

RAMANUJAN COLLEGE

UNIVERSITY OF DELHI

SOUTH CAMPUS IN KALKAJI 110019

PRACTICAL FILE

Data Structures

Submitted By: Yuvraj Singh

Examination Roll Number: 23020570056

Class Roll Number: 20231471

Submitted to: Dr. Sahil Pathak

Index

|  |  |  |  |
| --- | --- | --- | --- |
| S.NO. | Practical | Page No. | Remarks |
| 1 | Write a program to implement singly linked list as an ADT that supports the following operations:  i. Insert an element x at the beginning and at *ith* position of the singly linked list  ii. Remove an element from the beginning and from *ith* position of the singly linked list  ii. Search for an element x in the singly linked list and return its pointer | 4 |  |
| 2 | Write a program to implement doubly linked list as an ADT that supports the following operations:  i. Insert an element x at the beginning and at the end of the doubly linked list  ii. Remove an element from the beginning and from the end of the doubly linked list | 6 |  |
| 3 | Write a program to implement circular linked list as an ADT which supports the following operations.  i. Insert an element x in the list  ii. Remove an element from the list  ii. Search for an element x in the list and return its pointer | 8 |  |
| 4 | Implement Stack as an ADT and use it to evaluate a prefix/postfix expression. | 10 |  |
| 5 | Implement Queue as an ADT. | 12 |  |
| 6 | Write a program to implement Binary Search Tree as an ADT which supports the following operations:  i. Insert an element x  ii. Delete an element x  iii. Search for an element x in the BST  iv. Display the elements of the BST in preorder, inorder, and postorder traversal | 14 |  |
| 7 | Write a program to implement insert and search operation in AVL trees. | 17 |  |

# ***Acknowledgment***

**I would like to express my sincere gratitude to my instructor, Dr. Sahil Pathak, for providing me with the opportunity to work on this series of practical. Their invaluable guidance and support have been instrumental in understanding the core concepts of computer graphics.**

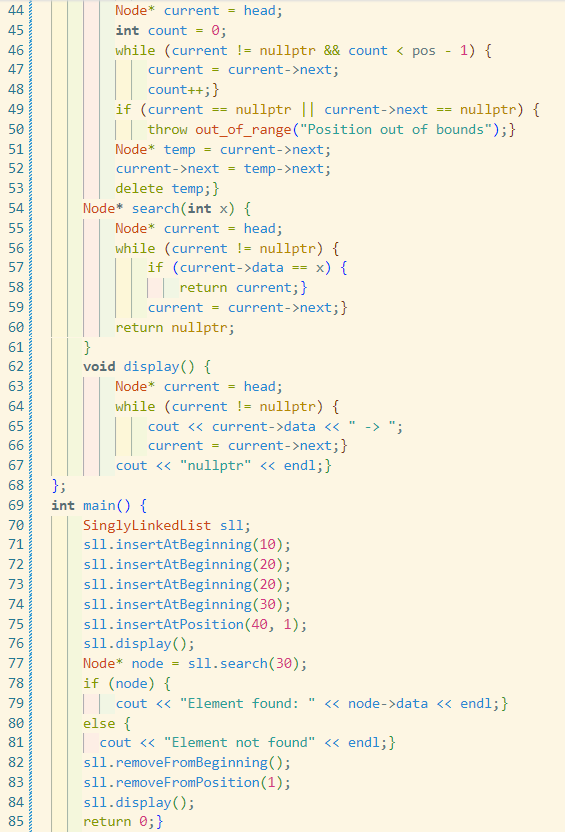
**I would also like to thank Ramanujan College, University of Delhi, for offering such a rich curriculum that fosters technical learning and hands-on experience. The practical applications of various algorithms and transformation techniques have deepened my knowledge in the field.**

**Furthermore, I would like to thank my peers for their encouragement and collaboration during the course of this project. Their inputs have enhanced my learning experience and contributed to a stimulating environment for problem-solving.**

1. Write a program to implement singly linked list as an ADT that supports the following operations:
   1. Insert an element x at the beginning and at *ith* position of the singly linked list
   2. Remove an element from the beginning and from *ith* position of the singly linked list
   3. Search for an element x in the singly linked list and return its pointer

