National University of Computer and Emerging Sciences, Lahore Campus

Course Name:	Data Structures	Course Code:	CS2001
Degree Program:	BS (CS, SE, DS)	Semester:	Fall 2022
Exam Duration:	60 Minutes	Total Marks:	20
Paper Date:	12-Nov-2022	Weight	15
Section:	ALL	Page(s):	5
Exam Type:	Midterm-II		

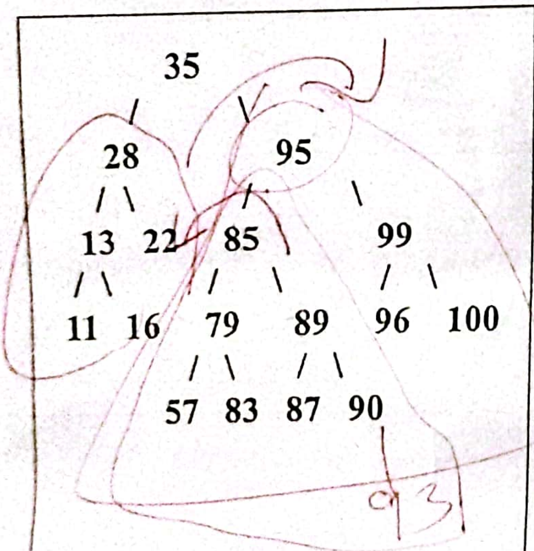
Student: Name: [redacted] **Roll No.** [redacted] **Section:** [redacted]

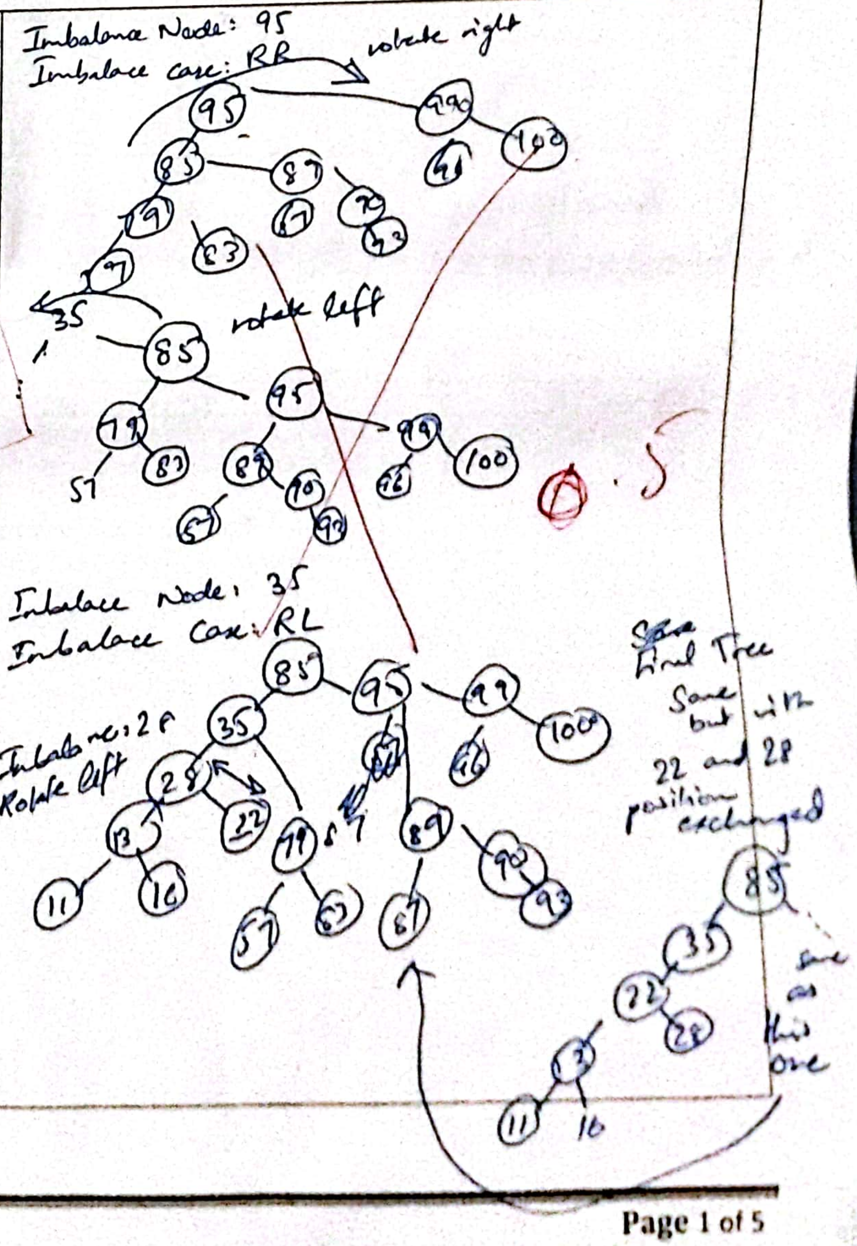
Instruction/Notes: Attempt all questions. Answer in the space provided. You can ask for rough sheets but will not be attached with this exam. **Answers written on rough sheet will not be marked.** Do not use pencil or red ink to answer the questions. In case of confusion or ambiguity make a reasonable assumption.

3+4=7

Question 1: [CLO: 1, 3] **(Marks: 5+5)**

a) Insert the value 93 in the following AVL tree and redraw the tree after rebalancing. You must show all working with the names of imbalance cases, nodes, and rotations performed.





Imbalance Node:	95, 85, 28
Imbalance Case:	RR, RL, RL
Rotations Performed:	3

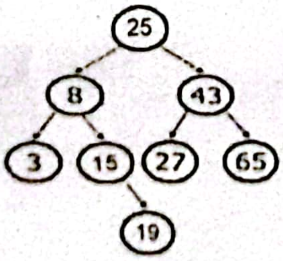
b) For each of the scenarios given below, suggest the most appropriate data structure chosen fr list and give appropriate reason of your choice.

(Arrays, linked-list (single, double, circular), Queue, Stack, tree(BST, AVL))

To create an index of important terms in a an electronic book	Arrays because arrays are good for keeping track of where items are.
To check whether a given seat number is reserved in a cinema or not	Arrays, as seat numbers are indexed and can be found in a ~~particular~~ easily through iterative search or directly.
To implement play list in repeat mode of a music player	Linked list (doubly) as the previous node can point to previous action taken and do it again.
To implement undo and redo functionality in a text editor	Stack, as you can push ~~&~~ and pop the same functionality over and over again.
To implement token system at bank.	Queue, as the it is FIFO structure, the people who have the first ~~token~~ turn in the system are the first to be served (out).

Question 2: [CLO: 3] **(Marks: 10)**

Write a **RECURSIVE** C++ function FINDVALUE in an Integer-based AVL tree class that takes a positive integer value as input and find the node with the smallest integer larger than or equal to the given input value. The function returns the node's data value and it must not take more than O(lgn) time.

EXAMPLE	
AVL- Tree 	Input is 8, then the value returned is 8 *As 8 is the smallest integer larger than or equal to the given value 8* Input is 26, then the value returned is 27 *As the integer 26 does not exists and 27 is the smallest integer larger than 26 in the given AVL tree* Input is 22, then the value returned is 25 *As 22 does not exists and 25 is the smallest integer larger than 22* Input is 67, then the value returned is -1 *As there is no integer equal to or larger than 67 in the given tree AVL tree*

```
Node * FINDVALUE ( Node *r , int value)
{
      if (r==nullptr) {return -1;}
    else if( ~~r->left~~ r->data ==value)
      { return r;}
     else if ( r->data < value)
       {     return FINDVALUE (r->left, value); }
      else if ( value > r->data)
        { return FINDVALUE (r->right, value); }

      if (
      if ( value < r-> left->data && (r->right->right->data >
            && r->right->data)
          { return r;}
        else if ( value < r-> right->data && (r->right->right->data
                 >r->right->data && r->right->left->data )>
                     r->right->data)
             { return r;}

          if ( value != r->data)
          { return -1;}

  void FIND VALUE( int value)
    { FINDVALUE (root, value);}
      }
```

Same

4