ΥΠΟΥΡΓΕΙΟ ΕΘΝΙΚΗΣ ΠΑΙΔΕΙΑΣ ΚΑΙ ΘΡΗΣΚΕΥΜΑΤΩΝ
ΕΙΔΙΚΗ ΥΠΗΡΕΣΙΑ ΔΙΑΧΕΙΡΙΣΗΣ ΕΠΕΑΕΚ
ΕΥΡΩΠΑΪΚΗ ΕΝΩΣΗ
ΣΥΓΧΡΗΜΑΤΟΔΟΤΗΣΗ
ΕΥΡΩΠΑΪΚΟ ΚΟΙΝΩΝΙΚΟ ΤΑΜΕΙΟ
Η ΠΑΙΔΕΙΑ ΣΤΗΝ ΚΟΡΥΦΗ
Επιχειρησιακό Πρόγραμμα
Εκπαίδευσης και Αρχικής
Επαγγελματικής Κατάρτισης

Η/Υ 432: Δομές Δεδομένων

Χειμερινό Εξάμηνο Ακαδημαϊκού Έτους 2005-2006

Παναγιώτα Φατούρου

1ο Σετ Ασκήσεων

Ημερομηνία Παράδοσης

Θεωρητικό Μέρος: 1ο Μέρος: Πέμπτη, 10/11, ώρα: 15:00. 2ο Μέρος: Δευτέρα 21/11, ώρα 12:00

Προγραμματιστικό Μέρος: Δευτέρα, 21/11, ώρα 12:00

Τρόπος Παράδοσης

Θεωρητικό Μέρος: Παραδίδονται στους βοηθούς του μαθήματος (Νικόλαος Καλλιμάνης και Μαρία Χριστοδουλίδου) το αργότερο μέχρι τις ημερομηνίες παράδοσης. Το γραφείο των βοηθών του μαθήματος είναι το Α25.

Προγραμματιστικό Μέρος: Θα χρησιμοποιήσετε το πρόγραμμα turnin. Η εντολή που θα εκτελέσετε είναι:
turnin lab1_05@cs432 <όνομα_αρχείου1.c> <όνομα_αρχείου2.c> ... <όνομα_αρχείουk.c>

Θεωρητικό Μέρος

1ο Μέρος

1. α. Θεωρήστε την ακόλουθη αναδρομική διαδικασία:

```
int rec(int n) {
    int n1, n2;
    if (n <= 5) then return n;
    else {
        n1 = rec(n-5);
        n2 = rec(n-3);
        return (n1+n2);
    }
}
```

Σας ζητείται να τρέξετε στο χαρτί (δηλαδή να κάνετε trace) την rec(n) για την περίπτωση που n = 19. Πρέπει να παρουσιάσετε όλες τις αναδρομικές κλήσεις της rec καθώς και τις τιμές των μεταβλητών n, n1 και n2 σε κάθε εκτέλεση της rec. [10%]

β. Για την ακόλουθη διαδικασία που περιγράφεται σε ψευδοκώδικα, εξετάστε αν τερματίζει σε πεπερασμένο πλήθος βημάτων για οποιαδήποτε ακέραια τιμή της παραμέτρου n:

```
a.  int g(int n) {
b.      if (n <= 1) return(1);
c.      else if (n <= 4) return ([n/2]);
d.      else if (n <= 20) return(g([n/2]));
e.      else if (n mod 2 == 0) return(g(n/2));
f.      else if (n mod 2 == 1) return(g([n/3]));
g.      else return(g(2*n));
}
```

Τι θα ίσχυε αν η γραμμή f. της διαδικασίας παραληφθεί; [10%]

2. α. Ποιες από τις ακόλουθες προτάσεις είναι σωστές και ποιες όχι; [10%]

i. $sqrt(n^5) \in O(n^2)$

ii. $\log(n^3) \in O(n \log n)$

iii. $sqrt(n) \log(sqrt(n)) \in O(n)$

iv. $5n^2+10n+3 \in O(n^{2/3})$

v. $5n^2+10n+3 \in O(n^3)$

vi. $5n^2+10n+3 \in \Theta(n^2)$

vii. $5n^2+10n+3 \in \Omega(n)$

γ. Βρείτε τη χρονική πολυπλοκότητα T(n) των παρακάτω αλγορίθμων. Βρείτε επίσης την τάξη της T(n) σε κάθε περίπτωση. [10%]

i. procedure f1(int n)

```
for i = 1 to n do {
    if (i mod 2 == 0) {
        for j = 1 to i do y = y+1;
        for j=n down to i do x = x+1;
    }
}
```

ii. procedure f2(int n)

```
for j = 1 to n {
    x = n;
    while (x > 0) do x = x-j;
    while k = j to n^2  do y = y-1;
}
```

Σημείωση: Το άθροισμα $H_n = 1+1/2+ 1/3 + ... + 1/n$ ονομάζεται αρμονικός αριθμός και είναι γνωστό ότι $H_n = O(\ln n)$.

