Birla Institute of Technology and Science Pilani, Hyderabad Campus
1st Semester 2022-23 Comprehensive Examination
BITS F232 (Foundations of Data Structures and Algorithms)
Max. Marks: 70, Duration: 3 Hrs., Type: Part Open, Date: 17/12/2022(AN)

Note: Write your answers serially in the answer sheet. Assume suitable data, if necessary. Write your answers legibly. Answer Part-A first. After submitting Part-A answer sheet, you are allowed to collect Part-B answer sheet and the reference materials. For the open book part only text, reference books, class notes, lecture slides, and lab sheets are allowed. You can take your own time in answering each part. There is no time limit for any part (within the three hours). Question paper consists of 7 questions.

Part-A (Closed Book): 40 Marks

Q.1 a) Let X be a quick sort code to sort integers in ascending order using the last element as pivot. Let s1 and s2 be the number of comparisons performed by X for input values {1, 2, 3, 4, 5} and {4, 1, 5, 3, 2} respectively. Find out if, s1 > s2 or s1 < s2 or s1 == s2? Provide reason for your answer in brief.

[2]

b) Assuming your friend has applied the first iteration of quick sort on an array and got the list as {1, 4, 0, 6, 8, 11, 10, 9}. He forgot which pivot he had chosen in the first iteration. Help him/ her in finding out a possible pivot he/ she might have used. Describe the rationale behind your proposal.

[2]

c) Ayush spends his weekends with children in a park. He wants to keep children happy by giving them cookies. He has bought a large number of cookies of different weights (in grams). A child will be happy if he/ she gets a cookie of equal or higher weight that he/ she desires. For example, if a child desires to eat a 10-gram cookie he/ she will be happy if Ayush gives any cookie that has a weight of 10-gram or more. Assuming that you are given with two lists, one containing weights of different cookies Ayush has bought, and the other containing desire/wish of children (in terms of grams), write an algorithm to find out how many children can be made happy by Ayush on a particular day. Assume that the cookies cannot be broken.

[3]

d) Mr Prashant lost his memory after attending the theory class on "Binary Search" topic. He forgot every algorithm that he was knowing earlier except the "Binary Search". Write a pseudocode or algorithm for him to find out the square root of a given number. You are free to consider floor or ceiling function, if needed.

[3]

Q.2 a) Assume that you are given with an array of integers. Write an algorithm (pseudo code) to find out all the Sub-arrays having a Sum equal to X. Your algorithm should run with a worst case time complexity of O(n).

Sample input: Array= {3, 6, -3, -1, -2, 4, 4, 2, -12, 6, -7}, Sum = 1
Sample output: 0-8, 3-5, 3-9

[3]

b) Compare the quick-sort and merge-sort algorithms in terms of their time and space complexity. Which is better in terms of time complexity? Which is better in terms of space complexity? Explain.

[2]

c) Given a string consisting of opening and closing parenthesis, give an O(n) algorithm (both time and space) to find the length of the longest balanced parenthesis in it. Describe your algorithm using pseudo code. Use an appropriate data structure. For example, for input: ((((()), Output should be: 4.

[3]

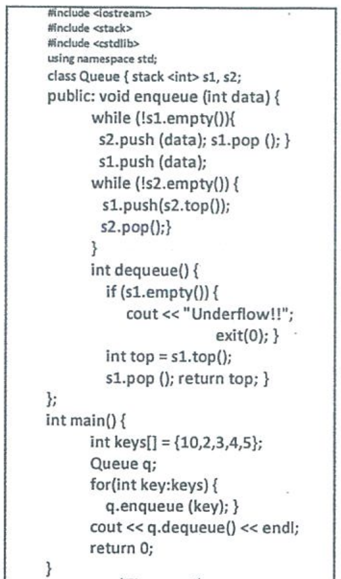
d) While writing the code in lab, your friend has made few mistakes in implementing a queue using the stack data structure as discussed in the class where the enqueue operation was made costly. When he runs the code, For an input: 10, 2, 3, 4, 5 he gets an output: 5. Actually, he should have got 10. Help him find out the error, if any. The code is given in **Fig.1**. Write down only the mistakes and its' correct counterpart.

[2]

Q.3 a) We have discussed Pre-order, In-order and Post-order traversal of a Binary tree in the class which are variations of depth first search. **Fig.2** gives implementation of level-order traversal using a queue (using extra space) which is a kind of breadth first search (BFS). Your task is to write the missing part of the code. Write only the code needed at Sections A and at B. Also, find out what is the run-time complexity?

