(7)

[This question paper contains 12 printed pages.]

Your Roll No.....

Sr. No. of Question Paper : 1076 **I**

Unique Paper Code : 2342012301

Name of the Paper : Data Structures

Name of the Course : **B.Sc. (H) Computer Science (DSC)**

Semester : III

Duration : 3 Hours Maximum Marks : 90

Instructions for Candidates

1. Write your Roll No. on the top immediately on receipt of this question paper.
2. The paper has **two** sections. **Section A** is compulsory.
3. Attempt any **four** questions from **Section B.**
4. All parts of a question must be answered together.

SECTION A

1. (a) Define a binary tree. How does it differ from a binary search tree (BST)? (2)

P.T.O.

(b) Show the final array track[ ] after performing the following function func()? (2)

```
int func()
{
    int track[] = {10, 20, 30, 40}, *striker;

    striker = track;
    track[1] += 30;
    *striker -= 10;
    striker++;
    return 0;
}
```

(c) Which data structure is more suitable for an application that requires frequent insertions and deletions to store and maintain data items: a linked list or an array? Justify the answer. (3)

(d) How can we determine if a circular queue is full? Explain along with C++ code. (3)

(e) Solve the recurrence relation using the master theorem : (4)

$$T(n) \rightarrow 4T(n/2) + \log n$$

(f) Consider the following function : (4)

```
int recursion(int x, int y)
 {
 if (x < 0)
{
 return -recursion(-x, y);
 }
else if (y < 0)
{
 return -recursion(x, -y);
 }
else if (x == 0 && y == 0)
{
 return 0;
 }
else
{
 return 100 * recursion(x / 10, y / 10)
 + 10 * (x % 10) + y % 10;
 }
}
```

What would be the output of recursion (10, 39) and recursion (62,–8)?

(g) A single array A [1... MAXSIZE] is used to implement two stacks. The two stacks grow from opposite ends of the array. Variables top1 and top2 (top 1< top 2) point to the location of the topmost element in each of the stacks. If the space is to be used efficiently, write the condition for "stack full". (4)

(h) Build a min-heap using the following data :

65, 60, 55, 50, 40, 33, 30, 22, 11

Show the heap after each insertion. (4)

(i) Consider an array with the following elements:

102, 280, 405, 513, 642, 746, 910, 958, 1004

Which searching technique (linear or binary) is more suitable and why? Will this technique still be appropriate if the same data is stored using a linked list? Justify your answer. (4)

SECTION B

2. (a) Write the C++ code for implementing a stack using the given class templates: (4)

```
template <class T> class Stack {
public:
        Stack();
        void push(T k);
        T pop();
        T topElement();
        bool isFull();
        bool isEmpty();
private:
    int top;
    T test_Stack[SIZE];
};
```

(b) Construct a binary tree from the given Inorder and Preorder traversals: (5)

Inorder : x, y, z, a, p, q, r

Preorder : a, y, x, z, q, p, r

Also write post order traversal.

(c) Consider a linear queue created using an array of size 4. Perform the following operations in the given order and show the status of the queue after every operation : (6)

Enqueue(4), dequeue(), dequeue(), Enqueue(5), Enqueue(6), Enqueue(8)

If the above queue was circular, show the final contents of the queue after performing the above operations.

3. (a) Sort the following set of elements using insertion sort. Show the contents of the array after every pass : (4)

34, 56, 12, 8, 92, 9, 44, 23

(b) Consider the linked list : (5)

8–>12–>91–>13–>42–>5–>NULL

Give the output of the following function List (head); where pointer head is initially pointing to element 8.

```
Node* List(Node* head) {
    Node* prev = nullptr;
    Node* curr = head;
    Node* next = nullptr;

    while (curr != nullptr) {
        next = curr->next;
        curr->next = prev;
        prev = curr;
        curr = next;
    }
    return prev;
}
```

(c) Insert the following keys into a binary search tree one by one in the given order: (6)

24, 30, 16, 43, 51, 65, 48, 75, 34, 4

Show all the steps involved. After that, Delete the key 16 using deletion by copying and show the resulting tree.

4. (a) Write a function to calculate the number of leaves in a binary tree. (4)

(b) Consider a stack of size 5. Perform the following operations in the given order and show the status of the stack after every operation: (5)

Push (4), pop(), pop(), push(5), push(6), push(8), peek(), pop(), pop(), push(90)

Show the final contents of the stack after performing the above operations.

