**Practical no:04**

**Program**:

#include<iostream>

using namespace std;

class node{

public:

node\* left;

node\* right;

int data;

};

class linked{

public:

node\* root=NULL;

node\* create(int val)

{

node\* t = new node();

t->data = val;

t->left = t->right = NULL;

return t;

}

node\* insert(node\* root, int val)

{

if(root == NULL)

{

root = create(val);

}

else if(val <= root->data){

root->left = insert(root->left, val);

}

else{

root->right = insert(root->right, val);

}

return root;

}

void inorder(node\* root)

{

if(root != NULL)

{

inorder(root->left);

cout << root->data << ' ';

inorder(root->right);

}

}

bool search(node\* root, int val)

{

if(root == NULL){

return false;

}

else if(root->data == val)

{

return true;

}

else if(val < root->data)

{

return search(root->left, val);

}

else{

return search(root->right, val);

}

}

void swap(node\* ptr)

{

if(ptr == NULL)

return;

swap(ptr->left);

swap(ptr->right);

node\* t = ptr->left;

ptr->left = ptr->right;

ptr->right = t;

}

node\* invert(node\* root)

{

swap(root);

return root;

}

int height(node\* root)

{

if(root == NULL)

{

return -1;

}

int lefth = height(root->left);

int righth = height(root->right);

return max(lefth, righth) + 1;

}

void mini(node\* root)

{

node\* t = root;

while(t->left != NULL)

{

t = t->left;

}

cout << "Minimum element is: " << t->data << endl;

}

};

int main()

{

linked l;

int n;

cout << "Enter the number of nodes: ";

cin >> n;

int arr[n];

cout << "Enter the elements: " << endl;

for(int i = 0; i < n; i++)

{

cin >> arr[i];

l.root = l.insert(l.root, arr[i]);

}

cout << "1. Insert the node:" << endl;

cout << "2. Display BST:" << endl;

cout << "3. Minimum Element:" << endl;

cout << "4. Height of the BST:" << endl;

cout << "5. Search element:" << endl;

cout << "6. Swap left and right nodes:" << endl;

cout << "7. Exit!" << endl;

int choice;

while(true)

{

cout << "\nEnter your choice: ";

cin >> choice;

switch(choice)

{

case 1:

char c;

cout << "Do you want to enter another number (y/n): ";

cin >> c;

if(c == 'y')

{

int num;

cout << "Enter number: ";

cin >> num;

l.root = l.insert(l.root, num);

}

break;

case 2:

cout << "Inorder traversal: ";

l.inorder(l.root);

cout << endl;

break;

case 3:

l.mini(l.root);

break;

case 4:

cout << "Height: " << l.height(l.root) << endl;

break;

case 5:

cout << "Enter the element to search: ";

int s;

cin >> s;

if(l.search(l.root, s))

cout << "Element is present!" << endl;

else

cout << "Element is not present!" << endl;

break;

case 6:

l.root = l.invert(l.root);

cout << "Inverted traversal: ";

l.inorder(l.root);

cout << endl;

break;

case 7:

cout << "Thanks!" << endl;

return 0;

default:

cout << "Invalid choice! Please try again." << endl;

break;

}

}

}

**Output:**

Enter the number of nodes: 5

Enter the elements:

30

20

10

25

40

1. Insert the node:

2. Display BST:

3. Minimum Element:

4. Height of the BST:

5. Search element:

6. Swap left and right nodes:

7. Exit!

Enter your choice: 2

Inorder traversal: 10 20 25 30 40

Enter your choice: 1

Do you want to enter another number (y/n): y

Enter number: 50

Enter your choice: 2

Inorder traversal: 10 20 25 30 40 50

Enter your choice: 3

Minimum element is: 10

Enter your choice: 4

Height: 2

Enter your choice: 5

Enter the element to search: 50

Element is present!

Enter your choice: 5

Enter the element to search: 66

Element is not present!

Enter your choice: 6

Inverted traversal: 50 40 30 25 20 10

Enter your choice: 7

Thanks!