured blocks of memory.</li> <li>• Know how to use developer tools for compilation, debugging, documentation and optimization of software. Particularly, compile multiple files with makefiles, link object files statically and dynamically, handle errors, analyze code statically and dynamically (valgrind and gprof) and collaborate with team members through versioning systems (SVN and GIT).</li> <li>• Implement routines that read and write structured binary files such as images, audio files, word processing documents or data from index systems.</li> <li>• Understand how to use basic commands for programmers in the UNIX operating system: file system, redirection and pipes, permissions and basic filters.</li> </ul>						
Prerequisites	CS 131			Required		None	
Course Content	i) Introduction to C for Programmers: types x86/x64, loops, selections, expressions, arrays, functions, IO, basic program organization, ii) Advanced C programming constructs: program anatomy and processes, memory and addresses (pointers, pointers and arrays, strings and examples), structures, unions and enumerations. Linear and non-linear programming data structures (dynamic memory allocation, lists, queues, doubly-linked lists, trees, applications and examples). iii) Advanced Compilation Topics and Tools: preprocessor directives, compiling multiple files with makefiles, static (.a) and dynamic (.so) linking of object files (.o), error handling (assert.h), static and dynamic code analysis (valgrind and gprof). iv) low-level programming (binary operators and examples, binary files and hexdump). v) Basic commands for programmers in the UNIX operating system: file system, redirection and pipes, permissions and basic filters.						
Teaching Methodology	Lectures (3 hours weekly), Recitation (1 hour weekly) and Laboratory sessions (1.5 hours weekly).						
Bibliography	<ol style="list-style-type: none"> <li>1. C Programming: A Modern Approach, K.N. King, 2nd Edition, ISBN-10: 0393979504, ISBN-13: 978-0393979503, 832 pages, W. W. Norton &amp; Company, 2008.</li> <li>2. Programming in C, 4th Edition, Stephen G. Kochan, ISBN-10: 0321776410, ISBN-13: 9780321776419, Addison-Wesley Professional, 600 pp, 2015.</li> <li>3. Your UNIX/Linux: The Ultimate Guide, 3rd Edition, Sumitabha Das, McGraw Hill, ISBN-13 9780073376202, 800 page, 2013.</li> <li>4. Η Language C σε Βάθος, Νίκος Χατζηγιαννάκης, Τρίτη Έκδοση, 978-960-461-208-6, Κλειδάριθμος, 2009.</li> </ol>						
Assessment	Final exam, midterm exam and homework						
Language	Greek						

Course Title	<i>Algorithms and Complexity</i>						
Course Code	<b>CS 236</b>						
Course Type	Compulsory						
Level	Undergraduate						
Year / Semester	2nd year/2nd semester						
Teacher's Name	Chr. Georgiou						
ECTS	7,5	Lectures / week	2 x 1.5 hours	Recitation / week	1 x 1 hour	Laboratories / week	1 x 1.5 hours
Course Purpose and Objectives	Familiarization with fundamental techniques of designing and analyzing algorithms. Familiarization with significant algorithms in various fields that have been suggested in the literature. Familiarization with techniques for implementing and empirically evaluating algorithms.						
Learning Outcomes	<p>Upon successful completion of this class, the student is expected to be able to:</p> <ul style="list-style-type: none"> <li>• Identify and use basic techniques for designing algorithms (backtracking, divide and conquer, dynamic programming, greediness, network flows).</li> <li>• Argue and prove the correctness of algorithms.</li> <li>• Analyze the worst-case complexity of algorithms.</li> <li>• Implement and experimentally evaluate the performance of algorithms.</li> <li>• Use the basic techniques for designing randomized Monte-Carlo algorithms (fingerprinting and random sample).</li> <li>• Solve matching problems by reducing them to the Max-flow problem.</li> <li>• Demonstrate advanced algorithmic thinking.</li> <li>• Demonstrate advanced problem-solving skills.</li> </ul>						
Prerequisites	CS 231		Required		None		
Course Content	Topics in the design and analysis of efficient algorithms and their complexity. Significant algorithms in Graph Theory, Algebra, Geometry, Number Theory and Combinatorics. General algorithmic techniques (e.g., divide-and-conquer, backtracking, dynamic programming). Fast Fourier Transform. Randomized and Parameterized algorithms. Inherent lower bounds on problem complexity.						
Teaching Methodology	Lectures (3 hours weekly), Recitation (1 hour weekly) and Laboratory sessions (1.5 hours weekly).						
Bibliography	<ol style="list-style-type: none"> <li>1. J. Kleinberg and É. Tardos, Σχεδίαση Αλγορίθμων, Εκδόσεις Κλειδάριθμος, 2008.</li> <li>2. T. H. Cormen, C. E. Leiserson, R. Rivest, and C. Stein, Εισαγωγή στους Αλγόριθμους, 2η έκδοση, Πανεπιστημιακές Εκδόσεις Κρήτης, 2016.</li> </ol>						
Assessment	Final exam, quizzes and homework (theoretical problems and programming assignments).						
Language	Greek						

Course Title	<b><i>Communications and Networks</i></b>						
Course Code	<b><i>CS 324</i></b>						
Course Type	Compulsory						
Level	Undergraduate						
Year / Semester	3rd year/1st semester						
Teacher's Name	V. Vassiliou						
ECTS	7,5	Lectures / week	2 x 1.5 hours	Recitation / week	1 x 1 hour	Laboratories / week	1 x 1.5 hours
Course Purpose and Objectives	Familiarization with fundamental topics in communication networks, with a focus on the Internet.						
Learning Outcomes	<p>Students successfully completing this course should be able to:</p> <ul style="list-style-type: none"> <li>• Explain the following core concepts of communication networks/computer networks: networking technologies and various network topologies. Wired and wireless networks. Layering networks: application layer, transport layer, Network Layer, link layer. Protocol basics. Applications and service quality.</li> <li>• Explain the following fundamentals in computer networks: protocol suite TCP/IP, Internet open systems. Core networking technologies such as routers, switches, repeaters. Protocols at application layer, transport layer, network layer, and link layer.</li> <li>• Demonstrates ability to solve networking problems and the evaluation of various Internet protocols with regard to performance.</li> <li>• Shows ability to use Internet simulators for understanding networking concept and in the design and evaluation of networks.</li> <li>• Shows ability to use Wireshark, a real time network monitoring tool and data traffic and protocol analysis, with the aim of assimilation of protocols and data traffic, but also for analysis of possible errors/problems in the functioning of the network.</li> <li>• To recognize new improved ways and mechanisms for network protocols.</li> <li>• To recognize and analyze new techniques and network technologies, like the Internet of Things.</li> </ul>						
Prerequisites	CS 131		Required		None		
Course Content	Introductory course in Computer Networks. The goal is the understanding and use of concepts related to fundamental issues in Communication Networks, using the Internet as an example. Deals with Networking layers, such as the application, transport, network, and data link layers. Open systems and internetworking. Networking technologies including wired and wireless Local Area Networks and network topologies. Algorithms, including routing and congestion control. Introduction to quality of service (QoS) and multimedia applications. Laboratory session includes practical exercises with wireshark and simulations.						
Teaching Methodology	Lectures (3 hours weekly), Recitation (1 hour weekly) and Laboratory sessions (1.5 hours weekly).						
Bibliography	1. J. F. Kurose and K. W. Ross, Computer Networking: A Top-Down Approach Featuring the Internet, 8th Edition, Addison-Wesley, 2020.						
Assessment	Final exam, midterm exam and homework (including laboratory exercises).						
Language	Greek						

Course Title	<b><i>Parallel Processing</i></b>						
Course Code	<b><i>CS 325</i></b>						
Course Type	Compulsory						
Level	Undergraduate						
Year / Semester	3rd year/2nd semester						
Teacher's Name	H. Volos						
ECTS	7,5	Lectures / week	2 x 1.5 hours	Recitation / week	1 x 1 hour	Laboratories / week	1 x 1.5 hours
Course Purpose and Objectives	Introduction to parallel programming and performance analysis of parallel programs						
Learning Outcomes	<ul style="list-style-type: none"> <li>• Familiarization with basic parallel architectures, including multicore processors and computing clusters.</li> <li>• Understanding the performance measurement and analysis methods for parallel programs</li> <li>• Learn parallel programming models</li> <li>• Introduction to the development and design of parallel programs</li> <li>• Familiarization with basic parallelism concepts (e.g. synchronization, coordination, load balancing)</li> <li>• Programming with different programming APIs (e.g. pthreads, OpenMP, CUDA, and MPI)</li> </ul>						
Prerequisites	CS 221, CS 232			Required		None	
Course Content	Parallel hardware architectures: the entire spectrum of parallel machines as appearing in Flynn's classification (SISD, SIMD, MISD, MIMD), including multicore processors, vector units, graphics processing units (GPUs), computing clusters. Parallel programming models: shared vs. non-shared memory, threads (Pthreads, OpenMP), nested parallelism (OpenMP), data parallelism (SIMD, SIMT, CUDA), message passing (MPI). Synchronization and coordination mechanisms: locking, atomicity, ordering, joins, barriers. Performance evaluation and analysis: speedup, efficiency, scalability analysis, complexity analysis, performance modeling. Parallel thinking: decomposition, assignment, orchestration, mapping. Basic parallel algorithms: elementary computation, matrix multiplication, sorting, prefix sum, sample scientific application.						
Teaching Methodology	Lectures (3 hours weekly), Recitation (1 hour weekly), and Laboratory sessions (1.5 hours weekly)						
Bibliography	1. Peter Pacheco, Matthew Malensek. An Introduction to Parallel Programming, 2nd Edition. Morgan Kaufmann, 2021						
Assessment	Final exam, midterm exam, and homework.						
Language	Greek						

Course Title	<b><i>Systems Security</i></b>						
Course Code	<b><i>CS 326</i></b>						
Course Type	Compulsory						
Level	Undergraduate						
Year / Semester	3rd year/2nd semester						
Teacher's Name	E. Athanasopoulos						
ECTS	7,5	Lectures / week	2 x 1.5 hours	Recitation / week	1 x 1 hour	Laboratories / week	1 x 1.5 hours
Course Purpose and Objectives	Introduction to systems security which covers a wide range of concepts. Primarily, the course helps students to become familiar with different research areas of modern systems security by covering topics such as applied cryptography, software vulnerabilities and exploitation, defenses, mobile security, web security, network security, privacy, and anonymity. Additionally, students will be able to perform engineering tasks in cryptography, software exploitation, network security, and program analysis.						
Learning Outcomes	<p>The student that has successfully completed the course would be able to:</p> <ul style="list-style-type: none"> <li>• Have a deep knowledge of modern applied cryptography, the main problems we solve with it, as well as the mechanics of cryptographic techniques.</li> <li>• Know the internals of symmetric encryption algorithms, such as DES, AES and RC4. The techniques will be also applied to programming assignments.</li> <li>• Understand the mathematical background of asymmetric encryption and, in particular, RSA.</li> <li>• Develop applications of public-key cryptography for digital signatures. In the context of these applications, the student will become familiar with cryptographic hash functions.</li> <li>• Understand the foundations of software exploitation and how vulnerabilities (e.g., buffer overflow) can be leveraged for compromising systems.</li> <li>• Construct real attacks in software with artificially injected vulnerabilities.</li> <li>• Analyze the execution of software on Linux using a debugger (gdb).</li> <li>• Understand how modern defenses against software exploitation work.</li> <li>• Understand how TLS works and the importance of Certificates.</li> <li>• Develop applications that capture and monitor network traffic.</li> <li>• Understand the basic mechanics of anonymity systems (e.g., TOR).</li> </ul>						
Prerequisites	CS 232	Required			None		
Course Content	Introduction to applied cryptography (early ciphers, modern symmetric and asymmetric ciphers, cryptographic hash functions and MACs, applications). Software vulnerabilities (buffer overflows, integer overflows, use-after-free). Control-flow attacks (code injection, code reuse). Defenses (non-executable pages, stack canaries, code randomization, CFI). Isolation techniques (SFI). Static and dynamic analysis of software. Mobile security (Android, iOS). Web security (cross-site scripting, CSRF, clickjacking, phishing). Network security (TLS, botnets, DDoS). Authentication and passwords. Privacy and anonymity (TOR).						
Teaching Methodology	Lectures (3 hours weekly), Recitation (1 hour weekly) and Laboratory sessions (1.5 hours weekly).						
Bibliography	<ol style="list-style-type: none"> <li>1. Alfred J. Menezes, Paul C. van Oorschot and Scott A. Vanstone. Handbook of Applied Cryptography, CRC Press, ISBN 0849385237. (<a href="http://cacr.uwaterloo.ca/hac/">http://cacr.uwaterloo.ca/hac/</a>)</li> <li>2. Ross Anderson. Security Engineering: A Guide to Building Dependable Distributed Systems, Second Edition, Wiley, ISBN 0470068523. (<a href="http://www.cl.cam.ac.uk/~rja14/book.html">http://www.cl.cam.ac.uk/~rja14/book.html</a>)</li> <li>3. Christof Paar and Jan Pelzl: Understanding Cryptography: A Textbook for Students and Practitioners, Springer.</li> <li>4. Δημοσιεύσεις.</li> </ol>						
Assessment	Final exam, midterm exam and homework (including laboratory exercises).						
Language	Greek						

Course Title	<i>Artificial Intelligence</i>						
Course Code	<i>CS 341</i>						
Course Type	Compulsory						
Level	Undergraduate						
Year / Semester	3rd year/1st semester						
Teacher's Name	Y. Dimopoulos						
ECTS	7,5	Lectures / week	2 x 1.5 hours	Recitation / week	1 x 1 hour	Laboratories / week	0
Course Purpose and Objectives	The purpose of this course is to introduce the fundamental principles and techniques that underpin the functioning of software systems exhibiting some form of “intelligence.” Upon completing the course, students will gain an understanding of the modern conception of Artificial Intelligence, the key problems it addresses, and the basic methods used to solve them. The course also explores current and emerging applications of Artificial Intelligence techniques across a range of domains.						
Learning Outcomes	Solid understanding of the fundamental ideas of modern Artificial Intelligence, the main problems that it studies, and the basic methods used to solve these problems. Familiarization with basic practical methods and tools of Artificial Intelligence.						
Prerequisites	CS 231		Required		None		
Course Content	Intelligent Agents. Search. Adversarial Search and Games. Constraint Satisfaction. Knowledge Representation and Reasoning. Planning. Machine Learning. Uncertainty.						
Teaching Methodology	Lectures (3 hours weekly) and Recitation (1 hour weekly).						
Bibliography	<ol style="list-style-type: none"> <li>1. S. Russel, P. Norvig, Artificial Intelligence: A Modern Approach, 4th Edition, Prentice Hall, 2021.</li> <li>2. D. Poole, A. Mackworth, Artificial Intelligence: Foundations of Computational Agents, 3rd Edition, Cambridge University Press, 2023.</li> <li>3. W. Ertel, Introduction to Artificial Intelligence, Springer, 2025.</li> <li>4. Ι. Βλαχάβας, Π. Κεφαλάς, Ν. Βασιλειάδης, Φ. Κόκκορας, Η. Σακελλαρίου, Τεχνητή Νοημοσύνη, Εκδόσεις Πανεπιστημίου Μακεδονίας, 2020.</li> </ol>						
Assessment	Final exam, midterm exam and homework (theoretical and programming assignments).						
Language	Greek						

Course Title	<b>Database Systems</b>						
Course Code	<b>CS 342</b>						
Course Type	Compulsory						
Level	Undergraduate						
Year / Semester	3rd year/1st semester						
Teacher's Name	D. Zeinalipour						
ECTS	7,5	Lectures / week	2 x 1.5 hours	Recitation / week	1 x 1 hour	Laboratories / week	1 x 1.5 hours
Course Purpose and Objectives	Introduction to the basic principles needed for the design and the use of a database. Provision of practical exercises in the application of these concepts with the use of an industrial system for database.						
Learning Outcomes	<p>On successful completion of the course students will be able to:</p> <ul style="list-style-type: none"> <li>• Obtain an in-depth understanding of concepts related to the design and utilization of a relational database management system by implementing relevant concepts in a commercial database management system.</li> <li>• Model the logical design of a Database application using the Entity-Relationship (ER) Model, the Enhanced Entity-Relationship (EER) Model and the Relational Data Model.</li> <li>• Implement a logical database design and manipulation logic using, at the beginning a formal language such as Relational Algebra, and then a real Data Definition Language (SQL-DDL) and Data Manipulation Language (SQL-DML).</li> <li>• Implement a logical database design using the SQL Data Definition Language (DDL) and learn how to manipulate the stored data using the Relational Algebra and the SQL Data Manipulation Language (DML).</li> <li>• Implement advanced database applications that include Constraints, Triggers, Views, Stored Procedures, User Defined Functions and Transactions. Develop sophisticated queries to extract information from complex datasets.</li> <li>• Access data stored in a database management system through a high-level programming language and the JDBC bridge.</li> <li>• Learn how to administer a relational database including backups, recovery, user access control and security.</li> <li>• Understand the theory and application of normalizing a relational schema with functional dependencies.</li> </ul>						
Prerequisites	CS 231	Required			None		
Course Content	Introduction to Databases. Organization and proper management of large quantities of data for use in applications. Database models such as the entity-relation model, the relational model, the network model and the hierarchical model.						
Teaching Methodology	Lectures (3 hours weekly), Recitation (1 hour weekly) and Laboratory sessions (1,5 hours weekly).						
Bibliography	<ol style="list-style-type: none"> <li>1. R. Elmarsı and S. Navathe, Fundamentals of Database Systems, 7th Edition, Addison-Wesley, 2015.</li> <li>2. R. Elmarsı and S. Navathe, Θεμελιώδεις Αρχές Συστημάτων Βάσεων Δεδομένων, 7η Έκδοση, 1ος τόμος, Εκδόσεις Διάλογος, 2016.</li> <li>3. R. Ramakrishnan and J. Gehrke, Database Management Systems, 3rd Edition, McGraw-Hill, 2003.</li> <li>4. C. Coronel, S. Morris, K. Crockett, C. Blewett, Database Principles: Fundamentals of Design, Implementation, and Management, 3rd Edition, 2020.</li> </ol>						
Assessment	Final exam, midterm exam and homework (assignments, laboratory quizzes, final project).						
Language	Greek						

Course Title	<b>Software Engineering</b>						
Course Code	<b>CS 343</b>						
Course Type	Compulsory						
Level	Undergraduate						
Year / Semester	3rd year/1st semester						
Teacher's Name	E. Constantinou						
ECTS	7,5	Lectures / week	2 x 1.5 hours	Recitation / week	1 x 1 hour	Laboratories / week	1 x 1.5 hours
Course Purpose and Objectives	Familiarization with and assimilation of the approaches, methodologies, models and tools used to develop quality software systems. Understanding of software requirements engineering, software architectures and software modeling. Understanding of software testing process and of software architecture design patterns. Understanding of software testing procedures. Use and application of architecture design patterns. Application of software creation methodologies on the construction of a real software system.						
Learning Outcomes	<p>The learning outcomes for the students are the following:</p> <ul style="list-style-type: none"> <li>• Understanding of phases for the development of quality software systems.</li> <li>• Familiarization with the different software process models.</li> <li>• Ability to collect software requirements.</li> <li>• Ability to effectively analyze modules for software development</li> <li>• Ability to design software systems.</li> <li>• Ability to implement functional software prototypes.</li> <li>• Understanding of object-oriented analysis and design of software systems.</li> <li>• Using effective methods for software systems development.</li> <li>• Using and applying architectural design patterns.</li> </ul>						
Prerequisites	CS 132		Required		None		
Course Content	Methods, tools, and procedures for the development and maintenance of large-scale software systems. Existing life-cycle models (e.g. waterfall model). Introduction to Agile development. Requirements analysis and specification techniques. Interfacing with stakeholders, as a team. Software development methodologies. Software development planning and risks. Unified Modelling Language (UML) and supported static and dynamic diagrams. Code transformation. Practical experience with CASE tools for modeling data and procedures (Modelio). Prototyping for Web applications (HTML, CSS). Architectural Design patterns (Model View Controller etc.). Software verification and validation. Test phases, unit testing and relevant frameworks. CASE tools. Project planning and management.						
Teaching Methodology	Lectures (3 hours weekly), Recitation (1 hour weekly) and Laboratory sessions (1,5 hour weekly).						
Bibliography	<ol style="list-style-type: none"> <li>1. I. Sommerville, Software Engineering, 10th Edition, Addison-Wesley, 2016.</li> <li>2. R. Pressman, Software Engineering: A Practitioner's Approach, 8th Edition, Mc Graw Hill, 2015.</li> <li>3. H. van Vliet, Software Engineering: Principles and Practice, 3rd edition, John Wiley &amp; Sons, 2008</li> <li>4. P. Stevens, R. Pooley, Using UML Software Engineering with Objects and Components, 2nd edition, Addison-Wesley, 2006.</li> </ol>						
Assessment	Final exam, midterm exam and semester team project.						
Language	Greek						

Course Title	<b><i>Diploma Project I and Diploma Project II</i></b>						
Course Code	<b><i>CS 400 and CS 401</i></b>						
Course Type	Compulsory						
Level	Undergraduate						
Year / Semester	4th year/7th – 8th semester						
Teacher's Name	CS Faculty						
ECTS	15	Lectures / week	-	Recitation / week	-	Laboratories / week	-
Course Purpose and Objectives	The "Diploma Project" course is a comprehensive capstone project for undergraduate students in the Computer Science department. This course involves the independent design, development, and evaluation of a software system, research study, or theoretical analysis. Under the supervision of a faculty advisor, students identify a problem or research question in the field of computer science, conduct a literature review, propose a solution, and implement or investigate it using appropriate methodologies and tools. The project concludes with a formal written report and a presentation of the findings to a panel of two faculty members.						
Learning Outcomes	<p>Through the Project, students should acquire the ability to:</p> <ul style="list-style-type: none"> <li>• Understand and possibly challenge the existing knowledge and practice at the forefront of the chosen specialization area</li> <li>• identify and demonstrate appropriate methodologies and know when to use them</li> <li>• define, articulate and use terminology, concepts, and theory in their field and know how to use them</li> <li>• use library and other tools to search for existing body of research relevant to their topic</li> <li>• identify and practice research ethics and responsible conduct in research</li> <li>• know and apply problem solving skills to constructively address research setbacks</li> <li>• work autonomously in an effective manner, setting and meeting deadlines</li> <li>• reflect on their own work, identifying lessons learned, strengths, and ways to improve</li> <li>• communicate confidently and constructively with graduate students and faculty as mentors</li> <li>• produce a well-structured written report that effectively communicates the problem, methodology, results, and conclusions of the project, following academic and professional standards.</li> </ul>						
Prerequisites	At least 156 ECTS of coursework		Required		None		
Course Content	Each student in the course has a project advisor. Faculty members announce projects towards the end of the previous semester, and once the selection procedure is completed (monitored by the Diploma Project Coordinator), a student-faculty assignment is announced.						
Teaching Methodology	Individual work under supervision						
Bibliography	Project-dependent						
Assessment	Written thesis, Software/System Prototype Assessment (if applicable), and oral presentation – evaluated by two members of the CS faculty (one being the project supervisor)						
Language	English/Greek						

Course Title	<b><i>Logic in Computer Sciences</i></b>						
Course Code	<b><i>CS 412</i></b>						
Course Type	Restricted Elective Course						
Level	Undergraduate						
Year / Semester	5 <sup>th</sup> or 6 <sup>th</sup> or 7 <sup>th</sup> or 8 <sup>th</sup> semester						
Teacher's Name	A. Philippou						
ECTS	7,5	Lectures / week	2 x 1.5 hours	Recitation / week	1 x 1 hour	Laboratories / week	0 hours
Course Purpose and Objectives	The main objective of the course is to prepare students for using logic as a formal tool in Computer Science. Furthermore, it aims to develop and cultivate formal and syllogistic reasoning and provide a thorough introduction to computational logic and its applications in Computer Science.						
Learning Outcomes	<p>Upon successful completion of the course, students should:</p> <ul style="list-style-type: none"> <li>• Understand the basic concepts of propositional and predicate calculus with emphasis on the applications of these concepts in Computer Science.</li> <li>• Be able to construct proofs and apply these skills to practical applications.</li> <li>• Be familiar with the Method of Resolution and its use in Logic Programming</li> <li>• Be familiar with concepts of linear and branching temporal logic and be able to use them to specify and verify computing systems.</li> <li>• Understand basic algorithms for formal verification and use formal verification tools.</li> <li>• Be familiar with Hoare specifications and their use in the analysis of the correctness of programs.</li> </ul>						
Prerequisites	CS 111		Required		None		
Course Content	Propositional Logic: Syntax, Semantics, Normal Forms, Decision Procedures, Proof Theory, Resolution. Predicate Logic: Syntax and Semantics, Proof Theory, Resolution, Logic Programming. Hoare Logic for program verification. Linear and Branching Temporal Logics: Syntax, Semantics, and model-checking algorithms.						
Teaching Methodology	Lectures (3 hours weekly) and Recitation (1 hour weekly).						
Bibliography	<ol style="list-style-type: none"> <li>1. M. Huth and A. Ryan, <i>Logic in Computer Science: Modeling and Reasoning about Concurrent Systems</i>, Cambridge University Press, 2000.</li> <li>2. M. Ben-Ari, <i>Mathematical Logic for Computer Science</i>, Springer-Verlag, 2nd Edition, 2003.</li> <li>3. U. Schoning, <i>Logic for Computer Scientists</i>, Springer-Verlag, 2nd Printing, 2008.</li> </ol>						
Assessment	Final exam, midterm exam and homework.						
Language	Greek						

Course Title	<b>Basic Principles of Programming Languages</b>						
Course Code	<b>CS 414</b>						
Course Type	Restricted Elective Course						
Level	Undergraduate						
Year / Semester	5 <sup>th</sup> or 6 <sup>th</sup> or 7 <sup>th</sup> or 8 <sup>th</sup> semester						
Teacher's Name	CS or Visiting Faculty						
ECTS	7,5	Lectures / week	2 x 1.5 hours	Recitation / week	1 x 1 hour	Laboratories / week	0 hours
Course Purpose and Objectives	The objective of the course is to study the basic principles of programming languages through the representation of basic programming notions into a uniform mathematical framework. At the same time, the course will follow the practical application of the notions under study into popular programming languages. In particular, the course will study the abstract syntax, the syntax, semantics, and the practical application of functional, procedural, object-oriented, and programming structures. Attention will be given to the static (type systems) and dynamic (execution behavior) properties of programming languages, as well as the notions of type soundness and execution safety. The course will introduce principles/notions of distributed programming and study modern execution safety problems in a distributed setting.						
Learning Outcomes	<p>At the end of the course a student should be able to:</p> <ul style="list-style-type: none"> <li>• Understand the different programming languages, including the most popular languages, through a unified programming framework.</li> <li>• Classify programming languages into procedural, object-oriented or functional, and into declarative or imperative, based on the programming structures they implement.</li> <li>• Learn the basic structures/characteristics of programming languages.</li> <li>• Be able to define the syntax, semantics, and type system of a simple programming language.</li> <li>• Understand the notions of type soundness and execution safety.</li> <li>• Use mathematical induction to prove simple safety properties in a programming language.</li> </ul>						
Prerequisites	CS 211, CS 231			Required		None	
Course Content	History of Programming Languages, the $\lambda$ -calculus. functional programming (The Haskell programming language, type system, type inference, pure vs impure programming languages), procedural and object-oriented programming (basic object-oriented programming notions, mathematical representation of the Java programming language), distributed programming (the actors programming model, process calculi as mathematical models for representing distributed programming).						
Teaching Methodology	Lectures (3 hours weekly), Recitation (1 hour weekly)						
Bibliography	<ol style="list-style-type: none"> <li>1. Benjamin C. Pierce, Types and Programming Languages, MIT Press, 2002.</li> <li>2. Graham Hutton, Programming in Haskell (2nd Edition), Cambridge University Press, 2016.</li> <li>3. John C. Mitchell, Concepts in programming languages, Cambridge University Press. 2003.</li> <li>4. Selected research articles</li> </ol>						
Assessment	Final exam, midterm exam and homework						
Language	Greek						

Course Title	<b>Computer Architecture</b>						
Course Code	<b>CS 420</b>						
Course Type	Restricted Elective Course						
Level	Undergraduate						
Year / Semester	5 <sup>th</sup> or 6 <sup>th</sup> or 7 <sup>th</sup> or 8 <sup>th</sup> semester						
Teacher's Name	Y. Sazeides						
ECTS	7,5	Lectures / week	2 x 1.5 hours	Recitation / week	1 x 1 hour	Laboratories / week	1 x 1.5 hours
Course Purpose and Objectives	Introduction to advanced principles of architecture, organization, operation and performance analysis of modern computer systems						
Learning Outcomes	<ul style="list-style-type: none"> <li>• Familiarization with computer technology trends</li> <li>• Benchmarking, performance measurement and analysis methods of modern computer systems</li> <li>• Power and energy of modern computing systems and techniques for their minimization</li> <li>• Understanding for hardware and software techniques used to improve the instruction level parallelism of cores found in modern central processing units</li> <li>• Learn advanced techniques to improve memory hierarchy performance</li> <li>• Microarchitectural security</li> <li>• Introduction to data level parallelism and basic organization of modern graphic processing units</li> <li>• Familiarization with thread level parallelism, parallel architectures with shared memory</li> <li>• Domain specific architectures and accelerators (e.g., TPUs, PIM)</li> <li>• Architecture of data centers</li> </ul>						
Prerequisites	CS 222		Required		None		
Course Content	Computer Technology, Moore's and Dennard's Law, market segments (IoT, client, server). Benchmarks, performance metrics (CPI/IPC, average and tail latency, QPS), means, reliability (FIT, MTTF availability, error detection/correction). Power (dynamic/static, C-states, P-states, idle and dvfs governor, turbo, core/uncore, frequency scalability, TDP, throttling, power delivery, Power, Parallelism and Specialization, Sustainability (Environmental impact of computer manufacturing, operation and end-of-life)). Review Pipeline and Cache Basics. Dynamic Instruction Level Parallelism (register renaming, out-of-order-execution, reorder-buffer, prediction, speculation, superscalar). Static ILP (scheduling, VLIW). Advanced Caches (mshrs, inclusion, prefetching, cache replacement policy). Security (side channels, hw transient and non-transient side channels). Data Level Parallelism (vector, SIMD, GPUs). Thread Level Parallelism and Shared Memory Multiprocessing (interconnect, cache coherence, snoopy and directory, consistency). Domain Specific Architectures and Accelerators. Data Center Architecture. Real Products: e.g., ICE Lake Core, Intel Memory Hierarchy, Nvidia GPU, Google TPU. Qualitative analysis of real machines and their performance data.						
Teaching Methodology	Lectures (3 hours weekly), Recitation (1 hour weekly) and Laboratory sessions (1.5 hours weekly).						
Bibliography	<ol style="list-style-type: none"> <li>1. J. Hennessy and D. Patterson, Computer Architecture: A Quantitative Approach, 6th Edition, Morgan Kaufmann, 2019.</li> <li>2. Selected research articles.</li> </ol>						
Assessment	Final exam, midterm exam and homework (project and exercises).						
Language	Greek						

Course Title	<b><i>Systems Programming</i></b>						
Course Code	<b><i>CS 421</i></b>						
Course Type	Restricted Elective Course						
Level	Undergraduate						
Year / Semester	5 <sup>th</sup> or 6 <sup>th</sup> or 7 <sup>th</sup> or 8 <sup>th</sup> semester						
Teacher's Name	D. Zeinalipour						
ECTS	7,5	Lectures / week	2 x 1.5 hours	Recitation / week	1 x 1 hour	Laboratories / week	1 x 1.5 hours
Course Purpose and Objectives	The main objective of this undergraduate course is to allow students develop complex system-level software in the C programming language while gaining an intimate understanding of the UNIX operating system (and all OS that belong to this family, such as Linux, the BSDs, and even Mac OS X). Topics covered will include the user/kernel interface, fundamental concepts of UNIX, user authentication, basic and advanced I/O, filesystems, signals, process relationships, and inter-process communication. Fundamental concepts of software development and maintenance on UNIX systems will also be covered. The taught concepts have an application to the whole family of UNIX OSs (e.g., Linux, MacOS, HP-UX, AIX, Solaris, Android, iOS, Raspbian) but also apply to Windows (e.g., Powershell and windows system calls).						
Learning Outcomes	<p>Upon successful completion of the course, the student will be able to:</p> <ul style="list-style-type: none"> <li>• Understand and combine intermediate and advanced utilities and stream editors in the UNIX operating system (and all OS that belong to this family, such as Linux, Android/iOS the BSDs, and Mac OS X).</li> <li>• Know how to setup, administer, maintain, upgrade and secure a networked UNIX operating system in the Cloud including the installation and configuration of networked services.</li> <li>• Explain and use the function of the common operating system kernel routines that are provided by an operating system and accessible from both a systems programming language and the shell.</li> <li>• Use operating system kernel calls from within a programming language to initiate and synchronize multiple threads/processes, interprocess communication, interact with the file system, set and respond to timers/interrupts.</li> <li>• Understand the inner workings of UNIX-like operating systems and obtain a solid foundation for an advanced course in operating systems, data management systems, distributed/parallel systems, and networked/cloud systems.</li> </ul>						
Prerequisites	CS 222		Required		None		
Course Content	i) Advanced commands of the UNIX operating system for administrators: filters with regular expressions, system utilities and stream editors (awk, sed). ii) Advanced shell programming with an emphasis on Bash: environment, control structures, debugging. iii) Low-level I/O system calls with C. iv) Process management: environment, control and signals, inter-process communication (IPC) with an emphasis on pipes and named pipes (FIFO), XSI IPC: semaphores, shared memory and message queues, network IPC (TCP Sockets) and the client/server model. v) Multithreaded programming, concurrency and performance aspects. Implementing network protocols from RFC documents. v) System security aspects, managing cluster computers and clouds: virtualization, data centers and Green IT.						
Teaching Methodology	Lectures (3 hours weekly), Recitation (1 hour weekly) and Laboratory sessions (1.5 hours weekly).						
Bibliography	<ol style="list-style-type: none"> <li>1. Your UNIX/Linux: The Ultimate Guide, 3rd Edition, Sumitabha Das, McGraw Hill, ISBN-13 9780073376202, 800 pp, 2013.</li> <li>2. UNIX and Linux System Administration Handbook 5th Edition, Evi Nemeth, Garth Snyder, Trent R. Hein, Ben Whaley, Dan Mackin, 1232 pages, Addison-Wesley, ISBN-10: 0134277554   ISBN-13: 978-0134277545, 2017.</li> <li>3. Computer Systems: A Programmer's Perspective, 3th Edition, Randal E. Bryant, David R. O'Hallaron, ISBN-10: 013409266X, ISBN-13: 9780134092669, Pearson, 1120 pp, 2016.</li> <li>4. Advanced Programming in the UNIX Environment, 3rd Edition, W. Richard Stevens, Stephen A. Rago, Addison-Wesley Professional, ISBN-10: 0321637739, ISBN-13: 9780321637734, 1024 pp, 2013.</li> </ol>						
Assessment	Final exam, midterm exam, programming assignments and laboratory quizzes.						
Language	Greek						

Course Title	<i>Advanced Networks</i>						
Course Code	<i>CS 422</i>						
Course Type	Restricted Elective Course						
Level	Undergraduate						
Year / Semester	6 <sup>th</sup> or 7 <sup>th</sup> or 8 <sup>th</sup> semester						
Teacher's Name	V. Vassiliou						
ECTS	7,5	Lectures / week	2 x 1.5 hours	Recitation / week	1 x 1 hour	Laboratories / week	1 x 1.5 hours
Course Purpose and Objectives	Extension of the basic knowledge about Computer Networks regarding architectures, techniques and protocols for the Internet.						
Learning Outcomes	<p>Upon successful completion of this course the student will have advanced knowledge in the following fields:</p> <ul style="list-style-type: none"> <li>• Explain the following core concepts of communication networks/computer networks: networking technologies and various network topologies. Wired and wireless networks. Layering networks: application layer, transport layer, Network Layer, link layer. Protocol basics. Applications and service quality, Basic network management.</li> <li>• Explain the following fundamentals in computer networks: protocol suite TCP/IP, Internet open systems. Core networking technologies such as routers, switches, repeaters. Internet security.</li> <li>• Demonstrate skills in solving networking issues network management issues, and analysis of communication protocols.</li> <li>• Demonstrate skills in deploying and analyzing various routing and congestion control algorithms.</li> <li>• Arguing, with regard to the infrastructure of a network and evaluates based on quality and other criteria the performance of networks.</li> <li>• Demonstrates ability to solve networking problems and the evaluation of various Internet protocols with regard to performance.</li> <li>• Shows ability to use Internet simulators for understanding networking concept and in the design and evaluation of networks.</li> <li>• Shows ability to use Wireshark, a real time network monitoring tool and data traffic and protocol analysis, with the aim of assimilation of protocols and data traffic, but also for analysis of possible errors/problems in the functioning of the network.</li> <li>• To seek to continuously evaluate new improved ways and mechanisms for network protocols.</li> <li>• To constantly seek and analyze new techniques and network technologies, like the Internet of Things, Software Defined Networking, and Network Virtual Functions.</li> </ul>						
Prerequisites	CS 324		Required		None		
Course Content	Advanced topics in Computer Networks and the Internet, such as: IPv6, Multicast Routing, QoS Routing, TCP Congestion Control, Performance Analysis, Multimedia Networking Applications, Realtime services and protocols, Quality of Service, MPLS, Traffic Engineering, Mobile and Wireless Networks, Issues in Security for Computer Networks. Introduction to advanced research topics (e.g. Internet of Things, wireless sensor networks, 5G). Introduction to Network Management, Software Defined Networks. Cloud and Fog Computing.						
Teaching Methodology	Lectures (3 weekly), Recitation (1 hour weekly) and Laboratory sessions (1.5 hours weekly).						
Bibliography	1. J. F. Kurose and K. W. Ross, Computer Networking: A Top-Down Approach, 8th Edition, Pearson, 2020.						
Assessment	Final exam, midterm exam and homework.						
Language	English						

Course Title	<i>Network and Information Security</i>						
Course Code	<b>CS 423</b>						
Course Type	Restricted Elective Course						
Level	Undergraduate						
Year / Semester	6 <sup>th</sup> or 7 <sup>th</sup> or 8 <sup>th</sup> semester						
Teacher's Name	V. Vassiliou						
ECTS	7,5	Lectures / week	2 x 1.5 hours	Recitation / week	1 x 1 hour	Laboratories / week	1 x 1.5 hours
Course Purpose and Objectives	Introduction to network and information security principles, understanding of basic areas in Cryptography, Authentication and Confidentiality. Gain of knowledge in methods for the evaluation of Software, Applications and Systems with respect to security. Application of tools for the protection of networks, applications and information.						
Learning Outcomes	<p>Upon successful completion of this course the student will have advanced knowledge in the following fields:</p> <ul style="list-style-type: none"> <li>• Identify some of the factors driving the need for network and information security</li> <li>• Demonstrates ability to understand of the issues involved in the field of information security and assurance</li> <li>• Navigate through the language of the field of network and information security.</li> <li>• Explain the CIA triad of Confidentiality, Integrity and Availability</li> <li>• Identify and classify computer and network security threats and attacks</li> <li>• Compare and contrast encryption systems and algorithms.</li> <li>• Encrypt and decrypt messages and sign and verify messages using well-known techniques</li> <li>• Acknowledge the ethical and legal considerations of network and information security.</li> </ul>						
Prerequisites	CS 324 and CS326			Required		None	
Course Content	Introduction to Security Threats and Attacks, Cryptographic Techniques (encryption, cryptanalysis, authentication, confidentiality), identification and authentication (Kerberos, PKI), Internet Application security protocols (PGP, SSL/TLS), Network security (Firewalls, IDS), Defending against threats on endsystems, Checking of networks and applications for vulnerabilities, Other issues in network and information security (privacy, ethics, legal framework).						
Teaching Methodology	Lectures (3 hours weekly), Recitation (1 hour weekly) and Laboratory sessions (1.5 hours weekly).						
Bibliography	<ol style="list-style-type: none"> <li>1. W. Stallings and L. Brown, Computer Security: Principles and Practice, 5th Edition, Pearson 2023</li> <li>2. W. Stallings, Network Security Essentials, Sixth Edition, Pearson, 2021.</li> <li>3. C. Kaufman, R. Perlman, M. Speciner, R. Perlner, Network Security: Private Communications in a Public World, 3rd edition, Pearson 2023</li> </ol>						
Assessment	Final exam, midterm exam and homework.						
Language	Greek						

Course Title	<b>Digital Signal Processing</b>						
Course Code	<b>CS 424</b>						
Course Type	Restricted Elective Course						
Level	Undergraduate						
Year / Semester	5 <sup>th</sup> or 6 <sup>th</sup> or 7 <sup>th</sup> or 8 <sup>th</sup> semester						
Teacher's Name	CS or Visiting Faculty						
ECTS	7,5	Lectures / week	2 x 1.5 hours	Recitation / week	1 x 1 hour	Laboratories / week	1 x 1.5 hours
Course Purpose and Objectives	Introduction to Digital Signal Processing (DSP) methods and applications						
Learning Outcomes	<p>Upon successful completion of this course the student will have advanced knowledge in the field of digital signal processing:</p> <ul style="list-style-type: none"> <li>• Have a comprehensive grounding in DSP concepts and algorithms plus practical information on the design and implementation of DSP systems.</li> <li>• Give a good understanding of DSP principles and their implementation and equips the delegate to put the ideas into practice and/or to tackle more advanced aspects of DSP</li> <li>• Have theoretical knowledge illustrated by application examples, by demonstrations and by work in the laboratory. using MATLAB</li> <li>• Gain a basic understanding of analysis, modeling and estimation of real world signals.</li> <li>• Solve basic problems within filtering, frequency analysis and signal modeling by combining earlier acquired knowledge and skills within mathematics, and basic digital signal processing.</li> <li>• Use Matlab to solve the above basic problems</li> <li>• Analyse and evaluate the properties of LTI systems in terms of z-transforms.</li> <li>• Understand the sampling theorem and perform sampling on continuous-time signals by applying advanced knowledge of sampling theory (i.e. aliasing, quantisation errors, pre-filtering).</li> <li>• Apply the concepts of all-pass and minimum-phase systems to analyse LTI systems and address complex design problems.</li> <li>• Evaluate design problems related to frequency selective processing and design FIR/IIR filters.</li> <li>• Construct systems for spectral estimation of real signals by applying advanced knowledge of Fourier techniques.</li> <li>• Judge implementation aspects of modern DSP algorithms.</li> <li>• Apply the relevant theoretical knowledge to design and analyse a practical discrete-time signal system, such as audio, bio-medical or wireless system.</li> </ul>						
Prerequisites	CS 111, MAS 029, MAS 012	Required			None		
Course Content	Discrete signals and systems, sampling of signals, frequency analysis of discrete systems and signals, z-transform, Fourier-Transform, Discrete Fourier Transform, and Fast Fourier Transform, digital filters, application examples.						
Teaching Methodology	Lectures (3 hours weekly), Recitation (1 hour weekly) and Laboratory sessions (1.5 hours weekly).						
Bibliography	<ol style="list-style-type: none"> <li>1. Alan Oppenheim, Ronald Schafer, John Buck, Discrete-time Signal Processing, Prentice-Hall, 2010.</li> <li>2. Alan Oppenheim, Ronald Schafer, John Buck, Zeitdiskrete Signalverarbeitung, 2. Auflage, Pearson Studium, 2004.</li> <li>3. Vinay Ingke, John Proakis, Digital Signal Processing using MATLAB, Third Edition, Cengage Learning, 2012.</li> <li>4. Σ. Θεοδορίδης, Digital Signal Processing, Εκδόσεις Πανεπιστημίου Πατρών, 1992.</li> <li>5. J. H. McClellan, R. W. Schafer and M. A. Yoder, DSP First, Prentice Hall, 1998.</li> <li>6. Matlab DSP packages</li> </ol>						
Assessment	Final exam, midterm exam and homework (laboratory exercises, additional exercises, final project).						
Language	Greek						

Course Title	<b><i>Internet Technologies</i></b>						
Course Code	<b>CS 425</b>						
Course Type	Restricted Elective Course						
Level	Undergraduate						
Year / Semester	5 <sup>th</sup> or 6 <sup>th</sup> or 7 <sup>th</sup> or 8 <sup>th</sup> semester						
Teacher's Name	A. Konstantinidis						
ECTS	7,5	Lectures / week	2 x 1.5 hours	Recitation / week	1 x 1 hour	Laboratories / week	1 x 1.5 hours
Course Purpose and Objectives	<p>The main objective of the course is to help students understand and become familiar with fundamental web technologies, as well as to develop skills in using web programming languages for designing and creating websites and web applications. Specifically, the course includes:</p> <ul style="list-style-type: none"> <li>• <b>Webpage Analysis:</b> The ability to analyze a webpage and identify its elements and attributes.</li> <li>• <b>Client-Side / Front-End Programming:</b> The creation of static and dynamic websites using HTML5, Cascading Style Sheets (CSS), and JavaScript (JS).</li> <li>• <b>Server-Side / Back-End Programming:</b> The development of applications that interact with databases, primarily using PHP, Java, and SQL.</li> <li>• <b>Client-Server Communication:</b> The implementation of communication mechanisms between the browser (client) and the web server using technologies such as Asynchronous JavaScript And XML (AJAX), XMLHttpRequest, Fetch API, and JSON.</li> <li>• <b>Web Application Programming:</b> The development of web applications focusing on RESTful APIs for data exchange over HTTP messages (e.g., GET, POST, PUT, DELETE) using prominent, popular frameworks (e.g. Spring Boot).</li> </ul>						
Learning Outcomes	<ul style="list-style-type: none"> <li>• Distinguish and Explain the Internet and the World-Wide Web (WWW):</li> <li>• Differentiate between the concepts of the Internet (a global network of interconnected computers) and the World-Wide Web (a system of interlinked documents and resources accessible via the Internet).</li> <li>• Understand Web Protocols, Components, and Web Services Programming:</li> <li>• Explain key web protocols (e.g., HTTP, HTTPS), identify core web components (e.g., web browsers, web servers), and understand the fundamentals of programming web services.</li> <li>• Develop Static and Dynamic Websites Using Client-Side Technologies:</li> <li>• Gain proficiency in using client-side web technologies such as HTML, Cascading Style Sheets (CSS), and JavaScript to design and develop static and dynamic websites.</li> <li>• Develop Server-Side Technologies for Database Communication:</li> <li>• Understand and utilize server-side web technologies, including PHP, Java, SQL, and JavaScript (e.g., Node.js), to enable communication with database systems for storing and retrieving data.</li> <li>• Establish Client-Server Communication Using AJAX:</li> <li>• Demonstrate the ability to use AJAX mechanisms (e.g., XMLHttpRequest, Fetch API) to enable asynchronous communication between client-side and server-side technologies.</li> <li>• Develop Web Applications Using RESTful APIs:</li> <li>• Exhibit skills in creating web applications that emphasize RESTful APIs for efficient data exchange over the HTTP protocol using prominent, popular frameworks (e.g. Spring Boot).</li> </ul>						
Prerequisites	CS 132 and CS 324		Required		None		
Course Content	Internet Fundamentals, WWW Fundamentals, Static and Dynamic Website Development, Client-Side Technologies (HTML, CSS, JavaScript, AJAX), Server-Side Technologies (PHP, Java, SQL), RESTful APIs.						
Teaching Methodology	Lectures (3 hours weekly), Recitation (1 hour weekly) and Laboratory sessions (1,5 hours weekly).						

Bibliography	<ol style="list-style-type: none"> <li>1. Web Programming with HTML, CSS, Bootstrap, JavaScript, React.JS, PHP, and MySQL Third Edition, 2022</li> <li>2. Hofstetter, Fred T. (2023). Computational Thinking on the Internet: Foundations, Web Design &amp; Cybersecurity (3rd ed.). Seattle: Kindle press.</li> <li>3. "Web Programming. Step by Step." M. Stepp, J. Miller, V. Kirst. Self published via lulu.com, 2014.</li> </ol>
Assessment	Final exam, midterm exam, and homework.
Language	Greek

Course Title	<b>Computer Graphics</b>						
Course Code	<b>CS 426</b>						
Course Type	Restricted Elective Course						
Level	Undergraduate						
Year / Semester	5 <sup>th</sup> or 6 <sup>th</sup> or 7 <sup>th</sup> or 8 <sup>th</sup> semester						
Teacher's Name	A. Aristidou / G. Chrysanthou						
ECTS	7,5	Lectures / week	2 x 1.5 hours	Recitation / week	1 x 1 hour	Laboratories / week	1 x 1.5 hours
Course Purpose and Objectives	Introduction to the basic principles of digital image synthesis. Explain how a 3-dimensional virtual world is defined starting from the geometry, the materials, the lights and cameras and how the 2-dimensional resulting image is produced by going through the graphics pipeline. Provision of both the theoretical foundations as well as practical skills through the use of industry standards, such as OpenGL or DirectX.						
Learning Outcomes	The course will provide students with an introduction to the basic principles of Computer Graphics and all the necessary knowledge they need to create images of virtual worlds starting from scratch. By following the course, students will learn how to define a three dimensional virtual world, giving the geometry, materials, textures, lights and camera parameters, and how to render the corresponding two-dimensional image. Students will learn how rendering is achieved either through the graphics pipeline or by ray-tracing, and they will see techniques for calculating local illumination, shadows and global illumination. At the practical laboratory, we will use graphics libraries such as OpenGL / WebGL and shaders, and implement an graphic application using UNITY game engine.						
Prerequisites	CS 232		Required		None		
Course Content	Scene construction, scene hierarchies, camera specification, projections of primitives, clipping, visible surface determination, polygon rasterisation (z-buffer), texture mapping, local and global illumination, shadows, ray tracing, radiosity, real-time acceleration techniques, GPU, and animation.						
Teaching Methodology	Lectures (3 hours weekly), Recitation (1 hour weekly) and Laboratory sessions (1.5 hours weekly).						
Bibliography	<ol style="list-style-type: none"> <li>1. Computer Graphics: Principles and Practice, J. F. Hughes, A. van Dam, M. McGuire, D. F. Sklar, J. D. Foley, S. K. Feiner, K. Akeley, Addison-Wesley Professional; 3rd edition, ISBN-13: 9780321399526, 2013.</li> <li>2. M. Slater, A. Steed and Y. Chrysanthou, Computer Graphics and Virtual Environments: From Realism to Real-Time, Addison Wesley, 2001.</li> <li>3. Θ. Θεοχάρης and Α. Μπεμ, Γραφικά – Αρχές and Αλγόριθμοι, Εκδοτικός Οίκος Συμμετρία, 1999.</li> </ol>						
Assessment	Final exam, midterm exam and homework.						
Language	Greek and English						

Course Title	<b>Mobile Computer Networks</b>						
Course Code	<b>CS 427</b>						
Course Type	Restricted Elective Course						
Level	Undergraduate						
Year / Semester	6 <sup>th</sup> or 7 <sup>th</sup> or 8 <sup>th</sup> semester						
Teacher's Name	P. Kolios						
ECTS	7,5	Lectures / week	2 x 1.5 hours	Recitation / week	1 x 1 hour	Laboratories / week	1 x 1.5 hours
Course Purpose and Objectives	The objective of this course is to introduce students into wireless mobile/local/cellular networks with an emphasis on the fundamental concepts and principles of the technology which are important for the design, application, evaluation and development of these systems. The course will also cover new architectures and topologies, existing and proposed standards as well as open research issues.						
Learning Outcomes	<p>Upon successful completion of this course the student will have advanced knowledge in the following fields:</p> <ul style="list-style-type: none"> <li>• Explain the following fundamental concepts of wireless and mobile networks: Wireless environment. Interference in a wireless environment. Basic principles of wireless data communication. Architectures and technologies of wireless networks and wireless communication. 2. Explain the following basic issues in wireless and mobile networks: Networking technologies classifications, such as wide area networks, metropolitan area networks, local area networks, body area networks. Infrastructure and Infrastructureless Networks, such as WLANs, Sensor Networks and the Internet of Things. Mobile networks, including 5G and beyond.</li> <li>• Demonstrate skills in deploying and analyzing various technologies of mobile/wireless networks.</li> <li>• Demonstrates ability to solve networking problems including analysis of protocols, and sizing and design issues in wireless networks.</li> <li>• Shows ability to use software defined radios (SDR) for the design and evaluation of networks.</li> <li>• Seek continuously new improved ways and mechanisms in wireless protocols/mobile networks.</li> <li>• Seek and analyze new techniques and technologies, networks such as the Internet of Things.</li> </ul>						
Prerequisites	CS 324		Required		None		
Course Content	Wireless environment, Interference and other problems in wireless communications, Architectures and technologies of wireless networks and wireless communication, Wireless MAC protocols, Wireless Local Area Networks (WLAN), Mobile Networks (emphasis on latest architectures, 3G, 4G, 5G), Mobility Management, wireless network technologies (ad-hoc, sensor, vehicular networks, Internet of Things), Open research issues and challenges						
Teaching Methodology	Lectures (3 hours weekly), Recitation (1 hour weekly) and Laboratory sessions (1.5 hours weekly).						
Bibliography	1. Wireless Communications Networks and Systems, C. Beard, W. Stallings, Pearson Higher Education, First Edition, January 2015.						
Assessment	Final exam, midterm exam, lab exercises and individual project.						
Language	Greek						

Course Title	<b><i>Internet of Things: Programming and Applications</i></b>						
Course Code	<b>CS 428</b>						
Course Type	Restricted Elective Course						
Level	Undergraduate						
Year / Semester	5 <sup>th</sup> or 6 <sup>th</sup> or 7 <sup>th</sup> or 8 <sup>th</sup> semester						
Teacher's Name	P. Kolios						
ECTS	7,5	Lectures / week	2 x 1.5 hours	Recitation / week	1 x 1 hour	Laboratories / week	1 x 1.5 hours
Course Purpose and Objectives	Introduction of the basic concepts for the design, implementation and programming of modern embedded systems and applications based on smart mobile devices, and sensor networks. Application development.						
Learning Outcomes	Familiarity with the basic concepts and methods of developing embedded systems. Introduction to mobile computing and programming on smart mobile devices. Application development using micro devices based on ARM microcontrollers, Arduino, and Rasperry Pi.						
Prerequisites	CS 222		Required		None		
Course Content	Programming of embedded systems. Introduction to the Internet of Things. Introduction to mobile devices, sensors and actuators for embedded system design. Interrupts and power consumption. Programming for embedded and mobile systems. Mobile computing. Architecture of Internet of Things, access networks. Programming in smartphones, embedded devices and applications for real-time/ online time series analysis. Network connectivity: Bluetooth, ZigBee, LoRA, NB-IOT. Cloud Computing. Connectivity on the cloud.						
Teaching Methodology	Lectures (3 weekly), Recitation (1 hour weekly) and Laboratory sessions (1.5 hours weekly).						
Bibliography	<ol style="list-style-type: none"> <li>1. Pascal Hirmer, Model-Based Approaches to the Internet of Things, Springer Cham., First Edition, 2023, ISBN 978-3-031-18883-1.</li> <li>2. Raj Kamal, Internet Of Things: Architecture and Design Principles McGraw Hill Education (India), ISBN-10: 93-5260-522-5, First Edition, 2017.</li> <li>3. Brunton SL, Kutz JN , Data-Driven Science and Engineering: Machine Learning, Dynamical Systems, and Control, Cambridge University Press, ISBN 978-1-108-42209-3, First Edition, 2019.</li> <li>4. E.A. Lee and S.A. Seshia , Introduction to Embedded Systems, MIT Press, ISBN: 978-0-262-53381-2, Second Edition, 2017.</li> <li>5. D. Hanes, G. Salgueiro, P. Grossetete, R. Barton, J. Henry , IoT Fundamentals: Networking Technologies, Protocols, and Use Cases for the Internet of Things, Cisco Press, ISBN-10: 1-58714-456-5, First Edition, 2017.</li> <li>6. P. Warden, D. Situnayake, TinyML: Machine Learning with TensorFlow Lite on Arduino and Ultra-Low-Power Microcontrollers, O'Reilly Media, First Edition, ISBN-10: 1492052043, 2020.</li> </ol>						
Assessment	Final exam, midterm exam, lab exercises and individual project.						
Language	Greek and English						

Course Title	<i>Theory and Practice of Compilers</i>						
Course Code	<b>CS 429</b>						
Course Type	Restricted Elective Course						
Level	Undergraduate						
Year / Semester	5 <sup>th</sup> or 6 <sup>th</sup> or 7 <sup>th</sup> or 8 <sup>th</sup> semester						
Teacher's Name	E. Athanasopoulos						
ECTS	7,5	Lectures / week	2 x 1.5 hours	Recitation / week	1 x 1 hour	Laboratories / week	1 x 1.5 hours
Course Purpose and Objectives	Fundamental principles of compiler design. Relation of translators to formal languages and automata theory. Lexical, syntactic and semantic analysis, code generation and optimization, etc. Practical exercises using lex and yacc.						
Learning Outcomes	<p>Upon completion of the course, students must be able to:</p> <ul style="list-style-type: none"> <li>• Describe the role of a compiler and distinguish the compilation phases and its differences and relationships with other related software such as a pre-processor, interpreter, debugger, linker, loader, and so on.</li> <li>• Describe the grammar of a programming language using regular expressions and production rules, recognize the importance of grammars for the development of a compiler, describe finite state automata, recognize their role as recognition engines, and parse trees,</li> <li>• describe the steps of running a syntax analyzer and how these can be created with the help of meta-tools,</li> <li>• describe the role of the symbol table and be able to choose the appropriate structure and organization to create it,</li> <li>• recognize the meaning of intermediate code, how it is created, and how it can be optimized,</li> <li>• recognize the characteristics of a final code and its execution environment.</li> </ul>						
Prerequisites	CS 211, CS 231			Required	None		
Course Content	Basic principles for designing translators. Correlation between standard languages, automated theory and translators. Dictionaries, syntactic and semantic analysis, code generation and optimization, etc. Practical exercises using lex and yacc.						
Teaching Methodology	Lectures (3 hours weekly), Recitation (1 hour weekly) and Laboratory sessions (1.5 hours weekly).						
Bibliography	<ol style="list-style-type: none"> <li>1. A. V. Aho, R. Sethi and J. D. Ullman, Compilers – Principles, Techniques, and Tools, Addison-Wesley, 1986.</li> <li>2. The Essence of Compilers, R. Hunter, Prentice Hall, 1999</li> <li>3. lex &amp; yacc, T. Mason and D. Brown, O'Reilly &amp; Associates Inc., 1990.</li> </ol>						
Assessment	Final exam, midterm exam and homework (theoretical and diagnostic assignments and semester project).						
Language	Greek						

Course Title	<i>Synthesis of Parallel Algorithms</i>						
Course Code	<i>CS 431</i>						
Course Type	Restricted Elective Course						
Level	Undergraduate						
Year / Semester	5 <sup>th</sup> or 6 <sup>th</sup> or 7 <sup>th</sup> or 8 <sup>th</sup> semester						
Teacher's Name	Chr. Georgiou						
ECTS	7,5	Lectures / week	2 x 1.5 hours	Recitation / week	1 x 1 hour	Laboratories / week	0 hours
Course Purpose and Objectives	Introduction to the fundamental techniques of parallel algorithm design and the use of these techniques in designing and analyzing parallel algorithms for basic problems.						
Learning Outcomes	<p>Upon successful completion of this class, the student is expected to be able to:</p> <ul style="list-style-type: none"> <li>• Identify and use basic techniques for designing parallel algorithms (pointer jumping, list ranking, Euler tour, tree raking, divide and conquer) for basic problems.</li> <li>• Identify the basic efficiency metrics (speedup, utilization) and complexity measures (parallel time and cost) of parallel algorithms.</li> <li>• Prove the correctness and analyze the worst-case complexity of parallel algorithms.</li> <li>• Identify the basic models of parallel computing (PRAM, Interconnecting networks, BSP) and know their basic parameters.</li> <li>• Employ basic techniques for the design and analysis of robust parallel algorithms in the F-PRAM and A-PRAM models of computation.</li> <li>• Implement and experimentally evaluate the performance of parallel algorithms using the XMT programming paradigm.</li> <li>• Demonstrate advanced algorithmic thinking.</li> <li>• Demonstrate advanced problem-solving skills.</li> </ul>						
Prerequisites	CS 231		Required		None		
Course Content	Introduction to parallel computing. Complexity and efficiency measurements of parallel algorithms. Parallel computing models. Basic techniques for the design of parallel algorithms. Efficient parallel algorithms in Combinatorics, Graph Theory, and Matrix Theory. Complexity analysis of algorithms on the Parallel Random Access Machine (PRAM). Comparison between various models of computation. Advanced topics (fault-tolerance, atomicity, synchronization, computational limitations of PRAM). The XMT programming paradigm.						
Teaching Methodology	Lectures (3 hours weekly) and Recitation (1 hour weekly).						
Bibliography	<ol style="list-style-type: none"> <li>1. J. Jaja, An Introduction to Parallel Algorithms, Addison-Wesley, 1992.</li> <li>2. C. Georgiou and A. A. Shvartsman, Cooperative Task-Oriented Computing: Algorithms and Complexity, Morgan &amp; Claypool, 2011.</li> </ol>						
Assessment	Final exam, midterm exam and homework (theoretical and programming assignments).						
Language	Greek						

Course Title	<b><i>Distributed Algorithms</i></b>						
Course Code	<b>CS 432</b>						
Course Type	Restricted Elective Course						
Level	Undergraduate						
Year / Semester	5 <sup>th</sup> or 6 <sup>th</sup> or 7 <sup>th</sup> or 8 <sup>th</sup> semester						
Teacher's Name	M. Mavronicolas						
ECTS	7,5	Lectures / week	2 x 1.5 hours	Recitation / week	1 x 1 hour	Laboratories / week	0 hours
Course Purpose and Objectives	Familiarization with the fundamental concepts of the Theory of Distributed Computing. Development of capabilities of designing, proving correct and analyzing distributed algorithms. Cultivation of syllogistic and mathematical approach to the field of distributed algorithms.						
Learning Outcomes	<p>The student who has successfully completed this course is expected to be able to:</p> <ul style="list-style-type: none"> <li>• Know the basic distributed computation models (shared memory versus messaging, determinism against randomization, concurrency, asynchronous and real-time concepts).</li> <li>• Know basic distributed algorithms and results of impossibility for fundamental problems such as mutual exclusion, agreement, synchronization, leader election, construction of minimal overlapping trees.</li> <li>• Demonstrate the accuracy and analyze the complexity of distributed algorithms.</li> <li>• Design distributed algorithms for problems using the algorithms and techniques taught in the course as structural elements.</li> <li>• Apply the lower limits and weaknesses learned in the course to new problems.</li> </ul>						
Prerequisites	CS 211, CS 231		Required		None		
Course Content	Formal models of distributed computing: shared memory versus message passing, determinism versus randomization, concepts of synchronism, asynchrony and real-time. Design and analysis of distributed algorithms and impossibility/improbability results for fundamental problems such as mutual exclusion, consensus, synchronization, leader election, construction of minimum spanning trees. Fault tolerance: Byzantine generals, wait-free algorithms, fault degrees. Formal methods for proving correctness of distributed algorithms. Advanced topics. Special emphasis throughout the course on lower and upper bounds on time and memory.						
Teaching Methodology	Lectures (3 hours weekly) and Recitation (1 hour weekly).						
Bibliography	1. H. Attiya and J. L. Welch, Distributed Computing: Fundamentals, Simulations and Advanced Topics, 2nd Edition, John Wiley and Sons Inc., 2003.						
Assessment	Final exam, midterm exam, homework, participation in class and attendance.						
Language	Greek						

Course Title	<b><i>Constraint Programming and Satisfaction</i></b>						
Course Code	<b>CS 433</b>						
Course Type	Restricted Elective Course						
Level	Undergraduate						
Year / Semester	5 <sup>th</sup> or 6 <sup>th</sup> or 7 <sup>th</sup> or 8 <sup>th</sup> semester						
Teacher's Name	Y. Dimopoulos						
ECTS	7,5	Lectures / week	2 x 1.5 hours	Recitation / week	1 x 1 hour	Laboratories / week	0 hours
Course Purpose and Objectives	A significant number of problems in Computer Science over a wide spectrum ranging from Computer Vision and Artificial Intelligence to the management of Computer Networks and Scheduling are special cases of Constraint Satisfaction problem. This course introduces the way to approach such problems and corresponding software. Students will be able to understand the structure and the behavior of Constraint Satisfaction problems and will get exposure to basic algorithms solving them. They will get experience over tools for Constraint Programming, the range of solvable problems and their applications to problem solving.						
Learning Outcomes	Knowledge of basic algorithms for solving Constraint Satisfaction and Optimization problems. Understanding local consistency, intelligent search, optimization and problem structure. Familiarization with practical tools and solving methods, and their application to industrial problems.						
Prerequisites	CS 111, CS 231		Required		None		
Course Content	Definition of constraint satisfaction problems. Constraint representation and complexity. Various forms of consistency. Backtracking and look-ahead techniques. Intelligent backtracking and condition for solution finding without backtracking. Heuristic and local methods for solution searching. Available commercial products. Study of problems from different application domains, their modeling and the complexity of various algorithms solving them.						
Teaching Methodology	Lectures (3 hours weekly) and Recitation (1 hour weekly).						
Bibliography	<ol style="list-style-type: none"> <li>1. R. Dechter, Constraint Processing, Morgan Keufmann, 2003.</li> <li>2. E. Tsang, Foundations of Constraint Satisfaction, Academic Press, 1993.</li> </ol>						
Assessment	Final exam, midterm exam and homework (programming assignments).						
Language	Greek						

Course Title	<b><i>Logic Programming and Artificial Intelligence</i></b>						
Course Code	<b><i>CS 434</i></b>						
Course Type	Restricted Elective Course						
Level	Undergraduate						
Year / Semester	5 <sup>th</sup> or 6 <sup>th</sup> or 7 <sup>th</sup> or 8 <sup>th</sup> semester						
Teacher's Name	CS or Visiting Faculty						
ECTS	7,5	Lectures / week	2 x 1.5 hours	Recitation / week	1 x 1 hour	Laboratories / week	1 x 1 hours
Course Purpose and Objectives	Familiarization with the basic concepts of Logic Programming and practical exercises in implementing them with the PROLOG language. Development of capabilities of applying Logic Programming to problems of Artificial Intelligence.						
Learning Outcomes	<p>Students of the course will have the following knowledge and skills:</p> <ul style="list-style-type: none"> <li>• Theoretical model of Logic Programming</li> <li>• Programming in PROLOG</li> <li>• The Refutation Theory as Failure in Logic Programming</li> <li>• Logic Programming Applications</li> <li>• Post-programming elements</li> <li>• Argumentation and Common Logic</li> <li>• Chronological reasoning in Common Logic</li> <li>• Programming under the GORGIA argument system</li> <li>• Developing Cognitive Systems</li> </ul>						
Prerequisites	CS 111	Required			None		
Course Content	Basic principles of Logic Programming and implementation using the language Prolog. Relation of Logic Programming to modern considerations regarding Artificial Intelligence. Solving application problems drawn from the fields of Artificial Intelligence and the Semantic Web, making use of Logic Programming and Constraint Logic Programming.						
Teaching Methodology	Lectures (3 hours weekly), Recitation and Laboratory sessions (1 hour weekly).						
Bibliography	1. L. Sterling and E. Shapiro, The Art of Prolog, 2nd Edition, The MIT Press, 1994.						
Assessment	Final exam, midterm exam and homework (programming assignments).						
Language	English						

Course Title	<b><i>Human Computer Interaction</i></b>						
Course Code	<b><i>CS 435</i></b>						
Course Type	Restricted Elective Course						
Level	Undergraduate						
Year / Semester	5 <sup>th</sup> or 6 <sup>th</sup> or 7 <sup>th</sup> or 8 <sup>th</sup> semester						
Teacher's Name	M. Konstantinidis						
ECTS	7,5	Lectures / week	2 x 1.5 hours	Recitation / week	1 x 1 hour	Laboratories / week	1 x 1.5 hours
Course Purpose and Objectives	Appreciation of the importance of designing good user interfaces and the relation of user interface design and the way users interact with computers. Experience with applying a well-known methodology for designing interactive systems, starting from identifying user's needs, to applying usability evaluation methods.						
Learning Outcomes	<p>At the end of the course the students are expected to be able:</p> <ul style="list-style-type: none"> <li>• Problem Identification and Research Skills: Analyze and identify a complex problem or research question in computer science and formulate a plan for addressing it using established methodologies.</li> <li>• Technical Implementation: Apply knowledge of computer science principles, programming, and software development to design and implement a solution or carry out research that addresses the chosen problem.</li> <li>• Project Management: Demonstrate effective project management skills, including planning, time management, resource allocation, and iterative development, to complete a substantial project within a set timeline.</li> <li>• Critical Analysis and Evaluation: Critically evaluate the performance, effectiveness, and limitations of the developed solution or research outcome, using both qualitative and quantitative metrics.</li> <li>• Communication Skills: Produce a well-structured written report that effectively communicates the problem, methodology, results, and conclusions of the project, following academic and professional standards.</li> <li>• Collaboration and Supervision: Work independently while also demonstrating the ability to incorporate feedback from supervisors and peers to refine the project.</li> <li>• Ethical Considerations: Understand and apply ethical principles and considerations, such as data privacy and responsible AI, in the execution of the project.</li> </ul>						
Prerequisites	None		Required		None		
Course Content	Analysis of the human as a computer system user (knowledge models, graphical animation, cognitive models). Interactive technologies (input-output devices, window environments, systems for collaborative support, virtual reality). Methodologies for the design of interactive systems. User-centered design and evaluation methods.						
Teaching Methodology	Lectures (3 hours weekly), Recitation (1 hour weekly) and Laboratory sessions (1,5 hours weekly).						
Bibliography	<ol style="list-style-type: none"> <li>1. Designing with the Mind in Mind: Simple Guide to Understanding User Interface Design Guidelines (3rd Edition) by Jeff Johnson (2020)</li> <li>2. About Face: The Essentials of Interaction Design, 4th Edition Alan Cooper, Robert Reimann, David Cronin, Christopher Noessel ISBN: 978-1-118-76657-6 September 2014 720 pages</li> <li>3. J. Preece, Y. Rogers and H. Sharp, Interaction Design: Beyond Human-Computer Interaction, John Wiley and Sons, 2002.</li> </ol>						
Assessment	Final exam, midterm exam and homework.						
Language	Greek						

Course Title	<i>Advanced Software Engineering</i>						
Course Code	<b>CS 441</b>						
Course Type	Restricted Elective Course						
Level	Undergraduate						
Year / Semester	5 <sup>th</sup> or 6 <sup>th</sup> or 7 <sup>th</sup> or 8 <sup>th</sup> semester						
Teacher's Name	G. Papadopoulos						
ECTS	7,5	Lectures / week	2 x 1.5 hours	Recitation / week	1 x 1 hour	Laboratories / week	1 x 1.5 hours
Course Purpose and Objectives	Familiarization and understanding of advanced principles, concepts and practices of software engineering. A number of contemporary areas of software engineering will be covered. This course further serves as a “roadmap” for advanced electives and graduate courses in software engineering.						
Learning Outcomes	<p>The learning outcomes for the students are the following:</p> <ul style="list-style-type: none"> <li>• Understand advanced techniques for managing complex software projects.</li> <li>• Develop team projects by applying the principles of team software development.</li> <li>• Be aware of the basic principles of software reuse through the different techniques available for this purpose.</li> <li>• Be able to apply the basic principles of Human-Machine Interaction in order to make applications accessible to users.</li> </ul>						
Prerequisites	CS 343			Required		None	
Course Content	Software reuse. Distributed software engineering. Service oriented architectures. Real-time systems. Project Management. Human-Computer Interaction for Software Engineering (interaction design, design rules, implementation and evaluation, universal design, user support).						
Teaching Methodology	Lectures (3 hours weekly), Recitation (1 hour weekly) and Laboratory sessions (1.5 hours weekly).						
Bibliography	<ol style="list-style-type: none"> <li>1. I. Sommerville, Software Engineering, 10th Edition, Addison-Wesley, 2016.</li> <li>2. A. Dix, J. Finlay, G. D. Abowd, R. Beale, Human Computer Interaction, 3rd Edition, Prentice Hall, 2004.</li> </ol>						
Assessment	Final exam, midterm exam, homework (theoretical problems – study / analysis of a software system) and laboratory exercise.						
Language	English						