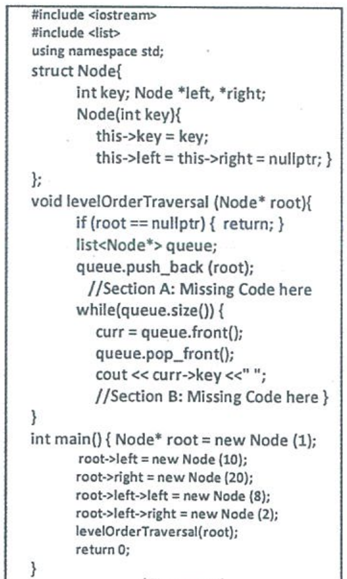
[3]

(P.T.O)



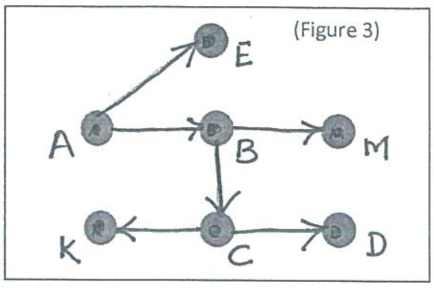
```
#include <iostream>
#include <stack>
#include <cstdlib>
using namespace std;
class Queue { stack <int> s1, s2;
public: void enqueue (int data) {
        while (!s1.empty()){
         s2.push (data); s1.pop (); }
         s1.push (data);
        while (!s2.empty()) {
         s1.push(s2.top());
         s2.pop();}
        }
        int dequeue() {
          if (s1.empty()) {
              cout << "Underflow!!";
                              exit(0); }
          int top = s1.top();
          s1.pop (); return top; }
};
int main() {
        int keys[] = {10,2,3,4,5};
        Queue q;
        for(int key:keys) {
          q.enqueue (key); }
        cout << q.dequeue() << endl;
        return 0;
}
```

(Figure 1)

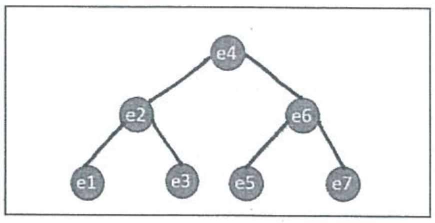


```
#include <iostream>
#include <list>
using namespace std;
struct Node{
        int key; Node *left, *right;
        Node(int key){
           this->key = key;
           this->left = this->right = nullptr; }
};
void levelOrderTraversal (Node* root){
        if (root == nullptr) { return; }
        list<Node*> queue;
        queue.push_back (root);
          //Section A: Missing Code here
        while(queue.size()) {
           curr = queue.front();
           queue.pop_front();
           cout << curr->key <<" ";
           //Section B: Missing Code here }
}
int main() { Node* root = new Node (1);
        root->left = new Node (10);
        root->right = new Node (20);
        root->left->left = new Node (8);
        root->left->right = new Node (2);
        levelOrderTraversal(root);
        return 0;
}
```

(Figure 2)



(Figure 3)



(Figure 4)

b) Design a hashmap to store the keys {34, 50, 98, 114, 146} using a table of size 16. Use a hash function as $h'(x) = x \bmod 16$, and $h(x) = (h'(x) + i^2) \bmod 16)$. Use quadratic probing to resolve the collisions, if any. Does the resolution scheme have a bearing on the table size? Explain your answer through this example. Also, show the hashtable contents with keys.

[2]

c) How do AVL trees help improve the performance over BSTs? Create an AVL tree with keys inserted into it in this order: **5, 17, 11**. After inserting all the keys if the tree is imbalanced (show with balance factors), find out what type of imbalance it is. Carry out the needed rotations to make it a balanced BST. At each step mention the rotation type, and also the balance factors of each node.

[3]

d) Given the In-order and Pre-order traversals of a **Binary tree** as below, draw the binary tree and find out the post-order traversal.
(**In-order**: dbeafkg, and **Pre-order**: abdekfg)

[2]

Q.4 a) Consider the graph shown in **Fig.3**. List down the nodes you will visit using BFS and DFS techniques. Why is the time complexity of iterating over the graph for these two traversal techniques O(V+E)? Give one-line answer.

[3]

b) Ankit likes AVL trees, but does not like rotations. He wants to create an AVL tree for the set of keys {8,9,10,11,12,13,14}. Find out the correct order of insertion of these keys into the AVL tree so that no rotations are needed. That is, which key should be inserted first, second, third and so on. Explain the rationale behind your answer.

[2]

c) Assume that the level order traversal of a binary **min-heap** H is: 10, 20, 16, 24, 40, 140, 42, 70, 64, 46. Now, if 18 and then 22 are inserted into H, what would be post-order traversal of H? Draw the Min-Heap clearly.

[3]

d) What is the average successful search time taken by binary search on a sorted array of 7 elements arranged in a binary search tree (BST) as shown in **Fig.4**. Show your calculations.

