

Matriculation Number: \_\_\_\_\_

CS1102

NATIONAL UNIVERSITY OF SINGAPORE

SCHOOL OF COMPUTING  
SEMESTER I AY2006/2007

CS1102: DATA STRUCTURES AND ALGORITHMS

Nov/Dec 2006

Time Allowed: 2 Hours

MATRICULATION NUMBER:

--	--	--	--	--	--	--	--	--	--

**INSTRUCTIONS TO CANDIDATES:**

1. Write your matriculation number in the space provided above. Also write your matriculation number at the top of each sheet in the exam paper. Shade your matriculation number on the OCR form. Remember to sign on the form.
2. This examination paper consists of **SEVEN (7) questions**. **Question 1 consists of 15 MCQ questions** and comprises **THIRTEEN (13)** printed pages including this front page.
3. Answer the **MCQ** questions by shading the **OCR** form and answer all of the other questions directly in the space given after each question. If necessary, use the back of the page.
4. Marks allocated to each question are indicated. Total marks for the paper is **100**.
5. This is a closed book examination and you can write in pencil.

EXAMINER'S USE ONLY				
Questions	Possible	Marks	Grader	Check
MCQ 1-15	30			
Q2	10			
Q3	12			
Q4	12			
Q5	10			
Q6	10			
Q7	16			
<b>Total</b>	100			

**Question 1 (MCQ – 30 marks)**

1) Which best describes the precondition of a method?

It is an assertion that

- a) describes precisely the conditions that must be true at the time the method is called.
- b) initializes the parameters of the method.
- c) describes the effect of the method on its postcondition
- d) explains what the method does.
- e) states what the initial values of the local variables in the method must be.

2) A list of items is to be maintained in random order. Operations performed on the list include

- I) insertion of new items at the front of the list
- II) deletion of old items from the rear of the list

A programmer considers using a linear singly linked list (LLL), a circular singly linked list (CLL), or an array to store the items. Which of the following correctly represents the run-time efficiency of (I) insertion and (II) deletion for this list?

You may assume that

- The array has sufficient slots for insertion
- Linear linked lists are implemented with a reference to the first node only.
- The most efficient algorithm possible is used in each case.

a) array: (I)  $O(n)$       (II)  $O(1)$   
LLL: (I)  $O(1)$       (II)  $O(n)$   
CLL: (I)  $O(1)$       (II)  $O(n)$

b) array: (I)  $O(n)$       (II)  $O(1)$   
LLL: (I)  $O(1)$       (II)  $O(n)$   
CLL: (I)  $O(1)$       (II)  $O(1)$

c) array: (I)  $O(1)$       (II)  $O(1)$   
LLL: (I)  $O(1)$       (II)  $O(1)$   
CLL: (I)  $O(1)$       (II)  $O(1)$

d) array: (I)  $O(n)$       (II)  $O(n)$   
LLL: (I)  $O(1)$       (II)  $O(n)$   
CLL: (I)  $O(1)$       (II)  $O(n)$

e) array: (I)  $O(1)$       (II)  $O(n)$   
LLL: (I)  $O(n)$       (II)  $O(1)$   
CLL: (I)  $O(n)$       (II)  $O(1)$

- 3) Which of the following, when used as the <body> of method sum, will enable that method to compute  $1 + 2 + 3 + \dots + n$  correctly for any  $n > 0$ ?

```
// precondition: n > 0
// postcondition: 1 + 2 + 3 + .... + n has been returned.
```

```
public static int sum(int n) {
    <body>
}
```

- I. return  $n + \text{sum}(n - 1)$ ;
- II if ( $n == 1$ ) return 1;  
else return  $n + \text{sum}(n - 1)$ ;
- III if ( $n == 1$ ) return 1;  
else return  $\text{sum}(n) + \text{sum}(n - 1)$ ;
- a) I only  
b) II only  
c) III only  
d) I, II and III  
e) I and II only

Question 4 – 5 refer to method t:

```
// precondition:  $n \geq 1$ 
public static int t(int n) {
    if ( $n == 1 \parallel n == 2$ ) return  $2 * n$ ;
    else return  $t(n - 1) - t(n - 2)$ ;
```