**Code:**

****

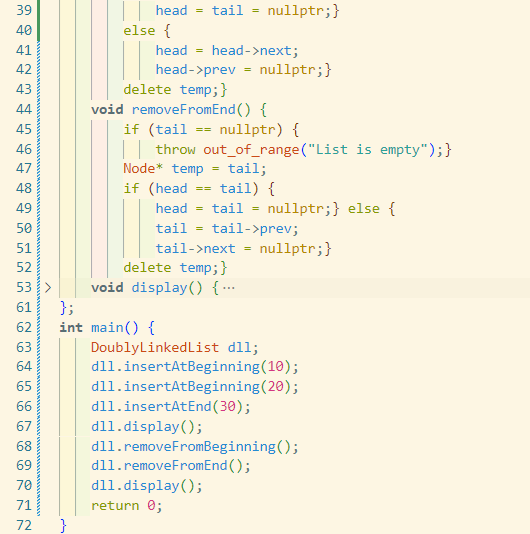
****

**Output:**

****

1. Write a program to implement doubly linked list as an ADT that supports the following operations:
   1. Insert an element x at the beginning and at the end of the doubly linked list
   2. Remove an element from the beginning and from the end of the doubly linked list

****

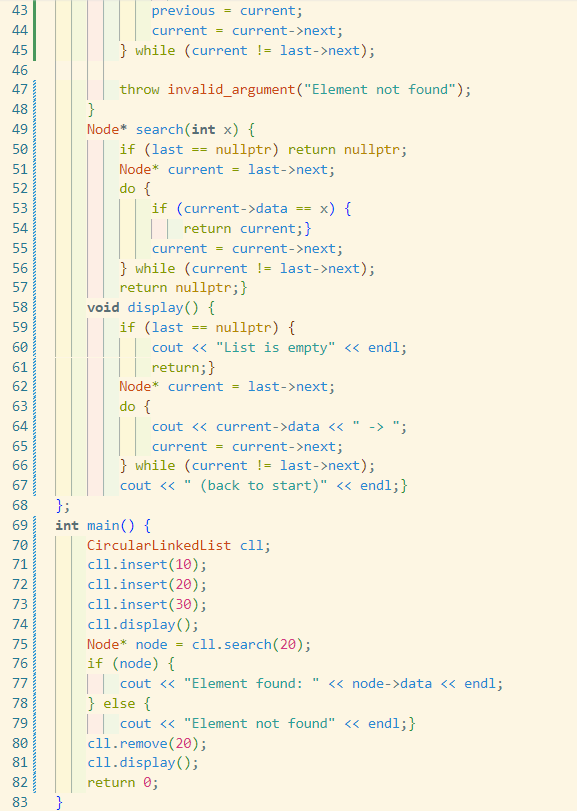
****

**Output:**

****

1. Write a program to implement circular linked list as an ADT which supports the following operations
   1. Insert an element x in the list
   2. Remove an element from the list
   3. Search for an element x in the list and return its pointer