Course Title	<b>Machine Learning</b>						
Course Code	<b>CS 442</b>						
Course Type	Restricted Elective Course						
Level	Undergraduate						
Year / Semester	5 <sup>th</sup> or 6 <sup>th</sup> or 7 <sup>th</sup> or 8 <sup>th</sup> semester						
Teacher's Name	Chr. Christodoulou						
ECTS	7,5	Lectures / week	2 x 1.5 hours	Recitation / week	1 x 1 hour	Laboratories / week	1 x 1.5 hours
Course Purpose and Objectives	Familiarization with theoretical and practical issues involved in Machine Learning. Study of machine learning methods as they have been developed in recent years. Implementation and assessment of Machine Learning systems.						
Learning Outcomes	<p>The learning outcomes for the students are the following:</p> <ul style="list-style-type: none"> <li>• develop an appreciation for what is involved in learning from data</li> <li>• understand a wide variety of learning algorithms including supervised, unsupervised as well as reinforcement learning algorithms and be able to implement them</li> <li>• understand how to apply a variety of learning algorithms to data</li> <li>• learn to be able to identify and apply the most appropriate machine learning technique to classification, regression and optimisation problems</li> <li>• understand the strengths and weaknesses of many popular machine learning approaches</li> <li>• be able to design and implement various machine learning algorithms in a range of real-world applications</li> </ul>						
Prerequisites	CS 231			Required		None	
Course Content	Introduction to Pattern Recognition, Multilayered Neural Networks and backpropagation learning algorithm, Deep Learning and Convolutional Neural Networks, Recurrent Neural Networks, Self-Organising Maps, Radial Basis Functions, Reinforcement Learning, Hopfield Networks & Boltzmann Machines. Survey of the developments in artificial intelligence, machine learning, expert systems, cognitive science, robotics and artificial neural networks, which contributed to the development of the theory of machine learning systems.						
Teaching Methodology	Lectures (3 hours weekly), Recitation (1 hour weekly) and Laboratory sessions (1,5 hours weekly).						
Bibliography	<ol style="list-style-type: none"> <li>1. C. M. Bishop, Neural Networks for Pattern Recognition, Oxford University Press, 1995.</li> <li>2. S. Haykin, Neural Networks and Learning Machines, 3rd Edition, Pearson Education, 2009.</li> <li>3. R. S. Sutton and A. G. Barto, Reinforcement Learning: an Introduction, MIT Press, 2nd Ed., 2018.</li> <li>4. A. Zhang, Z. C. Lipton, M. Li and A. J. Smola. Dive into Deep Learning, Cambridge University Press, 2023.</li> </ol>						
Assessment	Final exam, midterm exam and homework (lab exercises).						
Language	Greek						

Course Title	<b>Software Reuse</b>						
Course Code	<b>CS 443</b>						
Course Type	Restricted Elective Course						
Level	Undergraduate						
Year / Semester	5 <sup>th</sup> or 6 <sup>th</sup> or 7 <sup>th</sup> or 8 <sup>th</sup> semester						
Teacher's Name	G. Kapitsaki						
ECTS	7,5	Lectures / week	2 x 1.5 hours	Recitation / week	1 x 1 hour	Laboratories / week	1 x 1.5 hours
Course Purpose and Objectives	Understanding the usefulness of software reuse. Deepening in the different levels of reuse and understanding the differences between them. Use of software design patterns in practice. Understanding of Open Source Software and licenses. Effective reuse of content and data from software repositories.						
Learning Outcomes	<p>The learning outcomes for the students are the following:</p> <ul style="list-style-type: none"> <li>• Understanding the usefulness of software reuse.</li> <li>• Ability to differentiate between the different levels of reuse and understanding the differences between them.</li> <li>• Use of software design patterns in practice.</li> <li>• Understanding of Open Source Software and the differences between software licenses.</li> <li>• Ability to make the results of their work available to others via software repositories.</li> </ul>						
Prerequisites	CS 343			Required		None	
Course Content	Levels of reuse. Best practices for reuse. Evolution of reuse. Software repositories. Search and retrieval. Design patterns. Object-oriented programming standards. Open source software. Open source software licensing and legal issues. Organization policies and open-source based development. Outsourcing. Model-Driven Engineering principles. Service-Oriented Computing. Aspect-Oriented Programming. Content and data reuse from software repositories.						
Teaching Methodology	Lectures (3 hours weekly), Recitation (1 hour) and Laboratory sessions (1,5 hours weekly).						
Bibliography	<ol style="list-style-type: none"> <li>1. M. Ezran, M. Morisio, C. Tully, Practical Software Reuse, Practitioner Series, 2002.</li> <li>2. E. Freeman, E. Robson, B. Bates, K. Sierra, Head First Design Patterns, O'Reilly Media, 2004.</li> <li>3. C. Horstmann, A Practical Guide to Open Source Licensing, Wiley, 2nd Edition, 2006.</li> </ol>						
Assessment	Final exam, midterm exam, homework and semester project.						
Language	Greek						

Course Title	<b><i>Computational Intelligent Systems</i></b>						
Course Code	<b><i>CS 444</i></b>						
Course Type	Restricted Elective Course						
Level	Undergraduate						
Year / Semester	5 <sup>th</sup> or 6 <sup>th</sup> or 7 <sup>th</sup> or 8 <sup>th</sup> semester						
Teacher's Name	Chr. Christodoulou						
ECTS	7,5	Lectures / week	2 x 1.5 hours	Recitation / week	1 x 1 hour	Laboratories / week	1 x 1.5 hours
Course Purpose and Objectives	Global overview of Computational Intelligence and its applications in solving "real" problems in various disciplines such as decision making support, classification, prognosis and prediction, system optimization and recreational design. Moreover there will be an introduction to computational neuroscience/neuroinformatics as well as in cognitive science.						
Learning Outcomes	<p>The learning outcomes for the students are the following:</p> <ul style="list-style-type: none"> <li>• understand the fundamental concepts of computational intelligence</li> <li>• be able to design and implement genetic and evolutionary algorithms as well as fuzzy systems for solving practical problems</li> <li>• appreciate the advantages of genetic algorithm and evolutionary based approaches for optimisation problems, compared to traditional methodologies</li> <li>• understand the basic concepts of cognitive science like mind, brain, memory and behaviour and modelling approaches</li> <li>• understand and be able to explain the fundamental principles of information processing by neural systems</li> <li>• appreciate the importance of computational neuronal models in the quest of understanding the brain</li> <li>• understand the most important biophysical neuronal models and the different levels of description and complexity in computational neuronal modelling from the level of the single neuron to that of neuronal networks</li> <li>• understand how experimentally recorded physiological signals enable us to understand the functionality of neurons/systems in the brain and how statistical approaches help in the analysis of such data</li> </ul>						
Prerequisites	None		Required		None		
Course Content	Evolutionary Computing. Genetic Algorithms. Artificial Neural Networks. Fuzzy Systems. Artificial Life. Computational Neuroscience/Neuroinformatics; Hodgkin & Huxley and Integrate-and-Fire neuron models; Neural Coding; Hebbian Learning and Synaptic Plasticity; introduction to cognitive science. Development and Implementation of Computational Intelligence Systems.						
Teaching Methodology	Lectures (3 hours weekly), Recitation (1 hour weekly) and Laboratory sessions (1.5 hours weekly).						
Bibliography	<ol style="list-style-type: none"> <li>1. A. P. Engelbrecht, Computational Intelligence: An Introduction, John Wiley and Sons, 2nd Edition 2007.</li> <li>2. R. C. Eberhart and Y. Shi, Computational Intelligence: Concepts to Implementations, Elsevier, 2007.</li> <li>3. E. R. Kandel, Αναζητώντας τη Μνήμη, (Μετάφραση Α. Καραμανίδης), Πανεπιστημιακές Εκδόσεις Κρήτης, 2008.</li> <li>4. P. Dayan and L. Abbott, Theoretical Neuroscience: Computational and Mathematical Modelling of Neural Systems, MIT Press, 2001.</li> </ol>						
Assessment	Final exam, midterm exam and homework (laboratory exercises, additional exercises, final project).						
Language	Greek						

Course Title	<b>Digital Image Processing</b>						
Course Code	<b>CS 445</b>						
Course Type	Restricted Elective Course						
Level	Undergraduate						
Year / Semester	5 <sup>th</sup> or 6 <sup>th</sup> or 7 <sup>th</sup> or 8 <sup>th</sup> semester						
Teacher's Name	C. Pattichis						
ECTS	7,5	Lectures / week	2 x 1.5 hours	Recitation / week	1 x 1 hour	Laboratories / week	0
Course Purpose and Objectives	Introduction to the basic principles of Digital Image Processing: Digital Image and Video. Analysis and implementation of image and video processing and analysis algorithms and their application in industrial and biomedical systems.						
Learning Outcomes	<ul style="list-style-type: none"> <li>Familiarization with the basic concepts and methods of developing simple image and video processing and analysis systems.</li> <li>Introduction and understanding of image and video processing and analysis instruction set.</li> <li>Develop applications utilizing the above.</li> </ul>						
Prerequisites	CS 231, MAS 029		Required		None		
Course Content	Binary Image Representation. Image Histogram and Point Operations. Discrete Fourier Transform. Linear Image Filtering. Non Linear Image Filtering Pipeling. Image Compression. Image Analysis I. Image Analysis II. Digital Video Processing.						
Teaching Methodology	Lectures (3 hours weekly), Recitation (1 hour weekly).						
Bibliography	1. R. C. Gonzalez and R. E. Woods, Digital Image Processing, 4th Edition, Pearson, 2017.						
Assessment	Final exam, midterm exam and homework (laboratory exercises, additional exercises, final project).						
Language	Greek						

Course Title	<i>Advanced Database Systems</i>						
Course Code	<b>CS 446</b>						
Course Type	Restricted Elective Course						
Level	Undergraduate						
Year / Semester	6 <sup>th</sup> or 7 <sup>th</sup> or 8 <sup>th</sup> semester						
Teacher's Name	D. Zeinalipour						
ECTS	7,5	Lectures / week	2 x 1.5 hours	Recitation / week	1 x 1 hour	Laboratories / week	1 x 1.5 hours
Course Purpose and Objectives	Familiarization with advanced topics in the design and management of Databases (and special kinds of those). Exposure to significant open problems and research directions in the field of Databases.						
Learning Outcomes	<p>Upon successful completion of the course, the student will be able to:</p> <ul style="list-style-type: none"> <li>• Understand physical database design along with implementation issues and also devise appropriate ways to store and index data.</li> <li>• Demonstrate understanding of issues surrounding query optimization, concurrency control, parallelism and recovery in data management.</li> <li>• Implement the internal components of a database system in the programming language C++ under the Minibase architecture.</li> <li>• Analyze the time complexity of secondary memory algorithms.</li> <li>• Compare different techniques applicable to distributed databases.</li> </ul>						
Prerequisites	CS 342			Required		None	
Course Content	Theoretical approach to logical and physical design of databases. Algorithms for logical and physical design of databases. Primary and secondary indexing techniques. Advanced query processing and query optimization. Query parallelism. Concurrency control and recovery, integrity and security of data. Distributed databases and introductory concepts distributed transaction processing involving multiple and heterogeneous databases. Problems of interfacing a database with software.						
Teaching Methodology	Lectures (3 hours weekly), Recitation (1 hour weekly) and Laboratory sessions (1.5 hours weekly).						
Bibliography	<ol style="list-style-type: none"> <li>1. Database Management Systems: Paperback Edition, 3 Edition, Raghu Ramakrishnan and Johannes Gehrke, McGraw-Hill Publishers, Paper; 1065 pp, ISBN: 0-07-123057-2, 2003.</li> <li>2. Fundamentals of Database Systems, 7/E Ramez Elmasri, Shamkant B. Navathe, ISBN-10: 0133970779, ISBN-13: 9780133970, 2016.</li> <li>3. Principles of Distributed Database Systems, Özsu, M. Tamer, Valduriez, Patrick, 3rd Edition, 846 p., Springer Press, 2011.</li> </ol>						
Assessment	Final exam, midterm exam, programming exercises and presentation.						
Language	Greek						

Course Title	<b>Computer Vision</b>						
Course Code	<b>CS 447</b>						
Course Type	Restricted Elective Course						
Level	Undergraduate						
Year / Semester	5 <sup>th</sup> or 6 <sup>th</sup> or 7 <sup>th</sup> or 8 <sup>th</sup> semester						
Teacher's Name							
ECTS	7,5	Lectures / week	2 x 1.5 hours	Recitation / week	1 x 1 hour	Laboratories / week	1 x 1.5 hours
Course Purpose and Objectives	Familiarity with the basic concepts and methods of developing a simple computerized vision system. Introduction and understanding of commands in a programming language to support the required functionality for developing computational vision systems. Develop applications utilizing the above.						
Learning Outcomes	<ul style="list-style-type: none"> <li>• Familiarization with the basic concepts and methods of developing simple computer vision systems.</li> <li>• Introduction and understanding of basic computer vision instruction set.</li> <li>• Develop applications utilizing the above.</li> </ul>						
Prerequisites	CS 231, MAS 029		Required		None		
Course Content	Basic concepts and methodologies relating to the subject of Computer Vision. Image information, image processing, feature extraction. Image segmentation, clustering, multiple-image processing, case studies.						
Teaching Methodology	Lectures (3 hours weekly), Recitation (1 hour weekly) and Laboratory sessions (1.5 hours weekly).						
Bibliography	<ol style="list-style-type: none"> <li>1. D. Forsyth and J. Ponce, Computer Vision: A Modern Approach, 2nd Edition, Pearson, 2011.</li> <li>2. S.D. Prince, Computer Vision: Models, Learning and Inference, 2012.</li> <li>3. R. Hartley and A. Zeisserman, Multiple View Geometry, Cambridge University Press, 2003.</li> <li>4. C. Bishop, Pattern Recognition and Machine Learning, Springer-Verlag, 2007.</li> <li>5. O. Faugeras and Q.T. Luong, Geometry of Multiple Images, MIT Press, 2001.</li> </ol>						
Assessment	Final exam, midterm exam, programming exercises and presentation.						
Language	Greek						

Course Title	<b><i>Data Mining on the Web</i></b>						
Course Code	<b><i>CS 448</i></b>						
Course Type	Restricted Elective Course						
Level	Undergraduate						
Year / Semester	5 <sup>th</sup> or 6 <sup>th</sup> or 7 <sup>th</sup> or 8 <sup>th</sup> semester						
Teacher's Name	G. Pallis						
ECTS	7,5	Lectures / week	2 x 1.5 hours	Recitation / week	1 x 1 hour	Laboratories / week	1 x 1.5 hours
Course Purpose and Objectives	Introduction to data mining, Clustering, Classification, Association Rules, Link Analysis, Web communities, Web Personalization.						
Learning Outcomes	<ul style="list-style-type: none"> <li>• Understanding Big data systems like Map-reduce</li> <li>• Explain algorithms for massive data sets and methodologies in the context of data mining (i.e., Link analysis such as PageRank, spam detection and hubs-and-authorities, Computational advertising)</li> <li>• Demonstrate the ability to match various algorithms for particular classes of problems (i.e. Similarity search such as locality-sensitive hashtag, association rules, clustering, classification)</li> <li>• Apply and develop algorithms as a part of software development for mining big data</li> </ul>						
Prerequisites	CS 342	Required			None		
Course Content	Data mining on the Web refers to the automatic discovery of interesting and useful patterns from the data associated with the usage, content, and the linkage structure of Web resources. It has quickly become one of the most popular areas in computing and information systems because of its direct applications in e-commerce, information retrieval/filtering, Web personalization, and recommender systems. The primary focus of this course is on examining techniques from data mining to extract useful knowledge from Web data. This course will be focused on a detailed overview of the data mining process and techniques, specifically those that are most relevant to Web mining. Several topics will be covered such as Map-Reduce framework, Web data clustering, classification, association rules, recommendation systems, link analysis, social networks and Web advertising.						
Teaching Methodology	Lectures (3 hours weekly), Recitation (1 hour weekly) and Laboratory sessions (1,5 hour weekly).						
Bibliography	1. J. Leskovec, A. Rajaraman, J. D. Ullman, Mining of Massive Datasets, Cambridge University Press, 2020.						
Assessment	Final exam, midterm exam, homework (programming and theoretical assignments) and semester project.						
Language	Greek						

Course Title	<b><i>Professional Practice in Software Engineering</i></b>						
Course Code	<b><i>CS 449</i></b>						
Course Type	Restricted Elective Course						
Level	Undergraduate						
Year / Semester	5 <sup>th</sup> or 6 <sup>th</sup> or 7 <sup>th</sup> or 8 <sup>th</sup> semester						
Teacher's Name	G. Kapitsaki						
ECTS	7,5	Lectures / week	2 x 1.5 hours	Recitation / week	1 x 1 hour	Laboratories / week	1 x 1.5 hours
Course Purpose and Objectives	Embedding and practical application of the theoretical approaches and methodologies of Software Engineering for the development of a product-software system that serves the needs of an organization belonging to the local or international market. Practical use of software processes and tools, such as central and distributed version control systems (Git), testing at different levels, group communication, professionalism, and ethical conduct. Application of the Scrum development methodology.						
Learning Outcomes	<p>The learning outcomes for the students are the following:</p> <ul style="list-style-type: none"> <li>• Embedding and practical application of the theoretical approaches and methodologies of Software Engineering for the development of a product-software system that serves the needs of an organization belonging to the local or international market.</li> <li>• Applying software development steps effectively, via the use of the Scrum development methodology.</li> <li>• Applying effectively testing processes of software systems.</li> <li>• Ability to measure different aspects of software systems.</li> <li>• Strengthening of group communication, professionalism, and ethical conduct.</li> </ul>						
Prerequisites	CS 343	Required			None		
Course Content	Undertake and carrying out to completion a significant software project by small student groups (of about 2-6 students each). All phases in the development of software, especially on implementation and testing. Most of the specific projects come from the industrial sector. Roles and responsibilities in a software team. Version control systems (central and distributed, e.g., Git). Version control branching strategies. Use of the Scrum development methodology (Scrum review meeting, Scrum retrospective, etc.). Testing, test-driven development and automated testing. Software system analysis through software metrics (e.g., Lines of Code). Specialized issues depending on the project nature (e.g., deployment on web servers, GUI tools and frameworks etc.). Open Source software and licenses. Data privacy legislation and connection with software features.						
Teaching Methodology	Students are grouped in teams of 2-6 persons. Meetings/discussions are held regularly (weekly per team), Recitation (1 hour weekly) and Laboratory sessions (1.5 hours weekly).						
Bibliography	1. Selected articles from the literature and tools documentation.						
Assessment	Assessment of the product-software system, assessment of the corresponding documentation and deliverables, quizzes, oral presentation and final exam.						
Language	Greek or English						

Course Title	<i>Network Virtualisation and Management</i>						
Course Code	<i>CS 450</i>						
Course Type	Restricted Elective Course						
Level	Undergraduate						
Year / Semester	5 <sup>th</sup> or 6 <sup>th</sup> or 7 <sup>th</sup> or 8 <sup>th</sup> semester						
Teacher's Name	V. Vassiliou						
ECTS	7,5	Lectures / week	2 x 1.5 hours	Recitation / week	1 x 1 hour	Laboratories / week	1 x 1.5 hours
Course Purpose and Objectives	<p>The course aims to cover the latest approaches in network virtualization (e.g. Software Defined Networks, Network Function Virtualisation, Virtual Infrastructure Management, VNF-Virtual Network Functions, NFV Management and orchestration, Network Services) and the cloud (e.g. OpenStack), as well as more traditional techniques of network management (e.g. SNMP).</p> <p>The course will also introduce the Network Management Fundamentals, explain the different technologies that are used in network management (traditional, e.g. SNMP, and latest virtualised approaches, such as SDN/NFV), and how they relate to each other. It will also introduce different management reference models, such as Fault, Configuration, Accounting, Performance, and Security (FCAPS), the different building blocks of network management, the protocols used, the organization of data, and the management communication aspects.</p>						
Learning Outcomes	<p>On completion of the course the students will:</p> <ul style="list-style-type: none"> <li>• know basic terminology and concepts in network virtualization (e.g. NFV-Network Function Virtualisation, VNF-Virtual Network Functions, Network Services Orchestration, cloud-based network management) and be able to apply them to solve more advanced tasks.</li> <li>• know basic terminology and concepts in overall Network Management and Services.</li> <li>• be able to build, configure, and deploy an SDN-based network and setup open flow type networking.</li> <li>• have working knowledge of concepts: Network Virtualisation, Virtualised Network Services, Network Services Orchestration, and Network Slicing.</li> <li>• be able to build, configure, deploy, and monitor via SNMP an IP-network, with respect to computers and networking equipment.</li> <li>• be able to configure and deploy the monitoring of a network, computers and services.</li> <li>• be able to use basic management tools for administration of networks, computers and services.</li> <li>• be able to evaluate problems that may arise at management of networked systems.</li> </ul>						
Prerequisites	CS324		Required		None		

Course Content	<ul style="list-style-type: none"> <li>• Networking technologies overview: A. LANs, WLANs, WANs, PANs, MANs, etc...B. BGP, VPN, VLANs and DNS review, introduction to cloud computing.</li> <li>• Introduction to network management.</li> <li>• Brief review of network management standards, models, languages and SLAs.</li> <li>• Management functional areas (FCAPS): Fault Management, Configuration Management, Accounting Management, Performance Management, Security Management.</li> <li>• Network virtualisation and virtualised network management: need and fundamentals (NFV MANO, ETSI MANO, OPENBATTON)</li> <li>• Software Defined Networks and Open Flow (Mininet lab examples with ONOS).</li> <li>• Network Virtualization and Virtualised Services</li> <li>• (e.g. NFV-Network Function Virtualisation, VNF-Virtual Network Functions, Network Service Orchestration, cloud based network management).</li> <li>• Traditional and new network monitoring</li> <li>• MIB,OID and SMI, SNMPv1,v2,v3, RMON, MRTG, and new monitoring tools (e.g. NAGIOS, ZENOSS).</li> <li>• Network management protocols and tools (e.g. ipconfig, ping, ...), NMS (e.g. Nagios, Cacti, OpenView, NetView, CygNet, etc...), NetConf and YANG.</li> <li>• Performance Management:</li> <li>• Quality of Service (QoS) architectures and traffic regulation mechanisms, Network Security management.</li> <li>• Other Topics and Trends</li> <li>• Newer network management tools (e.g.XML based tools)</li> <li>• Cloud Services Management</li> <li>• New thinking and directions.</li> </ul>
Teaching Methodology	Lectures (3 hours weekly), Recitation (1 hour weekly) and Laboratory sessions (1.5 hours weekly).
Bibliography	<ol style="list-style-type: none"> <li>1. William Stallings, Foundations of Modern Networking: SDN, NFV, QoE, IoT, and Cloud, Pearson, 2022.</li> <li>2. Jim Doherty, SDN and NFV Simplified: A Visual Guide to Understanding Software Defined Networks and Network Function Virtualization, 1st Edition, Pearson, 2016,.</li> <li>3. Alexander Clemm, Network Management Fundamentals, 2006, Cisco Press, 2006</li> </ol>
Assessment	Final exam, midterm exam and homework.
Language	Greek or English

Course Title	<b>Software Analysis</b>						
Course Code	<b>CS 451</b>						
Course Type	Restricted Elective Course						
Level	Undergraduate						
Year / Semester	5 <sup>th</sup> or 6 <sup>th</sup> or 7 <sup>th</sup> or 8 <sup>th</sup> semester						
Teacher's Name	E. Athanasopoulos						
ECTS	7,5	Lectures / week	2 x 1.5 hours	Recitation / week	1 x 1 hours	Laboratories / week	0 hours
Course Purpose and Objectives	The course explores fundamental concepts in analyzing software of multiple forms and for different purposes. Many times, we need to analyze software for (a) locating bugs (debugging), (b) measuring performance bottlenecks (profiling), (c) adding instrumentation that enhances a program's behavior (e.g., add a security defense). The course exposes several techniques for working directly with the binary form of software (binary analysis and re-writing), as well as exploring and augmenting the source of C/C++ programs through the extension of modern compiler toolchains (LLVM).						
Learning Outcomes	<ul style="list-style-type: none"> <li>Analyse binaries in Unix.</li> <li>Instrument binaries in Unix.</li> <li>Create proof-of-concept debuggers.</li> <li>Modify modern compilers to enhance the functionality of existing C/C++ code.</li> </ul>						
Prerequisites	CS211, CS232			Required		None	
Course Content	ELF format of Unix binaries. Tools that can work and explore binaries in Unix (show different sections, symbols, shared libraries, etc.). How relocations and shared libraries work in binaries (e.g., the usage of GOT). Using ptrace(). Disassembling binaries using the Capstone framework. Re-writing binaries programmatically. Pre-loading binaries. Dynamic and static analysis of binary code. C/C++ instrumentation through LLVM passes. Applications of software analysis.						
Teaching Methodology	Lectures and hands-on sessions.						
Bibliography	<ol style="list-style-type: none"> <li>Practical Binary Analysis: Build Your Own Linux Tools for Binary Instrumentation, Analysis, and Disassembly. Dennis Andriess. ISBN-10: 1593279124.</li> <li>LLVM documentation.</li> <li>Published papers.</li> </ol>						
Assessment	Final exam, midterm exam, programming assignments.						
Language	Greek or English						

Course Title	<b><i>Datacenter Computing</i></b>						
Course Code	<b><i>CS 452</i></b>						
Course Type	Restricted Elective Course						
Level	Undergraduate						
Year / Semester	5 <sup>th</sup> or 6 <sup>th</sup> or 7 <sup>th</sup> or 8 <sup>th</sup> semester						
Teacher's Name	H. Volos						
ECTS	7,5	Lectures / week	2 x 1.5 hours	Recitation / week	1 x 1 hour	Laboratories / week	1 x 1.5 hours
Course Purpose and Objectives	The course aims to study the key principles and concepts that underlie a modern data center. The course will conduct a vertical study of datacenter technology covering the entire system stack, including hardware architectures, systems software, and application programming frameworks. The course will also explore cross-cutting concerns such as total cost of ownership, service level objectives, reliability and availability, energy efficiency, and privacy and security.						
Learning Outcomes	<p>At the end of the course the students are expected to be able:</p> <ul style="list-style-type: none"> <li>• Understand the basic ideas and design principles in data center systems.</li> <li>• Understand performance, scalability, availability, efficiency, power, and cost tradeoffs in data center systems.</li> <li>• Know about data center infrastructure technologies including virtualization, storage, and networking.</li> <li>• Know how to develop, deploy, and run data-center based applications to solve real problems.</li> </ul>						
Prerequisites	CS 222		Required		None		
Course Content	Horizontal scaling, server design, heterogeneous hardware and accelerators, total-cost of ownership (TCO) analysis, performance analysis, power management, resource management, virtualization, virtual machines and containers, serverless computing, cloud computing, microservices, data-center storage, data-center networking, side channels, trusted execution.						
Teaching Methodology	Lectures (3 hours weekly), Recitation (1 hour weekly) and Laboratory sessions (1,5 hours weekly).						
Bibliography	<ol style="list-style-type: none"> <li>1. Luiz André Barroso, Urs Hölzle, and Parthasarathy Ranganatha. <i>The Datacenter as a Computer: Designing Warehouse-Scale Machines</i>. Third Edition Morgan &amp; Claypool (2019).</li> <li>2. John Hennessy and David Patterson. <i>Computer Architecture: A Quantitative Approach</i>. Sixth Edition Morgan &amp; Claypool (2016)</li> <li>3. Selected articles from the international literature.</li> </ol>						
Assessment	Final exam, midterm exam and homework (theoretical and programming assignments).						
Language	Greek and English						

Course Title	<b>Software Evolution</b>						
Course Code	<b>CS 484</b>						
Course Type	Restricted Elective Course						
Level	Undergraduate						
Year / Semester	5 <sup>th</sup> or 6 <sup>th</sup> or 7 <sup>th</sup> or 8 <sup>th</sup> semester						
Teacher's Name	E. Constantinou						
ECTS	7,5	Lectures / week	2 x 1.5 hours	Recitation / week	1 x 1 hour	Laboratories / week	0 hours
Course Purpose and Objectives	Understanding of important challenges in software evolution. Understanding and implementation of methods and tools for solving software evolution problems. Application of methods and tools in existing software systems and explanation of results.						
Learning Outcomes	<p>The learning outcomes for the students are the following:</p> <ul style="list-style-type: none"> <li>• Understanding important challenges in software evolution.</li> <li>• Understanding and implementation of methods and tools for solving software evolution problems.</li> <li>• Application of methods and tools in existing software systems.</li> <li>• Understanding of the results obtained from existing software systems on software evolution after the systems analysis.</li> </ul>						
Prerequisites	CS 343			Required		None	
Course Content	Introduction to fundamental terms in software evolution and Lehman laws. Evolution of software requirements. Evolution of software architecture and retrieval of software architecture from the system's source code. Source code evolution and code clones. Introduction to software ecosystems, dependencies and evolution in software ecosystems. The human factor in software evolution. Software maintenance and bugs. Software smells and software refactoring. Mining software repositories and empirical software engineering.						
Teaching Methodology	Lectures (3 hours weekly) and Recitation (1 hour weekly).						
Bibliography	<ol style="list-style-type: none"> <li>1. Tom Mens and Serge Demeyer. 2008. Software Evolution (1st. ed.). Springer. (επιλεγμένα κεφάλαια)</li> <li>2. Tom Mens, Alexander Serebrenik, Anthony Cleve. 2014. Evolving Software Systems. Springer. (επιλεγμένα κεφάλαια)</li> <li>3. Tom Mens, Coen De Rover, Anthony Cleve, 2023. Software Ecosystems: Tooling and Analytics, Springer. (επιλεγμένα κεφάλαια).</li> <li>4. Selected articles from the literature.</li> </ol>						
Assessment	Final exam, midterm exam and homework (semester project).						
Language	Greek						

Course Title	<i>Special Issues in Computer Science: Mobile Computing and applications</i>						
Course Code	<b>CS 498</b>						
Course Type	Restricted Elective Course						
Level	Undergraduate						
Year / Semester	5 <sup>th</sup> or 6 <sup>th</sup> or 7 <sup>th</sup> or 8 <sup>th</sup> semester						
Teacher's Name	CS or Visiting Faculty						
ECTS	7,5	Lectures / week	2 x 1.5 hours	Recitation / week	1 x 1 hour	Laboratories / week	1 x 1.5 hours
Course Purpose and Objectives	<p>Upon completion of the course, students should be able to:</p> <ul style="list-style-type: none"> <li>• They understand the fundamentals of mobile computing, including mobile operating systems, wireless networks, and distributed applications.</li> <li>• They develop modern mobile applications that leverage sensors, network interfaces and cloud services, aiming at efficiency and user-friendliness.</li> <li>• They manipulate location and context-awareness technologies, such as GPS, Wi-Fi fingerprinting, BLE, NFC and RFID, to create intelligent and adaptive applications.</li> <li>• They apply mobile sensing and crowdsensing principles to real-world conditions, looking at aspects such as performance, energy consumption and privacy.</li> <li>• They evaluate mobile system architectures, selecting the appropriate tools and methods for different categories of applications.</li> <li>• They are familiar with research in the field of mobile computing, following recent scientific articles and publications in reputable journals and conferences (e.g. MDM, GeoInformatica, Pervasive Computing).</li> </ul>						
Learning Outcomes	<p>Upon successful completion of the course, the student will be able to:</p> <ul style="list-style-type: none"> <li>• It designs and implements mobile applications, using modern development tools (SDKs, APIs, frameworks).</li> <li>• Analyze and evaluate mobile and distributed application architecture for performance, security, and user experience.</li> <li>• It leverages detection and sensor technologies to develop context-aware applications that adapt to the environment and user needs.</li> <li>• It integrates cloud services, edge computing and databases into mobile applications for data storage, synchronization and management.</li> <li>• Understands the challenges and practical solutions related to energy, security, privacy, and accessibility in portable and wearable systems.</li> <li>• It monitors and evaluates current research trends in the field of mobile and ubiquitous computing, developing reading and understanding abilities of scientific articles.</li> </ul>						
Prerequisites	CS 231	Required			None		
Course Content	<p>This course introduces students to the basic principles and modern practices of mobile computing and mobile application development. Through a combination of theoretical teaching and laboratory exercises, students gain an essential understanding of mobile device software, wireless and ad-hoc networks, location services, as well as context-aware and sensory systems.</p> <p>Emphasis is placed on the design and development of modern mobile applications, which interact with cloud services, utilize embedded sensors and rely on technologies such as Bluetooth, Wi-Fi, RFID, NFC and cellular networks.</p> <p>The course also covers advanced and topical topics, such as wearable computing, mobile sensing and crowdsensing.</p> <p>Upon completion of the course, students will be able to develop efficient, scalable and user-friendly mobile applications, having gained experience with modern tools, platforms and technological standards that shape the ecosystem of mobile computing systems.</p>						
Teaching Methodology	Lectures (3 hours weekly), Recitation (1 hour weekly) and Laboratories (1.5 hour weekly).						