- 4) For the method call  $t(6)$ , how many calls to t will be made, including the original call?
- a) 6  
b) 7  
c) 11  
d) 15  
e) 25
- 5) The run time of method t is
- a)  $O(n)$   
b)  $O(n^2)$   
c)  $O(2^n)$   
d)  $O(n^3)$   
e)  $O(\lg n)$

- 6) Refer to method doSomething:

```
public static Object doSomething ( TreeNode root) {  
    if (root != null)  
        if (root.getRight() == null)  
            return root.getValue();  
        else  
            return doSomething(root.getRight());  
    return null;  
}
```

Which best describes what doSomething does?

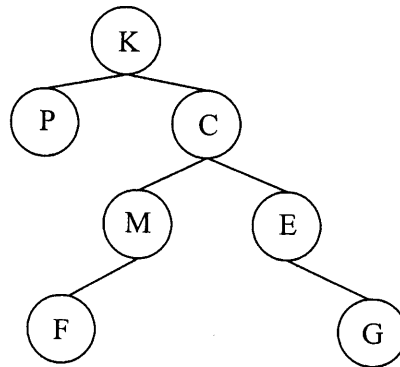
- a) It returns the largest element in a nonempty binary search tree.
  - b) It returns the largest element in a nonempty tree.
  - c) It returns an element at the highest level of a nonempty tree.
  - d) It returns the smallest element in a nonempty binary search tree.
  - e) It returns the smallest element in a nonempty tree.
- 7) Operations on a queue can be carried out at \_\_\_\_\_.
- a) its front only
  - b) its back only
  - c) both its front and back
  - d) any position in the queue
  - e) none of the above
- 8) A full binary tree with  $k$  leaves contains how many nodes?
- a)  $k$
  - b)  $k^2$
  - c)  $2^k$
  - d)  $\log_2 k$
  - e)  $2k - 1$
- 9) Each of the following lists of numbers is inserted, in the order given, into a binary search tree. Which list produces the most balanced tree?
- a) 2 4 7 5 8 10
  - b) 9 7 2 1 4 0
  - c) 5 1 2 6 3 4
  - d) 2 5 1 4 0 3
  - e) 6 4 1 8 10 5

10) The tree shown is traversed postorder and each element is pushed onto a character stack *s* as it is encountered. The following program fragment is then executed:

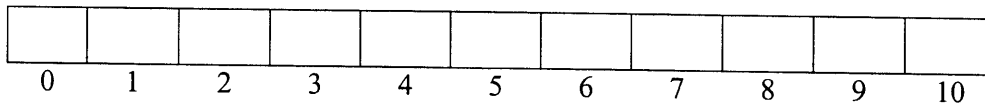
```
for (int j = 1; j <= 5; j++)
    ch = s.pop();
```

What value is contained in the variable *ch* after the segment is executed?

- a) M
- b) G
- c) K
- d) F
- e) P



11) The following key values are to be inserted into the hash table shown in the order given 10 28 2 7 45 25 40 29. The hash function is  $key \% 11$ . Collisions will be resolved with linear probing. Which array slot will 29 eventually occupy?



- a) 7
- b) 8
- c) 9
- d) 10
- e) 0

12) Which of the following is true of a priority queue?

- a) If elements are inserted in increasing order of priority (i.e, lower priority element inserted first), and all elements are inserted before any are removed, it work like a queue.
- b) If elements are inserted in increasing order of priority (i.e, lowest priority element inserted first), and all elements are inserted before any are removed, it works like a stack.
- c) If all elements are inserted before any are removed, it works like a queue.
- d) If elements are inserted in decreasing order of priority(i.e, highest priority element inserted first), and all elements are inserted before any are removed, it works like a stack.
- e) If elements are inserted in increasing order of priority, then it works like a queue whether or not all insertions precede any removals.

13) An unsorted list of integers is stored in an array. Having the list unsorted leads to inefficient execution for which of the following operations?

