LAB EXAM

DATA STRUCTURE AND ALGORITHMS

Write a Java program to

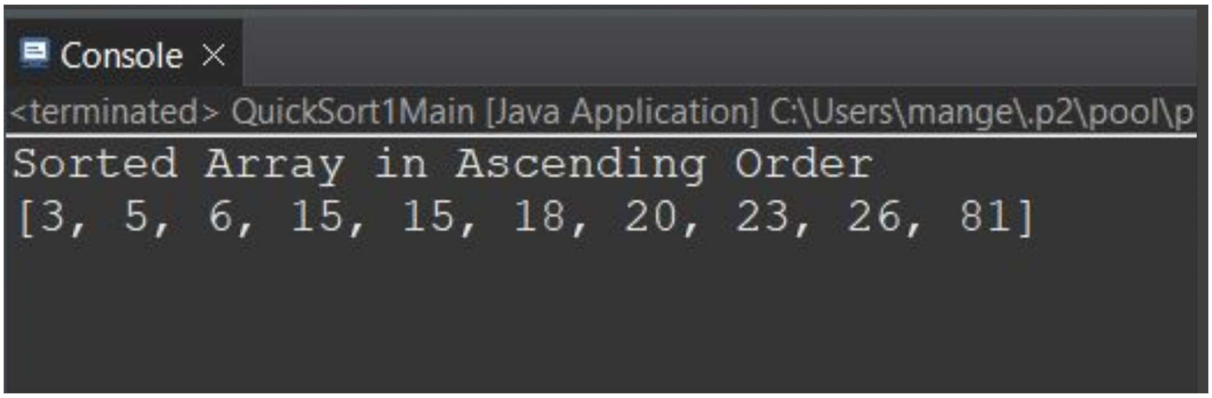
a. **Perform quick sort**

```
package com.quicksort.code;

    import java.util.Arrays;

    public class QuickSort {
   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);
            }
          }
      }
```

```
package com.quicksort.code;

      import java.util.Arrays;
    public class QuickSortMain {
     public static void main(String args[]) {
```

```
            int[] arr = { 23, 18, 26, 81, 6, 15, 3, 15, 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));
           }

}
```



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(2);
            Node node3= new Node(5);
            Node node4= new Node(6);
            Node node5= new Node(8);
            Node node7= new Node(7);
            Node node8= new Node(4);
            Node node9= new Node(10);

            rootNode.left=node4;
            rootNode.right=node8;
            node4.left=node3;
            node4.right=node5;
            node8.left=node7;
            node8.right=node9;
            return rootNode;
            }
```

```
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);

}

}
```