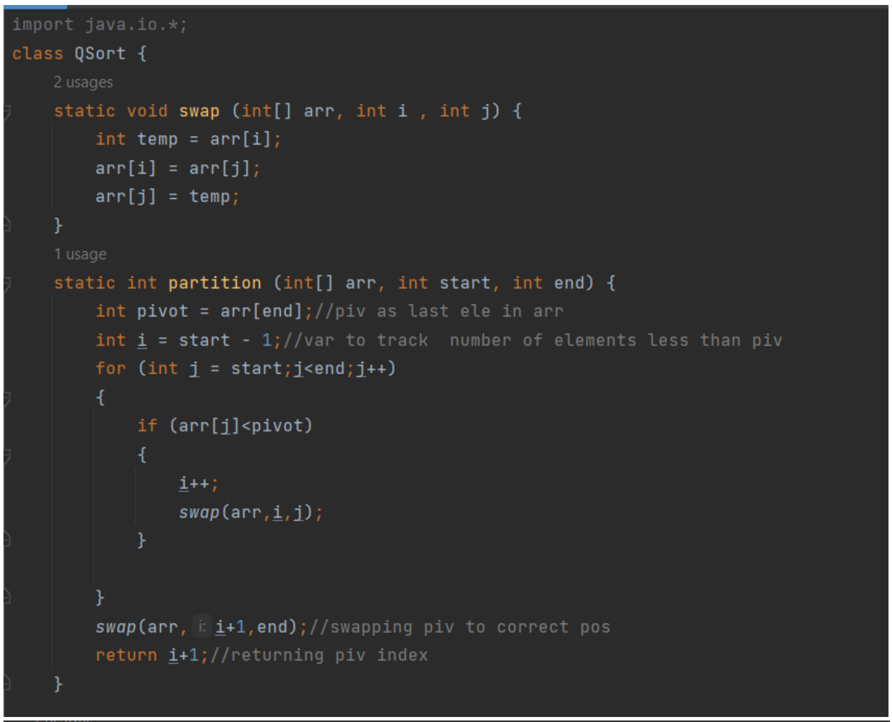
Data Structures and Algorithms

<u>Lab Assignment:</u> <u>Quick Sort and Merge Sort</u>

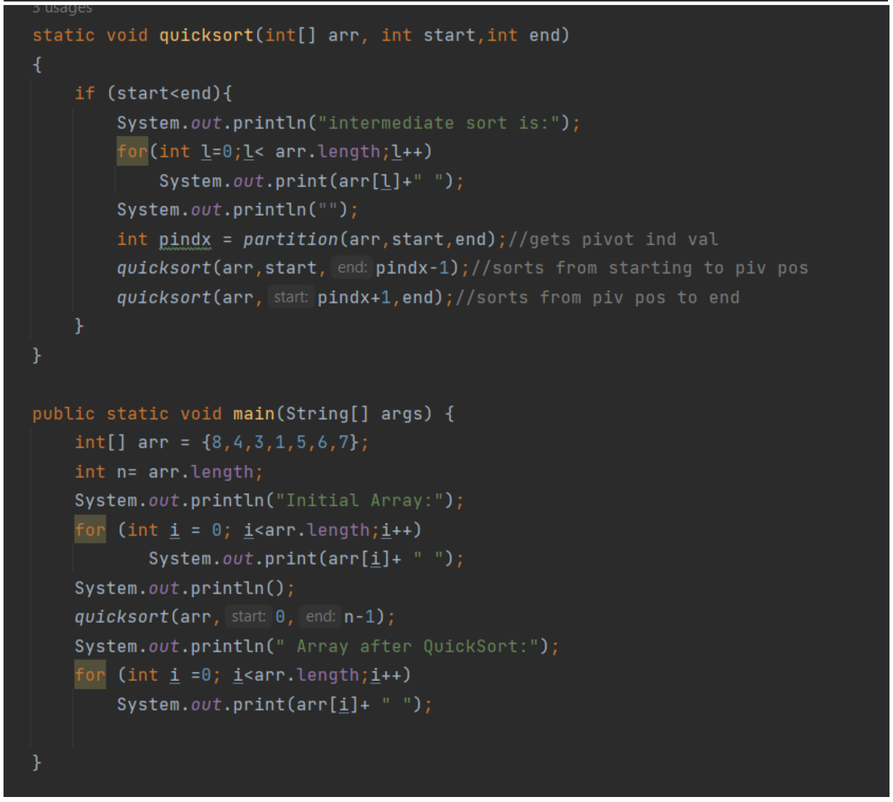
<u>Quick Sort:</u>

Input:



```
import java.io.*;
class QSort {
    2 usages
    static void swap (int[] arr, int i , int j) {
        int temp = arr[i];
        arr[i] = arr[j];
        arr[j] = temp;
    }
    1 usage
    static int partition (int[] arr, int start, int end) {
        int pivot = arr[end];//piv as last ele in arr
        int i = start - 1;//var to track  number of elements less than piv
        for (int j = start;j<end;j++)
        {
            if (arr[j]<pivot)
            {
                i++;
                swap(arr,i,j);
            }

        }
        swap(arr, i: i+1,end);//swapping piv to correct pos
        return i+1;//returning piv index
    }
```