Bibliography	<p>The bibliography may be enriched during the semester with additional articles and sources that will be posted on the course website.</p> <p>Recommended Supplementary Books:</p> <ol style="list-style-type: none"> <li>1. John Krumm (Επιμ.), Ubiquitous Computing Fundamentals, 1η Έκδοση, CRC Press, 2009. ISBN: 978-1-4200-9360-5</li> <li>2. John Krumm, Spatial Gems, Volume 1, Επιμέλεια: Andreas Züfle, Cyrus Shahabi, ACM Books, 2022. ISBN: 978-1-4503-9811-4</li> </ol> <p>Scientific Journals &amp; Conference Proceedings:</p> <ol style="list-style-type: none"> <li>1. IEEE International Conference on Mobile Data Management (MDM)</li> <li>2. GeoInformatica (Springer)</li> <li>3. Pervasive and Mobile Computing (Elsevier)</li> </ol>
Assessment	Homework, group project and presentation, midterm and final exam, participation during lectures.
Language	Greek

Course Title	<b><i>Special Issues in Computer Science: Introduction to Natural Language Processing</i></b>						
Course Code	<b>CS 499</b>						
Course Type	Restricted Elective Course						
Level	Undergraduate						
Year / Semester	5 <sup>th</sup> or 6 <sup>th</sup> or 7 <sup>th</sup> or 8 <sup>th</sup> semester						
Teacher's Name	CS or Visiting Faculty						
ECTS	7,5	Lectures / week	2 x 1.5 hours	Recitation / week	0 hours	Laboratories / week	1 x 1.5 hours
Course Purpose and Objectives	Develop a basic understanding of the concepts, techniques and applications of Natural Language Processing (NLP). Familiarity with the fundamental pre-processing steps and basic NLP tasks utilized in machine learning applications. To enhance the student's ability to analyze linguistic data, to select and apply appropriate methods of characterization and classification, and to understand the principles of methods of integrating words and deep learning models. Introduction to Large Language Models (LLMs), such as ChatGPT, and the possibilities they offer for advanced applications such as chatbots and information retrieval systems. Enhancement of analytical thinking through the study of modern applications and tools, preparing the student for applied work or postgraduate deepening in the field of NLP.						
Learning Outcomes	<p>A student successfully completing the course should:</p> <ul style="list-style-type: none"> <li>• Understand the basic tasks of Natural Language Processing and the types of data and challenges involved.</li> <li>• It applies pre-processing, language modeling, and feature extraction techniques to solve NLP problems.</li> <li>• It describes and compares Machine Learning and Deep Learning algorithms for tasks such as sorting, text representation, and information extraction.</li> <li>• It analyzes and evaluates the performance of NLP models using standard metric and experimental procedures.</li> <li>• It understands the use of modern NLP technologies such as Word Embeddings, Contextual Embeddings and Large Language Models (LLMs).</li> <li>• It designs simple applications using ready-made NLP tools, utilizing Python libraries.</li> </ul>						
Prerequisites	None		Required		None		
Course Content	The course introduces students to the basic concepts and applications of Natural Language Processing (NLP). Pre-processing and language modeling techniques are initially covered, followed by a presentation of basic NLP tasks, such as annotation of part of speech, entity recognition, and syntactic analysis. Then, methods of extracting and selecting attributes, machine learning text sorting techniques, and representations of words and documents are examined. The course concludes with an introduction to neural architectures, transformative models (e.g. BERT) and applications of Large Language Models (LLMs). Theoretical concepts are supported by lab exercises in Python using popular NLP tools.						
Teaching Methodology	Lectures (3 hours weekly) and Laboratories (1.5 hour weekly).						
Bibliography	<ol style="list-style-type: none"> <li>1. Jurafsky &amp; Martin, Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition.</li> <li>2. Selected articles from international literature.</li> <li>3. Educational notes and slides.</li> </ol>						
Assessment	Homework, midterm and final exam, participation during lectures.						
Language	Greek						

Course Title	<b><i>Industrial Placement</i></b>						
Course Code	<b><i>CS 500</i></b>						
Course Type	Restricted Elective Course						
Level	Undergraduate						
Year / Semester	3rd year/ Summer Semester						
Teacher's Name	H. Volos / E. Athanasopoulos / P. Kolios / M. Dikaiakos						
ECTS	7,5	Lectures / week	-	Recitation / week	-	Laboratories / week	-
Course Purpose and Objectives	Linking and applying in a real work environment the knowledge that students have acquired in the curriculum courses. Bringing students in direct contact with the new trends and needs of the industrial market as well as the demand for specific skills and competencies. Providing the students with a first experience and opportunity to adapt to a working environment and to recognize the requirements of a professional space as well as the possibility of demystifying some fears about this space: all necessary for the preparation for future entry into the industry market. In exceptional cases, it may be a precursor to future professional cooperation between the student and the employer.						
Learning Outcomes	<ul style="list-style-type: none"> <li>• Strengthening academic knowledge and professional development in real working conditions</li> <li>• Acquiring professional experience</li> </ul>						
Prerequisites	CS 342, CS 343			Required		None	
Course Content	The industrial placement provides students with employment for a short period of time in a real work environment and on a subject related to the curriculum of the Department of Computer Science.						
Teaching Methodology	-						
Bibliography	-						
Assessment	Final report (of total length between 6000 and 10000 words)						
Language	Greek						

Course Title	<b><i>Principles Enterpreunership and Innovation</i></b>						
Course Code	<b><i>BPA 369</i></b>						
Course Type	Compulsory						
Level	Undergraduate						
Year / Semester	4 <sup>th</sup> year/ 7 <sup>th</sup> semester						
Teacher's Name	D. Nikolaou						
ECTS	5	Lectures / week	2 x 1.5 hours	Recitation / week	0 hours	Laboratories / week	0 hours
Course Purpose and Objectives	<p>The aim of the course is to explain the process of innovation and entrepreneurship (inter-dependent concepts) in an understandable and simple way. Additionally, students will be guided through identifying a business opportunity to setting up and growing a business (including how to perform a successful exit).</p> <p>The lectures are based on academic theory, but the emphasis is focused on the practical application of this theory. The series of lectures is designed to familiarize students with theory and practice about entrepreneurship and the management of new businesses while simultaneously emphasizing the role played by new business ideas for the economy.</p> <p>Entrepreneurship is a mindset and a method of creating and developing economic activity by combining:</p> <ol style="list-style-type: none"> <li>(1) risk management,</li> <li>(2) creativity</li> <li>(3) innovation and</li> <li>(4) good management in the context of a new or existing organization.</li> </ol> <p>The main practical goal of the course is to develop a business plan. The primary tool utilized is the business model canvas.</p>						
Learning Outcomes	<p>The student/student is expected to:</p> <ul style="list-style-type: none"> <li>• define important concepts of entrepreneurship and innovation (knowledge)</li> <li>• recognize, explain (understanding) and implement the necessary strategies for the success of a company</li> <li>• make the decisions necessary for a business plan (implementation) by justifying them though collecting and processing information (analysis)</li> <li>• evaluate the level of implementation efficiency (application)</li> </ul>						
Prerequisites	None		Required		None		
Course Content	<p>Specifically, the following sections will be analyzed in depth:</p> <ul style="list-style-type: none"> <li>• Key Partners</li> <li>• Key activities</li> <li>• Key resources</li> <li>• Value creation</li> <li>• Customer Segments</li> <li>• Communication/Distribution channels</li> <li>• Customer Relationships</li> <li>• Cost structure and Expenses</li> <li>• Revenue streams.</li> </ul> <p>Additional topics include: business plans, venture capital firms, angel investors, intellectual property protection, sources and diffusion of innovation.</p>						
Teaching Methodology	Lectures, talks by entrepreneurs, visits to companies, case studies, project preparation						
Bibliography	<ol style="list-style-type: none"> <li>1. Scarborough, Norman M. and Cornwall, Jeffrey R. (2019), Essentials of Entrepreneurship and Small Business Management, Pearson</li> <li>2. Harvard Business School Case Pack</li> </ol>						
Assessment	<p>45% Final exam  30% Group assignment (business plan)  15% Activities (e.g. business model presentation, pitching)  10% Attendance and participation</p>						
Language	Greek						

Course Title	<b>General Advanced English</b>						
Course Code	<b>LAN 100</b>						
Course Type	Compulsory						
Level	Undergraduate						
Year / Semester	1 <sup>st</sup> year / 1 <sup>st</sup> semester						
Teacher's Name	Language Centre faculty						
ECTS	5	Lectures / week	2 x 1.5 hours	Recitation / week	0 hours	Laboratories / week	0 hours
Course Purpose and Objectives	<ul style="list-style-type: none"> <li>• Familiarize students with the conventions of academic writing</li> <li>• Help students differentiate between different writing types (narration/description, example, classification, comparison and contrast)</li> <li>• Enable students to produce written texts progressing from paragraphs to complete essays</li> <li>• Assist students in writing coherently and concisely</li> <li>• Enhance students' vocabulary</li> <li>• Help students foster an appreciation of the importance of dedicated systematic reading in a wide variety of genres</li> <li>• Promote students' critical thinking skills with respect to engaging with written sources, as well as creating authentic texts of their own</li> <li>• Encourage students to actively incorporate correct usage of grammar and mechanics in the English language through targeted practice activities</li> <li>• Enable students to express their ideas clearly both orally and in writing</li> </ul>						
Learning Outcomes	<ul style="list-style-type: none"> <li>• Recognize various writing types and identify their characteristics</li> <li>• Write clearly and coherently</li> <li>• Express personal views/opinions/thoughts with growing competency</li> <li>• Compose complete and developed essays ranging from 1 to 3 pages</li> <li>• Enhance comprehension skills through exposure to ongoing readings</li> <li>• Demonstrate critical thinking skills</li> <li>• Use content-specific academic vocabulary appropriately</li> <li>• Deliver an oral presentation to a group of peers</li> </ul>						
Prerequisites	None		Required		None		
Course Content	<p>This course is designed to guide students in building the required writing, vocabulary and grammar skills to function in an academic setting. Ongoing exposure to reading materials, as well as required vocabulary and grammar activities, will assist students in enhancing their writing skills.</p> <p>Specific writing assignments will guide students in understanding the academic writing process and academic writing conventions. Through close and critical reading of texts, students will analyze ideas, question sources and communicate their thoughts in a clear and effective way. Students will also be required to deliver an informative oral presentation.</p>						
Teaching Methodology	Task-based Learning, Communicative Learning, Collaborative Learning						
Bibliography	<ol style="list-style-type: none"> <li>1. Langan, J., &amp; Allbright, Z. L. (2018). College Writing Skills with Readings, 10th Edition. McGraw Hill, USA.</li> <li>2. Supplementary pack of readings and additional materials prepared by instructor</li> </ol>						
Assessment	3 essays, Final exam, Oral presentation, Participation and continuous assessment						
Language	English						

Course Title	<i>English for Computer Science</i>						
Course Code	<i>LAN 111</i>						
Course Type	Compulsory						
Level	Undergraduate						
Year / Semester	1 <sup>st</sup> year / 2 <sup>nd</sup> semester						
Teacher's Name	Language Centre faculty						
ECTS	5	Lectures / week	2 x 1.5 hours	Recitation / week	0 hours	Laboratories / week	0 hours
Course Purpose and Objectives	<ul style="list-style-type: none"> <li>• To enhance students' ability to communicate in the field of Computer Science</li> <li>• To develop reading, writing, listening and speaking skills</li> <li>• To enable students to effectively comprehend and summarize Computer Science related texts</li> <li>• To enable students to write memos and manual instructions</li> <li>• To broaden Computer Science related vocabulary</li> <li>• To enable students to prepare and present power point presentations on topics related to their major field of study</li> </ul>						
Learning Outcomes	<p>Read a variety of contemporary computer science related articles and identify relevant information and writing styles. Locate main ideas and paraphrase them. Differentiate main ideas from supporting details and express the ideas succinctly in students' own words. Demonstrate increasing competency in reading, writing, speaking and listening skills. Practice and employ relevant computer science vocabulary as well as being able to explain it in layman's terms. Identify specific format and language components employed in writing that is concise. Compose coherent and succinct writing by appraising sources for suitability and relevance. Examine examples of writing styles and formats to differentiate between effective and non-effective writing. Produce effective writing. Arrange and synthesize information in order to plan/ prepare/ produce oral tasks</p>						
Prerequisites	LAN 100 or equivalent		Required		None		
Course Content	This course aims at helping students communicate successfully in the field of computer science by teaching them the use of effective reading, writing, listening and speaking strategies. Students will work on understanding texts relevant to the discipline, writing concisely and coherently as well as improving speaking fluency by delivering oral presentations. Focus will be placed on increasing, understanding and using computer science terminology. Collaboration and interaction between students will be achieved through various group and task-based activities.						
Teaching Methodology	Task-based learning, Communicative, Collaborative						
Bibliography	Course pack						
Assessment	Midterm exam, final exam, oral presentations, assignments						
Language	English						

Course Title	<b>Calculus I</b>						
Course Code	<b>MAS 012</b>						
Course Type	Compulsory						
Level	Undergraduate						
Year / Semester	1 <sup>st</sup> year / 1 <sup>st</sup> semester						
Teacher's Name	G. Georgiou (or other Mathematics and Statistics faculty member)						
ECTS	5	Lectures / week	2 x 1.5 hours	Recitation / week	1 x 1 hour	Laboratories / week	0 hours
Course Purpose and Objectives	Introduction to the basic notions of Calculus and familiarization of the students with the basic theorems and techniques. The course is designed for Computer Science students.						
Learning Outcomes	The students learn how to calculate limits, differentiate and integrate. Emphasis is put on the applications of derivatives and integrals.						
Prerequisites	None		Required		None		
Course Content	Introductory notions. Functions. Limits and Continuity. Derivatives and applications. Integrals. Indefinite and definite integrals. Fundamental theorems of Calculus. Integration techniques. Area, Volume Length of a curve.						
Teaching Methodology	Lectures (3 hours weekly) and Recitation (1 hour weekly).						
Bibliography	<ol style="list-style-type: none"> <li>H. Anton, I. Bivens, and S. Davis, Calculus Late Transcendentals, 10th Edition, Wiley, 2013</li> <li>Γ. Γεωργίου, Χ. Σοφοκλέους, Σημειώσεις Μαθηματικών, Τόμος Α', Κατζηλάρης, 1995</li> </ol>						
Assessment	Final exam, midterm exam, quizzes, and class participation.						
Language	Greek						

Course Title	<b>Calculus II</b>						
Course Code	<b>MAS 013</b>						
Course Type	Compulsory						
Level	Undergraduate						
Year / Semester	1 <sup>st</sup> year / 2 <sup>nd</sup> semester						
Teacher's Name	A. Vidras (or other Mathematics and Statistics faculty member)						
ECTS	5	Lectures / week	2 x 1.5 hours	Recitation / week	1 x 1 hour	Laboratories / week	0 hours
Course Purpose and Objectives	Introduction of the students to basic notions of calculus in one variable, in particular power series, integrals and applications of them in geometry and science.						
Learning Outcomes	<p>The students are expected to</p> <ul style="list-style-type: none"> <li>Acquire a solid knowledge of one variable calculus,</li> <li>Be able to solve problems involving power series and integrals</li> <li>Use the theory developed in order to solve certain geometric and physical problems.</li> </ul>						
Prerequisites	MAS012		Required		None		
Course Content	Power series, Taylor and Maclaurin Series, Analytic functions, Indefinite and definite integrals, Geometric applications, Multivariable functions, Improper integrals.						
Teaching Methodology	Lectures (3 hours weekly) and Recitation (1 hour weekly).						
Bibliography	<ol style="list-style-type: none"> <li>H. Anton, I. Bivens, and S. Davis, Calculus Late Transcendentals, 10th Edition, Wiley, 2013</li> </ol>						
Assessment	Midterm and final exam.						
Language	Greek						

Course Title	<b><i>Linear Algebra</i></b>						
Course Code	<b><i>MAS 029</i></b>						
Course Type	Compulsory						
Level	Undergraduate						
Year / Semester	1 <sup>st</sup> year / 2 <sup>nd</sup> semester						
Teacher's Name	E. Ieronymou (or other Mathematics and Statistics faculty member)						
ECTS	5	Lectures / week	2 x 1.5 hours	Recitation / week	1 x 1 hour	Laboratories / week	0 hours
Course Purpose and Objectives	Introduction to the basic principles of Linear Algebra. Linear spaces, linear independency, base, dimension, inner product spaces. Linear systems of equations, matrices determinants, eigenvalues, eigenvectors. Gram-Schmidt normalisation. Introduction to Analytic Geometry.						
Learning Outcomes	The students get familiar with basic principles of Linear Algebra.						
Prerequisites	None		Required			None	
Course Content	Linear spaces, linear independency, base, dimension, inner product spaces. Linear systems of equations, matrices, determinants, eigenvalues, eigenvectors. Gram-Schmidt normalisation. Introduction to Analytic Geometry.						
Teaching Methodology	Lectures (3 hours weekly) and Recitation (1 hour weekly).						
Bibliography	<ol style="list-style-type: none"> <li>Γ. Γεωργίου, Γραμμική Αλγεβρα, Καντζιηλάρης, Λευκωσία, 1998.</li> <li>Χ. Σοφοκλέους, Στοιχεία Γραμμικής Αλγεβρας, 2016.</li> </ol>						
Assessment	Midterm and final exam.						
Language	Greek						

Course Title	<b><i>Introduction to Probability and Statistics</i></b>						
Course Code	<b><i>MAS 055</i></b>						
Course Type	Compulsory						
Level	Undergraduate						
Year / Semester	2 <sup>nd</sup> year / 3 <sup>rd</sup> semester						
Teacher's Name	X. Miscouridou (or other Mathematics and Statistics faculty member)						
ECTS	7	Lectures / week	2 x 1.5 hours	Recitation / week	1 x 1 hour	Laboratories / week	0 hours
Course Purpose and Objectives	To present to students of computer science basic ideas of probability and statistics which are relevant to computer science						
Learning Outcomes	With the completion of the course, the students should have the necessary background and knowledge in probability and statistics and be able to appreciate the applications in their field of computer science.						
Prerequisites	None		Required			None	
Course Content	Combinatorics, probability, conditional probability, Bayes' Theorem, classical probability problems (distribution of balls in containers, the birthday problem, etc.). Random variables, distributions (discrete and continuous). Independence. Expected value (and other properties), applications of distributions. Probability inequalities (Jensen's inequality, Markov's inequality, Chebyshev's rule). Estimators of unknown parameters. Central Limit Theorem. Introduction to stochastic processes, Markov chains, applications, random walks, Poisson process. Statistical functions, point estimation, confidence intervals, hypothesis testing, applications. Correlation of random variables, simple linear regression and applications. Use of a statistical package or statistical programming language (e.g., R) for the Statistics part. Students are required to familiarize themselves with the basic concepts of the course through a series of practical applications and exercises.						
Teaching Methodology	Lectures (3 hours weekly) and Recitation (1 hour weekly).						
Bibliography	<ol style="list-style-type: none"> <li>R. Hogg and A. Craig: Introduction to Mathematical Statistics, Prentice-Hall, 2012</li> <li>M Mitzenmacher and E. Upfal: Probability and Computing, Cambridge University Press, 2005.</li> </ol>						
Assessment	Final exam, midterm exam and homework (in R).						
Language	Greek						

### ***Courses for other Departments***

These courses are offered for students of other Departments. The content of such courses is appropriately formulated, aiming at the perception, by students of other sciences, of the importance of Informatics, its relationship with other sciences and the possibilities it offers. Each of the Computer Science courses for other Departments carries 5, 6 or 7 ECTS credits. These courses can be offered every semester or offered in "parallel classes", depending on the needs and possibilities of each case.

Course Title	<b><i>Introduction to Computer Science</i></b>						
Course Code	<b><i>CS 001</i></b>						
Course Type	Compulsory (for students of the SPS Department) and Elective						
Level	Undergraduate						
Year / Semester	Fall and Spring						
Teacher's Name	M. Hadjiaros						
ECTS	6	Lectures / week	2 x 1.5 hours	Recitation / week	0 hours	Laboratories / week	1 x 1.5 hours
Course Purpose and Objectives	Introduction to the basic concepts and the wide range of Informatics. Familiarization and comprehensive information of students with the structure and use of computers, computer programs, the Internet and applications of Informatics in other fields.						
Learning outcomes	.						
Prerequisites	None		Required		None		
Course Content	This course is addressed to students who do not belong to the Department of Informatics and aims to promote and explain basic needs of the science of Informatics. It seeks the study of computers and modern trends in the field of Informatics as well as its possible applications in various fields. At the same time, it attempts to give students of other specialties, who will be users of computers and especially of the Internet in their own workplace, the opportunity to appreciate the possibilities offered by Informatics. Also, through the material, students will come into contact with the dangers and malicious use of the Internet and how to protect themselves from them both on a personal and professional level. Through the laboratory exercises, students will become familiar with various software packages, which are considered useful in their academic and professional careers. More specifically, CS001 students will come into contact with internet safety, social networks, introduction to Web 2.0, search engines, bibliography management tools and avoiding sidetracking, using social networks to create a website and promote a small business						
Teaching Methodology	Lectures (3 hours weekly) and Laboratories (1.5 hour weekly).						
Bibliography	-						
Assessment	Final exam, midterm exam and homework (laboratory exercises).						
Language	Greek						

Course Title	<b><i>Introduction to Computer Science</i></b>						
Course Code	<b><i>CS 002</i></b>						
Course Type	Compulsory (for students of the EDU Department, primary and pre-primary)						
Level	Undergraduate						
Year / Semester	Fall and Spring						
Teacher's Name	M. Hadjiaros						
ECTS	5	Lectures / week	2 x 1.5 hours	Recitation / week	0 hours	Laboratories / week	1 x 1.5 hours
Course Purpose and Objectives	Introduction to the basic concepts and the wide range of Informatics. Familiarization and comprehensive information of students with the structure and use of computers, computer programs, the Internet and applications of Informatics in other fields.						
Learning outcomes	.						

Prerequisites	None	Required	None
Course Content	Foundations of Informatics, the main historical events that have contributed to its development and the possibilities of its use. Basic elements that make up Informatics and ways to utilize it in other sciences and applications. The Unix operating system. Practice with application packages, and the Unix environment. Basic principles of programming in a 4th generation language.		
Teaching Methodology	Lectures (3 hours weekly) and Laboratories (1.5 hour weekly).		
Bibliography	1. 1. B. A. Forouzan, Εισαγωγή στην Επιστήμη των Υπολογιστών, Εκδόσεις “ΚΛΕΙΔΑΡΙΘΜΟΣ”, 2003		
Assessment	Final exam, midterm exam and homework (laboratory exercises).		
Language	Greek		

Course Title	<b><i>Computer Science and Information Systems</i></b>						
Course Code	<b><i>CS 003</i></b>						
Course Type							
Level	Undergraduate						
Year / Semester	Fall and Spring						
Teacher's Name	CS or Visiting Faculty						
ECTS	6	Lectures / week	2 x 1.5 hours	Recitation / week	0 hours	Laboratories / week	1 x 1.5 hours
Course Purpose and Objectives	Familiarity with the most basic concepts of Informatics, Information Systems and Computer Systems. Contact with modern trends in the practice of Informatics. Practice the use of various software packages that are useful in academia and professionally.						
Learning outcomes	.						
Prerequisites	None	Required			None		
Course Content	This course is addressed to students who do not belong to the Department of Informatics and aims to promote and explain basic needs of the science of Informatics. It seeks the study of computers and modern trends in the field of Informatics as well as its possible applications in various fields. At the same time, it attempts to give students of other specialties, who will be users of computers and especially of the Internet in their own workplace, the opportunity to appreciate the possibilities offered by Informatics. Also, through the material, students will come into contact with the dangers and malicious use of the Internet and how to protect themselves from them both on a personal and professional level. Through the laboratory exercises, students will become familiar with various software packages, which are considered useful in their academic and professional careers. More specifically, CS003 students will come into contact with internet security, social networks, Web 2.0 tools, search engines, introduction to Cloud Computing, basic principles of programming, introduction to databases, basic principles of web design, use of social networks for website creation and promotion of small business and products (marketing).						
Teaching Methodology	Lectures (3 hours weekly) and Laboratories (1.5 hour weekly).						
Bibliography	-						
Assessment	Final exam, midterm exam and homework.						
Language	Greek						

Course Title	<b><i>Introduction to Programming</i></b>						
Course Code	<b><i>CS 031</i></b>						
Course Type	Compulsory (for students of MAS Department)						
Level	Undergraduate						
Year / Semester	Spring						
Teacher's Name	Chr. Christoforou						
ECTS	7	Lectures / week	2 x 1.5 hours	Recitation / week	1 x 1 hours	Laboratories / week	1 x 1.5 hours
Course Purpose and Objectives	<p>The course teaches the basic principles of programming and their application through the Python programming language. Emphasis is placed on structured programming, abstraction, implementation, control, and debugging of modular programs. The course also covers the foundation of algorithmic thinking, the understanding of basic data structures, and very basic concepts regarding the operation of computers.</p> <p>The general purpose of the course is to acquire the ability to solve simple problems through programming. More specifically, the objectives of the course are:</p> <ul style="list-style-type: none"> <li>• Understanding the basic principles of programming, algorithmic thinking, algorithmic techniques, and program structure</li> <li>• The design, implementation, testing and debugging of programs</li> <li>• Evaluating solutions to a problem</li> <li>• Learning a high-level programming language (Python)</li> </ul>						
Learning Outcomes	<p>The student who successfully completes this course, is expected to be able to:</p> <ul style="list-style-type: none"> <li>• Understand the basic concepts of computer science and describe the basic function of a computer system and its main units (CPU, memory, input/output).</li> <li>• Recognizes the importance of using computers in problem-solving and task optimization.</li> <li>• Understand the importance of data representation (e.g., binary system, data types) and the basic concepts of data structures.</li> <li>• Understand the principles of algorithmic thinking and how they are applied to program design and development.</li> <li>• Explain and implement the steps of analyzing a problem and designing the appropriate solution through programming.</li> <li>• Discuss programming issues in the Python programming language, design and program questions related to their basic degree, such as solving linear algebra exercises etc.</li> <li>• Determine the input and output data, the operations that should be implemented in the CPU during the execution of a project.</li> <li>• Understand the basic flow control, iteration, and function control structures in Python programs.</li> <li>• Implement programs in Python that use variables, selection structures, loops, functions, and simple data structures (such as lists and plurals).</li> <li>• Leverage basic Python libraries and file editing techniques for practical problems.</li> <li>• Discuss about implementing mathematical problems using the Python programming language.</li> <li>• Analyze the Python programming language and recognize the importance of using it.</li> </ul>						
Prerequisites	None		Required			None	

Course Content	<p>This course aims to teach the basic principles of programming and apply them through the Python programming language. The course material covers topics such as, what is an algorithm, a program, and what "proper programming" means. Also, the stages of solving a problem through a computer are studied and the stages of building a program are presented. The Python language is used to implement the solution on the computer. In the context of teaching the Python language, the structure of a standard program is given and the basic commands of the language regarding program flow (branching, iterating), data input/output, and calling functions are presented. It also describes the basic and complex data structures that the language provides. All of the above are accompanied by examples both during the lectures and during the workshops.</p> <p>Summary Table of Contents</p> <ul style="list-style-type: none"> <li>• Data Representation and Operation of a Computer</li> <li>• Introduction to the Python programming language</li> <li>• Variables, Operators and Numerical Expressions</li> <li>• Control Commands and Program Flow Control</li> <li>• Repetition Loops</li> <li>• Functions</li> <li>• Strings</li> <li>• Using Python Libraries</li> <li>• Data Structures</li> <li>• File Editing</li> </ul>
Teaching Methodology	Lectures (3 hours weekly), Recitation (1 hour weekly) and Laboratories (1.5 hour weekly).
Bibliography	<p>Mandatory bibliography:</p> <ol style="list-style-type: none"> <li>1. Cay S. Horstmann, Rance D. Necaise, Python for Everyone, Wiley, 2016</li> </ol> <p>Additional bibliography:</p> <ol style="list-style-type: none"> <li>2. Dimitris Leventas, Python Guide through Examples, TasPython, 2009 (<a href="http://python.org.gr/phocadownload/Tutorials/tutorial_by_example.pdf">http://python.org.gr/phocadownload/Tutorials/tutorial_by_example.pdf</a>)</li> </ol>
Assessment	Final exam, midterm exam, homework (programming assignments) and quizzes.
Language	Greek

Course Title	<b><i>Programming Methods for Problem Solving</i></b>						
Course Code	<b><i>CS 032</i></b>						
Course Type	Compulsory (for students of the BPE, AFN, ECO and CHE departments)						
Level	Undergraduate						
Year / Semester	Fall / Spring						
Teacher's Name	Chr. Kyriakou & N. Temene						
ECTS	6	Lectures / week	2 x 1.5 hours	Recitation / week	0 hours	Laboratories / week	1 x 1.5 hours
Course Purpose and Objectives	Acquisition of the ability to solve simple problems through programming. More specifically, the course teaches the basic principles of programming with emphasis on structured programming, subtraction, implementation, control and debugging of modular programs. The course also covers the foundation of algorithmic thinking, the understanding of basic data structures, and very basic concepts regarding the operation of computers.						
Learning Outcomes	<p>Upon successful completion of the course, students will be able to:</p> <ul style="list-style-type: none"> <li>• They understand basic principles of programming, algorithmic thinking, algorithmic techniques, and program structure.</li> <li>• Design, implement, test and debug modular programs</li> <li>• Evaluate solutions to a problem</li> <li>• Learn and use a high-level programming language (for this purpose learning the Python language has been chosen)</li> </ul>						
Prerequisites	None		Required		None		
Course Content	<ul style="list-style-type: none"> <li>• Data Representation and Operation of a Computer</li> <li>• Introduction to the Python programming language</li> <li>• Variables, operators and numerical expressions</li> <li>• Control commands (if, elif, else) and program flow control</li> <li>• Repeat loops (for, while)</li> <li>• Functions</li> <li>• Data Structures</li> <li>• File Editing</li> </ul>						
Teaching Methodology	Lectures (3 hours weekly) and Laboratories (1.5 hour weekly).						
Bibliography	<ol style="list-style-type: none"> <li>1. Python for Everyone, Cay S. Horstmann, Rance D. Nicaise, Wiley 2013.</li> <li>2. Python Guide Through Examples, Dimitris Leventeas, TasPython, 2009 - <a href="https://www.openbook.gr/odigos-python-mesw-paradeigmatwn/">https://www.openbook.gr/odigos-python-mesw-paradeigmatwn/</a></li> </ol>						
Assessment	Final exam, midterm exam, homework or quizzes.						
Language	Greek						

Course Title	<b>Introduction to Programming for Electrical and Computer Engineers</b>						
Course Code	<b>CS 034</b>						
Course Type	Compulsory (for students of the ECE Department)						
Level	Undergraduate						
Year / Semester	Spring						
Teacher's Name	CS or Visiting Faculty						
ECTS	7	Lectures / week	2 x 1.5 hours	Recitation / week	1 x 1 hours	Laboratories / week	1 x 1.5 hours
Course Purpose and Objectives	Learning problem-solving methods through programming. Acquisition of skills in solving problems in a procedural way and the foundation of algorithmic thinking. Foundation of basic programming principles, algorithmic techniques and program structures. Design, implement, test and debug modular programs. Understanding the important concepts of program abstraction and data abstraction. Application of the basic principles through the C programming language.						
Learning Outcomes	.						
Prerequisites	None		Required		None		
Course Content	Introduction to computers and programming languages. Problem solving and programming, problem specification, algorithms and programs, progressive refinement methodology, program and data abstraction. Software development process, top-down design, problem breakdown, reuse, testing and debugging strategies. Variables: names, values, addresses, basic formulas (numbers, characters, logical values), operators and expressions, constants, use of libraries. I/O functions. Processes (functions), parameters, calls, arguments, pass through value or address. Program flow, naming scope rules, variable/process call lifetime, program status. Procedural programming, algorithmic structures (sequence, selection, repetition, retroactivity), memory. Complex and enumerable data types, tables (one-dimensional and multidimensional), structures and records, indexes (index-type variables, address and indirect reference operators, index arithmetic, tables and indexes, indexes and functions). Introduction to dynamic memory reservation						
Teaching Methodology	Lectures (3 hours weekly), Recitation (1 hour weekly) and Laboratories (1.5 hour weekly).						
Bibliography	<ol style="list-style-type: none"> <li>1. J. R. Hanly, E. B. Koffman, Problem Solving and Program Design in C, 4th Edition, Addison-Wesley, 2003.</li> <li>2. B. W. Kernighan, D. M. Ritchie, Η Γλώσσα Προγραμματισμού C, Δεύτερη Έκδοση, Εκδόσεις ΚΛΕΙΔΑΡΙΘΜΟΣ, 1990.</li> </ol>						
Assessment	Final exam, midterm exam and homework (programming assignments).						
Language	Greek						

Course Title	<b>Data Structures and Algorithms for Electrical and Computer Engineers</b>						
Course Code	<b>CS 035</b>						
Course Type	Compulsory (for students of the ECE Department)						
Level	Undergraduate						
Year / Semester	Fall						
Teacher's Name	Chr. Kyriakou & N. Temene						
ECTS	7	Lectures / week	2 x 1.5 hours	Recitation / week	1 x 1 hours	Laboratories / week	1 x 1.5 hours
Course Purpose and Objectives	Study of the methods of organizing information, the algorithms that create and transform it and the analysis of the complexity of algorithms. Familiarity with data structures and their processing algorithms, appreciation of the importance of careful organization of information for their efficient investigation and processing, development of design and implementation skills of algorithms that minimize their execution time and space and familiarity with techniques for analyzing algorithm efficiency.						
Learning Outcomes	.						
Prerequisites	CS 034			Required		None	
Course Content	Advanced programming principles based on the C programming language: Recursion, Structures, Pointers and efficient memory and file management. In-memory representation of data structures. Data types and abstract data types. Algorithm complexity and middle and worst case analysis. Linear Data Structures: Lists, Stack and Queue using sequential and dynamic memory commitment. Applications of stacks and linked lists. Sorting algorithms SelectionSort, InsertionSort, MergeSort, QuickSort, and BucketSort. Tree Data Structures: Binary Trees, Binary Search Trees, Balanced Trees, B-Trees. Priority Queues and Piles. Graphs: representation, processing algorithms, topological classification and crossing algorithms. Fragmentation techniques, fragmentation functions, and conflict management methods.						
Teaching Methodology	Lectures (3 hours weekly), Recitation (1 hour weekly) and Laboratories (1.5 hour weekly).						
Bibliography	<ol style="list-style-type: none"> <li>1. R. F. Gilberg, B. A. Fourouzan, Data Structures: A Pseudocode Approach with C, Second Edition, Thomson Publishing.</li> <li>2. K. N. King, C Programming: A Modern Approach, Second Edition, W. W. Norton &amp; Company, 2008.</li> <li>3. N. Μισυρλής, Δομές Δεδομένων με C.</li> <li>4. M. A. Weiss, Data Structures and Algorithm Analysis in C, Addison Wesley, 1996.</li> </ol>						
Assessment	Final exam, midterm exam and homework (programming and theoretical assignments).						
Language	Greek						

Course Title	<b><i>Programming Methods for Problem Solving in Social Sciences</i></b>						
Course Code	<b><i>CS 036</i></b>						
Course Type	Compulsory (for students of the PSY Department)						
Level	Undergraduate						
Year / Semester	Spring						
Teacher's Name	CS or Visiting Faculty						
ECTS	5	Lectures / week	2 x 1.5 hours	Recitation / week	0 hours	Laboratories / week	1 x 1.5 hours
Course Purpose and Objectives	Acquisition of the ability to solve simple problems through programming. More specifically, the course teaches the basic principles of programming with emphasis on structured programming, subtraction, implementation, control and debugging of modular programs. The course also covers the foundation of algorithmic thinking, the understanding of basic data structures, and very basic concepts regarding the operation of computers.						
Learning Outcomes	<p>Upon successful completion of the course, students will be able to:</p> <ul style="list-style-type: none"> <li>• They understand basic principles of programming, algorithmic thinking, algorithmic techniques, and program structure.</li> <li>• Design, implement, test and debug modular programs</li> <li>• Evaluate solutions to a problem</li> <li>• Learn and use a high-level programming language (for this purpose learning the Python language has been chosen)</li> </ul>						
Prerequisites	None		Required		None		
Course Content	<ul style="list-style-type: none"> <li>• Data Representation and Operation of a Computer</li> <li>• Introduction to the Python programming language</li> <li>• Variables, operators and numerical expressions</li> <li>• Control commands (if, elif, else) and program flow control</li> <li>• Repeat loops (for, while)</li> <li>• Functions</li> <li>• Data Structures</li> <li>• File Editing</li> </ul>						
Teaching Methodology	Lectures (3 hours weekly) and Laboratories (1.5 hour weekly).						
Bibliography	<ol style="list-style-type: none"> <li>1. Python for Everyone, Cay S. Horstmann, Rance D. Nicaise, Wiley 2013.</li> <li>2. Python Guide Through Examples, Dimitris Leventas, TasPython, 2009 - <a href="https://www.openbook.gr/odigos-python-mesw-paradeigmatwn/">https://www.openbook.gr/odigos-python-mesw-paradeigmatwn/</a></li> </ol>						
Assessment	Final exam, midterm exam, homework (programming assignments) and quizzes.						
Language	Greek						

Course Title	<b><i>Data Manipulation in Python</i></b>						
Course Code	<b><i>CS 037</i></b>						
Course Type	Compulsory (for students of the ECO department)						
Level	Undergraduate						
Year / Semester	Spring						
Teacher's Name	CS or Visiting Faculty						
ECTS	6	Lectures / week	0 hours	Recitation / week	0 hours	Laboratories / week	2 x 2 hours
Course Purpose and Objectives	The purpose of the course is to provide an understanding of topics related to data analysis as well as practical contact with scientific tools and languages such as Python and SQL. Manipulating data is a repetitive task for data analysts. Reading a dataset, checking its properties, manipulating its content, and creating visualizations are often tedious tasks. Therefore, increasing efficiency in this process is beneficial for those who want to handle small or large datasets. Spreadsheet-based software does not have the ability to properly support this process, due to the lack of automation and repeatability. This lab course, which promotes the use of a high-level language such as Python, complemented by using SQL queries to extract data from databases, is ideal for handling datasets. This course is expected to train students to use SQL and Python effectively so that they are able to load and handle datasets efficiently.						
Learning Outcomes	<p>Upon successful completion of the course, students will be able to:</p> <ul style="list-style-type: none"> <li>• They use built-in functions of the Python programming language and perform operations on data structures (such as lists, dictionaries, pleiades).</li> <li>• Understand and use basic SQL queries to select, group, and aggregate data from relational databases.</li> <li>• They load datasets from databases and files through the Pandas library.</li> <li>• They produce basic statistical metrics (mean, standard deviation, median).</li> <li>• They apply data handling techniques such as data cleaning (correcting/deleting/filling in missing data), time series manipulation (indexing, sampling, time shifting, rolling statistics), data transformation (scaling, descaling).</li> <li>• They create graphical representations using datasets using different types of charts (2D, 3D).</li> </ul>						
Prerequisites	CS032 or previous knowledge of the Python programming language			Required		None	
Course Content	This lab course focuses on loading datasets from files (e.g. .csv) and relational database tables using SQL statements such as selection, data manipulation (cleansing, filtering, grouping, aggregating, changing sampling, scaling, descaling) of data in tabular format for exploratory analysis and visualization using important packages such as Numpy, Pandas, Matplotlib, Seaborn, and Plotly. The course does not cover statistics, data mining, machine learning, or predictive modeling. Its aim is to provide students with the means to effectively address common data handling practices in order to increase their effectiveness. These skills are useful for preparing a dataset which can then be used for exploratory analysis (statistical analysis and visualization) and predictive modeling.						
Teaching Methodology	The course offers teaching in a laboratory environment where students will be able to immediately apply the learned knowledge in practice. Students will be able to apply the taught material through laboratory exercises within the classroom with the help of the teacher.						
Bibliography	1. "Python for Data Analysis, 3rd Edition" by Wes McKinney, 2023.						
Assessment	Final exam, midterm exam, homework or quizzes. The final/midterm exams and quizzes will be in the laboratory.						
Language	Greek						

Course Title	<i>eHealth and Medical Informatics</i>						
Course Code	<i>CS 041</i>						
Course Type	Compulsory (for Medical School students)						
Level	Undergraduate						
Year / Semester	Fall						
Teacher's Name	I. Schiza						
ECTS	3	Lectures / week	1 x 3 hours	Recitation / week	0 hours	Laboratories / week	0 hours
Course Purpose and Objectives	Introduction of the doctor of the future to the new world of things of eHealth (Health) and Medical Informatics at local, European and international level. Consolidation of the legislative and social framework of Health. Explanation of the patient-centered approach to medical practice as a prerequisite for successful implementation of Health. Utilization of the possibilities provided by information and communication technologies in medical and clinical practice, mainly through the modeling of medical practice, procedures and knowledge. Management, standardization, presentation and use of medical data for the realization of Health.						
Learning Outcomes	.						
Prerequisites	None		Required		None		
Course Content	Introduction of the terms of health and its operating framework. Legislative, regulatory and social background. Methods for the utilization of information technology for the extraction of medical information and data from knowledge, data and medical information databases. Applications of IT systems used for the circulation of medical knowledge, the management of medical information, the appropriate use of an electronic patient file for patients and the support of a medical decision. Extensive reference to the legal framework that regulates this practice, in accordance with European and international directives.						
Teaching Methodology	Lectures/presentations (3 hours per week) Discussions/presentations, Structured visits to hospital clinics to collect data and explore ways to utilize them by medical staff.						
Bibliography	<p>There is no book that covers all the material that is expected to be taught in the course. There will be extensive use of the Internet even in the classroom. The following books are recommended.</p> <ol style="list-style-type: none"> <li>1. D. Koutsouris, S. Pavlopoulos, A. Prentza, Introduction to Biomedical Technology and Medical Systems Analysis, Tziola Publications, 2003.</li> <li>2. E. H. Shortliffe, J. J. Cimino (Eds.), Biomedical Informatics: Computer Applications in Health Care and Biomedicine, Springer Verlag; 3rd edition, 2006.</li> <li>3. Extensive use of the internet and e-libraries.</li> </ol>						
Assessment	Final exam, midterm exam, homework (studies and/or exercises).						
Language	Greek						

Course Title	<i>eHealth Seminars</i>						
Course Code	<i>CS 042</i>						
Course Type	Compulsory (for Medical School students)						
Level	Undergraduate						
Year / Semester	Spring						
Teacher's Name	I. Schiza						
ECTS	3	Lectures / week	1 x 2 hours	Recitation / week	0 hours	Laboratories / week	0 hours
Course Purpose and Objectives	Familiarization of the student with information and communication technologies (ICT) and their practical application in medical and clinical practice. Gain experience through the citation of practices that have been adopted by renowned doctors who have put eHealth into practice. Monitoring of practices such as applications in intensive care units, laparoscopic procedures, surgeries, robotic surgeries, remote diagnostics, remote monitoring, and live recording and utilization of relevant medical databases. Reflection of the student on what will happen in the field of medicine if ICT will continue to develop.						
Learning Outcomes	.						
Prerequisites	CS 041			Required		None	
Course Content	Selected presentations/lectures by at least six doctors, selected according to the topic and their availability, from Cyprus and abroad. Video conferencing tools may also be used for live presentations and communication with the physician for discussion and resolution of questions. When possible, students will also be given the opportunity to visit a medical unit on site.						
Teaching Methodology	Lectures/presentations/discussions (2 hours every two weeks), Possibility of structured visits to hospital clinics in consultation with the presenters.						
Bibliography	There is no book that covers the material that is expected to be taught in the course. The presenters will propose relevant literature that will be available on the internet.						
Assessment	Written exams and homework (individual and group work).						
Language	Greek						

## Postgraduate Programmes of Study

The Department of Computer Science offers Postgraduate Programmes of Study leading to Master's and Doctoral degrees in Computer Science.

Postgraduate studies in the Department are governed by relevant Postgraduate Study Rules, which have been approved by the Senate of the University (149th Senate Session, 22/5/2002). These rules are set out in the specific Annex B of this guide. Also, Appendices C and D list the Specifications that must be met by the final form of the Master's and Doctoral Theses respectively.

### *Master's Programmes*

The Department of Computer Science offers the following Master's Programme in **Greek**:

- **Master in Computer Science (MCS)**

The Department, in collaboration with the Department of Mathematics and Statistics and the Department of Business Administration and Public Administration, offers the Interdepartmental Master's Degree Programme in **English**:

- **Master of Science in Data Science (DSC)**

The Department, in collaboration with the Department of Electrical and Computer Engineering of the University of Applied Sciences, offers the Interdepartmental Postgraduate Programme in **English**:

- **Master in Artificial Intelligence (MAI)**

The Department, in collaboration with the Open University of Cyprus and the Department of Psychology of the University of Cyprus, offers the Distance Learning, Interuniversity Postgraduate Programme in **English**:

- **Master in Cognitive Systems**

According to the ECTS System, a minimum of 90 ECTS credits are required to obtain a Master's Degree in the Postgraduate Study Programme.

In Master's programmes in which the preparation of a Master's Thesis is required, the thesis should be stored in the Digital Library of the University of Cyprus Lekythos (<https://lekythos.library.ucy.ac.cy/>).

**Table of Specialization Courses of Postgraduate Programmes**

<b>Course Code and Title</b>	<b>Master in Computer Science</b>	<b>Master in Data Science</b>	<b>Master in Artificial Intelligence</b>
CS601 – Distributed Systems	√		
CS602/DSC516 - Cloud Computing	√	√	
CS603 – Using Software Architectures to Design and Implement Software Systems	√		
CS604 – Artificial Intelligence	√		
CS605 – Advanced Computer Architecture	√		
CS606 – Computer Networks and the Internet	√		
CS607 – Visual Computing	√		
CS622 – AI Entrepreneurship	√		
CS646/DSC513 – Advanced Topics in Databases	√	√	
CS649/MAI649 – Principles of Ontological Databases	√		√
CS650/MAI650 – Internet of Things	√		√
CS653 – Computer Games Software Technology	√		
CS656 – Computer Graphics: Modelling and Realism	√		
CS657 – Wireless Networks	√		
CS659 – Design on Embedded Systems	√		
CS660/DSC512 – Information Retrieval and Search Engines	√	√	
CS664 – Systems Analysis and Verification	√		
CS667/MAI647 – Computational Neuroscience	√		√
CS668/MAI644 – Mechanical Vision	√		
CS673 – Algorithmic Game Theory	√		
CS674 – Networks and System Security	√		
CS679 – Electronic Health	√		
CS681 – Advanced Software Reuse and Mining Software Repositories	√		
CS682/DSC517 – Data Security	√	√	
CS699 – Special Topics in Computer Science	√		
CS720 – Independent study	√		
DSC510 – Introduction to Data Science and Analytics		√	
DSC511 – Big Data Analysis		√	
DSC512/CS660 – Information Retrieval and Search Engines	√	√	
DSC513/CS646 – Advanced Topics in Data Management	√	√	
DSC514/MAI623 – Natural Language Processing		√	√
DSC515/MAI642 – Deep Learning		√	√
DSC516/CS602 – Cloud Computing	√	√	
DSC517/CS682 – Data Security	√	√	
DSC551 – Data Visualization		√	
MAI601 – AI Camp			√

Course Code and Title	Master in Computer Science	Master in Data Science	Master in Artificial Intelligence
MAI611 – Fundamentals of Artificial Intelligence			√
MAI612 – Machine Learning			√
MAI613 – Research Methodologies and Professional Practices in AI			√
MAI614 – AI on the Edge Webinars I (2 ECTS) and MAI632 AI on the Edge Webinars II (2 ECTS) Mandatory courses for MAI			√
MAI621 – Artificial Intelligence Ethics I			√
MAI622 – AI Entrepreneurship			√
MAI623/DSC514 – Natural Language Processing		√	√
MAI631 – Artificial Intelligence Ethics II			√
MAI641 – Master Thesis			√
MAI642/DSC 515 – Deep Learning		√	√
MAI643 – Artificial Intelligence in Medicine			√
MAI644/CS668 – Computer Vision	√		√
MAI645 – Machine Learning for Graphics and Computer Vision			√
MAI646 – Cognitive Programming for Human-centric AI			√
MAI647/CS667 – Computational Neuroscience	√		√
MAI648 – Human-centered Intelligent User Interfaces			√
MAI649/CS649 – Principles of Ontological Databases	√		√
MAI650/CS650 – Internet of Things	√		√
MAI652 – Autonomous Mobile Robots			√

The following is a description of the Masters:

### ***1. MASTER in COMPUTER SCIENCE***

The Master's programme in Computer Science is primarily designed for

- Graduates of Computer Science and related Sciences who seek to deepen their knowledge in Computer Science and develop research skills in specialized areas of Computer Science
- IT professionals who wish to expand and update their knowledge and acquire modern know-how in advanced information technologies.

Students who attend this programme can pursue doctoral studies after graduation.

The programme offers two titles:

- Master of Science in Computer Science (for students completing a 30 ECTS dissertation), and
- Master of Science in Computer Science – Professional (for students who meet the requirements of the programme without a thesis)

The duration of studies must be at least three semesters.

Students who choose Option 1 (with a thesis) are required to complete 90 ECTS corresponding to eight courses and the thesis. More specifically:

- Seven postgraduate courses of 8 ECTS (from the list of postgraduate courses with CS code).
- A postgraduate course (4 ECTS) (CS 670 Research Methods and Professional Practices in Computer Science).
- Master's Thesis (30 ECTS).

Students who choose Option 2 (without a thesis) are required to complete 92 ECTS corresponding to twelve courses. More specifically:

- Eleven postgraduate courses of 8 ECTS (from the list of postgraduate courses with CS code).
- A postgraduate course of 4 ECTS (CS 670 Research Methods and Professional Practices in Computer Science).

The programme includes the course Independent Study (CS 720). This course offers students, particularly those who are not completing a dissertation, the opportunity to gain comprehensive knowledge in a specific area of Computer Science under the guidance of a professor in the department.

Students have the opportunity to attend up to two courses from other related postgraduate programmes and be credited with up to 16 ECTS. These include the two programmes, recently created by our department, in Artificial Intelligence and Data Science.

### ***Admission Criteria:***

Candidates should have:

- Bachelor's degree in Computer Science or a related field from a recognized institution, with a general grade equivalent to at least "Very Good" (e.g., 6.5 based on the University of Cyprus grading)
- Relevant professional experience can be an additional qualification.

### **Brief Description of Master in Computer Science Courses**

Each description shows the name of the instructor offering the course in the academic year 2025/2026 or the instructor who suggested the course for courses not offered during the academic year 2025/2026.

Course Title	<b><i>Distributed Systems</i></b>						
Course Code	<b><i>CS 601</i></b>						
Course Type	Elective						
Level	Graduate						
Year / Semester	Fall						
Teacher's Name	Chr. Georgiou						
ECTS	8	Lectures / week	4 hours	Recitation / week	-	Laboratories / week	1.5 hours
Course Purpose and Objectives	Familiarization with fundamental concepts and principles of distributed systems in both breadth and depth. Development of capabilities of designing, analyzing and programming distributed systems and algorithms. Gaining an understanding of the possibilities they offer but also the problems faced in distributed systems.						
Learning Outcomes	<p>Upon successful completion of this class, the student is expected to be able to:</p> <ul style="list-style-type: none"> <li>• Understand the basic concepts and principles of distributed systems.</li> <li>• Demonstrate advanced ability in the design, analysis and programming of distributed systems and algorithms.</li> <li>• Master the different naming, distributed file, security and fault-tolerant systems under the client-server paradigm.</li> <li>• Understand the basic characteristics of Peer-to-Peer Systems as well as of Distributed Ledgers (Blockchains).</li> <li>• Implement and develop distributed applications and algorithms using the ZeroMQ concurrency framework.</li> </ul>						
Prerequisites	None		Required		None		
Course Content	Basic concepts and principles of distributed systems. Communication, processes and synchronization. Naming. Distributed file systems, distributed operating systems and middleware. Security and cryptography in distributed systems. Distributed shared memory and its consistency. Fault-tolerance. Distributed algorithms and distributed programming. Design and development of applications in distributed environments. Case-studies of specific distributed systems (e.g., Blockchains). Practical exposition with programming project and programming exercises.						
Teaching Methodology	Lectures (3 hours weekly) and Recitation (1 hour weekly), Laboratory Lecture (1.5 hours weekly).						
Bibliography	<ol style="list-style-type: none"> <li>1. A. S. Tanenbaum and M. van Steen, <i>Κατανεμημένα Συστήματα: Αρχές and Υποδείγματα</i>, Εκδόσεις ΚΛΕΙΔΑΡΙΘΜΟΣ, 2005.</li> <li>2. G. Coulouris, J. Dollimore, T. Kindberg, and G. Blair, <i>Distributed Systems – Concepts and Design</i>, 5th Edition, Pearson, 2011.</li> <li>3. Relevant published articles.</li> </ol>						
Assessment	Final exam, midterm exam, in-laboratory quizzes and homework (programming assignments).						
Language	Greek						

Course Title	<b>Cloud Computing</b>						
Course Code	<b>CS 602</b>						
Course Type	Elective						
Level	Graduate						
Year / Semester	Fall						
Teacher's Name	M. Dikaiakos						
ECTS	8	Lectures / week	3 hours	Recitation / week	1 hour	Laboratories / week	1.5 hours
Course Purpose and Objectives	<p>This course covers topics and technologies related to Cloud Computing (CC), focusing on state-of-the-art technologies, current research and emerging issues of relevance. Students will engage in the study of basic concepts, recent literature and experimentation with acknowledged technologies. The precepts and laboratories use primarily materials and learning content by Amazon Web Services Academy, which are designed to help students prepare for AWS Certification. Students who attend this course will gain an understanding of the Cloud Computing paradigm and the technical underpinnings of Cloud services. They will be able to describe and analyze key middleware components of Cloud services, to understand the main Cloud application development paradigms, and to use state-of-the-art Cloud service offerings for Data Science-related projects. Precepts and labs will help students prepare for AWS Certification.</p> <p>Students will review and explore, through lectures, discussions, videos, reading and writing assignments, labs, and practice, the following modules: i) Basic Concepts and Models of CC; ii) CC Building Blocks (Data centers and software); iii) Cloud Application Programming Paradigms.</p>						
Learning Outcomes	<p>The students who complete this course successfully will:</p> <ul style="list-style-type: none"> <li>• Master the fundamental concepts, the main enabling technologies and the key programming and application-development paradigms of modern Cloud Computing services.</li> <li>• Be able to design develop, deploy, and monitor highly scalable cloud-based applications by creating and configuring virtual machines, containers, microservices on the cloud.</li> <li>• Be familiar with techniques for big data analysis in Cloud Computing environments.</li> <li>• Compare, contrast, and evaluate the key trade-offs between multiple approaches to cloud system design, and Identify appropriate design choices when solving real-world cloud computing problems.</li> <li>• Write comprehensive case studies analyzing and contrasting different cloud computing solutions.</li> <li>• Make recommendations on cloud computing solutions for an enterprise.</li> </ul> <p>Be prepared to take the AWS Certification exams.</p>						
Prerequisites	Basic knowledge in Programming and Data Structures, Operating Systems, Networking, Parallelism.			Required		None	
Course Content	<p>Part I: Basic Concepts and Models</p> <ul style="list-style-type: none"> <li>• Module 1: Fundamental concepts, terminology, Cloud Computing Evolution.</li> <li>• Module 2: Cloud Computing models.</li> </ul> <p>Part II: Building Blocks</p> <ul style="list-style-type: none"> <li>• Module 3: Data centers and warehouse-scale computers.</li> <li>• Module 4: Virtualization, Containers, and Resource Management.</li> <li>• Module 5: Cloud Storage.</li> </ul> <p>Part III: Cloud Application Programming Paradigms</p> <ul style="list-style-type: none"> <li>• Module 6: Serverless Computing.</li> <li>• Module 7: Microservices and DevOps.</li> </ul>						

Teaching Methodology	Hybrid 1. Lectures with physical presence: 3 hours every week (13 x 3hrs) 2. Online precept: 1 hour every week (13 x 1hr). This contains online lectures, quizzes, and exercises of the course Amazon Cloud Foundations of the Amazon Web Services Academy program. 3. Lab sessions with physical presence in class (4 x 1.5 hrs) and online (9 x 1,5hr). This contains online lectures, hands-on labs, and quizzes of the course Amazon Web Services Introduction to Cloud the Amazon Web Services Academy program.
Bibliography	Readings for this class comprise chapters from books, papers from the scientific literature, and notes from AWS Academy. Required and additional readings will be posted on the course outline web page. In particular, we will use material from the following books: 1. Barroso, L. A., Holzle, U. & P. Raganathan (2018) The data center as a Computer. An Introduction to the Design of Warehouse-Scale Machines. Third Edition. In Synthesis Lectures on Computer Architecture (Vol. 2, Issue 1). Morgan & Claypool Publishers. 2. Marinescu, D. (2017) Cloud Computing: Theory and Practice. Morgan Kaufmann. 3. Foster I. and Gannon Dennis B. (2017) Cloud Computing for Science and Engineering. The MIT Press. 4. Peterson, Baker, Bavier, Williams and Davie (2022) Edge Cloud Operations: A Systems Approach. version v0.2. 5. Jeff Nickoloff and Stephen Kuenzli (2019) Docker In Action. Manning.
Assessment	Student progress is evaluated continuously through class participation and the assessment of in-class presentations, writing assignments, group project deliverables, and final exam.
Language	English

Course Title	<i>Using Software Architectures to Design and Implement Software Systems</i>						
Course Code	<b>CS 603</b>						
Course Type	Elective						
Level	Graduate						
Year / Semester	Fall						
Teacher's Name	G. Papadopoulos						
ECTS	8	Lectures / week	3 hours	Recitation / week	1 hour	Laboratories / week	1.5 hours
Course Purpose and Objectives	The fundamental concepts, principles and modern methods in the use of Software Architectures in the design and implementation of modern software systems will be taught. The role of Software Architectures in Software Engineering will be understood, with an emphasis on Software Reuse issues.						
Learning Outcomes	Understanding of the basic concepts of Software Architectures and related Architecture Description Languages. Design and implementation of modern software systems using a specific Architecture Description Language. Correlating the modeling phase with this implementation through the use of modern middleware platforms and software application development environments (Javabeans, .Net, Eclipse, etc.).						
Prerequisites	Undergraduate course in Software Engineering		Required			None	
Course Content	Basic concepts. Architectural Design. Connectors. Modeling. Visual Representation. Architectural Standards. Analysis and Implementation. Non-functional properties. Security and trust. Standards. The human factor. Specific application areas.						
Teaching Methodology	Lectures (3 hours weekly), Recitation (1 hour weekly) and Laboratory sessions (1.5 hours weekly).						
Bibliography	1. Taylor, R., Medvidovic, N., Dashofy, E., Software Architecture: Foundations, Theory, and Practice, 2010, Wiley 2. Bass, L., Clemens, P., Kazman, R., Software Architecture in Practice, 4th Edition, 2022, Pearson						
Assessment	Midterm and Final exam, lab exercises and group project.						
Language	Greek (or English)						

Course Title	<i>Artificial Intelligence</i>						
Course Code	<b>CS 604</b>						
Course Type	Elective						
Level	Graduate						
Year / Semester	Fall						
Teacher's Name	Y. Dimopoulos, Chr. Christodoulou						
ECTS	8	Lectures / week	3 hours	Recitation / week	1 hour	Laboratories / week	1.5 hours
Course Purpose and Objectives	This course covers specialized topics in Artificial Intelligence, such modeling and solving constraint satisfaction problems, symbolic learning, learning with various forms of neural networks including deep learning, and reinforcement learning.						
Learning Outcomes	Understand different methods of solving constraint satisfaction problems, such as finite domain solving, propositional satisfiability, and answer set programming. Learning in various types of Neural Networks including supervised and deep learning, unsupervised learning, as well as reinforcement learning. Familiarize with software systems for modeling and solving practical problems in reasoning and learning.						
Prerequisites	None		Required		None		
Course Content	Introduction to Artificial Intelligence. Topics in Constraint Satisfaction. Satisfiability and Optimization in Logic. Answer Set Programming. Topics in Machine Learning, Data Mining, and Reasoning under Uncertainty. Introduction to Artificial Neural Networks. Single layer and Multi layer Perceptrons. Backpropagation learning algorithm. Deep Learning and Convolutional Neural Networks. Recurrent Neural Networks. Self-organizing Maps. Radial-basis Function Networks. Reinforcement Learning. Hopfield Neural Networks and Boltzmann Machines as well as Restricted Boltzmann Machines.						
Teaching Methodology	Lectures (3 hours weekly), Recitation (1 hour weekly) and Laboratory sessions (1.5 hours weekly).						
Bibliography	<ol style="list-style-type: none"> <li>1. S. Russell and P. Norvig, <i>Artificial Intelligence: A Modern Approach</i>, Second Edition, Prentice Hall, 2002.</li> <li>2. S. Haykin, <i>Neural Networks and Learning Machines</i>, Third Edition, Pearson Education, 2009.</li> </ol>						
Assessment	Final exam, midterm exam and homework.						
Language	Greek						

Course Title	<i>Advanced Computer Architecture</i>						
Course Code	<b>CS 605</b>						
Course Type	Elective						
Level	Graduate						
Year / Semester	Fall						
Teacher's Name	Y. Sazeides / H. Volos						
ECTS	8	Lectures / week	3 hours	Recitation / week	1 hour	Laboratories / week	1.5 hours
Course Purpose and Objectives	Introduction to advanced principles of organization, operation and performance analysis of modern computer systems						
Learning Outcomes	<ul style="list-style-type: none"> <li>• Familiarization with computer technology trends</li> <li>• Performance measurement and analysis for modern computer systems</li> <li>• Understanding for techniques used to improve the instruction level parallelism of cores found sin modern central processing units</li> <li>• Learn advanced techniques to improve memory hierarchy performance</li> <li>• Microarchitectural Security, transient and non-transient attacks</li> <li>• Familiarization with thread level parallelism, parallel architectures with shared memory, and the problems of memory coherence and memory consistency</li> <li>• Introduction to techniques used for the organization of modern graphic processing units and to data level parallelism</li> <li>• Domain Specific Architecture for Machine Learning</li> <li>• Appreciation of the architecture of large scale computing systems (compute farms)</li> <li>• Familiarization with methods for measuring dependability of computer systems</li> <li>• Read and critique of research papers</li> </ul>						
Prerequisites	Undergraduate course equivalent to the CS221 (Computer Organization and Assembly Programming) and undergraduate course equivalent to the CS222 (Operating Systems).			Required		None	
Course Content	Performance evaluation and comparison, as well as benchmarking programs; Basic microarchitecture concepts of modern processors; Pipelining, instruction-level parallelism, prediction, speculation, and static/dynamic instruction scheduling; memory hierarchy; microarchitectural security; vector processors, simd, gpus, fine-grain-multithreading; multiprocessors, coherency, consistency; tpus, systolic arrays; case studies of modern processors; Current research trends in the area of computer architecture.						
Teaching Methodology	Lectures (3 hours weekly), Recitation (1 hour weekly) and Laboratory sessions (1.5 hours weekly).						
Bibliography	<ol style="list-style-type: none"> <li>1. J. Henessy and D. Patterson, Computer Architecture: A Quantitative Approach, 6th Edition, Morgan Kaufmann, 2020.</li> <li>2. Επιλεγμένα ερευνητικά άρθρα από τη Bibliography.</li> </ol>						
Assessment	Final exam, homework, class participation and paper review/discussion .						
Language	Greek						

Course Title	<b><i>Computer Networks and the Internet</i></b>						
Course Code	<b><i>CS 606</i></b>						
Course Type	Elective						
Level	Graduate						
Year / Semester	Fall						
Teacher's Name	V. Vassiliou						
ECTS	8	Lectures / week	3 hours	Recitation / week	1 hour	Laboratories / week	1.5 hours
Course Purpose and Objectives	Understanding (at a graduate level) of the basic concepts and matters regarding Computer Networks and the Internet. Familiarization with modern views of Computer Networks and exposure to the related open research problems.						
Learning Outcomes	<ul style="list-style-type: none"> <li>• Explain the following core concepts of communication networks/computer networks: networking technologies and various network topologies, network layering, protocol basics, applications and Quality of Service, new techniques in networking and network management.</li> <li>• Explain the following fundamentals in computer networks: protocol suite TCP/IP, Core networking technologies such as routers, switches. Protocols at application layer, design philosophy for reliable services at the transport layer. New technologies at network layer and link layer.</li> <li>• Demonstrate skills in solving networking issues and analysis of communication protocols.</li> <li>• Demonstrate skills in deploying and analyzing various routing and congestion control algorithms.</li> <li>• Arguing, with regard to the infrastructure of a network and evaluates based on quality and other criteria the performance of networks.</li> <li>• Demonstrates ability to solve networking problems and the evaluation of various Internet protocols with regard to performance.</li> <li>• Shows ability to use Internet simulators for understanding networking concept and in the design and evaluation of networks.</li> <li>• Shows ability to real time network monitoring tools and data traffic and protocol analysis, with the aim of assimilation of protocols and data traffic, but also for analysis of possible errors/problems in the functioning of the network.</li> <li>• To seek to continuously evaluate new improved ways and mechanisms for network protocols.</li> <li>• To constantly seek and analyze new techniques and network technologies, like the Internet of Things.</li> </ul>						
Prerequisites	Undergraduate course equivalent to the CS324 (Communications and Networks)			Required		None	
Course Content	Introduction to Internet and Networking Technologies. TCP/IP suite of protocols, Quality of Service (QoS), New Networking Architectures. Protocols and Standards (e.g. DiffServ, IPv6, MPLS). Network Performance Evaluation (e.g. queueing theory, and simulation tools). Traffic Modeling and Traffic Engineering. Congestion Control and Resource Allocation. Network Design and Optimization.						
Teaching Methodology	Lectures (3 hours weekly), Recitation (1 hour weekly) and Laboratory sessions (1.5 hours weekly).						
Bibliography	<ol style="list-style-type: none"> <li>1. L. Peterson and B. Davies, Computer Networks: A Systems Approach, Sixth Edition, Morgan Kaufmann, 2020.</li> <li>2. J. F. Kurose and K. W. Ross, Computers Networking – A Top Down Approach to the Internet, 8th Edition, Addison-Wesley, 2020.</li> </ol>						
Assessment	Final exam and homework (Individual or Group Project and laboratory exercises)						
Language	Greek						

Course Title	<b>Visual Computing</b>						
Course Code	<b>CS 607</b>						
Course Type	Elective						
Level	Graduate						
Year / Semester	Fall						
Teacher's Name	Y. Chrysanthou / A. Aristidou						
ECTS	8	Lectures / week	3 hours	Recitation / week	1 hour	Laboratories / week	1.5 hours
Course Purpose and Objectives	Teaching the basic principles of digital image processing, mechanical vision and computer graphics. These three areas meet in a multitude of recent applications due to the developments in hardware technology and related algorithms. Emphasis on industrial and bio-medical applications as well as on virtual reality applications.						
Learning Outcomes	<p>Upon completion of the course the students will be able to apply the basic 3D graphics algorithms as well as techniques for image analysis and computer vision. To this end, knowledge and skills will be acquired on the following topics:</p> <p>Basic Principles of Computer Graphics:</p> <ul style="list-style-type: none"> <li>• Representing objects in polygonal form</li> <li>• Geometric Transformations</li> <li>• Coordinate systems and projections</li> <li>• Scene graph, camera definition, graphics pipeline</li> <li>• Local and global illuminations</li> <li>• Real-time image generation</li> </ul> <p>Basic principles of image analysis and computer vision:</p> <ul style="list-style-type: none"> <li>• Feature extraction in images</li> <li>• Camera calibration</li> <li>• Perspective projection</li> <li>• Recovery 3D shape information based on multiple viewpoints</li> <li>• Selected topics in Computational Photography</li> </ul>						
Prerequisites	Programming in C, Basic Linear Algebra		Required		None		
Course Content	Binary image processing, intensity transformations, the discrete Fourier transform, linear and nonlinear filtering, image compression, image analysis, basic principles of video processing. Basic principles of 3Dgraphics: polygonal representations, transformations, local and world coordinate system, scene graph, camera and field of view specification, orthographic and perspective projection, clipping in 2D & 3D, polygon rasterization, back face elimination, visible surface determination with the Z-buffer method and Binary Space Partitioning Trees, local illumination - flat, Phong & Gouraud shading, real-time graphics, applications.						
Teaching Methodology	Lectures (3 hours weekly), Recitation (1 hour weekly) and Laboratory sessions (1.5 hours weekly).						
Bibliography	<ol style="list-style-type: none"> <li>1. Watt and F. Policarpo, The Computer Image, Addison–Wesley, 1998.</li> <li>2. R. C. Gonzalez and R. E. Woods, Digital Image Processing, Second Edition, Addison–Wesley, 2002.</li> <li>3. M.Slater, A. Steed and Y. Chrysanthou, Computer Graphics and Virtual Environments: From Realism to Real-Time, Addison-Wesley, 2001.</li> </ol>						
Assessment	Final exam, Midterm Exam, Exercises and Project.						
Language	Greek						

Course Title	<i>AI Entrepreneurship</i>						
Course Code	<i>CS 622</i>						
Course Type	Elective						
Level	Graduate						
Year / Semester	Spring						
Teacher's Name	M. Dikaiakos						
ECTS	8	Lectures / week	3 hours	Recitation / week	1 hour	Laboratories / week	-
Course Purpose and Objectives	<p>Progress in modern economies is shaped by scientific inventions and technological innovations that provide improved products or solutions to traditional needs, disrupt traditional business models or satisfy unanticipated consumer needs opening up new markets. Entrepreneurship is the primary mechanism through which inventions and innovations are brought to global markets.</p> <p>In recent years, the success of global Internet platforms, the rapid advance of digitalization across numerous sectors, and the ensuing abundance of digital data available on Internet have turned Artificial Intelligence (AI) and Machine Learning (ML) into major driving forces in innovative entrepreneurship, leading to unprecedented economic, social and cultural opportunities, challenges, and global impact.</p> <p>This course seeks to help students explore and master key concepts and challenges of relevance to AI and Data-driven entrepreneurship. The course introduces students to the world of AI entrepreneurship through case studies that demonstrate successes, failures and challenges. The course provides also an overview of and an introduction to key steps to develop a company, design a business model, explore product-market fit, manage intellectual property, and attract investment. Students will explore acknowledged innovation-driven entrepreneurship methodologies and experiment with them and associated tools to pursue the translation of their ideas into entrepreneurial endeavors. The course examines issues faced by Startup Founders and Chief Technology Officers who need to innovate at the boundaries of AI, Information Technology and Business by understanding all perspectives.</p>						
Learning Outcomes	<p>After taking this course, students should be able to:</p> <ul style="list-style-type: none"> <li>• Understand and explain the interplay between Big Data, Machine Learning and various application domains.</li> <li>• Evaluate technological ideas and apply the key stages of turning an idea or invention into a commercial product.</li> <li>• Apply the Business Model Canvas methodologies in Information Technology and Scientific application contexts.</li> <li>• Recognize and undertake the steps of the Disciplined Entrepreneurship methodology, and manage the key activities required to bring an innovative product or service to the market: product definition and market segmentation; value proposition analysis and high-level product specification; market and competition analysis; business model definition and revenue models; customer and user acquisition; minimum viable product definition and product implementation planning.</li> <li>• Understand the basics of fundraising and financing options for a startup.</li> <li>• Understand the basics of incorporation and company structure.</li> <li>• Understand the key challenges for attracting talent, establishing and managing a startup team.</li> <li>• Apply tools for project and team management, collaboration, ideation, rapid prototyping: Trello, Slack, SimpleMind, Proto.io, Github, Google AdService, Google Cloud, Heroku, etc.</li> <li>• Prepare pitch decks, and pitch in front of potential investors, an AI-related business idea/product/service.</li> </ul>						
Prerequisites	None		Required		None		

Course Content	<p>The course will comprise weekly live and recorded lectures by the professor and by invited speakers on various aspects of entrepreneurship and innovation. The students will be required to establish teams and work on an idea, producing a business plan and a prototype of an MVP, and several writeups for class readings and invited lectures. Lectures will cover Case Studies in AI Entrepreneurship, Basic concepts in Entrepreneurship and Innovation, and elements of Preparatory Analysis for establishing a startup company, Setting up a company, Value Proposition, Market Analysis and Competition, Business Modeling for AI Products and Services, Customer acquisition and Sales.</p> <p>Module 1: Innovation, Entrepreneurship and AI</p> <ul style="list-style-type: none"> <li>• Invention, Entrepreneurship, Innovation, Research, Start-ups, Ecosystems, Risk, Venture Capital</li> <li>• Intellectual Property Elements</li> <li>• Steps involved to turn an Invention to a Start-up</li> <li>• Explore success stories and failures of AI entrepreneurship; discuss visions for the future</li> </ul> <p>Module 2: Customer &amp; Market Exploration</p> <ul style="list-style-type: none"> <li>• Market segmentation – DE Step 1</li> <li>• Beachhead market selection – DE Step 2</li> <li>• End-user Profile Definition – DE Step 3</li> <li>• Total Addressable Market Size (TAM) of Beachhead – DE Step 4</li> <li>• Profile Persona development for the Beachhead – DE Step 5</li> <li>• Identify your Next 10 Customers – DE Step 9</li> </ul> <p>Module 3: Product &amp; Competition</p> <ul style="list-style-type: none"> <li>• Full Life Cycle Use Case – DE Step 6</li> <li>• High-level Product Specification – DE Step 7</li> <li>• Value Proposition: Definition and Quantification – DE Step 8</li> <li>• Define your Core – DE Step 10</li> <li>• Charting your Competitive Position – DE Step 11</li> </ul> <p>Module 4: Business Modeling</p> <ul style="list-style-type: none"> <li>• Design a Business Model – DE Step 15</li> <li>• Introduction to Platform Economy, Network effects, Platform-based services</li> <li>• Business Model Generation – Business Model Canvas</li> <li>• Set Your Pricing Framework – DE Step 16</li> <li>• Calculate Lifetime Value of Acquired Customer – DE Step 16</li> <li>• Cost of Customer Acquisition (COCA) Analysis – DE Step 18</li> </ul> <p>Module 5: Product Design/Prototyping</p> <ul style="list-style-type: none"> <li>• Design and test key assumptions – DE Steps 20, 21</li> <li>• Minimum Viable Business Product – DE Step 22</li> <li>• Product demonstration and customer-satisfaction assessment – DE Step 23</li> <li>• Lean Product Methodology Overview</li> </ul> <p>Module 6: Customer acquisition/Sales</p> <ul style="list-style-type: none"> <li>• Customer’s Decision-Making Unit Definition – DE Step 12</li> <li>• Map Process to Acquire Paying Customer – DE Step 13</li> <li>• Map the Process to Acquire a Customer – DE Step 18</li> </ul> <p>Module 7: Fundraising</p> <ul style="list-style-type: none"> <li>• Introduction to Start-up Financing and Fundraising</li> <li>• Pitching</li> </ul> <p>Module 8: Scaling Up</p> <ul style="list-style-type: none"> <li>• Calculate TAM Size for Follow-on Markets – DE Step 14</li> <li>• Develop a Product Plan – DE Step 24</li> </ul>
Teaching Methodology	Lectures (3 hours weekly), Recitation (1 hour weekly), Team Project (all semester).

