No of Pages : 3 Course Code : 15XW23

Roll No:
(To be filled in by the candidate)

PSG COLLEGE OF TECHNOLOGY, COIMBATORE - 641 004

SEMESTER EXAMINATIONS, MAY - 2016

MSc - SOFTWARE SYSTEMS Semester : 2

15XW23 DATA STRUCTURES AND ALGORITHMS

Time : 3 Hours **Maximum Marks : 100**

INSTRUCTIONS:

1. Answer **ALL** questions from GROUP – I.
2. Answer any **4** questions from GROUP – II.
3. Answer any **ONE** question from GROUP – III.
4. Ignore the box titled as **"Answers for Group III"** in the Main Answer Book.

GROUP - I Marks : 10 x 3 = 30

1. What do you mean by best and worst case time complexities of an algorithm? Analyze the following code to find the total frequency count and predict the time complexity.

```
for i = 1 to n {
    for j = 1 to 2*i { ...
                      k = j;
                      while (k >= 0) {...
                      k = k - 1; }
                     }      }
```

2. The following C function takes a singly-linked list as input argument. It modifies the list by moving the last element to the front of the list and returns the modified list. Some part of the code is left blank. Write the suitable statement(s) to replace the blank line.

```
Typedef struct node{
int value;
struct node *next;
} Node;
Node *move_to_front(Node *head) { Node *p, *q;
if((head==NULL)||(head->next==NULL)) return head;
q=NULL;  p=head;
while(p->next!=NULL) { q=p; p=p->next; }
………………
return head;  }
```

3. Consider the array A[2:5, 6:8, 2:4, -2:7] with base address 447700. Find the address of the element A[3,7, 3, 5] in row major order and column major order.

4. Understand the following code. Show the contents of stack and queue for each phase of operation.

```
/*s1 –stack with size 4 and q1-queue with size 4*/
x=1, y=2
one:
if (x%2!= 0) { Push(s1,y);     x=x+1;     y=y+10; }
else  {Enqueue(q1,y*2);       x=x+5;     y=y+1;  }
end if
if  s1 not full and q1 not full  then goto one; end if;
two:
if(y%2!=0 and x%2!=0) {pop(s1);  y=y+1; x=x+10}
else {dequeue(q1);  y=y+5;  x=x+1;}
end if
if s1 is not empty and q1 is not empty then goto two; end if;
```

5. Prove that the height of a binary tree that contains n, n ≥ 0, elements is at most n and at least $\log_2 (n+1)$.

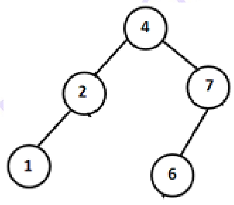
6. A binary tree T has 9 nodes. The inorder and preorder traversals of T yield the following.

   Inorder traversal : E A C K F H D B G

   Preorder traversal: F A E K C D H G B

   Draw the binary tree T. Write each step of the process to identify the structure of the binary tree. Also write the postorder of the binary tree.

7. Write the advantageous of threaded binary trees over binary trees. Draw the preorder threading and postorder threading for the following binary tree.



8. Sort the following elements using radix sort, and also write the time complexity to sort n elements.

   234, 32, 456, 12, 897

9. Construct a Huffman tree with the following data and calculate the weighted path length.

   34, 6, 78, 2, 4

10. How can you determine whether the given sorting algorithm is stable or not? Name any two stable sorting algorithms and any one unstable sorting algorithm.

GROUP - II **Marks : 4 x 12.5 = 50**

11. a) Explain the usage of stack in recursive function call. Show the stack value of each recursive call of the following code by considering the function call as f=pow(5,5). Solve the recurrence relation of the given code to find the time complexity of the code segment.

```
long pow(long x, int n)
{
      if (n==0)   return 1;
      if(n==1)   return x;
      if(isEven(n))      return pow(x*x,n/2);
      else                  return pow(x*x,n/2)*x;
}
```
(6)

    b) Write an algorithm to find a triple representation of a given sparse matrix, also design another algorithm to perform addition on any two triple representation matrices. Also compute the time complexity of each algorithm. (6.5)

12. a) Design a method for keeping two stacks within a single linear array, so that neither stack overflows until all of memory is used and an entire stack is never shifted to a different location within the array. Write algorithms push1, push2, pop1 and pop2 to manipulate the two stacks. (6.5)

    b) Write an algorithm to evaluate postfix expression using stack to reduce it into a single value. Illustrate each step of the algorithm using the following infix expression.

    E = (a+b) * (c/d) + d /f*g (6)

13. a) A deque is a data structure consisting of a list of items, on which the following operations are possible.

    push(x) : Insert item x on the front end of the deque pop() : Remove the front item from the deque and return it

inject(x): Insert item x on the rear end of the deque

eject() : Remove the rear item from the deque and return it

Write algorithms to support the deque that take O(1) time per operation using linked list. (6.5)

b) Explain the benefits of circular queue over linear queue? Write algorithms to perform insertion and deletion on a circular queue. (6)

14 a) How is a binary tree data structure represented in computer memory? Write any two representations with examples. Also write algorithms to perform the following on a binary tree. Analyze the time complexity of each operation.

i) Insert an element

ii) Delete an element (7.5)

b) Explain the process of constructing an expression tree from the following expression with an algorithm. Also illustrate each phase of the steps where stack is involved.

(A+B*C) – ((D*E+F)/G) (5)

15. Explain the concept of priority queue. Write algorithms to implement insertion and deletion operations of a priority queue using a linked list and a heap data structure. Illustrate each phase of insertion and deletion of each algorithm using the following elements. Also, compute the time complexity of each algorithm.

34, 6, 12, 7, 89, 100, 3, 45

GROUP - III **Marks : 1 x 20 = 20**

16. Explain the concept of linear data structures and mention the advantages of linked lists over sequential representation. Also discuss in detail on the usage of singly linked list, doubly linked list and circular list.

Write efficient algorithms to perform each of the following and compute the time complexity of each. Also, explain what type of linked list is suitable for each of the task with justification.

(i). Insert a new node either at beginning or end of the list. The choice is decided at runtime.

(ii). Remove the node from the list which is pointed by the pointer *p*

(iii).In a *self-adjusting* list, all insertions are performed at the front. A *self-adjusting* list adds a *find* operation, and when an element is accessed by a *find*, it is moved to the front of the list without changing the relative order of the other items.

(iv). Add any two polynomial expressions.

17 Explain the concept of hashing. Discuss the key terms hash function and hash collision.

Start with an empty hash table and insert the following keys using linear probing with b = 13 buckets and a hash function f(k) = k%b 7, 42, 25, 70, 14, 38, 8, 21, 34, 11.Also do the following.

(i) Draw the hash table for each insert.

(ii) What is the loading factor of your table after the last insert?

(iii) What is the maximum and average number of buckets examined in an unsuccessful search of your table?

(iv) What is the maximum and average number of buckets examined in a successful search?

(v) Draw a chained hash table for the given keys.

FD/RL **/END/**