LAB EXAM

Data Structure and Algorithms

Mangesh Pawar

1. Write a Java program to

a. Perform quick sort

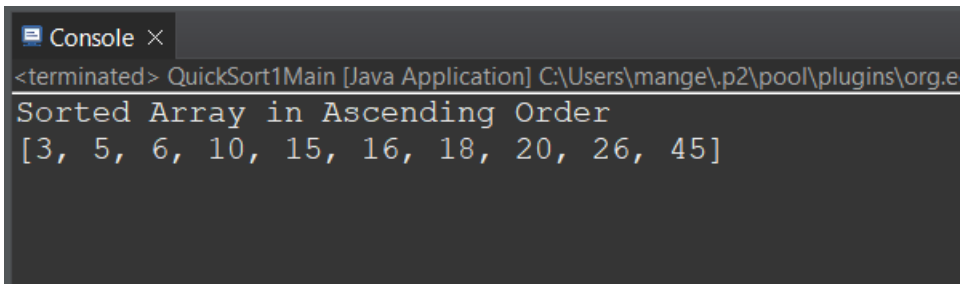
```
package com.sorting;
public class QuickSort1 {
static int partition(int arr[], int low, int high) {
int pivot = arr[high];
int i = (low - 1);
for (int j = low; j < high; j++) {
if (arr[j] <= pivot) {
i++;
int temp = arr[i];
arr[i] = arr[j];
arr[j] = temp;
}
}
int temp = arr[i + 1];
arr[i + 1] = arr[high];
arr[high] = temp;
return (i + 1);
}
static void quickSort(int arr[], int low, int high) {
if (low < high) {
int pi = partition(arr, low, high);
quickSort(arr, low, pi - 1);
quickSort(arr, pi + 1, high);
}
```

```
}
}
```

Main Method:

```
package com.sorting;
import java.util.Arrays;
public class QuickSort1Main {
public static void main(String args[]) {
int[] arr = { 45, 16, 26, 18, 6, 15, 3, 10, 5, 20 };
int n = arr.length;
QuickSort1.quickSort(arr, 0, n - 1);
System.out.println("Sorted Array in Ascending Order ");
System.out.println(Arrays.toString(arr));
}
}
```

Output:



b. Perform preorder tree traversal

```
package com.treetravesal;
public class PreTraversal {
Node root;
public static class Node {
int key;
Node left;
Node right;
public Node(int key) {
```

```
this.key = key;
}
}
public void preOrder(Node node1) {
if(node1!=null) {
System.out.print(" "+node1.key);
preOrder(node1.left) ;
preOrder(node1.right);
}
}
public static Node binaryTree() {
Node rootNode=new Node(6);
Node node3= new Node(3);
Node node4= new Node(4);
Node node5= new Node(5);
Node node7= new Node(7);
Node node8= new Node(8);
Node node9= new Node(9);
rootNode.left=node4;
rootNode.right=node8;
node4.left=node3;
node4.right=node5;
node8.left=node7;
node8.right=node9;
return rootNode;
}
}
```

Main Method:

```
package com.treetraversal.main;
import com.treetravesal.PreTraversal;
```

```
import com.treetravesal.PreTraversal.Node;

public class PreTraversalMain {

public static void main(String[] args) {

PreTraversal b = new PreTraversal();

Node rootNode= PreTraversal.binaryTree();

System.out.println("Display elements in preOrder arrangement:");

b.preOrder(rootNode);

}

}
```

Output: