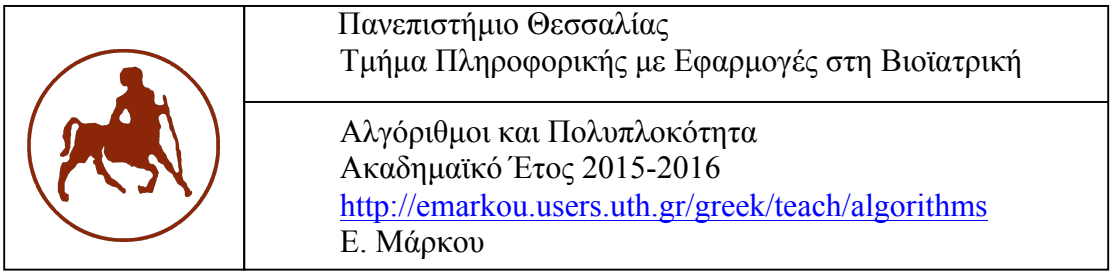

Πανεπιστήμιο Θεσσαλίας
Τμήμα Πληροφορικής με Εφαρμογές στη Βιοϊατρική

Αλγόριθμοι και Πολυπλοκότητα
Ακαδημαϊκό Έτος 2015-2016
http://emarkou.users.uth.gr/greek/teach/algorithms
Ε. Μάρκου

Σετ Ασκήσεων

Ημερομηνία Παράδοσης: Παρασκευή 20 Μαϊου 2016.

Τρόπος Παράδοσης: Χειρόγραφες ή εκτυπωμένες σελίδες που τις αφήνετε στη γραμματεία του τμήματος (κατά τις ώρες λειτουργίας της).

Άσκηση **1** **[20** μονάδες**]**

Θεωρήστε την ακόλουθη διαδικασία, όπου η swap(int a, int b) ανταλλάσσει την τιμή της μεταβλητής a με την τιμή της μεταβλητής b.

```
void MySort(table T[0..n-1]: array of integers)
{
        for m = 0 to n-2 do
        {
                j = m;
                for k = m+1 to n-1 do
                {
                        if T[k] < T[j] then j = k;
                }
                swap(T[m], T[j]);
        }
}
```

α. Παρουσιάστε σύντομη περιγραφή του τρόπου λειτουργίας της MySort(). Ποιο είναι το αποτέλεσμα της MySort();

β. Εκτελέστε με το χέρι τη MySort(T[0..5]) για την περίπτωση που T = [50, 20, 30, 10, 3, 15].

γ. Ποιά είναι η χρονική πολυπλοκότητα της MySort(); Βρείτε την τάξη στην οποία ανήκει (συμβολισμός Θ).

Άσκηση **2 [40** μονάδες**]**

Θεωρήστε τον ακόλουθο αναδρομικό αλγόριθμο, ο οποίος υπολογίζει το ελάχιστο στοιχείο ενός μη-ταξινομημένου πίνακα ακεραίων T[a..b] με $n = b-a+1 = 2^k$ ($k > 0$) στοιχεία, όπου a,b είναι θετικοί ακέραιοι που καθορίζουν τη θέση του πρώτου και του τελευταίου στοιχείου του πίνακα.

```
int FindMin(table T: array of int, int a, int b)
/* επιστρέφει το ελάχιστο στοιχείο του μη- ταξινομημένου πίνακα T */
{
        int middle;

        if (b-a == 0) return T[a];
        middle = κάτω ακέραιο μέρος{(a+b)/2};
        for j=a to middle do
                if (T[j] > T[j+middle]) then swap(T[j],T[j+middle]);
        min = FindMin(T, a, middle);
        return min;
}
```

α. Εκτελέστε με το χέρι την FindMin(T,1,8) για την περίπτωση που T= [8, 3, 25, 32, 15, 7, 20, 1]. Παρουσιάστε σύντομη περιγραφή του τρόπου λειτουργίας του αλγορίθμου.

β. Παρουσιάστε αναδρομική σχέση που να περιγράφει τη χρονική πολυπλοκότητα της FindMin(). Υπολογίστε την τάξη πολυπλοκότητας της FindMin().

γ. Δίνεται ο ακόλουθος αναδρομικός αλγόριθμος (FindMinMax()) που υπολογίζει και το ελάχιστο και το μέγιστο στοιχείο του πίνακα T. Μετά την εκτέλεση του αλγορίθμου το ελάχιστο στοιχείο του πίνακα βρίσκεται στη θέση T[a] ενώ το μέγιστο στοιχείο βρίσκεται στη θέση T[b].

```
void FindMinMax(table T, int a, int b)
{
        int middle;

        if (b-a==0) return;
        middle = κάτω ακέραιο μέρος{(a+b)/2};
        for j=a to middle do
                if (T[j] > T[j+middle-a+1]) then swap(T[j],T[j+middle-a+1]);
        FindMinMax(T, a, middle);
        FindMinMax(T, middle+1,b);
}
```

i. Εκτελέστε με το χέρι την FindMinMax(T,1,8) για την περίπτωση που T=[8,3,25,32,15,7,20,1]. Παρουσιάστε σύντομη περιγραφή του τρόπου λειτουργίας του αλγορίθμου.

ii. Παρουσιάστε αναδρομική σχέση T(n) που να περιγράφει τη χρονική πολυπλοκότητα της FindMinMax(). Τι τάξης είναι η πολυπλοκότητα της FindMinMax() και γιατί;

Άσκηση **3 [20** μονάδες**]**

α) Γιατί ο γνωστός άπληστος (Greedy) αλγόριθμος για το πρόβλημα του συνεχούς σακιδίου δεν οδηγεί αναγκαστικά σε βέλτιστη λύση αν εφαρμοστεί στο πρόβλημα του διακριτού σακιδίου (Discrete Knapsack); Δώστε ένα αντιπαράδειγμα.

β) Να εφαρμόσετε έναν αλγόριθμο δυναμικού προγραμματισμού για να λύσετε το πρόβλημα του διακριτού σακιδίου για πέντε αντικείμενα με βάρη (3, 2, 4, 5, 6) και αξίες (3, 2, 5, 6, 7) αντίστοιχα και μέγιστο βάρος σακιδίου 16.

Άσκηση **4 [20** μονάδες**]**

Έστω πίνακας θετικών ακέραιων A[1 … n] καθένας από τους οποίους έχει το πολύ k ψηφία (όπου το k είναι μια σταθερά). Να δείξετε ότι ο A μπορεί να ταξινομηθεί σε χρόνο O(n).