HY240: Δομές Δεδομένων
Εαρινό Εξάμηνο – Ακαδημαϊκό Έτος 2018-19
Παναγιώτα Φατούρου

1η Σειρά Ασκήσεων

Ημερομηνία Παράδοσης: Δευτέρα, 25 Φεβρουαρίου 2019
Τρόπος Παράδοσης: Οι ασκήσεις μπορούν να παραδοθούν είτε σε ηλεκτρονική μορφή χρησιμοποιώντας το πρόγραμμα turnin (δείτε οδηγίες στην ιστοσελίδα του μαθήματος), είτε στους βοηθούς του μαθήματος, τη Δευτέρα, 25 Φεβρουαρίου 2019, ώρα 13:00-14:00. Ασκήσεις που παραδίδονται μετά τις 14:00 της Δευτέρας, 25/2/2019 θεωρούνται εκπρόθεσμες. Εκπρόθεσμες ασκήσεις γίνονται δεκτές σε ηλεκτρονική μορφή (pdf) και η αποστολή τους πρέπει να γίνει χρησιμοποιώντας το πρόγραμμα turnin. Εναλλακτικά, εκπρόθεσμες ασκήσεις μπορούν να παραδοθούν στη διδάσκουσα.

Άσκηση 1 [20 μονάδες]

Έστω $T_1(n)$ το συνολικό πλήθος των φορών που εκτελείται η εντολή x = x + 1 από τη ρουτίνα FindMyCompexity1() και $T_2(n)$ το συνολικό πλήθος των φορών που εκτελείται η εντολή x = x + 1 από τη ρουτίνα FindMyComplexity2(). Υπολογίστε τα $T_1(n)$ και $T_2(n)$. Μελετήστε την τάξη (συμβολισμός O) των $T_1(n)$ και $T_2(n)$.

```
procedure FindMyComplexity1(integer n)
    for i = 1 to √n do
        for j = 1 to n do
            for k = 1 to √n – j +1 do x = x+1;
    }
```

```
procedure FindMyComplexity2(integer n)
        for (i=1; i <= n; i = i + 2)  {
            for (j = i; j <= n^2; j++)
                x = x+1;
        }
}
```

Άσκηση 2 [25 μονάδες]

α. Έστω f(n) και g(n) ασυμπτωτικά θετικές συναρτήσεις. Αποδείξτε τις ακόλουθες προτάσεις:

i. Αν f(n) = O(g(n)) και h(n) = O(g(n)), τότε $(c_1f+c_2h)(n) = O(g(n))$, για οποιαδήποτε σταθερές $c_1, c_2 \in R+$. [5M]

ii. $\max\{f(n),g(n)\} = \Theta(f(n)+g(n))$. [5M]

iii. Η f(n) είναι στο O(g(n)) αν και μόνο αν $g(n) = \Omega(f(n))$. [5M]

β. Μελετήστε την ασυμπτωτική πολυπλοκότητα (βάσει του συμβολισμού Θ) της συνάρτησης [10M]

$f(n) = 5n^3\log^2 n^3 + n^3\sqrt{n} - n^3 \log n - 3 \log^4 n$

Άσκηση 3 [20 μονάδες]

α. Επιλύστε τις ακόλουθες αναδρομικές σχέσεις με επαναληπτική αντικατάσταση.

i. $T(n) = 2T(n/2) + 5$ [5M]

ii. $T(n) = 2T(n-1) + 1$ [5M]

β. Για την αναδρομική σχέση του ερωτήματος α., σχεδιάστε το δένδρο αναδρομής. [5M]

γ. Για την αναδρομική σχέση του ερωτήματος β., επαληθεύστε, χρησιμοποιώντας μαθηματική επαγωγή, πως η λύση που βρήκατε με επαναληπτική αντικατάσταση είναι σωστή. [5M]

Άσκηση 4 [35 μονάδες]

Θεωρήστε τον ακόλουθο αναδρομικό αλγόριθμο, ο οποίος υπολογίζει το ελάχιστο στοιχείο ενός μη-ταξινομημένου πίνακα ακεραίων T[s...u] με n = u-s+1 = 2^k-1 (k > 0) στοιχεία, όπου s, u, u > s, είναι θετικοί ακέραιοι που καθορίζουν το πρώτο και το τελευταίο στοιχείο του πίνακα.

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

  if (u-s == 0) return T[s];
  middle = ⌊(s+u)/2⌋;
   for j = s to middle do  {
                if (T[j] > T[j+middle]) then
                        swap(T[j],T[j+middle]);
   }
   min = FindMin(T, s, middle);
   return min;
}
```

α. Ιχνηλατίστε την FindMin(T,1,8) για την περίπτωση που T= [10,2,20,30,17,8,19,1]. Παρουσιάστε σύντομη περιγραφή του τρόπου λειτουργίας του αλγορίθμου. [5M]

β. Παρουσιάστε αναδρομική σχέση που να περιγράφει τη χρονική πολυπλοκότητα της FindMin(). Τι τάξης είναι η πολυπλοκότητα της FindMin() και γιατί; [7M]

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[s] ενώ το μέγιστο στοιχείο βρίσκεται στη θέση T[u].

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

i. Ιχνηλατίστε την FindMinMax(T,1,8) για την περίπτωση που T= [10,2,20,30,17,8,19,1]. Παρουσιάστε σύντομη περιγραφή του τρόπου λειτουργίας του αλγορίθμου. [10M]

ii. Παρουσιάστε αναδρομική σχέση T(n) που να περιγράφει τη χρονική πολυπλοκότητα της FindMinMax(). Τι τάξης είναι η πολυπλοκότητα της FindMinMax() και γιατί; [13M]