

LAB ASSIGNMENT – 4

WAP TO IMPLEMENT THE QUICK SORT ALGORITHM.

QUICK SORT CODE:

```
package rec_Program;

import java.util.Arrays;
import java.util.Scanner;

public class QuickSort {

    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);
```

```
static void quickSort(int array[], int low, int high) {
```

```
    if (low < high)
```

```
    {
```

```
        int p = partition(array, low, high);
```

```
        quickSort(array, low, p - 1);
```

```
        quickSort(array, p + 1, high);
```

```
    }
```

```
}
```

```
public static void main(String[] args) {
```

```
    // TODO Auto-generated method stub
```

```
    Scanner sc = new Scanner(System.in);
```

```
    System.out.println("Enter the size of array");
```

```
    int n = sc.nextInt();
```

```
    int[] array = new int[n];
```

```
    System.out.println("Enter the elements of the Unsorted  
Array");
```

```
    for(int i = 0 ; i < n ; i++)
```

```
    {
```

```
        array[i] = sc.nextInt();
```

```
    }
```

```
    System.out.println("Unsorted Array");
```

```
    System.out.println(Arrays.toString(array));
```

```
quickSort(array, 0, n - 1);
```

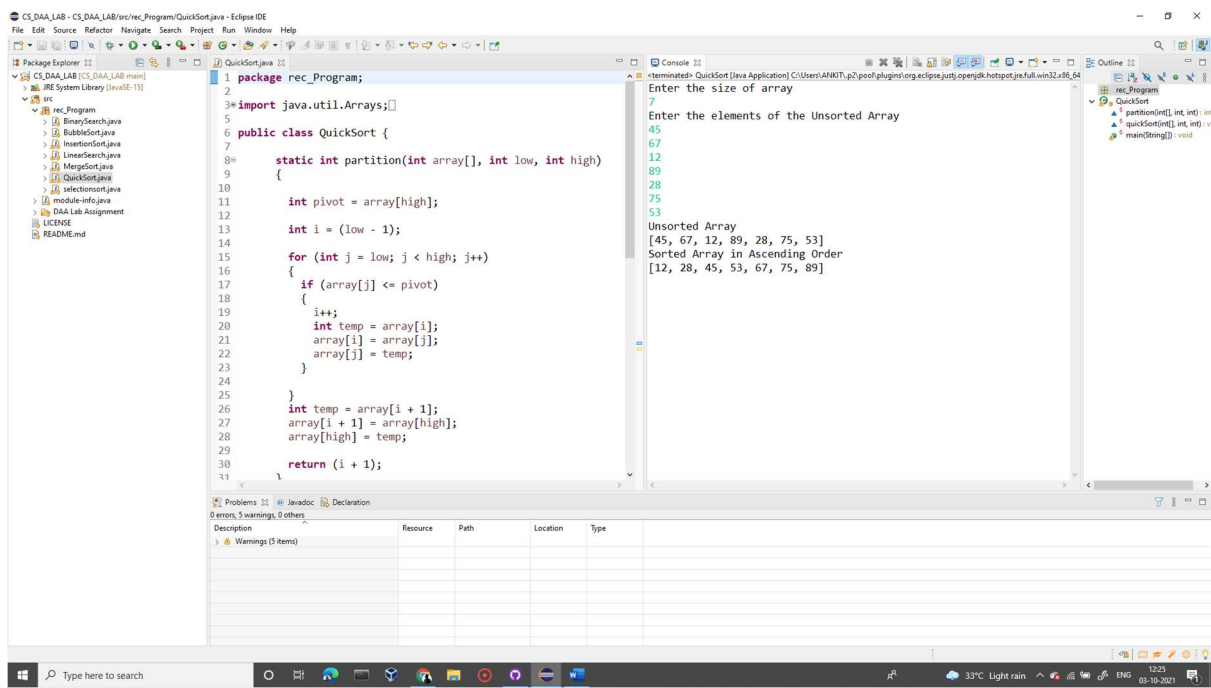
```
System.out.println("Sorted Array in Ascending Order ");
```

```
System.out.println(Arrays.toString(array));
```

```
}
```

```
}
```

QUICK SORT CODE OUTPUT:



The screenshot displays the Eclipse IDE interface. The main editor shows the source code for `QuickSort.java` within the `rec_Program` package. The code implements a recursive quicksort algorithm. The `partition` method is static and takes an array, low, and high indices as parameters. It selects a pivot at the high index and partitions the array around it. The `main` method prompts the user to enter the size of the array and the elements of the unsorted array. It then calls `quickSort` to sort the array and prints the sorted array in ascending order.

```
1 package rec_Program;
2
3 import java.util.Arrays;
4
5 public class QuickSort {
6
7     static int partition(int array[], int low, int high)
8     {
9         int pivot = array[high];
10
11         int i = (low - 1);
12
13         for (int j = low; j < high; j++)
14         {
15             if (array[j] <= pivot)
16             {
17                 i++;
18                 int temp = array[i];
19                 array[i] = array[j];
20                 array[j] = temp;
21             }
22         }
23         int temp = array[i + 1];
24         array[i + 1] = array[high];
25         array[high] = temp;
26         return (i + 1);
27     }
28 }
29
30
31 }
```

The console output shows the execution of the program. It prompts the user to enter the size of the array (7) and the elements of the unsorted array (45, 67, 12, 89, 28, 75, 53). The output displays the unsorted array, the sorted array in ascending order, and the main string.

```
Enter the size of array
7
Enter the elements of the Unsorted Array
45
67
12
89
28
75
53
Unsorted Array
[45, 67, 12, 89, 28, 75, 53]
Sorted Array in Ascending Order
[12, 28, 45, 53, 67, 75, 89]
```

Ankit Yadav

1900290120016