



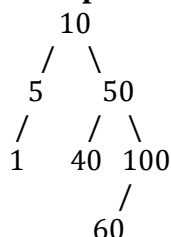
**EAST WEST UNIVERSITY**  
**Department of Computer Science and Engineering**  
**B.Sc. in Computer Science and Engineering Program**  
**Mid Term II Examination, Summer 2021 Semester**

**Course:** CSE207- Data Structures, Section-4  
**Instructor:** Tanni Mittra, Senior Lecturer, CSE Department  
**Full Marks:** 30 (25 will be counted for final grading)  
**Time:** 1 Hour and 30 Minutes

**Note:** There are **SIX** questions, answer **ALL** of them. Course Outcome (CO), Cognitive Level and Mark of each question are mentioned at the right margin.

- Suppose, you have an infix expression containing 12 characters. By following the algorithm of infix to postfix conversion, after performing operations on 11 characters you have following operator on stack,  $S = - ( /$ . Draw the state of the stack for each of the following cases considering as 12th character – [CO2,C3, Mark: 5]
  - A minus sign
  - A closing parenthesis
  - A number
  - A plus sign
  - A multiplication sign
- Consider a postfix expression  $ABC+-D*EF+/-$  and your student id. For example if your id is 2019-1-60-011 then ignore 0 of respective year (2019->219) and department Id 60(60->6). Then take value of the operand of above expression from your student id i.e.  $A=2, B=1, C=9, D=1, E=6, F=0$ . Now evaluate the value of the postfix expression using stack where the value of the operand is your student ID. [CO2,C3, Mark: 6]
- Consider a Binary Search Tree (BST) is already created and reference Node *\*root* contains the address of the root of the BST. **Write** a function *Descendent (Node \*data)* that will print all the descendants of a particular node and also print own left and/or right child of that particular node. For BST node consider the following Node class and you can use the functions of our created BST ADT. [CO2,C3, Mark: 5]

**Example:**



**Input:**

*Node = 50*

**Output:**

*Descendants : 40 100 60*

*Parent: 40 and 100*

**Class Node**

```

{
    int data;
    node
    *left,*right;
};
  
```

4. Find the contents of queue Q1 and stack S1 after the following code is executed with the data: 5, 7, 12, 4, -1, 4, 6, 0, 8, 67, 34, 23, 5, -2, 44, 0, 33, 22, 6, and 55? Show step by step output of the following code snippet. [CO2,C3, Mark: 5]

```

1. Q1 = createQueue; S1 = createStack;
2. Loop (not end of data)
3. Read number;
4. If (number >= 0)
5.   PushStack(S1, number)
6. Else
7.   Pop(S1, x);
8.   loop (not empty S1)
      8.1. Pop (S1,x);
      8.2. Enqueue(Q1,x);
9.   end loop;
10. End if;
11. End loop;

```

5. Suppose you have already developed a Stack ADT with push() and pop() operations. You already knew the parenthesis checking algorithm. Now consider you have to modify the parenthesis checking algorithm that will parse only a specific type of expression containing parenthesis like {} or (). Write a function called Modifyparen(char[] expression) that will parse the above mentioned specific type of expression [CO2,C3, Mark: 5]

6. a. Find minimum and maximum height of a binary tree having 31 numbers of nodes. [CO2,C2, Mark: 2]  
 b. Find minimum and maximum numbers of nodes of a binary tree of height 7. [CO2,C2, Mark: 2]