Bibliography	<ol style="list-style-type: none"> <li>1. Bill Aulet, <i>Disciplined Entrepreneurship</i>, Wiley, 2013.</li> <li>2. Bill Aulet, <i>Disciplined Entrepreneurship Workbook</i>, Wiley, 2017.</li> <li>3. Alexander Osterwalder et al, <i>Business Model Generation</i>, Wiley, 2010.</li> <li>4. Ash Fontana, <i>The AI-First Company: How to Compete and Win with Artificial Intelligence</i>, Penguin, 2021.</li> <li>5. Peter Thiel and Blake Masters, <i>Zero to One: Notes on Startups, or How to Build the Future</i>, Virgin Books, 2015.</li> <li>6. Cade Metz (2021). "The Genius Makers: The Mavericks Who Brought A.I. to Google, Facebook, and the World." Random House Business.</li> <li>7. Lee, Kai-Fu (2018). "AI Superpowers: China, Silicon Valley, And The New World Order." Houghton Mifflin Harcourt Company.</li> <li>8. Smith, B. and Browne C.A. (2019). "Tools and Weapons. The Promise and the Peril of the Digital Age." Penguin.</li> <li>9. O'Neil, C. (2016). "Weapons of Math Destruction: How Big Data Increases Inequality and Threatens Democracy." Crown.</li> <li>10. Alexander Osterwalder et al, "Value Proposition Design: How to Create Products and Services Customers Want." Wiley, 2014.</li> <li>11. Ben Horowitz, "The Hard Thing about Hard Things." Harper Business, 2014.</li> <li>12. Steven G. Blank, "The Four Steps to the Epiphany. Successful Strategies for Products that Win." Lulu, 2006.</li> <li>13. Clayton Christensen, "The Innovator's Dilemma: When New Technologies Cause Great Firms to Fail (Management of Innovation and Change)." Harvard Business Review Press, 2016.</li> <li>14. Jeff Bezos, "The Everything Store: Jeff Bezos and the Age of Amazon." Corgi, 2014.</li> <li>15. Geoffrey G. Parker, Marshall W. Van Alstyne and Angeet Paul Choudary, "Platform Revolution." W.W. Norton and Co., 2016.</li> <li>16. European Patent Office. <i>Inventors' Handbook</i>.</li> <li>17. Y Combinator's Resources, <a href="https://www.ycombinator.com/resources/">https://www.ycombinator.com/resources/</a></li> <li>18. Steve Blank, "How to build a startup?" Udacity, <a href="https://classroom.udacity.com/courses/ep245">https://classroom.udacity.com/courses/ep245</a></li> <li>19. Sam Altman, "How to start a startup?" <a href="http://startupclass.samaltman.com/">http://startupclass.samaltman.com/</a></li> </ol>
Assessment	Group project report and presentation, writing assignments.
Language	English

Course Title	<i>Advanced Topics in Databases</i>						
Course Code	<b>CS 646</b>						
Course Type	Elective						
Level	Graduate						
Year / Semester	Spring						
Teacher's Name	D. Zeinalipour						
ECTS	8	Lectures / week	3 hours	Recitation / week	1 hour	Laboratories / week	1.5 hours
Course Purpose and Objectives	The main objectives of this graduate-level course are to provide an in-depth understanding of advanced concepts and research directions in the field of databases. The course is organized in three parts: (i) Fundamentals of Database Systems Implementation; (ii) Distributed, Web and Cloud Databases; (iii) Spatio-temporal Data Management, Sensor Data Management, and (iv) other selected and advanced topics from the recent scientific literature.						
Learning Outcomes	<p>Upon successful completion of the course, the student will be able to:</p> <ul style="list-style-type: none"> <li>• Understand physical database design along with implementation issues and also devise appropriate ways to store and index data.</li> <li>• Demonstrate understanding of issues surrounding query optimization, concurrency control, parallelism and recovery in data management.</li> <li>• Develop the ability to express queries in different forms.</li> <li>• Understand contemporary issues and emerging technologies such as Distributed Databases, Big Data, NoSQL, Graph Databases, On-Line Analytical Processing and Data Warehouses.</li> <li>• Explain methods suitable for particular types of data such as temporal, multimedia or spatial data.</li> <li>• Critically read, analyze and understand the latest research developments and results in the field of data management.</li> </ul>						
Prerequisites	Undergraduate course equivalent to the CS342 (Database Systems)		Required		None		
Course Content	(i) Fundamentals of modern Database Management Systems (DBMSs): storage, indexing, query optimization, transaction processing, concurrency and recovery. (ii) Fundamentals of Distributed DBMSs, Web Databases and Cloud Databases (NoSQL / NewSQL): Semi-structured data management (XML/JSON, XPath and XQuery), Document data-stores (i.e., CouchDB, MongoDB), Key-Value data-stores (e.g., BerkeleyDB, MemCached), Introduction to Cloud Computing (GFS, NFS, Hadoop HDFS, Replication/Consistency Principles), "Big-data" analytics και επεξεργασία δεδομένων (MapReduce, Apache's Hadoop, PIG), Column-stores (e.g., Google's BigTable, Apache's HBase, Apache's Cassandra, DuckDB), Graph databases (e.g., Neo4J και Kuzu) and Overview of NewSQL (Google's F1 και VoltDB). (iii) Spatio-temporal data management (trajectories, privacy, analytics) and index structures (e.g., R-Trees, Grid Files) as well as other selected and advanced topics, including: Embedded Databases (sqlite), Data Management in Stream Management, Sensor Management and Mobile Management Systems with Temporal DBs (Postgres/TimescaleDB, InfluxDB) and Spatial DBs (PostGIS), Energy and Environmentally-aware data management.						
Teaching Methodology	Lectures (3 hours weekly), Recitation (1 hour weekly) and Laboratory sessions (1.5 hours weekly).						
Bibliography	<ol style="list-style-type: none"> <li>1. R. Elmasri, S. Navathe, Fundamentals of Database Systems, 7th Edition, Pearson, 2016.</li> <li>2. R. Ramakrishnan and J. Gehrke, Database Management Systems, 3rd Edition, McGraw-Hill, 2003.</li> <li>3. S. Abiteboul, I. Manolescu, P. Rigaux, M.-C. Rousset, P. Senellart, Web Data Management, Cambridge University Press, 2011.</li> <li>4. Petrov, Database Internals: A Deep Dive into How Distributed Data Systems Work, O'Reilly Media; 1st Edition, 2019.</li> <li>5. Seven Databases in Seven Weeks: A Guide to Modern Databases and the NoSQL Movement 2nd Edition, Pragmatic Bookshelf; 2nd edition, 2018.</li> <li>6. T. Özsu, P. Valduriez, Principles of Distributed Database Systems, 3rd Edition, Springer Press, 2011.</li> <li>7. Selected research articles from the international bibliography.</li> </ol>						

Assessment	Midterm, final exam and homework (assignments and presentation)
Language	Greek

Course Title	<i>Principles of Ontological Databases</i>						
Course Code	<b>CS 649</b>						
Course Type	Elective						
Level	Graduate						
Year / Semester	Spring						
Teacher's Name	A. Pieris						
ECTS	8	Lectures / week	3 hours	Recitation / week	1 hour	Laboratories / week	0 hours
Course Purpose and Objectives	<p>Nowadays we need to deal with data that is very large, heterogeneous, distributed in different sources, and incomplete. At the same time, we have very large amounts of knowledge about the application domain of the data in the form of ontologies that can be used to provide end users with flexible and integrated access to data. This gave rise to ontological databases, which lie at the intersection of traditional databases, and knowledge representation and reasoning. The purpose of the course is to introduce students to the principles of ontological databases and demonstrate the importance of studying data-intensive problems in a mathematically rigorous way, as well as the implications of such studies for real-life applications.</p>						
Learning Outcomes	<p>Upon completion of this course, the students will be able to:</p> <ul style="list-style-type: none"> <li>• Abstract relational data and relational queries from their physical implementation and formalize them in a rigorous way.</li> <li>• Analyze the complexity of querying relational data and isolate the source of complexity.</li> <li>• Explain the semantics of Datalog queries, analyze the complexity of evaluating Datalog queries, and model queries in a declarative way.</li> <li>• Abstract rule-based ontologies from their physical implementation and formalize them in a rigorous way.</li> <li>• Explain and use the main (forward- and backward-chaining) techniques underlying ontological query answering.</li> <li>• Analyze the complexity of ontological query answering and isolate the source of complexity.</li> </ul>						
Prerequisites	None		Required		<p>While there are no formal prerequisites, it is recommended that students have passed an introductory course in Databases (some familiarity with the relational model, and the main relational query languages). It is also recommended that students have some basic familiarity with computational logic (first-order logic), and complexity theory (standard complexity classes such as PTIME and NP).</p>		

Course Content	<p>The main purpose of the course is to introduce students to the principles of ontological databases. To this end, it is vital to first cover the principles of relational databases, without taking ontologies into account, on top of which the principles of ontological databases are built. In particular, the course will cover the following topics:</p> <ul style="list-style-type: none"> <li>• Relational Model: data model, relational algebra, relational calculus (first-order queries), first-order query evaluation, static analysis of first-order queries (satisfiability, containment, and equivalence).</li> <li>• Conjunctive Queries (CQs): syntax and semantics, CQ evaluation, static analysis of CQs (satisfiability, containment, and equivalence), minimization of CQs, acyclicity of CQs, evaluation of acyclic CQs (Yannakaki's algorithm), semantically acyclic CQs and their evaluation.</li> <li>• Adding Recursion – Datalog: inexpressibility of recursive queries, syntax and semantics of Datalog, Datalog query evaluation, static analysis of Datalog queries (satisfiability, containment, equivalence, and boundedness).</li> <li>• Ontological Databases: rule-based ontologies (syntax and semantics), combining relational databases with rule-based ontologies, ontological query answering (OQA), universal models, ontology-based data access.</li> <li>• Ontological Query Answering: forward-chaining (the chase procedure), backward-chaining (resolution-based query rewriting), linear rule-based ontologies (tractable data complexity, intractable combined complexity, fixed-parameter tractability).</li> <li>• Advanced Topics (time permitting): expressive rule-based ontology languages, chase termination (necessary and sufficient conditions), static analysis of ontological queries (containment and boundedness).</li> </ul>
Teaching Methodology	Lectures, discuss solutions to non-trivial problems given in advance (during the weekly recitation hour), review of recent research papers.
Bibliography	<ol style="list-style-type: none"> <li>1. S. Abiteboul, R. Hull, V. Vianu, Foundations of Databases, 1995</li> <li>2. M. Arenas, P. Barcelo, L. Libkin, W. Martens, A. Pieris, Database Theory, υπό συγγραφή – προκαταρκτική έκδοση διαθέσιμη στο σύνδεσμο <a href="https://github.com/pdm-book/community">https://github.com/pdm-book/community</a></li> <li>3. F. Baader, I. Horrocks, C. Lutz, U. Sattler, An Introduction to Description Logic, 2017</li> <li>4. L. Libkin, Elements of Finite Model Theory, 2012</li> </ol>
Assessment	<p>For a technical course of this type, which focuses on the mathematical side of (ontological) databases, exams do not allow us to properly evaluate the students' knowledge of the material. For a proper evaluation, students must be presented with non-trivial problems and tasks, rather than “toy” ones that can be solved in a limited time. Therefore, the assessment of the course consists of the following three components:</p> <ul style="list-style-type: none"> <li>• Engagement component: During the course, the students will be given 10 exercises that will cover the various topics described above. A serious attempt to solve an exercise will be awarded all the marks, no matter if the provided solution is correct. The solutions of the exercises will be discussed during the recitation hours.</li> <li>• Essay and in-class presentation on the principles of databases (without ontologies): Students will choose a research paper from a given list, and present (i) a summary of the paper and (ii) analysis and critical thoughts (criticism of the paper, discussion on follow-up papers that show how the ideas of the paper under review have influenced the field, ideas for future research directions). There will be also an in-class presentation based on the essay.</li> <li>• Essay and in-class presentation on the principles of ontological databases: As above.</li> </ul>
Language	Greek

Course Title	<b><i>Internet of Things</i></b>						
Course Code	<b><i>CS 650</i></b>						
Course Type	Elective						
Level	Graduate						
Year / Semester	Spring						
Teacher's Name	V. Vassiliou						
ECTS	8	Lectures / week	3 hours	Recitation / week	1 hour	Laboratories / week	1.5 hours
Course Purpose and Objectives	The main objective of the course will be to provide an overview of the building blocks of IoT such as sensors and smart devices, M2M communication, data collection and processing, and the role of people and applications.						
Learning Outcomes	<p>Upon completion of this course, students should be able to:</p> <ul style="list-style-type: none"> <li>• Explain the definition and usage of the term “Internet of Things” in different contexts</li> <li>• Understand and describe the key components that make up an IoT system</li> <li>• Apply the knowledge and skills acquired during the course to build and test a complete, working IoT system involving prototyping, programming and data analysis</li> <li>• Independently research the technological trends which have led to IoT</li> <li>• Understand where the IoT concept fits within the broader ICT industry and recognize possible future trends</li> <li>• Value the impact of IoT on society by analysing IoT systems with regard to sustainability, safety, integrity and ethics.</li> <li>• Appreciate the role of big data, cloud computing and data analytics in a typical IoT system</li> </ul>						
Prerequisites	CS 606		Required		None		
Course Content	General Principles and Architecture of IoT Systems, Devices, Detection and Response, Communication Technologies, IoT Communication Protocols, IoT Architecture, IoT Applications						
Teaching Methodology	Lectures (3 hours weekly), Recitation (1 hour weekly) and Laboratory sessions (1.5 hours weekly).						
Bibliography	<ol style="list-style-type: none"> <li>1. Internet of Things: A Hands-On Approach, by Arshdeep Bahga and Vijay Madisetti, 2016</li> <li>2. Internet of Things (IoT): Architectures, Protocols and Standards, by Simone Cirani, Gianluigi Ferrari, Marco Picone, and Luca Veltri, Wiley 2018</li> <li>3. IoT and Edge Computing for Architects – Second Edition" by Perry Lea, O'Reilly, 2020</li> </ol>						
Assessment	Final exam, midterm exam and homework.						
Language	English						

Course Title	<b>Computer Games Software Technology</b>						
Course Code	<b>CS 653</b>						
Course Type	Elective						
Level	Graduate						
Year / Semester	Spring						
Teacher's Name	G. Chrysanthou						
ECTS	8	Lectures / week	3 hours	Recitation / week	1 hour	Laboratories / week	1.5 hours
Course Purpose and Objectives	Providing the necessary knowledge to design and implement an electronic game. Teaching the design of the structure of an electronic game and its parts, the virtual simulation of physical models, the use of various techniques for realistic animation and deformation of the shape of objects and characters, the application of artificial intelligence principles to the behavioral design of autonomous characters as well as methods to optimize the intended software in order to efficiently perform real-time calculations. In the lab, learning to use the above techniques together with a game engine (e.g. Unity) to implement the components of a game and the composition of the final software.						
Learning Outcomes	<p>The student who has successfully completed this course is expected to be able to:</p> <ul style="list-style-type: none"> <li>• Design an Electronic Game</li> <li>• Understand the game engine software architecture</li> <li>• Know models for interactive cameras</li> <li>• Implement algorithms for collision detection</li> <li>• Implement route and route planning techniques</li> <li>• Know methods for animation</li> <li>• Understand what are autonomous characters with 'intelligence'</li> <li>• Perform virtual simulations of physical models</li> <li>• Visualize a virtual world using a game engine</li> <li>• Know the basic principles of game networking</li> </ul>						
Prerequisites	Students must have basic knowledge in C or C ++ programming, and mathematics in general. Basic knowledge of computer graphics will be helpful.			Required		None	
Course Content	Game structure and design, computer animation, movement and deformation, interactive cameras, visual simulation of physically-based models, special effects using particle systems, collision detection, articulated characters, navigation and other behavioural models for autonomous characters						
Teaching Methodology	Lectures (3 hours weekly), Recitation (1 hour weekly) and Laboratory sessions (1.5 hours weekly).						
Bibliography	<ol style="list-style-type: none"> <li>1. R. Parent, Computer Animation: Algorithms and Techniques, Morgan Kaufmann, 2002.</li> <li>2. Watt and M. Watt, Advanced Animation and Rendering Techniques, Addison-Wesley, 1992.</li> <li>3. I. Millington, Artificial Intelligence for Games, Morgan Kaufmann, 2006.</li> </ol>						
Assessment	Final exam, midterm exam and homework.						
Language	Greek						

Course Title	<b><i>Computer Graphics: Modeling and Realism</i></b>						
Course Code	<b><i>CS 656</i></b>						
Course Type	Elective						
Level	Graduate						
Year / Semester	Spring						
Teacher's Name	G. Chrysanthou						
ECTS	8	Lectures / week	3 hours	Recitation / week	1 hour	Laboratories / week	1.5 hours
Course Purpose and Objectives	This course goes beyond the basics of digital image synthesis, looking at issues such as photo-realistic rendering, modeling and animation. A big component for this is the creation of realistic and detailed models as well as the faithful simulation of light transport. We will see how these can be applied to virtual and augmented reality. Students will acquire both the theoretical foundations as well as practical skills since a significant part of the course is the student project.						
Learning Outcomes	By the end of the class, students will have an in-depth understanding of the process of creating realistic images in computer graphics. The students will understand concepts of general lighting algorithms, modelling and animation.						
Prerequisites	Programming in C, Linear Algebra, Introductory course in Computer Graphics			Required		None	
Course Content	Modeling, parametric and implicit surfaces, camera specification, projections of primitives. Graphics Pipeline. Local and global illumination, shadows, ray tracing and radiosity. Real-time rendering of large environments. Acceleration techniques.						
Teaching Methodology	Lectures (3 hours weekly), Recitation (1 hour weekly) and Laboratory sessions (1.5 hours weekly).						
Bibliography	<ol style="list-style-type: none"> <li>1. M. Slater, A. Steed and Y. Chrysanthou, <i>Computer Graphics and Virtual Environments: From Realism to Real-Time</i>, Addison-Wesley, 2001.</li> <li>2. A. Watt, <i>3D Computer Graphics</i>, Third Edition, Addison-Wesley, 2001.</li> </ol>						
Assessment	Final exam, midterm exam and homework (group project and exercises).						
Language	Greek						

Course Title	<b>Wireless Networks</b>						
Course Code	<b>CS 657</b>						
Course Type	Elective						
Level	Graduate						
Year / Semester	Spring						
Teacher's Name	V. Vassiliou / P. Kolios						
ECTS	8	Lectures / week	3 hours	Recitation / week	1 hour	Laboratories / week	1.5 hours
Course Purpose and Objectives	Introduction to wireless networks (mobile/local/cellular/Ad-hoc/Sensor) with an emphasis on the fundamental concepts and principles of the technologies which are important for the design, application, evaluation and development of these systems. The course will also cover new architectures and topologies, existing and proposed standards, as well as open research issues.						
Learning Outcomes	<p>Students successfully completing this course should be able to:</p> <ul style="list-style-type: none"> <li>• Explain the following fundamental concepts of wireless and mobile networks: Wireless environment. Interference in a wireless environment. Basic principles of wireless data communication, including resource allocation. Architectures and technologies of wireless networks and wireless communication. Infrastructure and Infrastructureless Wireless Topologies</li> <li>• Explain the following basic issues in wireless and mobile networks: Networking technologies classifications, such as wide area networks, metropolitan area networks, local area networks, body area networks. Infrastructure and Infrastructureless Networks, such as WLANs, Sensor Networks and the Internet of Things. Mobile networks, including 5G and beyond. Security issues, technologies, and techniques for wireless networks.</li> <li>• Demonstrate skills in deploying and analyzing various technologies of mobile/wireless networks.</li> <li>• Demonstrates ability to solve networking problems including analysis of protocols, and sizing and design issues in wireless networks.</li> <li>• Demonstrates ability in evaluating different technologies, techniques and protocols, especially through personalized project studies.</li> <li>• Shows ability to use network simulators (OPNET) for the design and evaluation of networks.</li> <li>• Seek continuously new improved ways and mechanisms in wireless protocols/mobile networks.</li> <li>• Seek and analyze new techniques and technologies, networks such as the Internet of Things, Nanonet</li> </ul>						
Prerequisites	Undergraduate course equivalent to the CS 324 (Communications and Networks)		Required		None		
Course Content	Wireless environment, Interference and other problems in wireless communications, basic principles of wireless local and metropolitan area networks, and cellular wireless networks. New architectures and technologies of wireless networks and wireless communication (e.g., ad-hoc and sensor networks, VANETS). Resource management techniques, Next Generation wireless networks, design and planning of wireless networks, protocols for wireless and mobile networks. Internet/Web of Things.						
Teaching Methodology	Lectures (3 hours weekly), Recitation (1 hour weekly) and Laboratory sessions (1.5 hours weekly).						
Bibliography	<ol style="list-style-type: none"> <li>1. H Karl and A. Willing, Protocols and Architectures for Wireless Sensor Networks, Wiley, 2007.</li> <li>2. J. Schiller, Mobile Communications, Second Edition, Addison-Wesley, 2003.</li> <li>3. K. Sohraby, D. Minoli and Taieb, Wireless Sensor Networks: Technology, Protocols, and Applications, 2006.</li> </ol>						
Assessment	Final exam and homework (including Individual or Group Project and laboratory exercises).						
Language	Greek						

Course Title	<b><i>Design on Embedded Systems</i></b>						
Course Code	<b><i>CS 659</i></b>						
Course Type	Elective						
Level	Graduate						
Year / Semester	Spring						
Teacher's Name	P. Kolios						
ECTS	8	Lectures / week	3 hours	Recitation / week	1 hour	Laboratories / week	1.5 hours
Course Purpose and Objectives	To offer advanced knowledge for system design using embedded computers.						
Learning Outcomes	Familiarity with the basic concepts and methods of developing embedded systems. Introduction to mobile computing and programming on smart mobile devices. Application development using micro devices based on ARM microcontrollers, Arduino, and Rasperry Pi.						
Prerequisites	Knowledge on the subjects of Digital Systems, Computer Organization and Assembly Programming		Required			None	
Course Content	A review of embedded system processors. Organization of embedded systems: CPUs, RAM, ROM, buses, peripherals, sensors, actuators, interfacing. Examples of widely used processors buses and peripherals. Interfacing with peripherals: sampling, interrupts, advantages and disadvantages. Process distribution between hardware and software. Tools for the development of embedded systems and real-time operating systems. Hands-on experience with the development and implementation of embedded systems.						
Teaching Methodology	Lectures (3 hours weekly), Recitation (1 hour weekly) and Laboratory sessions (1.5 hours weekly).						
Bibliography	<ol style="list-style-type: none"> <li>1. F. Vahid and T. Givargis, Embedded System Design: A Unified Hardware/Software Introduction, John Wiley &amp; Sons, 2002.</li> <li>2. W. Wolf, High-Performance Embedded Computing: Architectures, Applications and Methodologies, Morgan Kaufman.</li> <li>3. W. Wolf, Computers as Components: Principles of Embedded Computing System Design, Morgan Kaufman.</li> <li>4. P. Raghavan, A. Lad and S. Neelakandan, Embedded Linux System Design and Development, Auerbach Publications</li> </ol>						
Assessment	Final exam, midterm exam and homework.						
Language	Greek						

Course Title	<b>Information Retrieval and Search Engines</b>						
Course Code	<b>CS 660</b>						
Course Type	Elective						
Level	Graduate						
Year / Semester	Spring						
Teacher's Name	G. Pallis						
ECTS	8	Lectures / week	3 hours	Recitation / week	1 hour	Laboratories / week	1.5 hours
Course Purpose and Objectives	The objective of this course is to examine the main computer science principles that lie behind Google and other search engines. To this end, the course will focus on basic and advanced techniques for text-based information systems: efficient text indexing; Boolean and vector space retrieval models; evaluation and interface issues; text classification and clustering. The course will also focus on Web search including crawling, link-based algorithms, and Web metadata.						
Learning Outcomes	<ul style="list-style-type: none"> <li>• Apply information retrieval principles to locate relevant information in large collections of data</li> <li>• Understand and deploy efficient techniques for the indexing of document objects that are to be retrieved</li> <li>• Implement features of retrieval systems for web-based and other search tasks</li> <li>• Analyse the performance of retrieval systems using test collections</li> <li>• Make practical recommendations about deploying information retrieval systems in different search domains</li> <li>• Advanced topics such as natural language processing (NLP) in search, personalized search, and web search engines</li> <li>• LLM-based approaches to enhance search relevance and user experience</li> </ul>						
Prerequisites	Algorithms, Data Structures, Internet Technologies and Linear Algebra		Required			None	
Course Content	Introduction to Information Retrieval. Boolean Retrieval. Text encoding: tokenisation, stemming, lemmatisation, stop words, phrases. Dictionaries and Tolerant retrieval. Index Construction and Compression. Scoring and Term Weighting. Vector Space Retrieval. Evaluation in information retrieval. Relevance feedback/query expansion. Text classification and Naive Bayes. Vector Space Classification. Flat and Hierarchical Clustering. Web Search Basics. Web crawling and indexes. Link Analysis. NLP in search, personalized search, and web search engines. LLM-based approaches						
Teaching Methodology	Lectures (3 hours weekly), Recitation (1 hour weekly) and Laboratory sessions (1.5 hours weekly).						
Bibliography	<ol style="list-style-type: none"> <li>1. Christopher D. Manning, P. Raghavan and H. Schutze, An Introduction to Information Retrieval, Cambridge University Press, 2008.</li> <li>2. Research papers and Tutorials from ACM SIGIR Conference on Research and Development in Information Retrieval</li> </ol>						
Assessment	Final exam, midterm exam and homework.						
Language	Greek						

Course Title	<b><i>System Analysis and Verification</i></b>						
Course Code	<b><i>CS 664</i></b>						
Course Type	Elective						
Level	Graduate						
Year / Semester	Spring						
Teacher's Name	A. Philippou						
ECTS	8	Lectures / week	3 hours	Recitation / week	1 hour	Laboratories / week	1.5 hours
Course Purpose and Objectives	The course aims to develop a deep understanding of contemporary technologies and methodologies for the modeling, analysis, and verification of computer systems, enabling students to evaluate and apply appropriate techniques to ensure system reliability and correctness.						
Learning Outcomes	<p>Upon successful completion of this class, the student is expected to be able to:</p> <ul style="list-style-type: none"> <li>• Demonstrate familiarity with the main approaches in formal software verification and determine the appropriate contexts for applying each.</li> <li>• Understand and utilize formal modeling and specification languages for defining system behaviors and requirements.</li> <li>• Write formal requirement specifications using temporal logic to analyze and verify system properties.</li> <li>• Create models in the Promela modeling language and use the SPIN model checker to simulate and verify these models.</li> <li>• Construct models using timed automata and employ the UPPAAL model checker to simulate and verify time-sensitive system behaviors.</li> </ul>						
Prerequisites	None		Required		None		
Course Content	Formal techniques for system specification and analysis. Concurrent systems and interleaving and partial-order semantics. Transition systems and Kripke structures. Temporal logic (linear and branching). Automated verification and model-checking. Design-by-contract specification languages. Abstract Specifications. Run-time and static annotation checking. Real-time system analysis: timed automata and timed temporal logic. Application of the techniques via selected tools (e.g. SPIN, NuSMV, UPPAAL, Dafny, Frama-C).						
Teaching Methodology	Lectures (3 hours weekly), Recitation (1 hour weekly) and Laboratory sessions (1.5 hours weekly).						
Bibliography	<ol style="list-style-type: none"> <li>1. M. Huisman and A. Wuijs, Concise Guide to Software Verification - From Model Checking to Annotation Checking, Springer, 2023.</li> <li>2. C. Baier and J.-P. Katoen, Principles of Model Checking. MIT Press, 2008</li> <li>3. D. Peled, Software Reliability Methods, Springer-Verlag, 2001.</li> <li>4. Επιλεγμένα ερευνητικά άρθρα από τη διεθνή Bibliography.</li> </ol>						
Assessment	Final exam, midterm exam and homework.						
Language	Greek						

Course Title	<i>Computational Neuroscience</i>						
Course Code	<i>CS 667</i>						
Course Type	Elective						
Level	Graduate						
Year / Semester	Spring						
Teacher's Name	Chr. Christodoulou						
ECTS	8	Lectures / week	3 hours	Recitation / week	1 hour	Laboratories / week	1.5 hours
Course Purpose and Objectives	Computational Neuroscience is an emerging and dynamically developing field aiming to elucidate the principles of information processing by the nervous system. This course aims to develop and apply computational methods for studying brain and behaviour as well as understanding the dynamics of the conscious mind.						
Learning Outcomes	<p>The learning outcomes for the students are the following:</p> <ul style="list-style-type: none"> <li>• understand and be able to explain the fundamental principles of information processing by neural systems</li> <li>• appreciate the importance of computational neuronal models in the quest of understanding the brain and the fact that many aspects of neuroscience cannot be understood without appropriate computational modeling framework</li> <li>• understand the most important biophysical neuronal models and the different levels of description and complexity in computational neuronal modelling from the level of the single neuron to that of neural networks</li> <li>• understand neuronal dynamics and learn how high dimensional neuronal models can be reduced to low dimensional neural models</li> <li>• understand how experimentally recorded physiological signals enable us to understand the functionality of neurons/systems in the brain and how statistical approaches help in the analysis of such data</li> <li>• be able to implement/simulate basic computational neuronal models through programming</li> <li>• become familiar and be able to use various computational neuroscience simulation software packages for modelling complex biophysical models and experimentally observed phenomena</li> <li>• be able to grasp the importance of high level modelling abstraction from the underlying neuronal principles for understanding brain behaviours</li> <li>• critical reading and discussion of recently published scientific papers</li> </ul>						
Prerequisites	Linear Algebra, Differential Equations		Required		None		
Course Content	Introduction to Computational Neuroscience; basic neurobiology: from the brain to single neurons; biophysics of single neurons; synapses; dendrites and axons. Conductance-based neuron models: the generation of action potentials and the Hodgkin and Huxley equations. Spiking neuron models and response variability: leaky integrator and leaky integrate-and-fire (LIF) type neuron models; spike time variability. Two dimensional (2D) neuron models: reduction of the four dimensional (4D) HH model to a 2D model; phase plane analysis of 2D models/nullclines; FitzHugh-Nagumo model; neuronal dynamics. Modelling synapses/inputs to neurons. Neuron models beyond HH – more ion channels and their functions. Cable Theory: neuronal structure; passive/active membranes; modelling axons and dendrites; action potential propagation. Compartmental models. Neural coding: firing rate; rate code; temporal code; neural operational modes – temporal integration/coincidence detection. Synaptic plasticity: Hebbian learning; Spike-Timing Dependent Plasticity. Bottom-up/top-down modeling of the brain: modeling of self-control behaviour as an example of top-down modelling. Bottom-up/top-down modeling of the brain: modeling of self-control behaviour as an example of top-down modelling. Modelling consciousness.						
Teaching Methodology	Lectures (3 hours weekly), Recitation (1 hour weekly) and Laboratory sessions (1.5 hours weekly).						

Bibliography	<ol style="list-style-type: none"> <li>1. P. Dayan and L. Abbott, Theoretical Neuroscience: Computational and Mathematical Modelling of Neural Systems, MIT Press, 2001.</li> <li>2. D. Sterratt, B. Graham, A. Gilles and D. Willshaw, Principles of Computational Modelling in Neuroscience, Cambridge University Press, 2011.</li> <li>3. W. Gerstner, W. M. Kistler, R. Naud, L. Paninski, Neuronal Dynamics: From single neurons to networks and models of cognition, Cambridge: Cambridge University Press, 2014.</li> <li>4. C. Koch, Biophysics of Computation: Information Processing in Single Neurons, Oxford University Press, 1998.</li> <li>5. E. M. Izhikevich, Dynamical Systems in Neuroscience: the Geometry of Excitability and Bursting, MIT Press, 2007.</li> </ol>
Assessment	Final exam, midterm exam and laboratory exercises/oral presentations of selected research papers.
Language	Greek

Course Title	<b><i>Mechanical Vision</i></b>						
Course Code	<b><i>CS 668</i></b>						
Course Type	Elective						
Level	Graduate						
Year / Semester	Spring						
Teacher's Name	C. Pattichis / G. Chrysanthou						
ECTS	8	Lectures / week	3 hours	Recitation / week	1 hour	Laboratories / week	1.5 hours
Course Purpose and Objectives	The objective of this course is to understand the basic issues in mechanical vision and the major approaches that address them. Through the duration of the course, vision is treated as a process of inference from noisy and uncertain data in order to answer the question of how computers can understand the visual world of humans.						
Learning Outcomes	<p>Upon the course's completion, students should:</p> <ul style="list-style-type: none"> <li>• Understand the principles of vision.</li> <li>• Appreciate the commercial potential of computer vision, but also understand the limitations of current methods.</li> <li>• Identify computer vision's near-future goals.</li> <li>• Apply mathematical and statistical methods as a means to solve vision-related problems.</li> <li>• Be able to extract features of interest from images.</li> <li>• Understand the process of perspective projection, be able to calibrate cameras, and manipulate 3D and 2D transformations.</li> <li>• Be able to recover 3D shape information based on multiple viewpoints.</li> <li>• Understand the principle of inference techniques such as the Kalman filter and the particle filter, and their use in 3D tracking.</li> <li>• Be able to develop simple programs utilizing computer vision algorithms.</li> </ul>						
Prerequisites	MAS 029: Linear Algebra CS 231: Data Structures and Algorithms		Required			None	
Course Content	Βασικές έννοιες and μεθοδολογίες που αφορούν το αντικείμενο της Μηχανικής Όρασης. Σχηματισμός Εικόνας, επεξεργασία εικόνας, ανίχνευση χαρακτηριστικών, κατάτμηση εικόνων and ομαδοποίηση χαρακτηριστικών, επεξεργασία πολλαπλών εικόνων, μελέτη εφαρμογών.						
Teaching Methodology	Lectures (3 hours weekly), Recitation (1 hour weekly) and Laboratory sessions (1.5 hours weekly).						
Bibliography	<ol style="list-style-type: none"> <li>1. D. Forsyth and J. Ponce, Computer Vision: A Modern Approach, Prentice-Hall, 2003.</li> <li>2. R. Hartley and A. Zeisserman, Multiple View Geometry, Cambridge University Press, 2003.</li> <li>3. C. Bishop, Pattern Recognition and Machine Learning, Springer-Verlag, 2007.</li> <li>4. O. Faugeras and Q. T. Luong, Geometry of Multiple Images, MIT Press, 2001.</li> <li>5. B. Horn, Robot Vision, MIT Press, Cambridge, Massachusetts, 1986.</li> </ol>						
Assessment	Final exam, midterm exam and homework (programming assignments).						
Language	Greek						

Course Title	<b><i>Research Methodologies and Professional Practices in Computer Science</i></b>						
Course Code	<b><i>CS 670</i></b>						
Course Type	Compulsory						
Level	Graduate						
Year / Semester	Spring						
Teacher's Name	Y. Dimopoulos						
ECTS	4	Lectures / week	3 hours	Recitation / week	-	Laboratories / week	-
Course Purpose and Objectives	Introduction to the methods and tools of Computer Science research and technological culture. Familiarization with reading, reviewing and presenting of technical literature. Technical writing. Literature review of a research or technical topic.						
Learning Outcomes	Understand basic research methodologies and professional practices of Computer Science. Acquire skills related to reading, reviewing, summarizing and presenting scientific and technical literature.						
Prerequisites	None		Required		None		
Course Content	Lectures, research seminars and atomic assignments (summary of research seminars) and group study of a research subject under the supervision of a faculty member.						
Teaching Methodology	Lectures, ερευνητικά σεμινάρια, ατομικές εργασίες (περίληψη ερευνητικών σεμιναρίων) and ομαδική μελέτη ερευνητικού θέματος υπό την επίβλεψη μέλους ΔΕΠ.						
Bibliography	<ol style="list-style-type: none"> <li>Selected research articles from the international literature.</li> <li>Course Presentation Slides (introductory and research).</li> </ol>						
Assessment	Attendance and participation in lectures and a number of research seminars, written atomic studies, group study of a research subject and technical presentation of the group study. The course grade is Pass/Fail.						
Language	Greek						

Course Title	<i>Algorithmic Game Theory</i>						
Course Code	<b>CS 673</b>						
Course Type	Elective						
Level	Graduate						
Year / Semester	Spring						
Teacher's Name	M. Mavronicolas						
ECTS	8	Lectures / week	3 hours	Recitation / week	1 hour	Laboratories / week	-
Course Purpose and Objectives	Familiarity with algorithmic problems in Game Theory. The three basic axes for such familiarization are (i) the techniques for the design and analysis of algorithms in Game Theory, (ii) the fundamental complexity results for difficult problems in Game Theory, and (iii) the techniques for the analysis of computational systems with selfish components.						
Learning Outcomes	<p>Upon successful completion of this class, the student is expected to have learned:</p> <ul style="list-style-type: none"> <li>• The basic concepts related to strategic games (e.g., pure and mixed strategies, best responses, equilibrium).</li> <li>• Pure and mixed Nash equilibria, their refinements and generalizations, the classical existence theorems of equilibria, and the basic algorithms and complexity for computing equilibria.</li> <li>• The complexity classes PLS and PPAD.</li> <li>• The concept of the Price of Anarchy and its analysis for both general and specific games.</li> <li>• The application of Game Theory to realistic cases.</li> </ul>						
Prerequisites	Undergraduate course equivalent to the CS211 (Theory of Computation) and undergraduate course equivalent to the CS436 (Algorithms and Complexity)		Required		None		
Course Content	Strategic games. Pure and mixed strategies, utilities, best responses, equilibrium concepts. Pure and mixed Nash equilibria, their refinements and generalizations. Classical existence theorems of equilibria and their algorithmic aspects. Algorithms and complexity of equilibrium searching. The complexity classes PLS and PPAD and their relation to equilibrium computation. Bimatrix games and algorithms to compute their approximate equilibria. The Price of Anarchy and its variants. Analysis of the Price of Anarchy for both general and specific games (e.g., selfish routing games, congestion games, security games). Applications to realistic cases (e.g., social networks, Internet formation).						
Teaching Methodology	Lectures (3 hours weekly) and Recitation/Laboratory sessions (1 hour weekly).						
Bibliography	<ol style="list-style-type: none"> <li>1. M. Mavronicolas and P. Spirakis, <i>Algorithmic Game Theory</i>, Springer, 2011, (book draft).</li> <li>2. Selected research articles from the international literature</li> </ol>						
Assessment	Final exam, midterm exam and homework (theoretical assignments).						
Language	Greek						