2ο Μέρος

1. Θεωρήστε έναν τριγωνικό πίνακα A[i,j] με $1 \leq i, j \leq n$. Θέλουμε να αποθηκεύσουμε τα μη-μηδενικά στοιχεία του πίνακα A σε διάταξη **κατά στήλες** σε έναν μονοδιάστατο πίνακα P.

   i. Ποιο πρέπει να είναι το μέγεθος του πίνακα P; [3%]

   ii. Εξηγήστε σε ποια θέση του πίνακα P βρίσκεται το στοιχείο A[i,j]. [5%]

   iii. Περιγράψτε τη διαδικασία read_A(int i, int j), η οποία επιστρέφει την τιμή του A[i,j] (δεδομένου ότι ο A έχει υλοποιηθεί χρησιμοποιώντας τον P). [7%]

2. Να γραφούν διαδικασίες (σε ψευδοκώδικα) που ανταλλάζουν το πρώτο στοιχείο μιας συνδεδεμένης λίστας με το τελευταίο στοιχείο της, όταν:

   i. Η λίστα είναι απλά συνδεδεμένη.

   ii. Η λίστα είναι διπλά συνδεδεμένη.

   iii. Η λίστα είναι κυκλική (δηλαδή ο δείκτης next του τελευταίου στοιχείου της λίστας δεν είναι NULL αλλά δείχνει στο πρώτο στοιχείο της λίστας).

   Δώστε επίσης διαδικασίες διαγραφής ενός στοιχείου από την αρχή και το τέλος κάθε μιας από τις παραπάνω κατηγορίες λιστών.

   Ποιες είναι οι χρονικές πολυπλοκότητες καθεμιάς από τις παραπάνω διαδικασίες ως συνάρτηση του αριθμού n των στοιχείων της λίστας; [22%]

3. Να γραφούν διαδικασίες (σε ψευδοκώδικα) που αντιστρέφουν τη σειρά των στοιχείων μιας απλής συνδεδεμένης λίστας, μιας κυκλικής λίστας, μιας διπλά συνδεδεμένης λίστας και μιας διπλά συνδεδεμένης κυκλικής λίστας. Μελετήστε την πολυπλοκότητα των αλγορίθμων που προτείνετε. Προσπαθήστε οι αλγόριθμοι αυτοί να έχουν τη μικρότερη δυνατή πολυπλοκότητα. [23%]

Προγραμματιστικό Μέρος (προαιρετικό)

Για απλότητα, θεωρήστε πως οι λίστες, στοίβες και ουρές που ζητούνται στην συνέχεια αποθηκεύουν μόνο έναν ακέραιο στο info πεδίο τους. Δουλέψτε σε όσα από τα ακόλουθα προγράμματα σας ευχαριστεί. Το (Ε) σημαίνει «Εύκολο», το (Μ) σημαίνει «Μέτριο» και το (Δ) σημαίνει «Δύσκολο». Να σημειωθεί ότι ο αριθμός των προγραμμάτων που περιγράφονται στη συνέχεια είναι μεγάλος για να υλοποιηθούν όλα ως την ημερομηνία παράδοσης. Δουλέψτε σε κάποια μόνο από αυτά. Ένας καλός αριθμός θα ήταν να δοκιμάσετε 2 ασκήσεις από την κατηγορία (Ε), 1 από την κατηγορία (Μ) και μία από την κατηγορία (Δ) (ή αντί αυτής δύο ακόμη από την κατηγορία (Μ)). Καλές επιλογές είναι π.χ.: 2 – 4 - 6 – 9, 2-4-6-10, 1 – 3 – 4 – 10, 2-6-5-9, κ.α. Φυσικά, ο κάθε φοιτητής μπορεί να υλοποιήσει όσες και όποιες ασκήσεις θέλει (αφού είναι όλες προαιρετικές). Μοιράστε τη δουλειά των ασκήσεων που αποφασίσατε πως θα υλοποιήσετε σε δύο μέρη (με περίπου ίδια δουλειά κάθε ένα από αυτά) και επιστρέψτε το πρώτο μέρος την Πέμπτη, 10/11, ενώ το δεύτερο μέρος τη Δευτέρα 21/11. Για παράδειγμα το πρώτο μέρος μπορεί

να περιλαμβάνει τις ασκήσεις της κατηγορίας (Ε) και (Μ) και το δεύτερο μέρος την άσκηση της κατηγορίας (Δ).

