Φροντιστήριο

Ουρές, Στοίβες & Λίστες

Να γραφεί η αναδρομική εξίσωση του παρακάτω αλγορίθμου

```
FIBONACCI (n)
1.  u = 1
2.  f = 1
3.  if n ≤ 1 then
4.        return 1
5.  else
6.        return FIBONACCI(n − 1) + FIBONACCI(n − 2)
7.  end if
```

$$T(n) = \begin{cases} 1 & \text{if } n = 0,\ n = 1, \\ T(n-1) + T(n-2) & \text{if } n > 1. \end{cases}$$

Άσκηση 1

Γράψετε μεθόδους έτσι ώστε να υλοποιήσετε μια ουρά, δομή και συναρτήσεις isEmpty(), size(), enqueue, dequeue(), χρησιμοποιώντας δύο στοίβες.

Θεωρείστε ότι η στοίβα έχει υλοποιημένες τις μεθόδους isEmpty(), size(), push(), pop(), top(). Ποιος είναι ο χρόνος εκτέλεσης της κάθε μεθόδου;

Έστω ότι οι δύο στοίβες ονομάζονται in και out.

```
isEmpty()                                                    O(1)

   return    in.isEmpty() && out.isEmpty()

size()                                                       O(1)
   return    in.size() && out.size()

enqueue(x)                                                   O(1)
   in.push(x)
                                                             O(n)
dequeue(x)
     if(out.isEmpty()) {
         while(!in.isEmpty()){
              var = in.top();
              in.pop();
              out.push(var);
         }
     }
     if(out.isEmpty()) {
         return null;
     else {
          var = out.top();
          out.pop();
          return var;
     }
```

Άσκηση 2

Να γράψετε μία μη-αναδρομική μέθοδο η οποία να εξετάζει αν τα δεδομένα μιας απλά συνδεδεμένης λίστας είναι ταξινομημένα σε αύξουσα σειρά.

Έστω ότι η λίστα είναι υλοποιημένη σαν μία κλάση με αναφορά στον πρώτο κόμβο και κάθε κόμβος Node έχει δεδομένα (.data) και αναφορά στον επόμενο κόμβο (.next).

```
boolean isSorted()
        boolean result = true;
        if(!isEmpty()) {
                Node n = head;
                while ( n.next!=null ) {
                        if( n.data > n.next.data) {
                                result = false;
                                break;
                        }
                        n = n.next;
                }
        }
        return result;
```

Άσκηση 3

Δείξετε πως μπορούμε να υλοποιήσουμε τρεις στοίβες έχοντας μόνο ένα πίνακα. Μήνυμα ότι η στοίβα είναι γεμάτη θα πρέπει να παίρνουμε μόνο στην περίπτωση που ο πίνακας έχει γεμίσει. Ποιος είναι ο χρόνος εκτέλεσης κάθε μεθόδου;

Τρεις στοίβες μπορούν να υλοποιηθούν σε ένα πίνακα, με το να έχουμε την πρώτη να ξεκινά από την αρχή του πίνακα και να προσθέτει προς το τέλος, τη δεύτερη να ξεκινά από το τέλος και να προσθέτει προς την αρχή και την τρίτη κάπου στο μέσο και να προσθέτει προς το τέλος.

Εάν η τρίτη στοίβα συγκρουστεί με οποιαδήποτε από τις άλλες θα πρέπει να μεταφερθεί.

Μια στρατηγική είναι να την μετακινήσουμε έτσι ώστε το κέντρο της (κατά το χρόνο της μετακίνησης) να είναι στο μέσο των κορυφών των δύο άλλων στοιβών.

Η διαγραφή ενός στοιχείου (pop) από μια στοίβα παίρνει χρόνο O(1) ενώ η εισαγωγή ενός στοιχείου (push) σε μια στοίβα παίρνει χρόνο O(n).