Course Title	<i>Networks and System Security</i>						
Course Code	<i>CS 674</i>						
Course Type	Elective						
Level	Graduate						
Year / Semester	Spring						
Teacher's Name	V. Vassiliou						
ECTS	8	Lectures / week	3 hours	Recitation / week	1 hour	Laboratories / week	1.5 hours
Course Purpose and Objectives	Understanding network and information security principles, Understanding of basic areas in Cryptography, Authentication and Confidentiality. Gain of knowledge in methods for the evaluation of Software, Applications and Systems with respect to security. Application of tools for the protection of networks, applications and information.						
Learning Outcomes	<ul style="list-style-type: none"> <li>• Identify some of the factors driving the need for network and information security</li> <li>• Demonstrates ability to understand of the issues involved in the field of information security and assurance</li> <li>• Navigate through the language of the field of network and information security.</li> <li>• Explain the CIA triad of Confidentiality, Integrity and Availability</li> <li>• Identify and classify computer and network security threats and attacks</li> <li>• Compare and contrast encryption systems and algorithms.</li> <li>• Encrypt and decrypt messages and sign and verify messages using well-known techniques</li> <li>• Acknowledge the ethical and legal considerations of network and information security.</li> </ul>						
Prerequisites	Introductory graduate course equivalent to CS606 (Computer Networks and the Internet)			Required		None	
Course Content	Introduction to security threats and attacks. Cryptographic and cryptanalysis techniques. Key exchange management (PKI). Network and Internet security protocols (IPSec, SSL/TLS). Identification and authentication standards (Kerberos, AAA). Systemsecurity (Firewalls, IDS). Specific threats on end-systems (viruses, worms, trojan horses, stack overflow, rootkits). Identification of security vulnerabilities in software and operating systems. Checking of networks and applications for vulnerabilities, introduction to computer systems forensics. Security policies. Security management, ethical and legal issues in system security.						
Teaching Methodology	Lectures (3 hours weekly), Recitation (1 hour weekly) and Laboratory sessions (1.5 hours weekly).						
Bibliography	<ol style="list-style-type: none"> <li>1. C. Kaufman, R. Perlman, M. Speciner, R. Perlner, Network Security: Private Communications in a Public World, 3rd edition, Pearson 2023</li> <li>2. C. P. Pfleeger and S. L. Pfleeger, Security in Computing, Fifth Edition, Pearson 2015.</li> </ol>						
Assessment	Final exam, midterm exam and homework (studies and/or laboratory assignments).						
Language	Greek						

Course Title	<b><i>Electronic Health</i></b>						
Course Code	<b><i>CS 679</i></b>						
Course Type	Elective						
Level	Graduate						
Year / Semester	Fall						
Teacher's Name	C. Pattichis						
ECTS	8	Lectures / week	3 hours	Recitation / week	1 hour	Laboratories / week	-
Course Purpose and Objectives	To introduce the student to the medical and clinical environment from the perspective of medical informatics and exploit the possibilities of using information technologies for modeling, prototyping, presenting and using the relevant data. To study and develop practical skills in building relevant intelligent information systems.						
Learning Outcomes	<p>To discover the new world of eHealth (health). To analyze the prospects that will be created at local, European and international level. To analyze and harness the legislative and social context of health. Assess the potential of information and communications technologies in medical science, mainly through the modeling of medical practice, processes and knowledge creation. To classify, standardize and help the physician use medical health data to avoid the disease.</p> <p>Include that Electronic Health (EH) is part of a new class of acts, a philosophical approach to medical practice and the relationship between a doctor and a patient. Understanding the necessity for a transition from the Medical Center to the Athenian-based approach to medicine. Introducing students to the potential of information technologies in medicine and clinical practice, mainly through the design of medical practice, processes and knowledge, ways of managing, standardizing and presenting information.</p>						
Prerequisites	None		Required		None		
Course Content	Information retrieval from medical databases, data, medical records, live signals, and data mining using intelligent techniques. Study of application systems that are currently in use for managing medical data and suggest ways for better handling and building, medical knowledge bases, electronic health record, and decision support systems for the medical profession.						
Teaching Methodology	Lectures (3 hours weekly) and Discussions/Presentations (1 hour weekly).						
Bibliography	<ol style="list-style-type: none"> <li>1. J. H. van Bommel and M. Musen, Handbook of Medical Informatics, (Edts), Springer, 1997.</li> <li>2. E. H. Shortliffe (Editor), L. M. Fagan, G. Wiederhold and L. E. Perreault Medical Informatics: Computer Applications in Health Care and Biomedicine, Publisher: Springer Verlag; 2nd edition (November 2000).</li> <li>3. L Burke, and B. Weill, Information Technology for the health professionals, Prentice Hall, 2000.</li> </ol>						
Assessment	Final exam, midterm exam and homework (studies and/or laboratory assignments).						
Language	Greek						

Course Title	<i>Advanced Software Reuse and Mining Software Repositories</i>						
Course Code	<i>CS 681</i>						
Course Type	Elective						
Level	Graduate						
Year / Semester	Spring						
Teacher's Name	G. Kapitsaki						
ECTS	8	Lectures / week	3 hours	Recitation / week	1 hour	Laboratories / week	1.5 hours
Course Purpose and Objectives	Understanding the usefulness of software reuse. Deepening in software design patterns. Understanding the usefulness of data mining for software and being able to perform the process of collecting, preprocessing and processing data from software repositories and Q&A sites.						
Learning Outcomes	<p>The learning outcomes for the students are the following:</p> <ul style="list-style-type: none"> <li>• Understanding the usefulness of software reuse and of its advanced topics.</li> <li>• Deepening in the different levels of reuse and understanding the differences between them.</li> <li>• Use of software components in practice.</li> <li>• Understanding of mining techniques for software.</li> <li>• Having obtained an overview of research methods on software reuse.</li> <li>• Understanding of software evolution mechanisms and practical use of software mining mechanisms from social coding platforms.</li> </ul>						
Prerequisites	Basic understanding of object-oriented programming and software engineering process.		Required		None		
Course Content	Levels of reuse. Best practices for reuse. Software design patterns. Object-oriented patterns. Software repositories (e.g. GitHub). Search and retrieval. Data extraction and mining. Data mining steps. Data preprocessing and processing (e.g. in the R programming language). Use of dedicated APIs. Q&A sites, e.g. Stack Exchange, and data mining. Open source software. Open source licensing and legal issues. License compliance. Selecting licenses. Latest developments and research works.						
Teaching Methodology	Lectures (3 hours weekly), Recitation (1 hour weekly) and Laboratory sessions (1.5 hours weekly).						
Bibliography	<ol style="list-style-type: none"> <li>1. T. Diamantopoulos, A. L. Symeonidis, Mining Software Engineering Data for Software Reuse, Springer, 2020.</li> <li>2. M. Ezran, M. Morisio, C. Tully, Practical Software Reuse, Practitioner Series, 2002.</li> <li>3. Head First Design Patterns, O'Reilly Media, 2004.</li> <li>4. C. Horstmann, A Practical Guide to Open Source Licensing, Wiley, 2nd Edition, 2006.</li> <li>5. Selected research papers and articles.</li> </ol>						
Assessment	Final and midterm exam, and homework (practical exercises and research assignments).						
Language	Greek						

Course Title	<b>Data Security</b>						
Course Code	<b>CS 682</b>						
Course Type	Elective						
Level	Graduate						
Year / Semester	Fall						
Teacher's Name	E. Athanasopoulos						
ECTS	8	Lectures / week	3 hours	Recitation / week	1 hour	Laboratories / week	-
Course Purpose and Objectives	Processing data is often realized through systems that can operate under hostile conditions, where adversaries try to monetize access to sensitive data. In this course we provide a short introduction of data security, and we review the basic arsenal we have for protection. We cover a large portion of applied cryptographic primitives and protocols that facilitate secure transmission of data. We then proceed and review how systems that process data can be attacked and protected. Finally, we discuss advanced attacks, and potential defenses, for systems that are based on Machine Learning.						
Learning Outcomes	<ul style="list-style-type: none"> <li>Review basic cryptographic primitives (focus on asymmetric encryption, such as Diffie-Hellman key exchange, RSA, Elliptic curves).</li> <li>Discuss Quantum computers (Shor's and Grover's algorithm).</li> <li>Understand how Transport Layer Security (TLS) is realized and how modern attacks attempt to bypass TLS.</li> <li>Understand how attacks that focus on exfiltrating data work and what are the available defenses.</li> <li>Review ML-based attacks and defenses.</li> </ul>						
Prerequisites	None		Required		None		
Course Content	Applied cryptography concepts (AES, RSA, Elliptic Curves, SHA256/SHA3, MACs). Quantum computers (Shor's and Grover's algorithm). Transport Layer Security (TLS) and attacks. Encrypted Search. Attacks for exfiltrating data from systems and possible defenses (oblivious memory, differential privacy, k-anonymity). ML-based attacks (adversarial input generation, membership inference attack) and defenses. GDPR.						
Teaching Methodology	Lectures (3 hours weekly), Recitation (1 hour weekly).						
Bibliography	<ol style="list-style-type: none"> <li>Handbook of Applied Cryptography, <a href="http://cacr.uwaterloo.ca/hac/">http://cacr.uwaterloo.ca/hac/</a></li> <li>Understanding Cryptography, <a href="http://www.crypto-textbook.com">http://www.crypto-textbook.com</a></li> <li>Published papers</li> </ol>						
Assessment	Final exam, midterm exam, bibliographic project and presentation.						
Language	Greek or English						

Course Title	<i>Special Topics in Computer Science</i>						
Course Code	<b>CS 699</b>						
Course Type	Elective						
Level	Graduate						
Year / Semester							
Teacher's Name	CS or Visiting Faculty						
ECTS	8	Lectures / week	3 hours	Recitation / week	1 hour	Laboratories / week	-
Course Purpose and Objectives	This course enables to introduce cutting-edge topics within Computer Science that are not addressed by existing courses. As the field of Computer Science evolves rapidly with the emergence of new technologies, this course provides students with a unique opportunity to explore the latest advancements and trends.						
Learning Outcomes	Determined by the course instructor, approved by the Graduate Studies Committee and the Department Council.						
Prerequisites	None		Required		None		
Course Content	Determined by the course instructor, approved by the Graduate Studies Committee and the Department Council.						
Teaching Methodology	Lectures (3 hours weekly), Recitation (1 hour weekly).						
Bibliography	Determined by the course instructor, approved by the Graduate Studies Committee and the Department Council.						
Assessment	Determined by the course instructor, approved by the Graduate Studies Committee and the Department Council.						
Language	Greek						

Course Title	<b><i>Independent Study</i></b>						
Course Code	<b><i>CS 720</i></b>						
Course Type	Restricted Elective Course						
Level	Graduate						
Year / Semester	Spring						
Teacher's Name	CS Faculty						
ECTS	8	Lectures / week	-	Recitation / week	-	Laboratories / week	-
Course Purpose and Objectives	<p>The independent study offers students an opportunity to gain comprehensive knowledge in a specific area of Computer Science under the guidance of a faculty advisor. This course emphasizes independent work and involves the design, development, and evaluation of a software system, research study, or theoretical analysis.</p> <p>Under faculty supervision, students will:</p> <ul style="list-style-type: none"> <li>• Identify a problem or research question within the field of Computer Science.</li> <li>• Conduct a review of the state-of-the-art.</li> <li>• Propose and implement a solution or conduct an investigation using appropriate methodologies and tools.</li> </ul> <p>The project culminates in a formal written report and a presentation of the findings to the supervising faculty.</p> <p>The study's theme (problem or research question) is collaboratively determined by the student and the supervising faculty. The PSC must approve:</p> <ul style="list-style-type: none"> <li>• Before the start of the course, the chosen independent study theme.</li> <li>• At the end of the course, the outcomes and assessment.</li> </ul>						
Learning Outcomes	<ul style="list-style-type: none"> <li>• Understand and potentially challenge the existing knowledge and practice related to the project problem</li> <li>• Identify and demonstrate appropriate methodologies and know when to use them</li> <li>• Define, articulate and use terminology, concepts, and theory in their field and know how to use them</li> <li>• Use library and other tools to search for existing body of research relevant to their topic</li> <li>• Identify and practice research/development ethics and responsible conduct in research or development</li> <li>• Know and apply problem solving skills to constructively address research/development setbacks</li> <li>• Work autonomously in an effective manner, setting and meeting deadlines</li> <li>• Reflect on their own work, identifying lessons learned, strengths, and ways to improve</li> <li>• Communicate confidently and constructively with faculty as mentors</li> <li>• Produce a well-structured written report that effectively communicates the problem, methodology, results, and conclusions of the project, following academic and professional standards</li> </ul>						
Prerequisites	Successful completion of at least 30 ECTS of postgraduate courses		Required		None		
Course Content	<p>This course provides an opportunity for study under the guidance of a faculty member on topics in Computer Science not covered in-depth by other postgraduate courses offered by the Department.</p> <p>Students interested in undertaking an independent study are encouraged to discuss potential topics with one or more faculty members in the department. Once a faculty member agrees to supervise a specific independent study, they will notify the Postgraduate Studies Committee (PSC) to approve the student's enrollment in this course.</p>						
Teaching Methodology	Individual work under supervision						
Bibliography							
Assessment	Written report, Software/System Prototype Assessment (if applicable), and oral presentation. Assessed by supervising faculty numerically						
Language	Greek or English						

## 2. MASTER in ARTIFICIAL INTELLIGENCE (MAI)

The aim of the English-language master's programme in Artificial Intelligence (AI), offered since September 2022, and since September 2023 has evolved into an interdepartmental with the Department of Electrical and Computer Engineering, is to be a modern programme, which will contain a strong interdisciplinary component, as required by human-centred, explainable and responsible artificial intelligence. Compulsory courses include courses on artificial intelligence and ethics, as well as artificial intelligence and entrepreneurship. Providing career advice to students is a top priority, with the aim of helping all graduates to successfully pursue an AI-related career, possibly setting up their own start-ups.

The development of this new master's programme was co-funded by the European Union (Connecting Europe Facility (CEF) – Telecommunications Sector). Europe's initiative to fund new master's programmes in the field of artificial intelligence demonstrates the importance that Europe attaches to recent developments in this field, providing solutions to global problems related to every aspect of human life, contributing to growth and competitiveness. Europe urgently needs more AI professionals, entrepreneurs and researchers, capable of paving the way for new innovations for the good of society and pushing the frontiers of the field to new challenges. This new generation of AI graduates must be fully familiar with the latest technological developments in terms of the breadth and depth of technical knowledge in AI, be connected to industry and be fully aware of the ethical issues involved.

The project's cooperation network includes three European universities (University of Cyprus, University of Bologna, and Ruse Angel Kanchev University), three Centres of Excellence (KIOS, CYENS, Eratosthenes), the Cyprus Institute, and a large number of high-tech SMEs, including 3AeHealth LTD, DENOVA, MLPS AD, AC Goldman, Deriv and Artemis Industries. Cultivating close links with industry provides significant added value to the educational experience.

The intended learning outcomes of the programme are the following:

- **Fundamental Principles of Intelligent Systems** – Understanding the basic principles that define intelligent software systems and keeping up to date with the latest developments in Artificial Intelligence.
- **Integrated Understanding of Machine Learning** – Gain a comprehensive understanding of the principles of machine learning that guide scientific and industrial innovations in Artificial Intelligence.
- **Business Perception in Artificial Intelligence and Data** – Understanding key concepts and challenges related to entrepreneurship in Artificial Intelligence and data utilization.
- **Practical Applications and Ethical Issues** – Application of research methods and tools to Artificial Intelligence, taking into account professional practices, regulatory frameworks and ethical issues.

### Programme Structure

Semester/Courses	ECTS Credits
Fall Semester 1	30
MAI611 Fundamentals of Artificial Intelligence	8
MAI612 Machine Learning	8
MAI613 Research Methodologies and Professional Practices in AI	4
MAI614 AI on the Edge Webinars I	2
Elective course 1	8
Spring Semester 1	30
MAI621 Artificial Intelligence Ethics I	6
MAI622 AI Entrepreneurship	8
Elective course 2	8
Elective course 3	8

MAI602 Internship (Student participation is optional)	8
MAI601 AI Camp	4
MAI602 Research/industrial internship (Based on evaluation, the internship could be considered as an elective course)	8
Fall Semester 2	30
MAI631 Artificial Intelligence Ethics II	4
MAI632 AI on the Edge Webinars II	2
Elective course 4	8
MAI641 Master Thesis or Elective course 5 and Elective course 6	16
Total ECTS	90-98

**Admission Criteria:**

- Bachelor's Degree in Science/Applied Sciences (Computer Science, Mathematics, etc.), Engineering (Electrical Engineering, Biomedical Engineering, etc.) or Cognitive Science
- Previous report on Artificial Intelligence issues or relevant work experience
- Programming knowledge
- Knowledge of the English language

Full information about the programme and course descriptions can be found on the <https://www.cs.ucy.ac.cy/index.php/education/postgrad/master-in-artificial-intelligence> and <https://mai4car.eu> websites.

**3. MASTER in DATA SCIENCE (DSC)**

The Department of Mathematics and Statistics, the Department of Business Administration and Public Administration and the Department of Computer Science participate in the postgraduate programme in Data Science. The programme is offered since September 2021 and is taught in English. The area of Data Science is about extracting knowledge from large amounts of data. Data Science is today a key field for the strategy of modern organizations, creating a growing need for highly qualified data scientists. The programme aims to provide students with a strong understanding of basic and advanced methods in statistical inference, machine learning, data visualization, and data mining, which are essential skills for a data scientist.

Completion of the programme requires 90 ECTS credits and 1.5 years of study. The programme offers three specializations (computer science, statistics, business analysis), where students will be able to choose the direction they want at the end of the second semester of their studies.

**Admission Criteria:**

The minimum qualifications required are:

- University degree from a **recognized institution** in a relevant field (Statistics, Computer Science, Mathematics, Engineering, Economics, Business Administration, Physics) with a **minimum grade of 6.5/10**.
- In case the degree has not yet been issued, the application can be submitted provisionally without it but **must be submitted before the start of classes (September)**.
- **Very good knowledge of the English language** (relevant examinations and/or certificates that have been obtained and prove the required level of English language can be found on the following website: <https://www.ucy.ac.cy/graduateschool/admission-requirements-2/>). This is not necessary for candidates who received their degrees from programmes taught in English.

Additional academic criteria required to apply are:

- Successful completion of Probability, Statistics or Econometrics courses and basic Mathematics courses (e.g. basic calculus, linear algebra, etc.)  
Successful completion of Computer Science courses (e.g., programming principles in Python, R, etc.)

Full information about the programme and course descriptions can be found on <https://datascience.cy/> website.

#### **4. MASTER in COGNITIVE SYSTEMS**

It is a distance learning programme that has been offered since September 2017 and is taught exclusively online in collaboration with the Open University of Cyprus and the Department of Psychology of the University of Cyprus. The programme is taught in English.

Cognitive Systems are a new generation of systems that aim to collaborate with their users at a level cognitively compatible with the common understanding of people, in order to provide personalized and customized services, where the system and the human being learn and adapt to each other's capabilities. The need for the development of these cognitive systems has been widely recognized. The Watson machine opened up the area of Cognitive Computing, and we are now seeing the development of Cognitive Assistants, such as Siri, Cortana, Alexa, Google Assistant and others, by every major computer company in the world.

Students are required to choose:

- 3 basic courses (COS511, COS512, COS513), where at least 2 are introductory courses (COS511, COS512), during the 1st Semester of study.
- Elective courses, where one-third must come from the area of Cognitive Psychology and one-third from the area of Computer Science.

#### ***Admission Criteria:***

The programme is aimed at students with a first degree in the fields of STEM (Science, Technology, Engineering, and Mathematics) or a first degree in Cognitive Science or Psychology. The programme requires basic knowledge in the field of mathematics (discrete mathematics, formal logic, probability/statistics, calculus) and computers (algorithms, basic programming). This should be demonstrated in a prospective student's application by listing relevant courses taken in previous degrees or by providing proof of completion of relevant online courses (e.g. MOOCs).

Highly motivated applicants from other disciplines are also encouraged to apply, but should expect to make extra efforts to supplement their previous studies by following additional introductory elements linked to individual courses so that they can gain the required background for these courses.

All applicants must prove that they meet the University's English language requirements by submitting the relevant certificates along with their application (minimum IELTS grade 5.5 or any other equivalent). Details on the specific English language requirements can be found here:

[https://www.ouc.ac.cy/images/Verification\\_of\\_english\\_language\\_knowledge\\_2025.pdf](https://www.ouc.ac.cy/images/Verification_of_english_language_knowledge_2025.pdf)

Full information about the programme and how to apply for the programme can be found on <http://cogsys.ouc.ac.cy website>.

## ***PhD in Computer Science***

The PhD is the most advanced degree conferred by our department and is designed for scholars who are driven to expand the boundaries of our field. The program provides a rigorous environment where candidates transition from students to independent researchers. By engaging in original research and working alongside our faculty, doctoral students address complex challenges of the modern world and emerge as the next generation of academic and industry leaders.

The PhD program in Computer Science aims to:

- Prepare students to undertake high quality research in Computer Science.
- offer students the opportunity to acquire deep knowledge in one or more fields of Computer Science.
- Prepare graduates able to pursue careers in positions of responsibility in either academia or industry, where they will effectively drive the development and application of new methods and ideas.
- Offer the students with education in the widest sense of the term, and cultivate the desire for continuous learning, which, in turn, leads to maturity and develops the facilities for independent and critical thinking.
- Help our graduates in order to acquire a deep understanding of Computer Science, both as a science, and in terms of its more general applications and effects on society.
- Prepare effective science communicators and teachers for both academia and industry.

With the completion of a PhD a graduate is expected to:

- Develop the ability to independently, identify, formulate, tackle and solve research problems in Computer Science.
- Have acquired deep knowledge in one or more fields of Computer Science.
- Possess powerful tools for addressing a wide range of topics.
- Demonstrate in-depth understanding of a breadth of disciplines in Computer Science and be largely familiar with the dominant research directions and cutting edge problems.
- Understand how principles and methods from Computer Science are used in modern interdisciplinary research areas.
- Exhibit versatility and innovative thinking in addressing and managing open questions in a variety of contexts, as an essential asset for careers in research, industry, commerce, education and the public sector.
- Develop skills such as: oral and written scientific communication, fluent use of scientific English, use of information/communication technology, organization and planning of group work.

## ***Program Structure***

For the completion of the program, students are required to complete 240 ECTS as follows:

- **At least 60 ECTS** of graduate level coursework (successful completion of at least 60 ECTS in courses at the graduate level). A master's level diploma or equivalent title partially or completely exempt students from this requirement.
- **At least 120 ECTS** for the research part of the program (research stage – 4 semesters, 30 ECTS per semester).
- **At least 60 ECTS** for the comprehensive examination, preparation and presentation of the research proposal and the writing of a Doctoral Thesis (dissertation stage – at least 2 semesters, 30 ECTS per semester).

The minimum duration of the program is six (6) semesters (3 years), while the maximum duration is sixteen (16) semesters (8 years)

The structure of the program of study is as follows:

PROGRAMME REQUIREMENTS	ECTS
Compulsory:	
(1) Research Methodologies	4
(2) Comprehensive Examination	0
(3) Research Stages	120
(4) Dissertation Proposal	0
(5) Writing Stages	60
Elective Courses	56
Total ECTS	240

### ***Applications and Admission Criteria***

The applications for the PhD program can be submitted to the Postgraduate School for Fall or Spring admission as follows:

- for admission to the Fall Semester, by 31 March every year
- for admission to the Spring Semester, by 31 October every year

Applicants who hold a recognized university degree, as well as persons who are expected to obtain a degree before the beginning of the curriculum, are eligible to apply.

Candidates shall submit the application form online, including the information below, within the submission deadline:

- A curriculum vitae.
- A certified photocopy of a recognized degree or qualification that has been approved as equivalent to a university degree by the Cyprus Council of Recognition of Higher Education Qualifications (KYSATS). If the degree/qualification has not been awarded, a certificate of expected graduation must be submitted
- A certified photocopy of the academic transcript for all curricula.
- A brief statement (up to two pages) stating the research goals and interests.
- A preliminary research proposal for admission to a doctoral program
- At least two (2) letters of recommendation

### ***Evaluation of the Application***

The Postgraduate School examines whether the candidates meet the basic requirements for admission and then forwards the applications to the departmental coordinators of the Postgraduate Studies.

The applications for Doctoral studies at the Department of Computer Science are examined by a four-member Postgraduate Studies Committee (PSC) consisting of the Graduate Program Coordinator and three other Members of the Academic Staff. The members of the PSC are appointed by the Council of the Department.

- The PSC examines the applications for admission and ranks candidates accordingly. The PSC has the right to invite candidates to an interview, without being obliged to do so for all candidates.
- For the PhD in Computer Science, the department requires the applicants to have a University Degree in Computer Science (or a relevant discipline) with a high grade.
- Additional admission criteria usually apply, such as: (a) fit between candidate's research interests and faculty expertise (b) number of students' publications in scientific journals (c) students' participation in seminars, symposia, research

- programmes (d) presentations by students at conferences (e) professional experience, etc.
- For students who have a university degree in a subject other than Computer Science/Informatics but are otherwise eligible to enter the PhD program, a personal interview is conducted to ascertain whether the candidate has the appropriate knowledge in Information Technology to be accepted as a PhD student or whether the candidate can be accepted in the program but with the provision to acquire necessary background knowledge through successfully completing relevant undergraduate and/or postgraduate courses.
  - The PSC recommends to the Council of the Department the names of the students to be admitted.
    - The PSC may also recommend transfer of graduate credit for qualified applicants.
  - Final approval for the admission of students to the Graduate Program is given by the Board of the Department, which reserves the right not to fill all the positions announced.
  - The candidates are informed about the outcome of their application.
  - The Chairperson of the Department, notifies the Dean of the relevant Faculty, the Postgraduate School and the Rector's Council regarding the students who have been admitted.

### ***Additional Information***

Additional information about the program and its rules can be found in the Appendices of the Department's Prospectus as well as in the University's Rules for Postgraduate Studies.

## Short Biographical Notes of Academic Staff

**Andreas Aristidou**, Associate Professor. He completed his undergraduate studies at the National and Kapodistrian University of Athens, Greece (BSc in Informatics and Telecommunications, 2005), postgraduate studies at King's College London, UK (MSc in Mobile and Personal Communications, 2006), graduating with honors, and at the University of Cambridge, UK (PhD in Information Engineering, 2011) as a Cambridge European Trust scholar. He has worked as a postdoctoral researcher at Reichman University in Israel, Shandong University in China, the University of Cyprus, and the Cyprus University of Technology. He was also a visiting professor at the University of Nicosia. He is a senior member of the Association of Computing Machinery (ACM), the Institute of Electrical and Electronic Engineers (IEEE), Eurographics, and the Scientific and Technical Chamber of Cyprus, where he served on the research committee. He is a member of the editorial board of the international journals *The Visual Computer (TVC)* and *Heritage*, and a guest editor at *Advances in Applied Clifford Algebras (AACAA)*. He has received numerous scholarships, distinctions, and research grants from highly competitive local, European, and international agencies. His main research interests lie at the intersection of computer graphics, virtual reality, and computer vision. He specializes in character animation, using techniques like machine learning, artificial intelligence, and generative models to create virtual humans. His work also extends to 3D analysis, visualization, and classification of human motion, with applications in games and simulators, digital cultural heritage, VR/AR/MR environments, and the application of Conformal Geometric Algebra in computer graphics. He has participated in several EU-funded projects and collaborates with European creative industries to design innovative algorithms for character motion synthesis and character retargeting.

**Elias Athanasopoulos** is an associate professor in Computer Science with the University of Cyprus. He received his BSc in Physics from the University of Athens and his PhD in Computer Science from the University of Crete. Before joining University of Cyprus, he was an assistant professor with Vrije Universiteit, Amsterdam. Elias was a Microsoft Research PhD Scholar, as well as a Marie Curie fellow with Columbia University and FORTH. His research interests are system security and privacy. He has several publications in all major systems, system-security, and privacy venues, such as *IEEE Security and Privacy (Oakland)*, *IEEE European Security and Privacy*, *ACM CCS*, *Usenix Security*, *Usenix ATC*, *NDSS*, *EuroSys*, *PETS*, and *ACM TOPS*.

**Chris Christodoulou**, Professor. Undergraduate studies at Queen Mary and Westfield College, University of London, UK (BEng in Electronic Engineering, 1991) and Birkbeck College, University of London, UK (BA in German, 2008). Graduate studies at the Kings College, University of London, UK (Ph.D. in Electronic Engineering, 1997). He has worked as a Postdoctoral Research Associate at the Kings College, University of London, UK (1995-97), and he has taught as a Lecturer (Assistant Professor) at Birkbeck College, University of London, UK (1997-2005). His current research interests include computational neuroscience, neural networks and machine learning.

**Giorgos Chrysanthou**, Professor. He directs the Graphics and Hypermedia lab and is Research Director at the newly established CYENS Centre of Excellence (Research Centre in Interactive Media, Smart Systems and Emerging Technologies). He studied in England (BSc and PhD from Queen Mary College) and worked for several years as a researcher and lecturer at University College London. He has been scientific coordinator or member of research teams of 25 research projects, related to graphics, virtual reality and cultural heritage and funded by various national and European sources. He has published more than 80 articles in journals and international conferences in the above fields and is one of the authors of the book "*Computer Graphics and Virtual Environments: From Realism to Real-Time*", (Addison-Wesley 2001). He was a member of the editorial boards of the journals *Computer Graphics Forum* (2017 - today), *Computers & Graphics* (2011 - today) and *Eastern Mediterranean*

Archaeology and Heritage Studies (2011-2016). He has organized 6 international conferences and served as chairman of the program committee in another 5. His most recent research interests include realistic real-time graphics, 3D representations and applications in various fields, including cultural heritage.

**Eleni Constantinou**, Assistant Professor. Undergraduate and postgraduate studies at the Aristotle University of Thessaloniki (BSc in Computer Science, 2005 and 2007) and PhD in Computer Science (2015). She has worked as a postdoctoral researcher at the University of Mons in Belgium (2016-2019) and as an Assistant Professor at the Technical University of Eindhoven in the Netherlands (Eindhoven University of Technology, 2019-2022). Her research interests include software evolution, software ecosystems and mining software repositories. She has published several articles in premier international conferences (MSR, ICSME, SANER, ESEM, etc.) and journals (EMSE, JSS, IEEE Software, etc.). She has participated in the program committees of leading international conferences (ICSE, ESEC/FSE, MSR, ICSME, SANER, etc.), journals (IEEE TSE, ACM TOSEM, JSS, EMSE, IST, IEEE Software, etc.) and in the organizing committee of leading conferences (ICSE, ESEC/FSE, MSR, ICSME, ICPC, etc.). He has received ACM SIGSOFT distinguished paper award at MSR'23 conference and IEEE TCSE distinguished paper award at SANER'22 conference, distinguished reviewer awards at MSR'19 and ICSME'20 conferences and an outstanding contribution award for the organization of the MSR'24 conference as program co-chair.

**Marios D. Dikaiakos**, is Professor of Computer Science at the University of Cyprus and founding director of the University's Centre for Entrepreneurship, since 2015. He served as Head of the Computer Science Department from 2010-2014. He is also the founding Director of the Laboratory for Internet Computing. Dikaiakos received his Ph.D. in Computer Science from Princeton University (1994), an M.A. in Computer Science from Princeton (1991), and a Dipl.-Ing. degree in Electrical Engineering from the National Technical University of Athens (summa cum laude, 1988). Dikaiakos' research focuses on Cloud Computing, Online Social Network Analysis and Data Science. His research work includes over 200 publications in books, journals and international conference proceedings, edited volumes and conference proceedings, and the direction of research projects at the University of Cyprus with a total budget exceeding 6 million Euros for the period 2000-2024. He has also served as program chair, co-chair and program committee member in numerous international scientific conferences; independent evaluator of European academic and research institutions; reviewer and evaluator for research proposals submitted to the E.U. and to several national European science foundations; and independent observer of proposal evaluation exercises of the E.U. under Framework Program 7. He is a Senior Member of the ACM, was a member of ACM's Distinguished Speakers Program (2012-2015), a member of the IEEE Computer Society and the Technical Chamber of Greece.

**Yannis Dimopoulos**, Professor. He received a PhD from the Department of Informatics of Athens University of Economics and Business in 1992. He has held research positions at the Max-Planck for Computer Science, and the University of Freiburg. His research interests are in the areas of Argumentation, Answer Set Programming, Planning, Constraint Satisfaction and Machine Learning. He publishes his work in international journals and top conferences such as IJCAI, AAI, AAMAS, etc. He serves regularly on the program committees of various conferences, and as a reviewer for international journals.

**Chryssis Georgiou**, Professor. Undergraduate studies at the University of Cyprus, Cyprus (B.Sc. in Mathematics, 1998). Graduate studies at the University of Connecticut, USA (M.Sc., 2002; Ph.D., 2003, both in Computer Science and Engineering). His research interests span the Theory and Practice of Fault-tolerant Distributed and Parallel Computing with a focus on Algorithms and Complexity. He has published more than 115 articles in journals and conference proceedings in his area of study and he has co-authored two books on Distributed Cooperative Computing. In August 2021, he co-edited a book

on the Principles of Blockchain Systems. Prof. Georgiou has served on several Program Committees of conferences in Distributed Computing and on the Steering Committees of EATCS DISC (2008-2010, 2010-2012) and of ACM PODC (2013-2015). He has also served a three-year term as the Chair of the Steering Committee of ACM PODC (2021-2024). In 2015, he served as the General Chair of PODC 2015, and in 2017 he served as the Track Program Committee co-Chair (Stabilizing Systems: Theory and Practice Track) of SSS 2017. In 2018, 2019, 2021 and 2022 he served as the co-Chair of the workshop ApPLIED. In 2020, he served as the PC co-Chair of NETYS 2020 and in 2023 as the PC co-Chair of the Algorithms and Theory Track of Euro-Par 2023. From January 2018 to December 2023, he served on the Editorial Board of Information Processing Letters. Prof. Georgiou's research has been funded by the University of Cyprus, the Cyprus Research and Innovation Foundation, and the European Commission.

**Georgia Kapitsaki** is an Associate Professor at the Department of Computer Science of the University of Cyprus. Her research interests include Software Engineering, Open Source Software, Human Aspects in Software Engineering, and Privacy Enhancing Technologies. She received her PhD from the National Technical University of Athens, Greece (2009). She has worked as a software and telecommunications engineer in Germany (2005, 2009-2010) and as a visiting researcher at the Otto von Guericke University of Magdeburg and TU Delft. She has been involved in the organisation of international conferences (e.g. ICSME 2022, SAC 2019, ICSR 2016, etc.) and has served as a member of the program committee (e.g., MSR 2024, SANER 2024, ESEM 2023). She has published over 60 papers in international journals and conferences. She has received funding from European research projects (e.g. Support4Resilience, SocioCoast).

**Elpida Keravnou-Papailiou** is Professor of Computer Science at the Department of Computer Science of the University of Cyprus and Co-Coordinator of the MSc Artificial Intelligence programme whose development was co-funded by EU under the MAI4CAREU project (INEA/CEF/ICT/A2020/2267423). She studied at Brunel University, UK (B. Tech. in Computer Science, 1982; Ph.D. in Cybernetics, 1985). She started her academic career in the Department of Computer Science at University College London in 1985. In 1992 she took up an academic position in the Department of Computer Science at the University of Cyprus where she served as Vice-Rector for Academic Affairs from 2002 to 2006, as Dean of the School of Pure and Applied Sciences from 1999 to 2002 and as the Chairperson of the Department of Computer Science, from 1994 to 1998 as the Department's first Chair and subsequently during the period from 2016 to 2022. She served as a Member of the Governing Board (June 2012 –June 2018) and of the Executive Committee (July 2014 –June 2018) of the European Institute of Innovation and Technology (EIT-[www.eit.europa.eu](http://www.eit.europa.eu)). She is a Fellow of the 2021 class of the International Academy of Health Sciences Informatics (IAHSI) and of the European Academy of Sciences and Arts (EASA). She has carried out research in the areas of knowledge engineering, expert systems, deep knowledge models, diagnostic reasoning, temporal reasoning, artificial intelligence in medicine, intelligent data analysis in medicine and hybrid decision support systems. She is an Editor of the scientific journal *Artificial Intelligence in Medicine* (Elsevier) since the launch of the journal in 1989. During the period from 2003 to 2005 she served as Chairperson of the Artificial Intelligence in Medicine Europe (AIME) Board. She is a Senior Program Committee Member of the biannual conferences AI in Medicine. She served as the first Rector of the Cyprus University of Technology during the period 2012-2015.

**Panayiotis Kolios** is an Assistant Professor at the Computer Science Department of the University of Cyprus. He received his BEng degree in Telecommunications Engineering from King's College London in 2008. He then joined the Centre for Telecommunications Research at King's College as a PhD student, funded by an EPSRC DTA scholarship, where he received his PhD degree in 2011. His interests are on both basic and applied research on networked intelligent systems. Some examples of

systems that fall into the latter category include intelligent transportation systems, autonomous aerial systems and the plethora of cyber-physical systems that arise within the Internet of Things. Particular emphasis is given to the domain of emergency management in which natural disasters, technological faults and cyber-attacks could cause disruptions to the aforementioned systems that need to be effectively handled. Tools used to study these systems include graph theoretic approaches, algorithmic development, mathematical and dynamic programming, as well as combinatorial optimization. He has been involved in numerous projects in this area of expertise with the total funding managed exceeding 5 million euros. He has published more than 200 publications in journals, conferences, and book chapters.

