DSA LAB EXAM

1. Write a Java program to

a. Perform quick sort

b. Perform preorder tree traversal

Solutio :

```
package com.labexam;

import java.util.Arrays;

public class exam {

public static void main(String[] args) {

int[] array = {5, 3, 8, 1, 4, 6, 9, 2, 7};

// Perform quick sort

quickSort(array, 0, array.length - 1);

System.out.println("Sorted array: " + Arrays.toString(array));

// Create a binary tree

Node root = new Node(5);

root.left = new Node(3);

root.right = new Node(8);

root.left.left = new Node(1);

root.left.right = new Node(4);

root.right.left = new Node(6);

root.right.right = new Node(9);

root.left.left.left = new Node(2);

root.left.left.right = new Node(7);

// Perform preorder tree traversal

System.out.print("Preorder traversal: ");
```

```
preorderTraversal(root);
}
public static void quickSort(int[] array, int low, int high) {
if (low < high) {
int partitionIndex = partition(array, low, high);
quickSort(array, low, partitionIndex - 1);
quickSort(array, partitionIndex + 1, high);
}
}
public static int partition(int[] array, int low, int high) {
int pivot = array[high];
int i = low - 1;
for (int j = low; j < high; j++) {
if (array[j] <= pivot) {
i++;
int temp = array[i];
array[i] = array[j];
array[j] = temp;
}
}
int temp = array[i + 1];
array[i + 1] = array[high];
array[high] = temp;
return i + 1;
}
public static void preorderTraversal(Node root) {
if (root != null) {
```

```
System.out.print(root.data + " ");

preorderTraversal(root.left);

preorderTraversal(root.right);

}

}

static class Node {

int data;

Node left;

Node right;

public Node(int data) {

this.data = data;

left = null;

right = null;

}

}

}
```

Output :