1. Υλοποιήστε σε C μια συνδεδεμένη στοίβα και μια συνδεδεμένη ουρά. Η υλοποίηση της ουράς έχει ήδη συζητηθεί στο μάθημα: γράψτε το πρόγραμμα και δείτε το να εκτελείται. Η στοίβα υλοποιείται με εντελώς αντίστοιχο τρόπο. (ΕΕ)
2. Υλοποιήστε σε C μια απλά συνδεδεμένη λίστα. Οι ακόλουθες λειτουργίες θα πρέπει να υλοποιηθούν (η σειρά με την οποία εμφανίζονται στη συνέχεια καθορίζει και τη δυσκολία τους σε αύξουσα διάταξη): (Ε)
    i. InsertToFront: Παίρνει ως παράμετρο έναν ακέραιο n. Εισάγει ένα νέο κόμβο που περιέχει την τιμή n ως πρώτο στοιχείο της λίστας.
    ii. PrintList: Διατρέχει τη λίστα και τυπώνει όλα τα στοιχεία της.
    iii. LookUp: Ψάχνει για ένα στοιχείο με τιμή n (όπου το n είναι ακέραιος, παράμετρος στη LookUp).
    iv. DeleteFromFront: διαγράφει το πρώτο στοιχείο της λίστας.
    v. DeleteFromEnd: διαγράφει το τελευταίο στοιχείο της λίστας.
    vi. Delete: Παίρνει ως παράμετρο έναν ακέραιο n, ψάχνει στη λίστα για τον κόμβο που περιέχει την τιμή n και αν ο κόμβος αυτός υπάρχει τον διαγράφει. Ο προς-διαγραφή κόμβος μπορεί να βρίσκεται οπουδήποτε μέσα στη λίστα.
3. Υλοποιήστε μια απλά συνδεδεμένη ταξινομημένη λίστα. Οι λειτουργίες που πρέπει να υλοποιηθούν είναι οι Insert (για την εισαγωγή ενός νέου κόμβου), LookUp (για την αναζήτηση μιας τιμής), Delete (για τη διαγραφή ενός κόμβου) και PrintList (για εκτύπωση των στοιχείων της λίστας). (Ε)
4. Υλοποιήστε μια διαδικασία που θα συνενώνει δύο απλά συνδεδεμένες ταξινομημένες λίστες σε μια. Η λίστα που προκύπτει πρέπει να είναι επίσης ταξινομημένη και να περιέχει τα στοιχεία και των δύο λιστών. (Μ)
5. Υλοποιήστε μια διπλά συνδεδεμένη λίστα. (Μ)
6. Υλοποιήστε μια απλά συνδεδεμένη κυκλική λίστα. Τροποποιήστε κατάλληλα τον κώδικα που γράψατε για την προγραμματιστική άσκηση 2. (Ε)
7. Υλοποιήστε μια διπλά συνδεδεμένη κυκλική λίστα. Τροποποιήστε κατάλληλα τον κώδικα που γράψατε για την προγραμματιστική άσκηση 5. (Μ)
8. Υλοποιήστε όσες από τις λειτουργίες που περιγράφονται στη θεωρητική άσκηση 4 δεν έχετε ακόμη υλοποιήσει για τα είδη λιστών που επιλέξατε να υλοποιήσετε στις προηγούμενες ασκήσεις. (Μ)
9. Υλοποιήστε τις λειτουργίες που περιγράφονται στη θεωρητική άσκηση 5 για τα είδη λιστών που επιλέξατε να υλοποιήσετε στις προηγούμενες ασκήσεις. (Δ)
10. Τροποποιήστε τον κώδικα που γράψατε για την προγραμματιστική άσκηση 2, ώστε η λίστα να αποθηκεύει πολυώνυμα της μορφής $a_n x^n + \ldots + a_1 x + a_0$. Το πρόγραμμα σας θα πρέπει να έχει τη δυνατότητα να αποθηκεύει περισσότερα από ένα πολυώνυμα (δηλαδή περισσότερες από μια λίστες). Χρησιμοποιείστε πίνακα ή λίστα για να αποθηκεύετε τους δείκτες στα πρώτα στοιχεία των λιστών αυτών. Υλοποιήστε

τη διαδικασία MultPol, που θα υπολογίζει το γινόμενο δύο πολυωνύμων και θα το αποθηκεύει σε μια νέα λίστα. Τα δύο πολυώνυμα που θα συμμετέχουν στον υπολογισμό του γινομένου θα επιλέγονται από το χρήστη (στον οποίο θα πρέπει να τυπώνονται όλα τα υπάρχοντα αποθηκευμένα πολυώνυμα και να του ζητείται να επιλέξει δύο εξ αυτών).