Άσκηση 3

Ημερομηνία Παράδοσης: 7 Μαΐου 2008

Σημειώσεις:

1. Στις απαντήσεις που θα παραδώσετε σημειώστε στην πρώτη σελίδα το ονοματεπώνυμό σας, τον αριθμό μητρώου σας και το τμήμα σας.
2. Οι ασκήσεις πρέπει να γίνουν ατομικά. Οποιαδήποτε μορφή αντιγραφής απαγορεύεται.
3. Η παρούσα άσκηση πρέπει να παραδοθεί το αργότερο μέχρι την αρχή του μαθήματος της 7ης Μαΐου, δηλαδή μέχρι τις 15:15. Καθυστερημένες ασκήσεις δε θα γίνουν δεκτές.
4. Σε περίπτωση που έχετε ερωτήσεις στείλτε email στην ηλεκτρονική λίστα του μαθήματος: hy240-list@tem.uoc.gr

Πρόβλημα 1 [15 μονάδες] Θεωρήστε την ακολουθία ακεραίων αριθμών 34, 56, 20, 78, 31, 14, 30, 34, 11. Ποιά θα είναι η μορφή ενός σωρού (heap) H αν αρχικά ο σωρός είναι άδειος και στη συνέχεια προσθέσουμε τους ακεραίους αριθμούς με τη σειρά που σας δίδονται;

Δείξτε τη μορφή του σωρού τόσο ως δέντρο, όσο και ως πίνακα μετά από κάθε εισαγωγή.

Πρόβλημα 2 [40 μονάδες] Θεωρήστε την ίδια ακολουθία ακεραίων αριθμών όπως στο προηγούμενο πρόβλημα. Ποιά θα είναι η μορφή ενός δυαδικού δέντρου αναζήτησης αν αρχικά το δέντρο είναι άδειο και στη συνέχεια προσθέσουμε τους ακεραίους αριθμούς με τη σειρά που σας δίδονται;

Θεωρήστε το ίδιο πρόβλημα αν το δέντρο είναι:

1. δέντρο AVL και
2. μελανέρυθρο δέντρο (red-black tree).

Δείξτε όλα των ενδιάμεσα στάδια των εισαγωγών. Στην περίπτωση του μελανέρυθρου δέντρου σημειώστε και το χρώμα των κόμβων.

Πρόβλημα 3 [10 μονάδες] Θεωρήστε ένα μελανέρυθρο δέντρο T, του οποίου το *μαύρο ύψος* είναι $\mathrm{bh}(T)$ και ο αριθμός των εσωτερικών του κόμβων είναι n. Δείξτε ότι $2^{\mathrm{bh}(T)} \leq n+1 \leq 4^{\mathrm{bh}(T)}$.

Πρόβλημα 4 [15 μονάδες] Υπολογίστε το ελάχιστο και μέγιστο δυνατό ύψος ενός μελανέρυθρου δέντρου με περιέχει 7 κλειδιά. Σχεδιάστε μελανέρυθρα δέντρα με 7 κλειδιά, όπου το πρώτο έχει το ελάχιστο και το δεύτερο έχει το μέγιστο ύψος που υπολογίστατε. Εξηγήστε γιατί τα δέντρα που σχεδιάσατε έχουν τα ύψη που υπολογίσατε.

Πρόβλημα 5 [20 μονάδες] Θεωρήστε ένα σωρό που υλοποιείται ως δέντρο. Οι δομές που αντιστοιχούν στους κόμβους του σωρού, αλλά και το σωρό σας δίνονται παρακάτω.

```
typedef struct HeapNode {
  HeapNode* leftChild;
  HeapNode* rightChild;
  HeapNode* parent;
  Key key;
} HeapNode;

typedef struct Heap {
  HeapNode* root;
  unsigned int size;
} Heap;
```

Στην ρίζα του σωρού ο δείκτης `parent` έχει την τιμή `NULL`. Το ίδιο ισχύει και για τους δείκτες `leftChild` και `rightChild` αν δεν έχουν ως παιδιά εσωτερικούς κόμβους.

Θεωρήστε τώρα ότι θέλουμε να μειώσουμε την τιμή του κλειδιού ενός κόμβου. Εξηγήστε πώς μπορεί να γίνει αυτό χωρίς να αφαιρέσετε τον κόμβο και να τον ξαναπροσθέσετε με το νέο κλειδί. Υπολογίστε το κόστος του αλγορίθμου σας και δώστε κώδικα σε `C` που να υλοποιεί τον αλγόριθμό σας. Ο κώδικάς σας θα πρέπει να έχει το function prototype:

```
void decrease_key(Heap* h, Key old_key, Key new_key);
```

όπου `old_key` είναι το παλιό κλειδί και `new_key` το νέο κλειδί, το οποίο και ξέρετε ότι είναι μικρότερο του παλιού. Μπορείτε να υποθέσετε ότι όλα τα κλειδιά του σωρού είναι ανά δύο διαφορετικά. Για να συγκρίνετε τα κλειδιά σας δίνεται η συνάρτηση σύγκρισης:

```
int compare(Key k1, Key k2);
```

η οποία επιστρέφει 0, αν τα κλειδιά `k1` και `k2` είναι ίσα, κάποιον θετικό ακέραιο αν το κλειδί `k1` είναι μεγαλύτερο του κλειδιού `k2` και κάποιον αρνητικό ακέραιο αν το κλειδί `k1` είναι μικρότερο του κλειδιού `k2`.

Τί συμβαίνει στην περίπτωση που δεν γνωρίζουμε ότι το νέο κλειδί είναι μικρότερο του παλιού; Ο αλγόριθμος σας είναι εφικτός στην περίπτωση αυτή; Πώς μπορούμε στην περίπτωση αυτή να τροποποιήσουμε το σωρό με αποδοτικό τρόπο;

Σύνολο μονάδων: 100