

Voortentamen Data Structures and Algorithms

29 September 2009

Task 1.

For 1. (a)–(c) the answer suffices, no justification is needed.

- (a) Consider the following algorithm for searching the element k in an unordered array containing n integers.

```
find( $k, A, n$ ):  
  for  $i = 0$  to  $n - 1$  do  
    if  $A[i] == k$  then return true  
  done  
  return false
```

What is the best-case and worst-case time complexity in big O-notation?

(3 points)

- (b) Consider the following algorithm computing the maximum subarray sum of an array containing n integers.

```
maxSubarray( $A, n$ ):  
   $max = 0$   
  for  $left = 0$  to  $n - 1$  do  
     $sum = 0$   
    for  $right = left$  to  $n - 1$  do  
       $sum = sum + A[right]$   
      if  $sum > max$  then  $max = sum$   
    done  
  done  
  return  $max$ 
```

What is the worst-case time complexity of this algorithm in big O-notation?

(3 points)

- (c) Fill in the question mark (with a possible simple, but not overestimating expression):

- (i) $5 \cdot n^2 + n^3 + 2 \in O(?)$
- (ii) $2 \cdot n + 5 \cdot \log_2 n \in O(?)$
- (iii) $2^n + 7 \cdot n \cdot \log_2 n \in O(?)$

(5 points)

- (d) Is an $O(n)$ algorithm always faster than an $O(n^2)$ algorithm? Justify your answer.
(4 points)

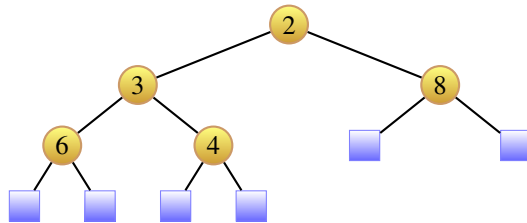
Task 2.

- (a) We implement a priority queue using a sorted array. What is the time complexity of **enqueue** and **dequeue** in terms of big O -notation? Justify your answer.
(5 points)
- (b) Again, consider a priority queue using a sorted array. What is the time complexity of priority queue sort (sorting a list using the priority queue)? Justify your answer.
(5 points)
- (c) Give the merge-sort tree for sorting the following list: 7, 5, 2, 8, 1, 4, 3, 6, 9.
(8 points)

Task 3.

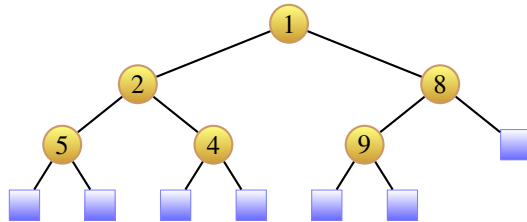
Draw the final heap after every addition/removal step; intermediate steps are not needed.

- (a) Add the keys 5 and 1 (in this order) to the following heap:



(7 points)

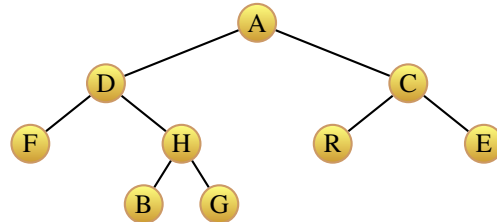
- (b) Remove the key 1 from the following heap:



(5 points)

Task 4.

(a) Consider the following tree:



Give the order of the nodes when visited with:

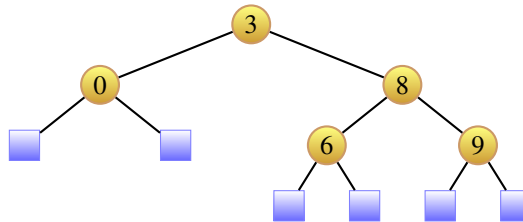
- (i) inorder traversal, and
- (ii) preorder traversal?

(6 points)

(b) Give a pseudo-code for postorder traversal of a tree.

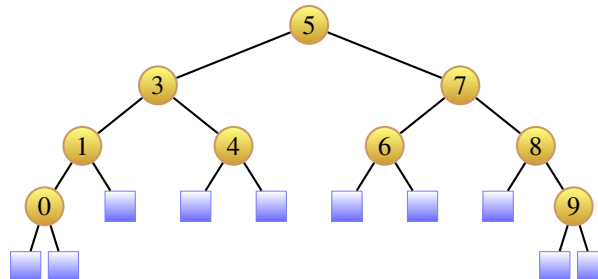
(5 points)

(c) Add the key 4 to the following AVL tree: draw the tree before and after each rotation step, and indicate the balance factor the nodes involved in rebalancing.



(10 points)

(d) Remove the key 5 from the following AVL tree:



(5 points)

Task 5.

- (a) Make a hash table of size 9. Use the hash function $h(k) = k \bmod 7$. Add to the initial empty hash table the following numbers in this order:

15, 10, 8, 3, 22

and use the following 3 different ways for solving collisions:

- (i) chaining
- (ii) linear probing
- (iii) double hashing with secondary hash function $d(k) = 4 - (k \bmod 4)$

(10 points)

- (b) What is the

- (i) best-case,
- (ii) average (expected), and
- (iii) worst-case

time complexity of searching a key k in a hash table?

(justify the worst-case complexity, for (i) and (ii) the answers suffice)

(9 points)

Task 6. Optional Tasks for Extra Points

- (a) Justify why the inorder traversal of a search tree yields elements in ascending order.

(5 points)

- (b) What is the minimal number of inner nodes of an AVL tree of height 8?

(5 points)

The grade is the (total amount of points plus 10) divided by 10.