- I. Inserting a new element
- II. Searching for a given element
- III. Computing the mean of the elements

- a) I only
- b) II only
- c) III only
- d) I and II only
- e) I, II, and III

14) Refer to method whatsIt:

```
public static int whatsIt ( TreeNode tree) {
    int x, y;
    if (tree = null) return -1;
    else {
        x = 1 + whatsIt(tree.getLeft());
        y = 1 + whatsIt(tree.getRight());
        if ( x >= y)
            return x;
        else
            return y;
    }
}
```

Method whatsIt returns -1 for an empty tree. What does method whatsIt do when invoked for a nonempty tree?

- b) It returns the largest value in the tree.
- c) It returns the number of nodes in the subtree that has the greatest number of nodes.
- d) It returns the level of the tree.
- e) It returns 1 plus the level of the tree.
- f) It returns either the leftmost value or the rightmost value of a tree, whichever is larger,

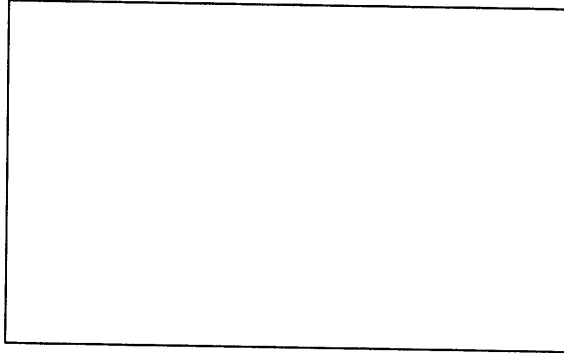
15) A connected undirected graph that has  $n$  vertices and exactly  $n - 1$  edges \_\_\_\_\_.

- a) cannot contain a cycle
- b) must contain at least one cycle
- c) can contain at most two cycles
- d) must contain at least two cycles
- e) none of the above

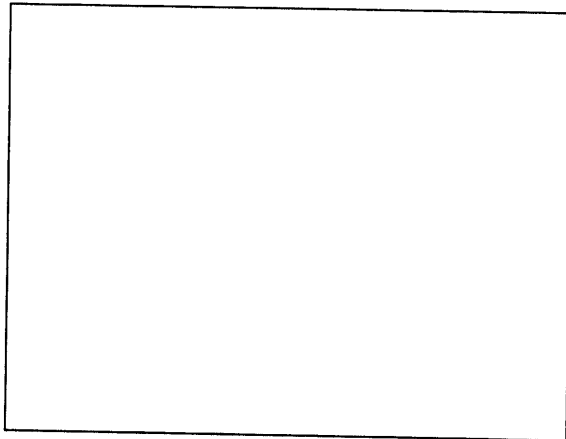
**Question 2 (10 marks)**

Study the following segments of codes and provide the running times using big-O notation.

a)  $f(n)$  {  
    if ( $n/2 > 1$ )  
         $f(n-1) + f(n-1)$ ;  
}



b) for ( $j = 0; j < n; j++$ )  
    for ( $k = n; k > j; k++$ )  
         $sum++$ ;



**Question 3 (12 marks)**

There are  $n$  people. The entry  $K[i][j]$  of  $K[1..n][1..n]$  is 1 when person  $i$  knows person  $j$  and 0 otherwise. (We assume that  $K[i][i] = 1$  for every  $i$ .) A celebrity is a person who is known by everyone and does not know anyone besides himself/herself. For example, in the left diagram below person 3 is a celebrity. There is no celebrity in the situation in the right diagram:

	1	2	3	4
1	1	1	1	0
2	0	1	1	0
3	0	0	1	0
4	1	0	1	1

	1	2	3	4
1	1	1	0	1
2	0	1	1	0
3	1	0	1	0
4	1	0	1	1

We can easily prove that there is at most one celebrity in any situation. Use this ideas to construct a  $O(n)$  algorithm which finds a celebrity or concludes that there is none.

**Question 4 (12 marks)**

Execute the following sequence of operations on

- a) An empty binary search tree      b) An empty AVL tree

When you delete a node with two children, replace it with the inorder predecessor and delete the inorder predecessor. Show only the final trees after the sequence of operations are executed.

Insert(13), Insert(98), Insert(34), Insert(81), Insert(57), Delete(13),  
Insert(62), Insert(77), Insert(49), Delete(81), Insert(95), Insert(25),  
Delete(34), Delete(77)

a) BST

b) AVL

**Question 5 (10 marks)**

The following is a hash table containing some elements which are hashed into the table using the following hash function and using quadratic probing to resolve collisions. Note that an X indicates that the slot was occupied by an element which has been deleted from the table.

$$H(\text{key}) = \text{key} \% 13$$

39	X			30	5			34		62		
0	1	2	3	4	5	6	7	8	9	10	11	12

How many more elements can we put into the table before we might not be able to find an empty slot for the next element? (3 marks)

Ans:

In order to avoid the situation where we are not able to find an empty slot, we re-hash the values into another table of size 17 and use double hashing to resolve collisions. If the primary hash function and the secondary hash function are  $H(\text{key}) = \text{key} \% 17$  and  $H(\text{key}) = 13 - \text{Key} \% 13$  respectively, show the table after the values are taken from left to right from the original table above and hash into the new table. You should also add in the following 3 keys: 56, 88, 72 (7 marks)

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16

**Question 6 (10 marks)**

- a) Given the following values: 33, 45, 12, 65, 78, 26, 54.  
 Create a max heap by adding the values one at a time. Show the final heap.  
 (3 marks)

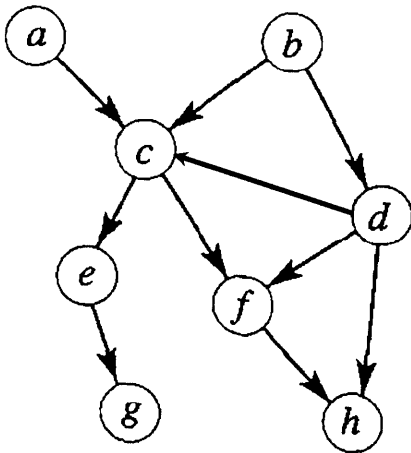
- b) with the same values, create a max heap by heap construction. Again show only the final heap. (3 marks)

- c) Using the heap created in b), perform heap sort. Show the steps (4 marks)


**Question 7 (16 marks)**

- a) Define what is meant by a topological sort of a directed graph. Explain why the graph has no cycles if such a topological sort exists. (2 marks)

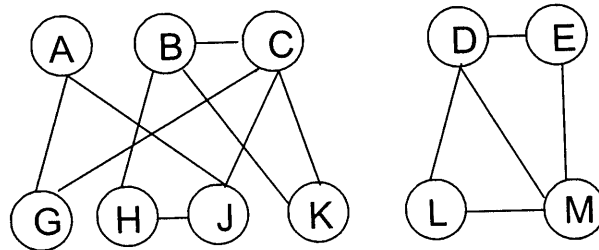
- b) Give the topological sequence of the following directed graph. (4 marks)



- c) A (finite) directed graph is called a *tree* if it has a distinguished node, the *root*, such that there is a unique path from every other node to the root. If a directed graph with  $n$  nodes is a tree, what is the number of edges in the graph? (2 marks)

- d) Consider now an undirected graph  $G$ . Recall that the two graph traversal techniques depth-first-search (DFS) and breadth-first-search (BFS) each compute a collection of trees such that every edge  $(u, v)$  of such a tree is an (undirected) edge of  $G$ , and if  $u$  and  $v$  are nodes in different trees, then there is no edge from  $u$  to  $v$  in  $G$ .

Apply both DFS and BFS to the following unconnected graph. Whenever the algorithm has a choice, assume that the node that comes earlier in the alphabet is considered first (like B before K). Give the resulting DFS- and BFS-tree(s) for the graph, which you may simplify by drawing the tree(s) with fewer crossing edges than in the original graph. (8 marks)



DFS

BFS

**END-OF-PAPER**