(c) Give the output of the following code : (6)

```
#include <deque>
#include <iostream>
using namespace std;

void showdq(deque<int> g)
{
    deque<int>::iterator it;
    for (it = g.begin(); it != g.end(); ++it)
        cout << '\t' << *it;
    cout << '\n';
}

int main()
{
    deque<int> dd;
```

```
    dd.push_back(10);
    dd.push_front(20);
    dd.push_back(30);
    dd.push_front(15);

    cout << "The deque dd is : ";
    showdq(dd);

    cout << dd.size();
    cout << dd.front();
    cout << dd.back();

    dd.pop_front();
    showdq(dd);

    dd.pop_back();
    showdq(dd);

    return 0;
}
```

5 (a) Create an AVL tree by inserting : (4)

14, 23, 26, 10, 9, 8

Show the tree after each insertion,

(b) Write the C++ code snippet for inorder traversal of the binary search tree. (5)

(c) Suppose a character'array to be sorted (into alphabetical order) by MIN–HEAPSORT initially contains the following sequence of letters : (6)

Show how would they arranged after BUILD-MIN-HEAP is over. What is the number of comparisons done to construct this heap?

6. (a) Solve the recurrence relation using the recurrence tree method: (4)

$$T(n) \rightarrow 2T(n/2) + n\log n$$

(b) Consider the following lists : (5)

List1: 5->7->9->11->13->15->NULL (Head Pointer LI is pointing to starting node element 5)

List2: 2->4->6->8->10->12->NULL (Head Pointer LI is pointing to starting node element 2)

What will be the output of op_Lists(L1, L2).

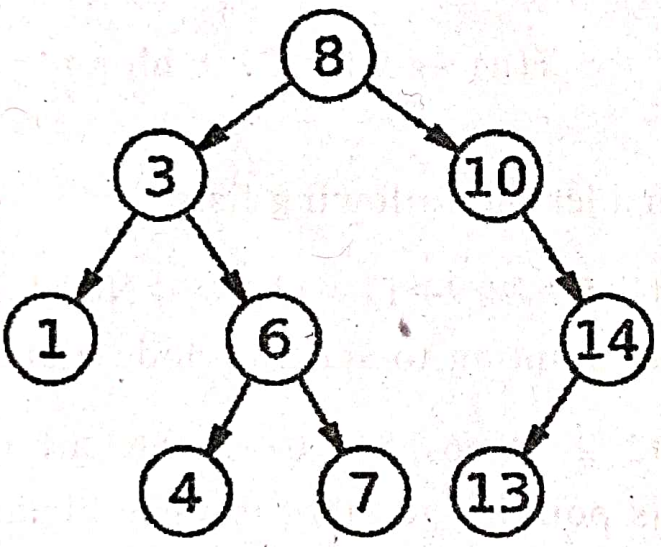
```
void op_Lists(Node* list1, Node* list2) {
    if (!list1) {
        list1 = list2;
        return;
    }

    Node* temp = list1;
    while (temp->next) {
        temp = temp->next;
    }
    temp->next = list2;
}
```

(c) Convert the following expression from prefix to postfix : (6)

+ – * 2 2 / 16 8 5

7. (a) Consider the following binary search tree and answer the questions given below : (4)



(i) Height of the tree

(ii) Number of internal nodes

(iii) Breadth first traversal

(iv) Number of leaves

(b) Sort the following functions in decreasing order of asymptotic (Big-O) complexity: (5)

(i) $f_1(n) = n^2 \cdot \log n$

(ii) $f_2(n) = n^{1.5} + 10^6$

(iii) $f_3(n) = 2^n$

(iv) $f_4(n) = n^3/1000$

(v) $f_5(n) = n(n-1)/2$

Justify your answer.

(c) Suppose the following class definitions of a circular single linked list are given: (6)

```
class Node
{
      int info;
      Node *next;
      Node(int i) {info = i; next=NULL;}
}
class IntCSLL
{
   Node *head, *tail;
   delete()
        // This function deletes a node from the head of the
circular linked list
   {
         Node *Temp = head;
         head->next = head;
         head = head->next;
         delete Temp;
     }

insert(int info)
         // This function inserts a node at the beginning of
a single linked list
     {
          Node *pNode = new Node(info);
          pNode = head;
     }
```

```
traverse()
          // This function traverses the single linked list
    {
       while (head != NULL)
       {
            cout << head->Info;
            head = head->next;
       }
    }
```

Find the errors in delete(), insert() and traverse() functions given in the above code (if any). Write the corrections.