Test-2 Examination- 2020-2021

B.Tech., Odd Semester

Course Title: Data Structures

Course Code: 15B11CI311

Maximum Marks: 20

Maximum Time: 01 hr + (15 minutes for uploading)

1. [CO3] [1 mark] Consider a situation in which counting sort which is used in radix sort is replaced by a quick sort, does it have any impact on radix sort. Justify in terms of correctness/working not in terms of time complexity?
2. [CO2] [1 mark] When can double hashing become and behave like linear probing?
3. [CO3] [1 mark] In a merge sort array is divided into two equal parts. What will be the recurrence relation if the array is divided into five equal parts?

4. [CO4][1 mark] What will be the minimum and maximum heights of a k-ary tree with 40 nodes? Consider K = 4.
5. [CO2] [1 mark] Consider a hash function h(t)= t %n, if we increase the hash table size by twice, is it a necessary and sufficient condition to decrease collision by half. Justify.
6. [CO3] [1 mark] Quick sort algorithm used to sort the records of students. Each record contains roll number, name, year and record to be sorted based on the roll number. There are two sections S1 and S2. In S1 records are stored in sorted order whereas in S2 records are unordered. Suppose you apply quick sort to sort the record in both section. Which section takes more number of comparison and why?

Consider the following pseudo code in which X, Y and Z [0…….k], where length of X and Y is n and k is the maximum element held by X. X has some random elements; Y does not have any element. Now consider a My_sort( ) function

```
My_sort(X, Y, k)
{
        for i ←0 to k                          //line 1
                do Z[i] =0                     //line 2
        for j ←1 to length[X]                  //line 3
                do Z[X[j]] ←Z[X[j]]+1          //line 4
        for i←1 to k                           //line 5
                do Z[i] ← Z[i] +Z[i-1]         //line 6
        for j ←1 to length[X]                  //line 7
                do                             //line 8
                        Y [Z[X[j]]] ←X[j]      //line 9
```

Z[X[j]]←Z[X[j]]-1 //line 10

}

7. [CO3] [1 mark] Does the above sorting algorithm is stable or not, justify.

8. [CO3] [1 mark] If the answer of question 7 is **yes** then point out the code which make it stable, if the answer is **no** then what changes could be made to make the above algorithm stable.

9.[CO3] [1 mark] Suppose an array A contains odd numbers from 51 to 100 in sorted order and B contains even numbers from 1 to 47 in sorted order. What will be the number of comparisons required to merge these two arrays using the merge procedure of merge sort?

10. [CO4] [1 mark] The following function traverses a binary tree. What will be the output of this function for the following binary tree, represented in an array? "\0" representing NULL value.[1 mark]

$$[3, 7, 9, 12, 6, \backslash 0, \backslash 0, 18, 16, \backslash 0, 15, \backslash 0, \backslash 0, \backslash 0, \backslash 0]$$

```
void traverse (node *root) {
  if (root == NULL)
       return;
  stack<node *> Stack;
Stack.push(root);
  while (!Stack.empty()) {
       node *temp = Stack.top();
cout<< temp->data<< " ";
Stack.pop();
      if (temp->lchild)
           Stack.push(temp->lchild);
      if (temp->rchild)
           Stack.push(temp->rchild);   } }
```

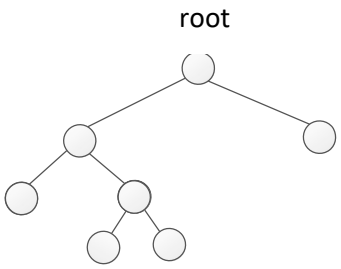
Q11. [CO3] [4 Marks] Suppose there is an array that contains many duplicates elements in the range of 5 to 9. Write a function for partition of modified quick sort that partition the array into three parts such that left sub-part contains the elements smaller than pivot, middle part contains element equal to the pivot and right sub part contains elements greater than pivot. Also write a recursive function for modified quick sort.

Example: A= {5, 9, 7, 6, 5, 8, 7, 8, 5, 8, 9, 6, 9, 7}. After applying the partition algorithm in the first pass content of the array will be: (5, 6, 5, 6, 5) (7, 7, 7) (8, 9, 8, 9, 8, 9)

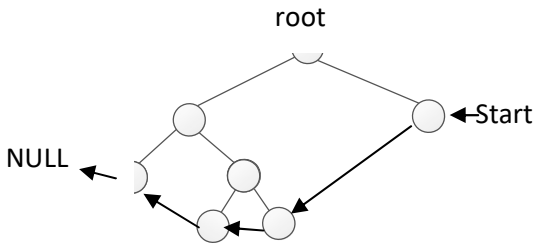
Q12. [CO4][3 marks] Consider the following data structure of a binary tree:

```
struct node {
int data;
node *lchild, *rchild, *next;
bool leaf;
};
```

Let a binary tree is given to you (representative tree is shown below) in which the "leaf" data member value of each node is one if it is a leaf node otherwise zero.



Construct a function which will take the root of the given binary tree and connect a leaf node to the next leaf node to its right and forms a linked list of all the leaf elements as shown below (Use the "next" data member for the same).



Besides, the root pointer of the binary tree, maintain the start pointer (as shown in the figure above) of the linked list formed with leaf elements of the binary tree. The function will return the created start pointer. Further use this start pointer to print all the leaves nodes.

Q13. [CO2][3 marks] Consider a hash table with size s=11 and a hash function with H(x) = x%11. The keys are 11, 12, 19, 20, 52, 44, 56, 37, 60 are inserted in order.

I. [1 mark] perform the insertion in the hash table with $H_1(x)$= x%11 and $H_2(x)$= 7-(x%7)
II. [1 mark] Suppose if separate chaining is used then what will the cost of unsuccessful search.
III. [1 mark] In H(x) = x % s, certain values of s are avoided, why? Can you give any example of any such value? Consider x is in decimal system and s is size of the hash table.

.