Εργασία 2 – Σκελετοί Λύσεων

Άσκηση 1

Επεξεργαζόμαστε μια-μια τις μέρες ως εξής: Τοποθετούμε την τιμή της μετοχής για την πρώτη ημέρα σε μία στοίβα και θέτουμε B[1] = 1. Για κάθε επόμενη μέρα ψάχνουμε στη στοίβα και κάνουμε pop όλες τις ημέρες με μικρότερη τιμή μετοχής. Η ζητούμενη τιμή του B είναι το άθροισμα των τιμών B αυτών των ημερών + 1.
Τοποθετούμε στη στοίβα την ημέρα αυτή.
Υποθέτουμε την ύπαρξη υλοποίησης στοίβας με πράξεις MakeEmpty, IsEmpty, Push, Pop, Top. Ο αλγόριθμος έχει ως εξής:

```
i=1;
MakeEmpty(S);
B[1] = 1;
Push(S, 1);
for (i = 2; i <= n ; i++)
     B[i] = 1;
     while (!IsEmpty(S))
          d = Top(S);
          if (A[i] >= A[d])
               Pop (S);
               B[i] =+ B[d];
          else
               break;

     Push(S, i);
```

Ο χρόνος εκτέλεσης της διαδικασίας δίνεται από το άθροισμα:

$$\sum_{i=2}^{n}(t_i+1)$$

όπου t_i είναι ο αριθμός των Pop που εκτελούνται στο βρόχο while κατά την i-οστή εκτέλεσης του βρόχου for. Αφού κάθε στοιχείο εισάγεται και εξάγεται από τη στοίβα ακριβώς μια φορά έχουμε επίσης ότι

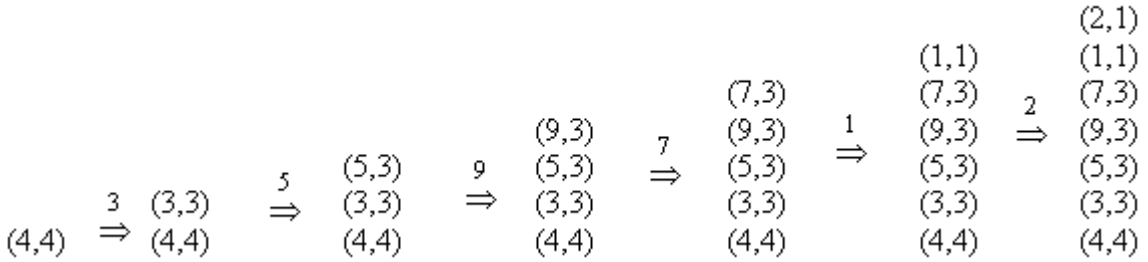
$$\sum_{i=2}^{n} t_i \le n-1$$

Επομένως ο χρόνος εκτέλεσης της διαδικασίας είναι της τάξης Θ(n)

Άσκηση 2

Σε κάθε θέση της στοίβας αποθηκεύουμε ζεύγη στοιχείων όπου το πρώτο μέλος κάθε ζεύγους αντιστοιχεί στο στοιχείο που εισήχθηκε, και το δεύτερο μέλος του ζεύγους στο μέχρι στιγμής ελάχιστο στοιχείο της στοίβας.

Για παράδειγμα οι διαδοχικές εισαγωγές 4,3,5,9,7,1,2 έχουν ως αποτέλεσμα τις πιο κάτω καταστάσεις



Κατά συνέπεια, ανά πάσα στιγμή μπορούμε να βρούμε και να επιστρέψουμε το ελάχιστο στοιχείο της στοίβας σε χρόνο σταθερό (επιστρέφοντας το δεύτερο μέλος του ζεύγους που βρίσκεται στον κόμβο κορυφής της στοίβας).

Άσκηση 3

Οι τρεις ουρές μπορούν να υλοποιηθούν σε ένα πίνακα με το να έχουμε την $1^η$ να μεγαλώνει από την αρχή του πίνακα προς το τέλος, τη $2^η$ από το τέλος προς την αρχή και την $3^η$ από κάποιο στο μέσο του πίνακα προς μια τυχαία κατεύθυνση. Εάν η $3^η$ συγκρουστεί με την $1^η$, θα πρέπει να μετακινηθεί είτε η πρώτη προς την αρχή του πίνακα (δεδομένου ότι κάποια στοιχεία έχουν διαγραφεί και έχουν αδειάσει θέσεις) είτε η $3^η$ στο μέσο των κεφαλών των δύο άλλων ουρών. Το ποια από τις δύο είναι καλύτερο να μετακινηθεί μπορούμε να το αποφασίσουμε βάση των κενών θέσεων που θα δημιουργηθούν. Αντίστοιχα, ενεργούμε στην περίπτωση που η $3^η$ ουρά συγκρουστεί με τη $2^η$.
Δεδομένης της μετακίνησης των στοιχείων, υπάρχει περίπτωση να πρέπει να μετακινηθεί μια ουρά μεγέθους τάξεως n λόγω σύγκρουσης. Σύγκρουση θα έχουμε μόνο στην περίπτωση εισαγωγής ενός στοιχείου, άρα η εισαγωγή ενός στοιχείου σε μια ουρά από O(1) γίνεται O(n). Οι υπόλοιπες μέθοδοι της ουράς δεν επηρεάζονται και παραμένουν ως έχουν.