**Marios Mavronicolas**, Professor. Undergraduate studies in Electrical Engineering at the National Technical University of Athens, Greece (Diploma in Electrical Engineering, Summa cum Laude; November 1985). Graduate studies in Computer Science at Harvard University, USA (M.A., 1988; Ph.D., 1992). He has been Visiting Faculty at the University of Oxford (2025). He has taught at the University of Paderborn, Germany, as a Guest Professor (2006 through 2007), at the University of Connecticut, USA, as Assistant Professor (1999), and at the University of Crete, Greece, as Visiting Assistant Professor (1992-93). He is a member of the Editorial Board of the esteemed scientific journal *Theoretical Computer Science* since 1997. He has been co-chair of the Program Committee of the 20th Conference on Web and Internet Economics (WINE 2024), chair of the Program Committee of the 13th International Symposium on Algorithms and Complexity (CIAC 2023), chair of the Program Committee of the 2nd International Symposium on Algorithmic Game Theory (SAGT 2009), co-chair of the Program Committee of the 2nd International Workshop on Web and Internet Economics (WINE 2006) and chair of the Program Committee of the 11th International Symposium on Distributed Algorithms (WDAG 1997). He has been Vice-Rector of International Affairs, Finance and Administration at the University of Cyprus (2010-2014). He has also been Chair and Vice-Chair of the Cyprus Council of Recognition of Higher Education Qualifications (KYSATS) (2009-2015). His research interests span the field of Algorithms and Complexity, with special focus on Algorithmic Game Theory, Economics and Computation, and Distributed Computing, areas in which he has published extensively in the most competitive scientific journals and international conferences of *Theoretical Computer Science*, where he is a (co)-author of more than 150 research publications. Recently his research interests have been extended to Theoretical Artificial Intelligence.

**Mihalis Nicolaou**, Assistant Professor. Previously, he was Associate Professor at the Computation-based Science and Technology Research Center at The Cyprus Institute and has held faculty and visiting researcher positions at the University of London, and Rutgers University. He received the B.Sc. degree from the University of Athens, Greece (Dept of Informatics and Telecommunications, 2008), and the M.Sc. and Ph.D. degrees from the Department of Computing, Imperial College London, U.K (2009, 2014). He is interested in developing machine learning algorithms that are robust, efficient, generalizable and interpretable. His work spans a wide range of applications; from computer vision, multimodal learning, and natural language processing, to tackling inter-disciplinary challenges in domains of critical impact, such as climate and earth observation, health, and science. He has received several awards for his research and has published more than 80 research articles in leading venues (e.g. NeurIPS, ICML, ICLR, IEEE TPAMI). His work has been supported by national, European, and industrial grants.

**George Pallis** is associate professor at the Computer Science Department, University of Cyprus and Associate Director of Laboratory of Internet Computing. He received his BSc (2001) and Ph.D. (2006) degree in Department of Informatics of Aristotle University of Thessaloniki (Greece). His research interests include Distributed and Internet Computing with particular focus on Big Data Analytics, and Cloud/Edge/Fog Computing. Dr. Pallis is one of the programme directors of the Data Science Master

programme at University of Cyprus. He is principal institutional investigator in research projects funded by EC, Research Promotion Foundation in Cyprus, and industry (e.g., Google) and has totally attracted more than 5.5M euro. Dr. Pallis has published over 100 papers in international journals (e.g., IEEE TKDE, IEEE TCC, IEEE TSC, ACM TOIT, Scientific Reports etc), magazines (e.g., CACM, IEEE Internet Computing) and conferences (e.g., INFOCOM, IPDPS, ICDCS, SEC, IoTDi, ICWSM, BIG DATA etc) and he is contributor of two international DIN (German Institute for Standardization) standards. His papers have been cited more than 5500 times with h-index 29 (Source: Google Scholar). Dr. Pallis has served as PC-Co-chair of CloudCom 2018 and CCGrid 2019 and General chair in IEEE/ACM SEC 2024 and General co-chair IEEE IC2E2024 conference. Dr. Pallis has also served in numerous Program and Organization Committees for international conferences (WWW, ICDCS, CONEXT, BIG DATA etc.) and he received the best paper awards in IEEE CloudCom 2024, IEEE/ACM UCC 2023, IEEE IoTDi 2022, IEEE ISCC 2022, IEEE BIG DATA 2016, ICSOC 2014 and the best demo award in the ACM/IEEE Symposium on Edge Computing. In 2019, Dr. Pallis served as guest editor for the edge computing special issue on the prestigious Proceedings of the IEEE journal. Dr Pallis is appeared in the World's Top 2% Scientists list published by Stanford University and he is golden core member of IEEE Computer Society. He was Editor in Chief (EIC) in the IEEE Internet Computing magazine (2019-2023) and Associate Editor in the IEEE Transactions on Cloud Computing. Currently, he is EIC Emeritus in the IEEE Internet Computing magazine, Associate Editor in IEEE Reliability Magazine and Associate Editor in the Computing Journal (Springer).

**George A. Papadopoulos** holds the (tenured) rank of Full Professor in the Department of Computer Science, University of Cyprus. His research interests include Advanced Software Engineering, Ubiquitous Computing, Cloud Computing, Parallel and Distributed Programming Models, Technology Enhanced Learning, Medical Informatics, Assistive Technologies, Context Aware and Recommender Systems, and Internet Technologies. He has published over 200 papers as book chapters or in internationally refereed journals and conferences, he is a current or past member in the Editorial Board of 18 international journals and is serving or has served as a Chair or Steering or Program Committee member in more than 200 international conferences. Professor Papadopoulos is a recipient of a 1995 ERCIM-HCM scholarship award. He has been involved or is currently participating, as coordinator or partner, in more than 100 internationally and nationally funded projects (total budget for his participation is more than 11 MEURO) and has been invited by the E.U. and national agencies as an Expert Evaluator or Reviewer more than 100 times. He is the Director of the Software Engineering and Internet Technologies (SEIT) Laboratory.

**Constantinos S. Pattichis**, Professor. He has 30 years of experience in eHealth and connected health, medical imaging, and explainable AI, and more recently in mHealth interventions based on X Reality applications. He has been involved in numerous projects in these areas with a total funding managed exceeding 20 million euros. He has published 152 journal publications, 265 conference papers, 3 monographs, 30 chapters in books and co-editor of 4 edited volumes, 22 journal special issues and 20 conference proceedings. He was the Technical Leader of the EU Digital Covid Certificate Platform for the issuance of the corresponding certificates for vaccination which has been used by more than 1 million users. He is also leading the “Deployment of Generic Cross Border eHealth Services in Cyprus”, under the EU Health and Digital Executive Agency (HaDEA). He is a Member of the European Academy of Sciences and Arts, Fellow of IEEE, IET, International Academy of Medical and Biomedical Engineering (IAMBE) and European Alliance for Medical & Biological Engineering & Science.

**Anna Philippou**, Professor. Undergraduate studies at the University of Oxford, UK (B.A. in Mathematics and Computation, 1992). Graduate studies at the University of Warwick, UK (M.Sc. in Parallel Computers and Computation, 1993 (with Distinction); Ph.D. in Computer Science, 1997). She

has worked as a Teaching Assistant at the University of Warwick, UK (1993-1996) and as a Postdoctoral Research Fellow at the University of Pennsylvania, USA (1997-1998). Her research interests include Concurrency Theory and its Applications, Privacy, Foundations of Mobile, Distributed, Reversible, and Probabilistic Systems, Formal Methods for Safety-Critical Systems, and Algorithmic Game Theory. She has published her work in top journals and conference proceedings in her research field. She served in the Program Committees of top conferences on Formal Methods and co-chaired the Program Committees of the international conferences TACAS 2009 and FORTE 2022. She has also served as the General Chair of ETAPS 2010 and FM 2016. Her research activity has been funded by the Cyprus Research Promotion Foundation and the European Commission.

**Andreas Pieris**, Assistant Professor. Undergraduate studies at the University of Cyprus (B.Sc. in Computer Science, 2006). Graduate studies at the University of Oxford (M.Sc. in Mathematics and Foundations of Computer Science, 2007; D.Phil. in Computer Science, 2011). He has worked at the University of Edinburgh (Associate and Assistant Professor, 2016 - 2021), the Vienna University of Technology (Postdoctoral Researcher, 2014 - 2016), and the University of Oxford (Postdoctoral Researcher, 2011 - 2014). His research interests focus on database theory with emphasis on uncertain data, knowledge representation and reasoning, and logic in computer science. He has published numerous papers in leading international conferences and journals. He has also served on the program committees of several conferences, including the top-tier database theory and artificial intelligence conferences.

**Yiannakis Sazeides**, Professor. Educated at Oakland University, USA (B.Sc., Computer Engineering, 1991). Postgraduate studies at Cornell University, USA, (M. Eng., USA Electrical Engineering, 1992) and University of Wisconsin-Madison, USA (Ph.D., Electrical Engineering, 1999). He has worked in high-performance processor R&D labs at HP, Compaq and Intel. He has taught at the Computer Science of the University of Cyprus as a Visiting Lecturer (2000-2001). His research interests include Computer Architecture with particular emphasis on Reliability, Memory Hierarchy, Dynamic Program Behavior and Data Center Modeling and Optimization.

**Vasos Vassiliou** is an Associate Professor at the Computer Science Department of the University of Cyprus. He is the co-Director of the Networks Research Laboratory and the Group Leader of the Smart Networked Systems Research Group of the CYENS Research Center, situated in Nicosia, Cyprus. He held positions as a Lecturer and Visiting Lecturer at the Computer Science Department of the University of Cyprus (2011-2019, 2005-2011, 2004-2005 respectively) and as an Assistant Professor at the Computer Science Department of Intercollege (Sep. 2002-Jan. 2004). Dr. Vassiliou holds a Ph.D. from the Georgia Institute of Technology, Atlanta, Georgia, USA with specialization in Telecommunications and Mobile Network protocols. He is a graduate of the Higher Technical Institute (HND 1993) and the University of South Florida (B.Sc. 1997) both in Electrical Engineering. His graduate studies were sponsored by a partial CASP scholarship from the Cyprus Fulbright Commission at the Georgia Institute of Technology (M.Sc., 1999 and Ph.D., 2002 in Electrical and Computer Engineering). He has published several articles in International Conferences and Journals and has co-authored a number of book chapters. His research interests include Next Generation Network Architectures, Mobile Protocols, Mobile Networks, Wireless Communications, QoS, and Traffic Engineering for computer and telecommunication networks. He is a Senior Member of IEEE and ACM and participates as a reviewer and Technical Program Committee member in several international conferences.

**Haris Volos**, Assistant professor. Undergraduate studies at the National Technical University of Athens (Dipl.-Ing. in Electrical and Computer Engineering, 2005). Graduate studies at the University of Wisconsin-Madison (M.Sc., 2007 and Ph.D., 2012, both in Computer Sciences). He has worked as a researcher at Hewlett Packard Labs, Palo Alto (2013-2018) and as a software engineer at Google, Mountain View (2018-2019). His research focuses on computer architecture and systems, with a

particular emphasis on the interaction between hardware architecture and systems software, data-centric computing, and heterogeneous memory systems. He has multiple publications in all major computer architecture conferences and journals, such as IEEE/ACM ISCA, IEEE/ACM MICRO, ACM ASPLOS, IEEE CAL, and ACM TACO.

**Demetris Zeinalipour**, Professor. Basic studies at the University of Cyprus, Cyprus (Bachelor of Computer Science, 2000). Postgraduate studies at the University of California, Riverside, CA, USA (MSc, 2003 and PhD, 2005, in Computer Science and Computer Engineering). He has worked at the University of California (Teaching and Research Staff, 2000-2005), the University of Cyprus (Visiting Lecturer, 2005-2007), the Open University of Cyprus (Lecturer, 2007-2009), and the University of Cyprus at all academic ranks (2009-present). His research interests include data management in computing systems and computer networks, particularly Mobile, Sensor and Spatio-Temporal Data Management; Big Data Management in Parallel and Distributed Architectures; Network, Blockchain and Telco Data Management; Crowd, Web 2.0 and Indoor Data Management; Data Privacy Management; Data Management for Sustainability.

# ANNEX A: Rules for Diploma Projects

## 1. GENERAL

A *Diploma Project* is prepared by a final-year student, usually during the seventh and eighth semesters of study in accordance with the Programme of Studies of the Department. The part of Diploma Project prepared in the first semester is called *Diploma Project I* and the part corresponding to the second semester *Diploma Project II*. These two parts are prepared and completed in *CS400 – Diploma Project I* and *CS401 – Diploma Project II*, respectively.

A Diploma Project corresponds to seventeen and a half (17.5) *ECTS credits*, which are credited to the student upon successful completion.

The student registers for the Diploma Project with the approval of the Academic Advisor.

The Department Council appoints a member of the Faculty of the Department, who ensures and coordinates the entire process of developing and assessing the Diploma Projects. This Faculty member is known as the *Diploma Projects Coordinator*.

All the forms and guides mentioned in this section are digitally available on the website of the Department. The Department maintains a *Digital Library for the Diploma Projects* for archiving purposes.

## 2. SUBSTANCE, FORM AND EVALUATION CRITERIA

### 2.1 Substance

Each *Diploma Project* must contain sufficient information that reflects the student's initiative, independent study and productivity (originality, in the broad sense).

The Diploma Project may be of theoretical or practical nature or a combination of both. A Diploma Project may include, for example, an application of existing techniques, extension of known methods in theory, software, hardware or applications areas, development of a prototype system, addressing theoretical problems, a survey review or study of a theoretical or practical area, etc.

### 2.2 Form

A *Diploma Project* must be a comprehensive document structured in chapters and must follow the rules of the technical guidance report called *Standards for Preparation the Diploma Project*.

The Diploma Project should include an introduction to the subject, an analysis of the importance of the project, a description of the related work, a review the work in the area of the topic, a description of the methodology used, listing, classification and evaluation of the results of the work and finally conclusions and suggestions for possible extension of the work.

*Diploma Projects* where software was created and/or used should also include the code of the software in a specific Annex, a description and analysis of the software a separate section, and instructions for the use of the software. The code of such software may not be used as the Thesis document.

### 2.3 Evaluation Criteria

The main criteria for the evaluation of the *Diploma Project* are the following:

- [a] Quality of work (e.g. accuracy and completeness of analysis, appropriateness of methodology, validity of theoretical results, software quality, implementation, consistency of material presented and association of ideas).

- [b] Degree of the objectives achieved of the Diploma Project.
- [c] Degree of understanding by students of the area of the Diploma Project topic.
- [d] Quality of the written language of Diploma Project (e.g. structure and organization, clarity, ease of reading and understanding).
- [e] Quality of Presentation of the Diploma Project. (e.g. oral speech, the adequacy and suitability of multimedia used (such as slides), proper utilization of the allowed time of presentation, and most importantly to point the contribution of Diploma Project through the presentation).

### 3. PROCEDURE

#### 3.1 Preparation

##### 3.1.1 Submission and Announcement of Topics

Each member of the Faculty submits to the Department, end of March of each year, a number of Diploma Project topics greater than the ratio of the number of students per Faculty member. Each Faculty member reserves the right not to supervise a number of Diploma Projects larger than the nearest integer above the ratio of students per Faculty member.

Each topic has a title. It is expected that a brief description of each topic is made available to the students by the proposing faculty. (It is also expected that the faculty members will update and renew appropriately the description of the projects on their personal websites).

The *Diploma Projects Coordinator* establishes a list of the submitted Diploma Project topics and announces the Diploma Project list to the faculty members of the Department and to the final-year students approximately in middle of April.

##### 3.1.2 Choice of Topic

The successful completion of at least 156 ECTS is a prerequisite for the assignment of a Diploma Project.

Each student should choose a Diploma Project topic. To this end, the students shall discuss with the corresponding faculty members that have offered the topics, in meetings during a specified period of time. Throughout the process of submission of preferred topics, the students obtain the agreement of the faculty member for the supervision of the corresponding project. This Faculty member will be the *Diploma Project Supervisor* for that student.

With the selected choice of the topics, the student completes a special *Registration Form*, available electronically on the website of the Department which includes the title, description, and any specialized software / hardware or other resources necessary for the preparation of the Diploma Project. The *Registration Form* is signed by the Faculty member, the Academic advisor of the student and it is deposited to the Department no later than the period of registration of students for the semester the student prepares the *first part of the Diploma Project*.

In case of not being able to select topic, the student returns the Registration Form to the Department, signed by him/her and the Academic Advisor. In this case, with the filing of the form, the *Diploma Project Coordinator* contacts the Faculty members of the Department, to ensure that every student who submitted the registration form without topic to select a topic before the end of the submission period. In case this is not possible, a special meeting of the Department Council is called to assure that each student is given a topic. The student must choose the topic selected by the Council meeting

Concurrently with the submission of the registration form, the student must register for the CS 400 course.

### **3.1.3 Change of Subject**

Changing the topic of the Diploma Project (with the same or another supervisor) is possible within the first three (3) weeks of the semester the student enrolled for the *first part of the Diploma Project* (Diploma Project I). To this end, using the same procedure that was followed for the submission of the original Diploma Project topic, the student fills in the *Special Registration Form* to the Department in order for the *coordinator of Diploma Projects* to approve it.

## **3.2 Preparation**

### **3.2.1 Supervision**

The supervision of a student who prepares a Diploma Project is the responsibility of the Diploma Project Supervisor. The monitoring and controlling of the Diploma Project progress is done through regular meetings between the student and the supervisor. In some cases, a co-supervisor may be indicated. The supervisor must be a PhD holder and may either come from the Department of Computer Science or from an institute or organization outside the Department.

### **3.2.2 Interim Evaluation**

During the examination period of the first semester in which the student enrolled in the Diploma Project, the student submits to the *Supervisor* a brief Progress Report. Following the submission of the Progress Report, the Supervisor shall submit a written assessment of the progress of the student to the Department, which is also sent to the student. The possible grades for this assessment are Success or Failure. The grade of the Diploma Project I is the student's grade for the course CS400 which has been enrolled for the current semester.

In the second semester the student can enrol in the Diploma Project II, CS401, only if the student has succeeded with CS400 course. In case of a Failure grade results in the student enrolling again in the first part of the Diploma Project (Diploma Project I: CS400) in the same or different topic (with the same or different supervisor).

## **3.3 Evaluation**

The final evaluation of the Diploma Project takes place towards the end of the semester where the student is registered for *Diploma Project II*.

### **3.3.1 Second Assessor**

The Diploma Project is assessed by the *supervisor*, and the co-supervisor (if assigned), together with another member of the Faculty, known as the *Second Assessor*. The Diploma Project Coordinator with the collaboration of the Supervisors, publishes a *list of Assessors and Program of Diploma Project presentations*.

In special cases, the Coordinator of the Diploma Project may approve as a *Second Assessor* a Visiting Faculty Member of the Department or a Member of the Special Teaching Staff of the Department or a Faculty member from other Department of the University or/and other University in Cyprus or abroad. The approval may be justified in cases of close affinity of the Second Assessor Diploma on the subject of the Diploma Project.

### **3.3.2 Presentations**

The Diploma Project Coordinator publishes the *Program of Diploma Project presentations* for three days during the week immediately following the examination period. The Program of Diploma Project presentations must be made publicly available to all faculty members of the department and the presentations are open to the public.

Each student who is expected to complete his/her Diploma Project II based on the Academic Advisor judgment must be listed in the *Diploma Project Presentation list*. These students present their Diploma Project in public in front of their *Academic Advisor* and their *Second Assessor*.

The students who have been excluded from the *Program of the Presentations* of the Diploma Project receive a grade equal to *CD (Continuation of Diploma Project)* and must continue working on the same Diploma Project so in the immediately following semester to complete their Diploma with success. Therefore, these students must register for the same course, *CS401*

### **3.3.3 Grading**

After the presentation the *Supervisor*, in consultation with the *Second Assessor*, submits the grade for the Diploma Project with written justification comments, in a specific *Assessment Form*, and in accordance with the procedure for filing course grades. The Assessment Form of the Diploma Project contains different *evaluation criteria* that need to be filled with numerical grades.

A *Failure* grade results in the student enrolling again in the first part of the Diploma Project (*Diploma Project I*) in a different topic with a different *supervisor*. A *Success* grade in Diploma Project receives a numerical grade according to the Rules Studies.

The grade of the Diploma Project is the grade for the courses *CS401* which is enrolled.

The *Coordinator of the Diploma Project* handles the cases where there is a disagreement on the grade for the Diploma Project between the *Supervisor* and the *second assessor*.

### **3.3.4 Submit in Digital Format**

Within ten days after the presentation and evaluation of the Diploma Project, the student shall submit to the Department its Diploma project in digital form. Failure to submit timely may result in delay of the student's graduation which may cause the graduation of the student impossible in the current semester.

### **3.3.5 Final Grade Submission**

The coordinator of the Diploma Project submits the student's grade for the Diploma Project to the Academic Affairs and Student Welfare in time.

*These regulations were approved at the 106<sup>th</sup> meeting of the Council of the Department of Computer Science, on 6 December 2010.*

## **ANNEX B: Rules for Graduate Studies**

The Department of Computer Science offers *Graduate Degree Programmes* that lead to *Master in in Computer Science* and *PhD in Computer Science*. The programmes are published at the Department's Guidebook. The *Graduate Programme Committee* coordinates the programmes based on the General Rules of the University of Cyprus.

### **[1] Admission Requirements**

- I.
  - a. The department announces once a year the maximum number of graduate students it can accept for the upcoming academic year starting in September. Applications arrived after the submission deadline is accepted only in case the number of graduate students is not met
  - b. The applicants must submit an application, a CV, official transcripts from all colleges/universities attended, a personal statement about his/her goals and interests, and two letters of recommendations (possibly by faculty members of their college or university, mailed directly to the department). It is not necessary for the applicant to hold an undergrad or college degree at the time he submits his application. However, the applicant must get / have a degree in Computer Science or related field of study before he joins the graduate programme. The average grade (GPA) of the applicant should be 6.5/10 (or equivalent to that) and the degree should be from an accredited college / university (as it is defined by the General Rules of University of Cyprus).
  - c. Applications are examined by the Graduate Programme Committee. The Graduate Programme Committee holds the right, based on its judgment, to call the applicant for personal interview or ask for more information. The Graduate Programme Committee writes an evaluation and admission report and submits it to the Department Council for approval. The Department Council holds the right to accept fewer students than it had announced. The department submits its report to the school it belongs to.
- II. The department assigns an Academic Advisor to every new graduate student. To a PhD student the department assigns a Research Advisor. The Research Advisor is assigned by the Department Council, after suggestion of the Graduate Programme Committee, and an agreement between the student and a faculty member. The Research Advisor watches over the research or any other work of the student and provides any necessary guidance.
- III. For students whose undergraduate degree was not related to Computer Science there is a possibility to be asked to take some undergraduate courses offered by the department. The student must pass these undergraduate courses in order to continue the graduate programme.

### **[2] Master Degree**

- I. To get a Master Degree in Computer Science, each student must complete successfully courses (a total of 60 ECTS units) from the Graduate Programme and complete a Master Thesis of 30 ECTS units, under the supervision of a Research Advisor. The assignment of a Research Advisor is based on the rules of the University Senate and is done before the student submits the Thesis Proposal. The student can transfer up to 15 ECTS units from courses completed successfully from similar graduate programmes. Theses completed at other graduate schools are not transferable

- II. The Thesis must deal with a research topic or a technical issue. It must be of some original contribution or show a thorough and clear understanding of some special topic. Full-time students normally complete the thesis in 6 months study time. When the student has completed successfully courses totalling to 30 ECTS units, he can submit a Form (signed also by his Research Advisor) stating his Master Thesis topic.
- III. The Master Thesis is submitted at the department and defended within the time period decided by the Department Council.
  - a. When a Master Thesis is submitted the Chair of the Department appoints a Thesis Examination Committee of three members. The head of the committee is the student's Research Advisor. It is possible a member of this committee to be a faculty member of other department or to not be a faculty member but to hold a PhD Degree or have a reputation in the field of study. The Committee can have also external members who do not hold a faculty position. However, the membership must be approved by the Department Council and the external member must hold a PhD Degree or must have a reputation in the field of study.
  - b. The Master Thesis is defended in a presentation before the Thesis Examination Committee. The head of the Thesis Examination Committee is responsible for the procedures that must be followed during the defence.
  - c. The Thesis Examination Committee can accept (even with conditions) or reject a Master Thesis. The Thesis Examination Committee writes its decision in a Master Thesis Evaluation Form and submits it to the Department Council for approval. When the Thesis is accepted the department informs the Student Affairs Office for the graduation procedure of the student. When the thesis is rejected, the student can follow the suggestions of the committee and resubmit it for the first and the last time. The submission and thesis defence procedure is followed from the beginning.
- IV. The minimum graduation time to get a Master Degree is three semesters and the maximum graduation time is eight semesters.

### [3] **PhD Degree**

- I. The basic requirements to get a PhD Degree are:
  1. (a) Successful completion of at least 60 ECTS units of graduate level courses. A student with a Master or equivalent degree is partially or fully exempt from this requirement.  
  
(b) Successful completion of the Comprehensive Examination which must be taken no later than the fifth semester of his/her studies. The student submits to the department a request to take the Comprehensive Examination.
  2. The Department assigns a Research Advisor to the student that has completed the ECTS units mentioned above.
  3. The structure and the subjects of the Comprehensive Examination are decided by the Department. The Comprehensive Examination is general and attempts to measure the student ability to complete the degree. The topics that are examined are Theory, Software, Hardware, and Applications.
  4. The Comprehensive Examination is presented before the Comprehensive Examination committee which is elected by the Postgraduate Committee of the department after it has been requested by the student's academic research supervisor. The formation of the Comprehensive Examination committee must be approved by the Department

committee. The academic research advisor of the student is the chair of the Comprehensive Examination committee.

5. The Comprehensive Examination consists of three phases. The student should obtain a passing grade in all of them to ensure success in the examination. The three phases are the following:
  - i. Submit to the Comprehensive Examination Committee a written literature review on the student's Research area. The review should have the length and the quality of a published survey article and must demonstrate the student's adequate knowledge and understanding of the topic under investigation and the open problems. To prepare the review, the Research Supervisor may give the student an indicative bibliography. The review is graded by the members of the Comprehensive Examination Committee with a Pass / Fail. When a review receives a passing grade, the student may proceed to an oral examination.
  - ii. The student should make an oral presentation of his/her survey to the committee. The presentation lasts 50-60 minutes, including questions addressed by the committee and the audience. The presentation is open to the public and should be announced to the members of the department and the University.
  - iii. The oral examination of the student, which is made by the members of committee, is closed to the public. The purpose of the examination is to further investigate whether the candidate has the skills and capability to conduct Doctoral-level research work in Computer
6. The Comprehensive Examination Committee submits for approval to the department a list of students that have succeeded in the examination.
7. A student that fails to pass the first comprehensive examination must take the examination the next time it is offered. A second fail of the student in the Comprehensive Examination disqualifies the student from the PhD programme of the department.
8. Second failure in the comprehensive examination implies exclusion from the PhD candidature in the Department.

## II.

1. The Post Graduate Programme Committee, taking into consideration the suggestions of the Research Supervisor, appoints a three-member Research Committee for the PhD candidate. The chair of the Research Committee is the Research Supervisor of a student. At most, one member of the Research Committee may be a faculty member of another department.
2. The student is allowed to change his/her Research Supervisor. He/she must submit for approval by the Department Board a request explaining in detail the reasons he/she wishes to change supervisor.

## III.

1. When the student has passed the Comprehensive Examination he/she submits a written Doctoral Thesis Proposal to his/her Research Committee. The proposal is also presented orally before the Committee members.
  - i. The Doctoral Thesis Proposal must have the following structure:
    1. Introduction
    2. Motivation

3. Problem Statement – Hypothesis
4. Approach
5. Roadmap
6. Related Work
7. Preliminary Work
8. Work to be done
9. Timeline
10. Future work

- ii. b. The Research Committee must examine the Doctoral Thesis Proposal before the end of the next semester. The proposal might be accepted or recommended for resubmission. The final acceptance of the proposal must be given before the start of the seventh year of the student study time. Otherwise, the student case (whether to continue or to terminate his PhD candidacy) is discussed at the Department Council.

IV.

1. Every Doctoral student must submit to the Research Supervisor an annual progress report. This report must be submitted before the end of each academic year.
2. The annual progress report is graded by the Coordinator of the Graduate Studies and the Research advisor of the student with a Pass/Fail. The grade is submitted to Student Affairs and is marked in the student's records to which level the student progress in his/her studies (research, writing)

V.

1. Every PhD candidate completes a Dissertation which must be an original and important contribution to the field of study.
2. The written format of the Dissertation is described in the Department's printed forms. The Dissertation can be submitted no sooner than four semesters before student's admission and completing successfully his Comprehensive Examination.

VI.

The student submits to the department six (6) copies of the Dissertation and a request (signed also by his Research Advisor) for the formation of a Dissertation Examination Committee. After this procedure, the graduate Programme Committee, with suggestions from the Research Advisor, appoints an Examination Committee (based on the rules of University's Senate). The Graduate Programme Committee assures a copy of the dissertation is sent to the members of the Examination Committee. The dissertation is defended by the candidate before the Examination Committee. Note that the three faculty members of the Department are usually the three members of the Research Committee

VII.

The Chair of the Examination Committee decides the date of the Dissertation Defence. The Defence must take place within three months from the Dissertation submission. All members of the Examination Committee must be present at the Defence. The defence procedure is based on the rules of the University's Senate.

1. The approval of at least four members of the Examination Committee is required to award the degree. In this case, the Examination Committee has the right to request changes or additions, which considers as necessary. The procedure for monitoring the changes or additions is decided by the Examination Committee and is written clearly in the Evaluation Report.
2. The Examination Committee hands over to the Department Chair a written Evaluation Report of the Dissertation and of the Doctoral Candidate in general, together with its recommendations, in a printed form. The Chair verifies that all the rules and procedures have been correctly followed and hands in the report to the Senate.

VIII. A maximum of eight (8) academic years is allowed for the completion of the Doctoral Programme.

Approved by the Senate, 149<sup>th</sup> Meeting, 22/5/2002

## ANNEX C: Specifications for Preparation of the Master Thesis

**Abstract.** An abstract is *required*. The body of the abstract may not exceed 400 words in length. Please see the sample abstract page for the format.

**Minimum Margins.** The minimum acceptable margins for all pages of the thesis and the abstract are 3.8 centimeters (1.5 inches) on left and 2.5 centimeters (1 inch) on the top, bottom, and right.

**Paper Requirement.** All pages of the thesis must be printed on 21 x 29.7 centimeter (8.27 x 11.69 inch) white paper. This is the regular A4 paper format.

**Font and Point Size.** Recommended fonts include Arial, Times New Roman, and Helvetica with a point size of either 11 or 12 (preferably 11).

**Printing.** Either laser printing or photocopying of high quality is acceptable. Inkjet printing is *not* acceptable as it is water soluble. Only *one side* of the paper is to be used for printing. Printing should be consistently clear and dark.

**Spacing.** The text of the thesis should be *double spaced*. Long quotations, footnotes, appendices, and references may be single spaced.

**Color.** It is recommended that you *do not* rely on color to convey information (e.g., use symbols or labels rather than colors to differentiate the lines on a graph, or use stripes and cross-hatching instead of colors to distinguish areas on a map).

**Photographs and Graphics.** Photographs and graphics in the dissertation should be printed or photocopied directly on the paper as high quality black and white images. Scanned images must print clearly. If color must be used, *only* color laser or color photocopy printing is acceptable.

**Use of materials copyrighted by others.** IMPORTANT: Any material included that goes beyond “fair use” requires written permission of the copyright owner. It may be useful to include these in the thesis as an appendix.

**Pagination.** Preliminary pages (i.e., the approval page, acknowledgments, table of contents, and the like) are to be numbered consecutively using lower case *Roman numerals*. All pages of the text, appendices (if any), and references must be numbered consecutively using *Arabic numerals*. The page number of the first page of each Chapter must be centered on the bottom of the page. In all other pages, the page number must be placed on the top of the page, aligned to the right.

**Landscape pages.** The top of a landscape page should be at the left margin and the bottom at the right margin. The page number is to be in the same relative position as on the portrait pages. An easy way to apply page numbers to landscape pages is to run them through the printer twice – once for the text, table, or figure (landscape orientation) and once for the page number (portrait orientation).

**Sequence of the main components of the dissertation.** The appropriate order of the major sections of the thesis follows: the title page, the abstract, the copyright page (if needed), the approval page, acknowledgments, table of contents, the text, bibliography/references, and appendices (if any).

**Bibliography/References.** The ACM (<http://www.acm.org/pubs>) or the IEEE (<http://standards.ieee.org/resources>) reference style should be followed.

**Footnotes and Endnotes.** No specific rules. The format that is prescribed by your advisory committee should be followed.

\* \* \*

It is recommended that you use your full legal name on the abstract, the title page, the copyright page (if appropriate), and on the approval page. Make certain that your name appears exactly the same way in all places.

Take a moment to check that all pages in all copies of your thesis are accounted for and in the proper order before submitting final copies to the Departmental Secretary. *One hardcopy* and *an electronic version* (in pdf format) of the thesis should be submitted to the Departmental Secretary.

\* \* \*

For students that plan to write their thesis using the Latex authoring language, they can download the style file “ucythesis.sty” that creates the required Master thesis format automatically from <http://www.cs.ucy.ac.cy/~chrysis/master-specs.html>

## ANNEX D: Specifications for Preparation of the Doctoral Dissertation

**Abstract.** An abstract is *required*. The body of the abstract may not exceed 400 words in length. See the sample abstract page for the format.

**Minimum Margins.** The minimum acceptable margins for all pages of the dissertation and the abstract are 3.8 centimeters (1.5 inches) on left and 2.5 centimeters (1 inch) on the top, bottom, and right.

**Paper Requirement.** All pages of the dissertation must be printed on 21 x 29.7 centimeters (8.27 x 11.69 inch) white paper that is *at least 25% cotton and 20lbs. weight*. This is a special A4 paper format that can be obtained at bookstores.

**Font and Point Size.** Recommended fonts include Arial, Times New Roman, and Helvetica with a point size of either 11 or 12 (preferably 11).

**Printing.** Either laser printing or photocopying of high quality is acceptable. Inkjet printing is *not* acceptable as it is water soluble. Only one side of the paper is to be used for printing. Printing should be consistently clear and dark.

**Spacing.** The text of the dissertation should be *double spaced*. Long quotations, footnotes, appendices, and references may be single spaced.

**Color.** It is recommended that you *do not* rely on color to convey information (e.g., use symbols or labels rather than colors to differentiate the lines on a graph, or use stripes and cross-hatching instead of colors to distinguish areas on a map).

**Photographs and Graphics.** Photographs and graphics in the dissertation should be printed or photocopied directly on the paper as high quality black and white images. Scanned images must print clearly. If color must be used, only color laser or color photocopy printing is acceptable.

**Use of materials copyrighted by others.** IMPORTANT: Any material included that goes beyond “fair use” requires written permission of the copyright owner. It may be useful to include these in the dissertation as an appendix.

**Pagination.** Preliminary pages (i.e., the approval page, acknowledgments, table of contents, and the like) are to be numbered consecutively using lower case *Roman numerals*. All pages of the text, appendices (if any), and references must be numbered consecutively using *Arabic numerals*. The page number of the first page of each Chapter must be centered on the bottom of the page. In all other pages, the page number must be placed on the top of the page, aligned to the right.

**Landscape pages.** The top of a landscape page should be at the left margin and the bottom at the right margin. The page number is to be in the same relative position as on the portrait pages. An easy way to apply page numbers to landscape pages is to run them through the printer twice – once for the text, table, or figure (landscape orientation) and once for the page number (portrait orientation).

**Sequence of the main components of the dissertation.** The appropriate order of the major sections of the dissertation follows: the abstract, the title page, the copyright page (if needed), the approval page, acknowledgments, table of contents, the text, the bibliography/references, and appendices (if any).

**Bibliography/References.** The ACM (<http://www.acm.org/pubs>) or the IEEE (<http://standards.ieee.org/resources>) reference style should be followed.

**Footnotes and Endnotes.** No specific rules. The format that is prescribed by your advisory committee should be followed.

\* \* \*

It is recommended that you use your full legal name on the abstract, the title page, the copyright page (if appropriate), and on the approval page. Make certain that your name appears exactly the same way in all places.

Take a moment to check that all pages in all copies of your dissertation are accounted for and in the proper order before submitting final copies to the Departmental Secretary. *Two hardcopies* and *an electronic version* (in pdf format) of the dissertation should be submitted to the Departmental Secretary.

\* \* \*

For students that plan to write their dissertation using Latex or MS Word authoring programs, they can download samples from <http://www.cs.ucy.ac.cy/~chryssis/phd-specs.html>.