| U.S.N. | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|

# B.M.S. College of Engineering, Bengaluru-560019

**Autonomous Institute Affiliated to VTU**

## April 2024 Semester End Main Examinations

**Programme: B.E.** **Semester: III**
**Branch: Artificial Intelligence and Machine Learning** **Duration: 3 hrs.**
**Course Code: 22AM3PCDST** **Max Marks: 100**
**Course: Data Structures**

**Instructions**: 1. Answer any FIVE full questions, choosing one full question from each unit.
2. Missing data, if any, may be suitably assumed.





### UNIT - I

1 a) What are the various dynamic memory allocation functions supported in C. How do they help in managing data and memory efficiently within data structures? Explain with examples for each along with Syntax. **10**

b) Write a C program to perform insert rear, delete rear, search for a key element and display operations on a Singly Linked List with a header node. **10**

### UNIT - II

2 a) Using the diagrammatic representations and C coding sequence, explain how to perform the following operations on a Circular Doubly Linked List (CDLL): **10**

i) Insert an element after the given key element
ii) Concatenate 2 CDLLs

b) Define Stack. Write a C program to demonstrate the various basic stack operations listed in its Abstract Data Type (ADT). **10**

**OR**

3 a) Write C functions to perform the following operations on a Doubly Linked List (DLL) **10**

i. Insert a node at the front end
ii. Delete a node from the rear end
iii. Count the number of nodes
iv. Display

b) Using stack data structure, convert the given infix expressions to their equivalent postfix forms. Tabulate the conversion steps and elaborate on the same. **10**

i) ( A + B ^ D ) / ( E – F ) + G
ii) A * ( B + D ) / E – F * ( G + H / K )

## UNIT - III

| | | | |
|---|---|---|---|
| 4 | a) | What is the advantage of circular queue over linear queue? Write a C program to implement the working of a circular queue on integer data type using dynamic memory allocation functions along with modules for insert(), delete(), display(), Overflow() and underflow(). | **10** |
| | b) | List the real time applications which can be programmed easily using recursion than the iterative solutions. | **4** |
| | c) | Implement a recursive call to compute factorial of a given number and illustrate the stack trace pictorially. | **6** |

## UNIT - IV

| | | | |
|---|---|---|---|
| 5 | a) | Using the given inorder - DJGBHEAFKIC and postorder - JGDHEBKIFCA tree traversal sequences of a Binary Tree, construct an equivalent Binary Search Tree and write the preorder traversal sequence of the resultant tree. | **10** |
| | b) | Using graphical representations, explain the concept of Threaded Binary Tree, its types, storage representations and working in detail. | **10** |

## UNIT - V

| | | | |
|---|---|---|---|
| 6 | a) | Construct a balanced search tree by applying the concept of AVL tree rotations for the given sequence of elements: 15, 25, 35, 30, 32, 12, 9 | **10** |
| | b) | Write the Red-Black tree algorithm and explain its working using an example of your own. | **10** |

**OR**

| | | | |
|---|---|---|---|
| 7 | a) | Construct a balanced search tree by applying the concept of Red Black trees for the given order of elements: 5, 80, 10, 65, 15, 55, 25 | **10** |
| | b) | Explain splay trees and its various rotations supported to enhance the search efficiency of a Binary Search Tree. | **10** |

******