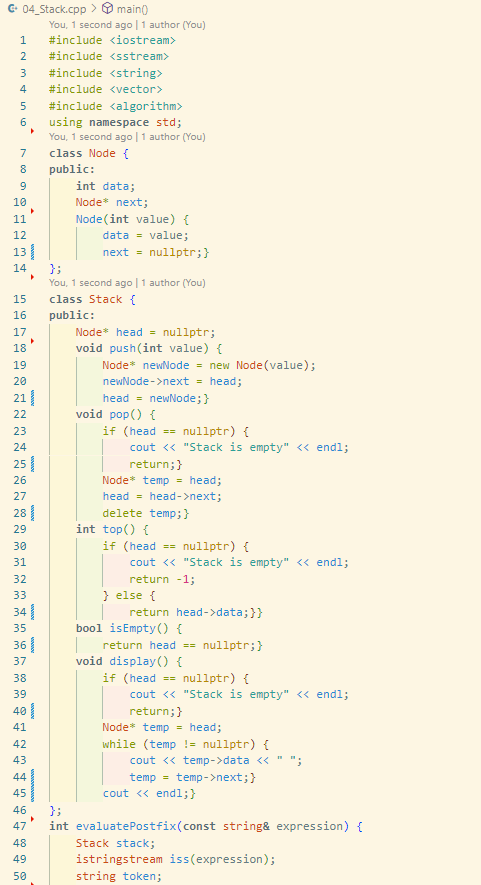
****

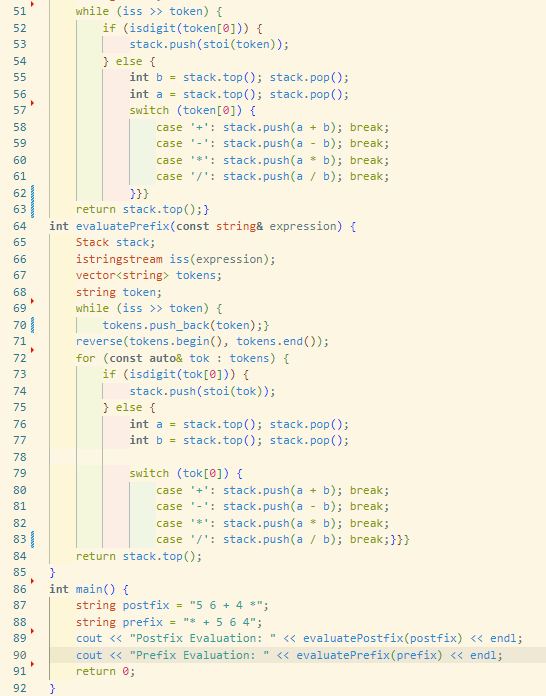
****

**Output:**

****

4. Implement Stack as an ADT and use it to evaluate a prefix/postfix expression.

****

****

**Output:**

****

5. Implement Queue as an ADT.

**Code:**





**Output:**

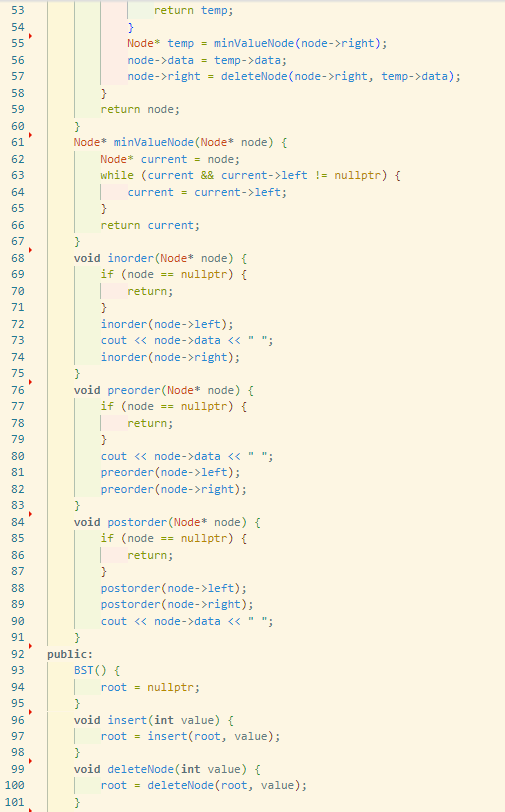
****

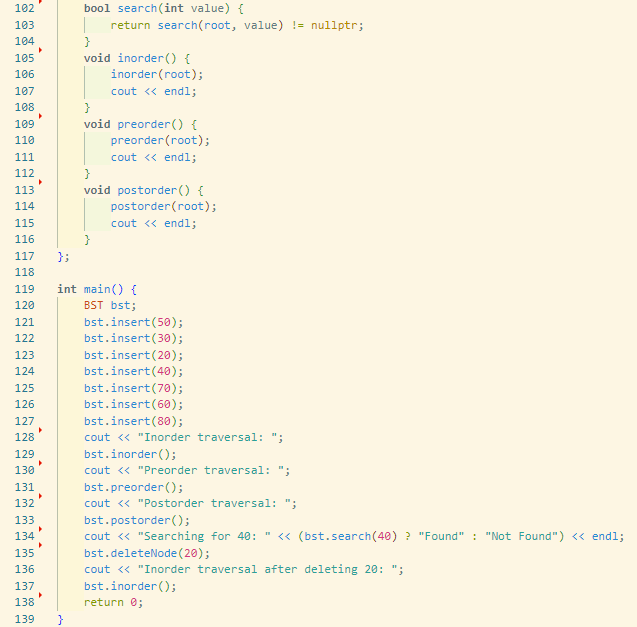
6. Write a program to implement Binary Search Tree as an ADT which supports the following operations:

1. Insert an element x
2. Delete an element x
3. Search for an element x in the BST
4. Display the elements of the BST in preorder, inorder, and postorder traversal

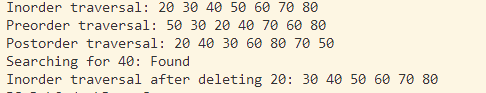
**Output:**

****

****

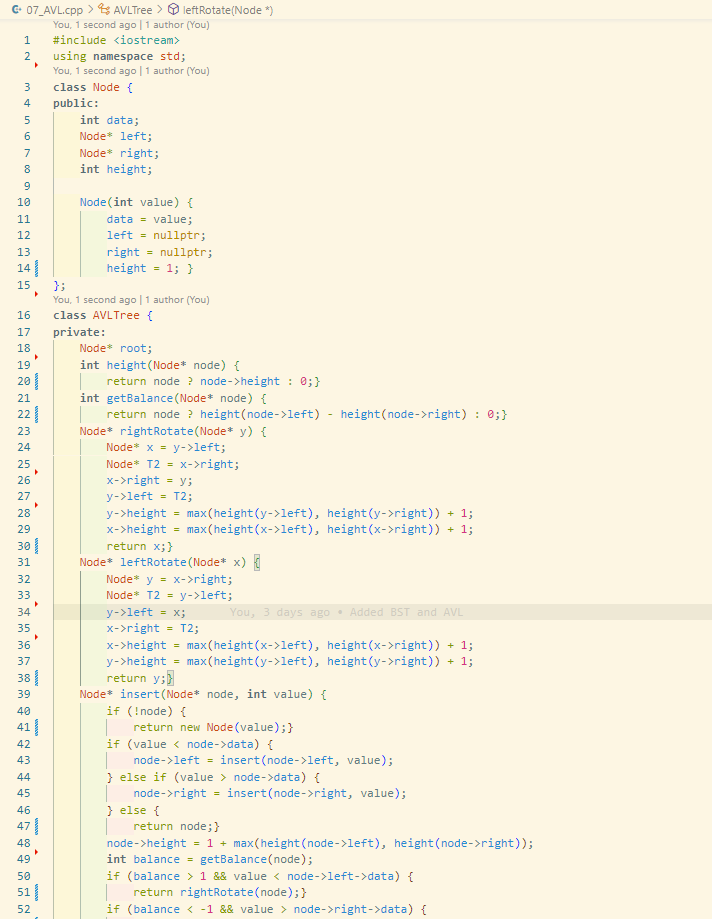
****

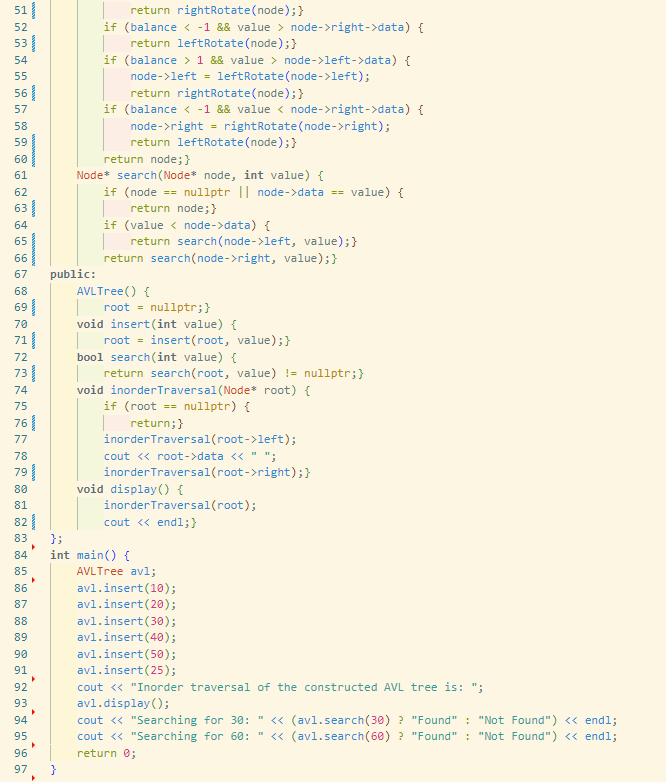
**Output:**

****

7. Write a program to implement insert and search operation in AVL trees.

**Code:**

****

****

**Output:**

****