[2]

Part-B (Open Book): 30 Marks (Reference materials are allowed for this part)

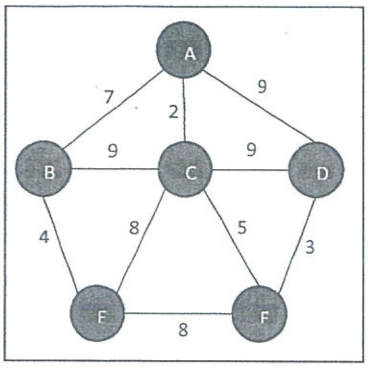
Q.5 a) A hash table of length 10 uses open addressing with hash function $h(x) = x \bmod 10$, and linear probing for collision handling. After inserting 6 values into an empty hash table, the table is as shown in **Fig. 5**. Find out whether any of these two sequences give a possible order in which the key values could have been inserted into the table. Explain your answer.
Sequence 1: 21, 46, 42, 34, 62, 23, 33 and Sequence 2: 46, 34, 42, 23, 62, 33, 21.

[4]

(P.T.O)

0	
1	21
2	42
3	23
4	34
5	62
6	46
7	33
8	
9	

(Figure 5)



(Figure 6)

```
#include <iostream>
#include <deque>
using namespace std;
class Stack {
private: deque <int> dq;
public:void push(int elem);
        void pop(); int top(); int size();
        bool empty();
};
void Stack:: push (int elem) {
     // enqueue this elem at the rear
        // (A: Code here)
     // dequeue and, enqueue
        for(int i=0; i < size()-1; i++) {
         //(B: Code here)
        }
}
```

(Figure 7: continued in Fig.8)

```
void Stack::pop() {
        int topElem = dq.front();
        dq.pop_front(); }
int Stack::top() { return dq.front(); }
int Stack::size() { return dq.size(); }
bool Stack::empty() {
        return size() == 0; }
int main() {
        Stack stack;
        stack.push(5); stack.push(6);
        stack.push(7);
        cout << stack.top() << '\n';
        stack.pop();
        cout << stack.top() << '\n';
        stack.pop();
        cout << stack.top() << '\n';
        return 0; }
```

(Figure 8)

b) Use Kruskal's algorithm discussed in the class to compute the Minimum cost Spanning Tree (**MST**) for the graph shown in **Fig.6.** Clearly show the steps involved to get the final MST.

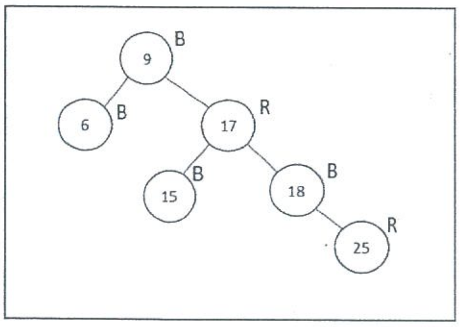
[4]

c) Given a deque as shown in **Fig.7** and **Fig.8** simulating a stack. The implementation consists of all the basic functionalities declared within the Stack class: push(), pop(), top(), size(), empty(). You need to write down the missing code (//A: Code here, and //B: Code here). What would be the output after you complete?

[4]

Q. 6 a) A Red-Black Tree (**RBT**) is as shown in **Fig.9**. Characters shown next to the nodes (storing keys) defines their color i.e. 'B' for black and 'R' for Red. Insert two keys, first 40 and then 75. After every insertion find out if the tree has become imbalanced. If yes, provide a remedy. Name what type of activities (rotation type, recoloring etc.) are performed at each step to get the final balanced tree.

[4]



```
int fun1 (int n) {

  if (n==0 || n ==1)

     return n;

  else

     return (2*fun1(n-1)+

               3*fun1(n-2));

}
```

```
int fun2 (int n) {
   int i; int X[N], Y[N], Z[N];
   X[0]=Y[0]=Z[0]=0;
   X[1]=1; Y[1]=2;Z[1]=3;
   for(i=2; i<=n; i++) {
      X[i]=Y[i-1]+Z[i-2];
      Y[i]=2*X[i]; Z[i]=3*X[i];
   }
   return X[n];
}
```

(Figure 9) (Figure 10) (Figure 11)

b) Fig.10 and **Fig.11** depict two functions fun1, and fun2. Find out the run-time complexity of these two functions. Which one will take more time to execute for a large value of 'n' and why? Explain.

[4]

c) In principle, it is possible to implement the List ADT via a singly linked list instead of a doubly linked list. Indeed, doing so uses less space as well. So, why are doubly linked lists a better choice? Explain.

[2]

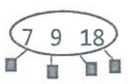
Q.7 a) Using a bottom up approach in Dynamic Programming (**DP**), find out the Longest Common Subsequence (LCS) of the below two sequences:
Sequence1: shoot
Sequence2: so
(Note: As discussed in the class, you need to show the contents of the table and find out the length of LCS and the characters forming LCS)

[3]

b) Apply Knuth Morris Pratt (**KMP**) pattern matching algorithm to find out if the below pattern is present in the given string. Show the contents of LPS table clearly which you might need to solve the problem.
String : aaaaba
Pattern : aaab

[3]

c) Given a (2,4) tree in **Fig. 12** storing 3 keys. Show the updates you would like to do on the tree to insert 10, 25, 31 and 45 in that order. Give reasons for your updates.



(Figure 12) [2]

--------~----------