Άσκηση 4

a. Ξεκινούμε με 2 μετρητές, ο $1^{ος}$ στην αρχή του πίνακα A και ο $2^{ος}$ στο τέλος του. Ενόσω ο $1^{ος}$ μετρητής βρίσκεται σε 0 τον μετακινούμε προς τα δεξιά. Παρόμοια, ενόσω ο $2^{ος}$ μετρητής βρίσκεται σε 1 τον μετακινούμε προς τα αριστερά. Όταν ο $1^{ος}$ μετρητής φτάσει σε 1 και ο $2^{ος}$ σε 0 ανταλλάσουμε τα δύο αυτά στοιχεία.

   Συνεχίζουμε να μετακινούμε τους μετρητές και να ανταλλάσουμε στοιχεία όπως πιο πάνω μέχρι οι δύο μετρητές να συναντηθούν. Σε αυτό το σημείο ο πίνακας είναι ταξινομημένος.

   Ο χρόνος εκτέλεσης του αλγορίθμου είναι Θ(n) διότι επεξεργαζόμαστε κάθε στοιχείο μόνο μια φορά.

b. Έχοντας 3 διακριτά στοιχεία μπορούμε να ταξινομήσουμε τον πίνακα A με το να τρέξουμε τον αλγόριθμο του μέρους A δύο φορές. Την πρώτη φορά θα ανταλλάξουμε τα 0 που βρίσκει ο $2^{ος}$ μετρητής με τα 1 ή 2 που βρίσκει ο $1^{ος}$. Με αυτό τον τρόπο

μετακινούμε όλα τα 0 στην αρχή. Τη δεύτερη φορά θα ξεκινήσουμε από τη θέση που συναντήθηκαν οι δύο μετρητές και θα ανταλλάσουν τα 1 και 2 αντίστοιχα.

Ο χρόνος εκτέλεσης του αλγορίθμου είναι Θ(n) διότι τρέχουμε τον αλγόριθμο του μέρους Α δύο φορές, άρα ο χρόνος εκτέλεσης είναι 2n = Θ(n).

Άσκηση 5

Έστω ο πίνακας A[n] και ακέραιος k. Ο αλγόριθμος έχει ως εξής:

i. Ταξινομούμε τον πίνακα Α σε αύξουσα σειρά με τον αλγόριθμο MergeSort.
ii. Αρχικοποιούμε δύο μεταβλητές l και r να δείχνουν στην αρχή και το τέλος του πίνακα αντίστοιχα, και μια μεταβλητή sum με το άθροισμα των τιμών των δύο θέσεων, δηλαδή, θέτουμε l = 0, r = n-1 και sum = A[l] + A[r].
iii. Εφόσον οι l και r δεν έχουν διασταυρωθεί επαναλαμβάνουμε τα πιο κάτω βήματα:
    a. Αν sum = k επιστρέφουμε ΝΑΙ.
    b. Αν sum > k, προχωρούμε το δεξιό δείκτη προς τα αριστερά και υπολογίζουμε τη νέα τιμή του sum (η οποία με αυτό τον τρόπο θα μειωθεί). Δηλαδή, θέτουμε r = r – 1 και sum = A[l] + A[r] και επαναλαμβάνουμε το βήμα.
    c. Αν sum < k, προχωρούμε τον αριστερό δείκτη προς τα δεξιά και υπολογίζουμε τη νέα τιμή του sum (η οποία με αυτό τον τρόπο θα αυξηθεί).
    Δηλαδή, θέτουμε l = l + 1 και sum = A[l] + A[r] και επαναλαμβάνουμε το βήμα.
iv. Επιστρέφουμε ΟΧΙ.

Ο χρόνος εκτέλεσης της διαδικασίας είναι O(n lg n). (Το βήμα (i) απαιτεί χρόνο εκτέλεσης O(n lg n), τα βήματα (ii) και (iv) χρόνο O(1) και το βήμα (iii) χρόνο O(n)).