**Assignment 5**

**Name** : Komal Potdar

**Roll No**.: 92

**PRN No**.: 12320165

**Div**: CS B SY

**Batch**: 3

**1. Write a menu driven  program in C to perform following  operations  on BST**

**a. Create**

**b. Insert**

**c. Inorder traversal**

**d. Mirror tree**

**e. Height of the tree**

**f. Level wise display**

**g. Leaf node display**

**h.Exit**

|  |
| --- |
| #include <stdio.h>  #include <stdlib.h>  struct Node {  int data;  struct Node \*left;  struct Node \*right;  };  struct Node\* createNode(int data) {  struct Node\* newNode = (struct Node\*)malloc(sizeof(struct Node));  newNode->data = data;  newNode->left = NULL;  newNode->right = NULL;  return newNode;  }  struct Node\* insert(struct Node\* root, int data) {  if (root == NULL)  return createNode(data);  if (data < root->data)  root->left = insert(root->left, data);  else if (data > root->data)  root->right = insert(root->right, data);  return root;  }  void inorderTraversal(struct Node\* root) {  if (root != NULL) {  inorderTraversal(root->left);  printf("%d ", root->data);  inorderTraversal(root->right);  }  }  struct Node\* mirrorTree(struct Node\* root) {  if (root == NULL) return NULL;  // Swap left and right subtrees  struct Node\* temp = root->left;  root->left = mirrorTree(root->right);  root->right = mirrorTree(temp);  // Print the mirror tree  printf("%d ", root->data);  return root;  }  int height(struct Node\* root) {  if (root == NULL) return 0;  int leftHeight = height(root->left);  int rightHeight = height(root->right);  return (leftHeight > rightHeight ? leftHeight : rightHeight) + 1;  }  void printLevel(struct Node\* root, int level) {  if (root == NULL) return;  if (level == 1) printf("%d ", root->data);  else if (level > 1) {  printLevel(root->left, level - 1);  printLevel(root->right, level - 1);  }  }  void levelWiseDisplay(struct Node\* root) {  int h = height(root);  for (int i = 1; i <= h; i++) {  printLevel(root, i);  printf("\n");  }  }  void displayLeafNodes(struct Node\* root) {  if (root == NULL) return;  if (root->left == NULL && root->right == NULL)  printf("%d ", root->data);  else {  displayLeafNodes(root->left);  displayLeafNodes(root->right);  }  }  void menu() {  printf("\na. Create\nb. Insert\nc. Inorder Traversal\nd. Mirror Tree\ne. Height of the Tree\nf. Level Wise Display\ng. Display Leaf Nodes\nh. Exit\n");  }  int main() {  struct Node\* root = NULL;  int choice, data;  do {  menu();  printf("Enter your choice: ");  scanf(" %c", &choice);  switch(choice) {  case 'a':  printf("Enter the root value: ");  scanf("%d", &data);  root = createNode(data);  break;  case 'b':  printf("Enter the value to insert: ");  scanf("%d", &data);  root = insert(root, data);  break;  case 'c':  printf("Inorder Traversal: ");  inorderTraversal(root);  printf("\n");  break;  case 'd':  printf("Mirror Tree: ");  mirrorTree(root);  printf("\n");  break;  case 'e':  printf("Height of the Tree: %d\n", height(root));  break;  case 'f':  printf("Level Wise Display:\n");  levelWiseDisplay(root);  break;  case 'g':  printf("Leaf Nodes: ");  displayLeafNodes(root);  printf("\n");  break;  case 'h':  printf("Exiting...\n");  break;  default:  printf("Invalid choice! Please enter a valid option.\n");  }  } while(choice != 'h');  return 0;  } |

Output:





