Lab Exam - **Pranit Tondvalkar**

22.01.2022 **220960920085**

1. Write a Java program to

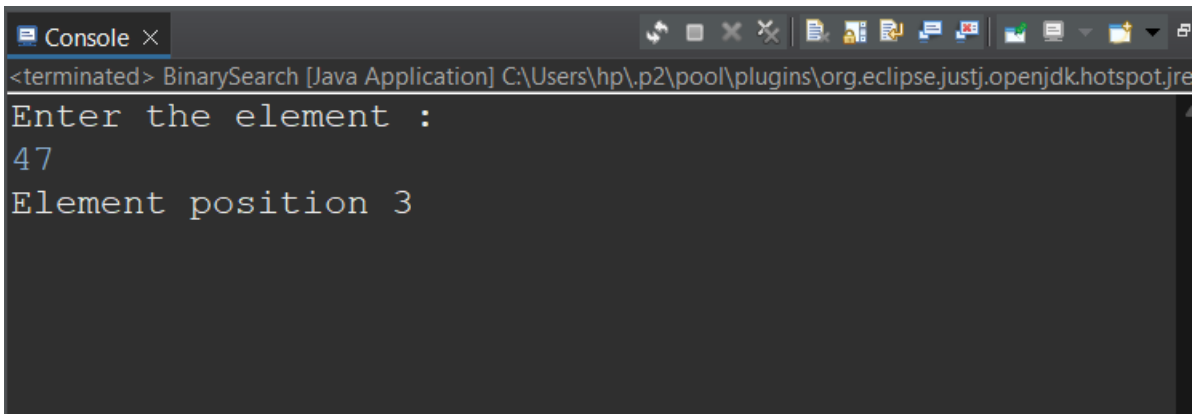
a. Perform binary search operation

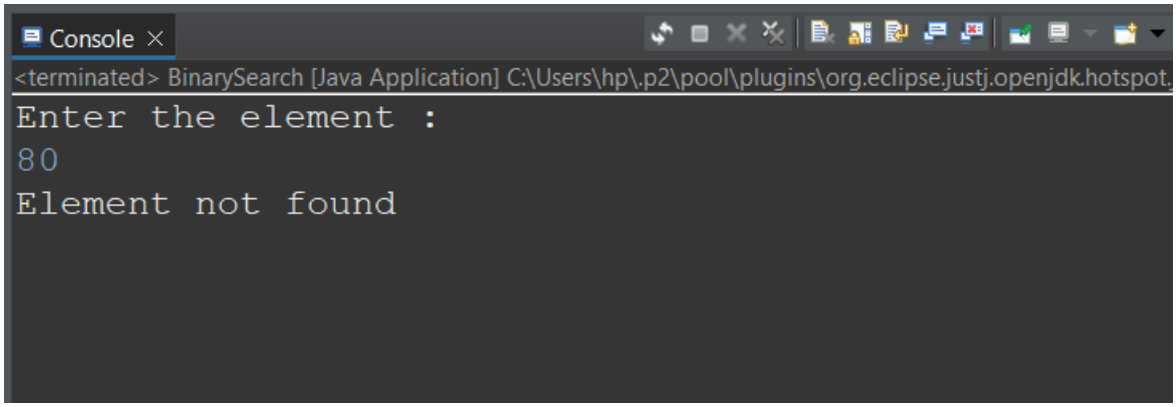
- **Class BinarySearch –**

```
import java.util.Scanner;

public class BinarySearch {

public int search(int arr[], int low, int high, int
element)

{

if(low == high) {

if(arr[low] == element)

return low;

else

return -1;

}

int mid = (low + high) / 2;

if (element == arr[mid])

return mid;

if (element > arr[mid])

return search(arr, (mid + 1), high, element);

else
```

```
return search(arr, low, (mid - 1), element);

}

public static void main(String[] args) {

BinarySearch obj = new BinarySearch();

int arr[] = {18, 36, 47, 58, 69, 100};

Scanner sc = new Scanner(System.in);

System.out.println("Enter the element : ");

int element = sc.nextInt();

int n = arr.length - 1;

int position = obj.search(arr, 0, n, element);

if(position >=0)

System.out.println("Element position "+(position+1));

else

System.out.println("Element not found");

}

}
```

- **Output –**





b. Execute tree traversal in postorder –

- **Class Node –**

```
package com.TreeTraversalPostOrder.entity;

public class Node {

int data;

public Node left;

public Node right;

public Node(int item)

{

data = item;

left = right = null;

}

}
```

- **Class BinaryTree –**

```
package com.TreeTraversalPostOrder.entity;

public class BinaryTree {

public Node root;
```

```
public void postOrder(Node node) {

if (node == null) {

return;

}

postOrder(node.left);

postOrder(node.right);

System.out.print(node.data + " ");

}

}
```

- **Main Method-**
- **Class PostOrderMain –**

```
package com.TreeTraversalPostOrder.main;

import com.TreeTraversalPostOrder.entity.BinaryTree;

import com.TreeTraversalPostOrder.entity.Node;

public class PostOrderMain {

public static void main(String[] args) {

BinaryTree tree = new BinaryTree();

tree.root = new Node(15);

tree.root.left = new Node(28);

tree.root.right = new Node(37);

tree.root.left.left = new Node(48);

tree.root.left.right = new Node(52);

System.out.println("Postorder traversal of binary
tree is :");
```

```
tree.postOrder(tree.root);

}

}
```

- **Output –**