**6. Detailed code with Relevant comments:->**

package tree;

import java.util.Scanner;

public class bstpractise {

// node class is created

public class node {

int data;

node l;

node r;

// constructor is created with following values

node(int data) {

this.data = data;

this.l = null;

this.r = null;

}

}

public static node root = null;

// function to create Binary search tree

public node create(node root) {

Scanner sc = new Scanner(System.in);

int data = sc.nextInt();

while (data != -1) {

root = insert(root, data);

data = sc.nextInt();

}

return root;

}

// function to insert node in Binary search tree

public node insert(node root, int d) {

if (root == null) {

root = new node(d);

return root;

} else if (d < root.data) root.l = insert(root.l, d);

else {

root.r = insert(root.r, d);

}

return root;

}

// function to Seach element in Binary search tree using iterative approach

public void iterativesearch(node root, int dta) {

node temp = root;

int flag = 0;

while (temp != null) {

if (temp.data == dta) {

System.out.println(dta + " Is present");

flag = 1;

break;

} else if (dta < root.data) {

temp = temp.l;

} else {

temp = temp.r;

}

}

if (flag == 0) System.out.println(dta + " is not present");

}

// function to Seach element in Binary search tree using recurssive approach

public void recursivesearchInBst(node root, int d7) { // re

if (root == null) {

System.out.println(d7 + " is NOT present");

return;

}

if (d7 == root.data) {

System.out.println(d7 + " is present");

} else if (d7 < root.data) {

recursivesearchInBst(root.l, d7);

} else {

recursivesearchInBst(root.r, d7);

}

}

// function to insert Element in Binary search tree

public node inBst(node root, int d1) {

if (root == null) {

root = new node(d1);

return root;

}

if (d1 > root.data) {

root.r = inBst(root.r, d1);

} else {

root.l = inBst(root.l, d1);

}

return root;

}

static node prev = null; // Helper variable for checking if the tree is a BST

//function to check if the binary tree is a BST or NOT

public boolean isBST(node root) {

if (root != null) {

if (!isBST(root.l)) {

return false;

}

if (prev != null && root.data <= prev.data) {

return false;

}

prev = root;

return isBST(root.r);

}

return true;

}

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

bstpractise t = new bstpractise();

int ch;

root = null;

// for a while loop

while (true) {

// List of choise

System.out.println("1.Create");

System.out.println("2.Search by iterative method");

System.out.println("3.Search by recursion method");

System.out.println("4.Adding new element in BST");

System.out.println("5.Check whether tree is BST or Not");

System.out.println("0.Exit");

System.out.print("Enter choise = ");

ch = sc.nextInt();

switch (ch) {

case 1:

System.out.println("Enter data to create BST");

root = t.create(root);

System.out.println();

break;

case 2:

System.out.println("Enter the num that you want to search in tree ");

int dta = sc.nextInt();

t.iterativesearch(root, dta);

System.out.println();

break;

case 3:

System.out.println("Enter the num that you want to search in tree ");

int data = sc.nextInt();

t.recursivesearchInBst(root, data);

System.out.println();

break;

case 4:

System.out.println("Enter data that you want to insert");

int x = sc.nextInt();

root = t.inBst(root, x);

System.out.println("data is added");

System.out.println();

break;

case 5:

boolean t1 = t.isBST(root);

if (t1) {

System.out.print("Tree is BST");

} else {

System.out.print("Tree is not BST");

}

System.out.println();

break;

case 0:

System.exit(0);

break;

}

}

}

}

**7. Results :->**

1.Create

2.Search by iterative method

3.Search by recursion method

4.Adding new element in BST

5.Check whether tree is BST or Not

0.Exit

Enter choise = 3

Enter the num that you want to search in tree

9

9 is present

Enter choise = 3

Enter the num that you want to search in tree

101

101 is NOT present

1.Create

2.Search by iterative method

3.Search by recursion method

4.Adding new element in BST

5.Check whether tree is BST or Not

0.Exit

Enter choise = 2

Enter the num that you want to search in tree

3

3 Is present

Enter choise = 2

Enter the num that you want to search in tree

100

100 is not present

1.Create

2.Search by iterative method

3.Search by recursion method

4.Adding new element in BST

5.Check whether tree is BST or Not

0.Exit

Enter choise = 5

Tree is BST

1.Create

2.Search by iterative method

3.Search by recursion method

4.Adding new element in BST

5.Check whether tree is BST or Not

0.Exit

Enter choise = 0

Process finished with exit code 0

1.Create

2.Search by iterative method

3.Search by recursion method

4.Adding new element in BST

5.Check whether tree is BST or Not

0.Exit

Enter choise = 4

Enter data that you want to insert

18

data is added