END SEMESTER EXAMINATION, JUNE-2024
DATA STRUCTURE AND ALGORITHMS (CSE 2001)

Programme: B.Tech. **Semester: 2nd**
Full marks: 60 **Time: 3 hours**

Subject Learning Outcome	*Taxonomy Level	Question Number	Marks
Ability to state and explain the basic programming syntax, semantics, building blocks.	L1, L2	2(a,b), 4(a,b,c)	10
Ability to develop java programs using programming constructs like conditional statements, looping, array, methods and class.	L1, L2, L3	2c, 3(a,b,c), 7c	10
Ability to analyze, debug and test the programs and correctly predict their outputs.	L2, L3	1(a,b,c), 10c	8
Ability to differentiate the behaviors of different data structures and their memory representations.	L3, L4	5(a,b,c), 9(a,b,c), 10a)	14
Ability to choose appropriate data structures that efficiently model the problem of interest.	L3, L4	6(a,b,c), 7(a,b)	10
Ability to apply advanced programming techniques for developing solutions of different problems.	L3, L4	8(a,b,c), 10b	8

*Bloom's taxonomy levels: Remembering (L1), Understanding (L2), Application (L3), Analysis (L4), Evaluation (L5), Creation (L6)

Answer all questions. All questions carry equal marks. All bits of each question carry equal marks.

1. (a) What will the output of the code given below with method call *show(3)*. **[2]**

```
public static void show(int n) {
    if(n>0) {
            show(n-1);
            System.out.print(n+" ");
            show(n-1);
    }}
```

(b) Find the time complexity of the following code using Big 'Oh' notation. [2]

```
public static void show(int n)  {
    for(int i=1;i<=n;i++)  {
            for(int j=1;j<=n;j*=2)  {
                System.out.println(i+" "+j);
            }
    }}
```

(c) What will be the output of the code in 1(b) with the method call **show(5);** in main method. [2]

2. (a) Can a method be both static and abstract? Justify your answer. [2]

(b) Define a class called **Book** with instance variables BName, BEdition, and BPrice. Use constructor to initialize the instance variables of the class. Add another instance method display() to display the book information. [2]

(c) Write a tester class for Book class of 2(b) to create an array of 5 objects and display the book details which have the maximum price. [2]

3. (a) Design a Java program for managing student attendance and checking the eligibility for exams. Create a **Student** class with attributes such as 'totalClassesHeld' and 'classesAttended'. Implement methods to calculate the percentage of classes attended and to check if the student is eligible to sit for exams based on the attendance criteria (i.e., attendance should be at least 75%). [2]

(b) Write a Java program to create a banking system with the following: an interface **Account** with abstract methods void deposit(int amt) and void withdraw(int amt). Two classes **SavingsAccount**, and **CurrentAccount** implements the Account interface. Both these classes are with attributes accountNumber, balance and override the methods of interface. You can add appropriate constructors and a display() method. Write a tester class to create object of the classes SavingsAccount and CurrentAccount, set their details and display after some deposit, withdraw operations. [2]

Faculty of Engineering and Technology (ITER), Siksha 'O' Anusandhan Deemed to be University

(c) Create a class **Date** with attributes dd, mm, yy and a method displayDate(). Write a Java program to create two Date objects and display the later Date. [2]

4. (a) Create a generic class **Calculator** with a method divide(T n1, T n2). Write a Java program to divide two appropriate numbers. [2]

(b) Identify possible exceptions that may occure in 4(a). [2]

(c) Write a Java program to compute and print square root of an integer. If the numer is -ve your program should throw an user defined exception **NegativeNumberException** and print a message "Negative Number Not Allowed". [2]

5. (a) Convert the following infix expression into postfix using stack.
G*H^I+(J-K^L/M)*N [2]

(b) Evaluate the following postfix expression using stack.
12, 2, 3, *, /, 3, 4, *, *, 2, 2, +, /
What is the stack top element after third * operation. [2]

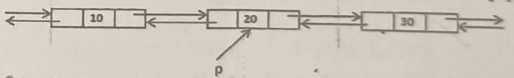
(c) Count the number of push and pop operations performed in the evaluation process of 5(b). [2]

6. (a) Consider a class **Node** for single linkedlist that store student information such as regNo and mark. Write a Java method to insert a new node at the end with header *void insEnd()*. [2]

(b) Write the loop statements to insert five such nodes in a linkedlist by calling the *void insEnd()* method written in 6(a) and display the linkedlist. [2]

(c) Write the java statements to find reference of both last and second last node in a single linkedlist using only one loop. [2]

7. (a) Write the Java statements to delete the node that is referred by the reference variable **p** in a double linkedlist as shown below. [2]



(b) Write the Java statements to insert a new node before the node referred by **p** in 7(a). [2].

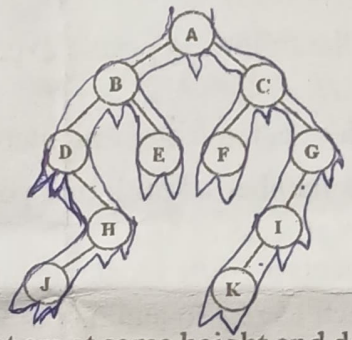
(c) Write a Java method to insert a new node at any node number in a double linkedlist. [2].

8. (a) Write a stack-based algorithm to reverse a string. Describe how the algorithm works with the input "INDIA". [2]

   (b) Write Java methods to implement insert operation in a linear queue using linkedlist. [2]

   (c) Let's consider the following operations in the array implementation of a linear queue of max size (MAX=5): ins(10), ins(20), ins(30), ins(40), del(), ins(50), ins(60), del(), ins(70), ins(80), del(), del(). ins() and del() are methods for insert and delete operations. Find the front, rear index values. Also find queue[front] and queue[rear]. [2]

9. (a) Write the in-order and post-order traversal sequence for the following diagram. [2]



   (b) List all the nodes that are at same height and depth in the tree given in 9(a). Find the degree of these nodes. [2]

   (c) Given the following information, construct the binary tree.
   Pre-order: PQZSUWRTVXY
   In-Order : ZQUWSPTXVRY [2]

10. (a) Draw the binary search tree by performing the following operations in the given sequence: ins(50), ins(20), ins(70), ins(10), ins(35), ins(30), ins(32), del(20), ins(34), ins(33), ins(60), ins(80), ins(65), ins(75), ins(63), del(70). [2]

    (b) Write a recursive method to implement binary search in an array.
    Method header: *int binarySearch(int a[], int lb, int ub)* [2]

    (c) What is/are the necessary conditions for binary search? Compare the number of comparisons required to search an element in worst case for linear search and binary search considering size of the array as 1024. [2]