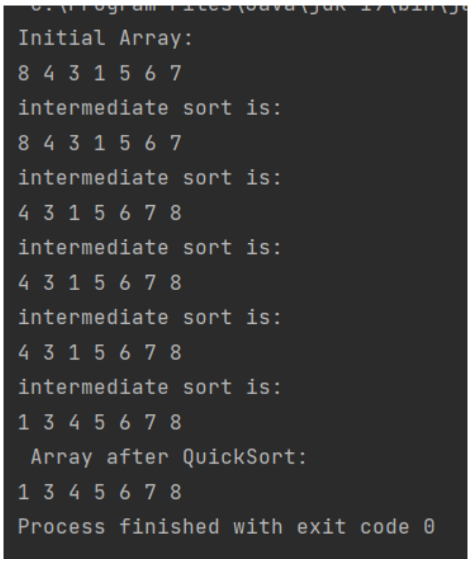


```
  3 usages
  static void quicksort(int[] arr, int start,int end)
  {
      if (start<end){
          System.out.println("intermediate sort is:");
          for(int l=0;l< arr.length;l++)
              System.out.print(arr[l]+" ");
          System.out.println("");
          int pindx = partition(arr,start,end);//gets pivot ind val
          quicksort(arr,start, end: pindx-1);//sorts from starting to piv pos
          quicksort(arr, start: pindx+1,end);//sorts from piv pos to end
      }
  }

  public static void main(String[] args) {
      int[] arr = {8,4,3,1,5,6,7};
      int n= arr.length;
      System.out.println("Initial Array:");
      for (int i = 0; i<arr.length;i++)
             System.out.print(arr[i]+ " ");
      System.out.println();
      quicksort(arr, start: 0, end: n-1);
      System.out.println(" Array after QuickSort:");
      for (int i =0; i<arr.length;i++)
          System.out.print(arr[i]+ " ");

  }
```

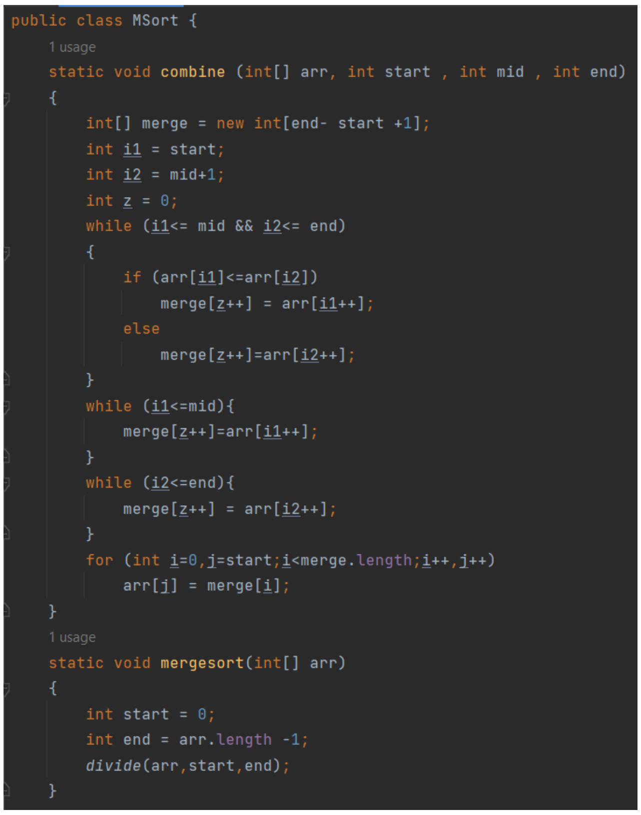
Output:



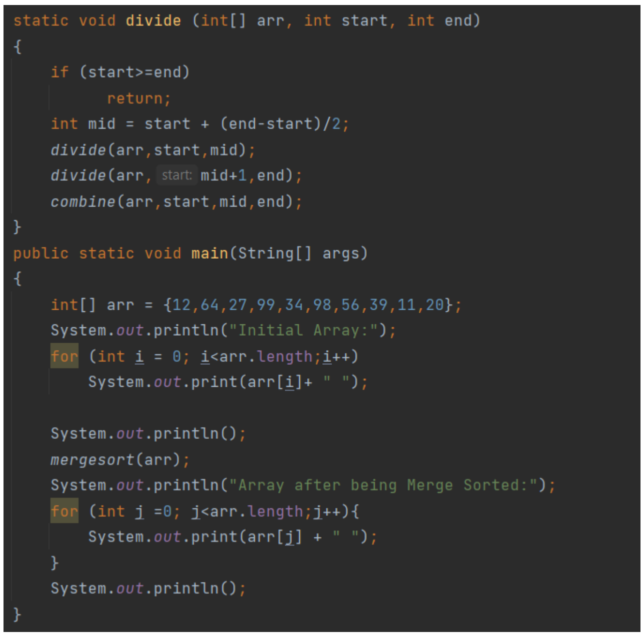
```
 C:\Program Files\Java\jdk-17\bin\ja
Initial Array:
8 4 3 1 5 6 7
intermediate sort is:
8 4 3 1 5 6 7
intermediate sort is:
4 3 1 5 6 7 8
intermediate sort is:
4 3 1 5 6 7 8
intermediate sort is:
4 3 1 5 6 7 8
intermediate sort is:
1 3 4 5 6 7 8
 Array after QuickSort:
1 3 4 5 6 7 8
Process finished with exit code 0
```

Merge Sort:

Input:



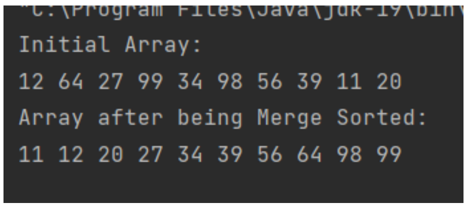
```java
public class MSort {
    1 usage
    static void combine (int[] arr, int start , int mid , int end)
    {
        int[] merge = new int[end- start +1];
        int i1 = start;
        int i2 = mid+1;
        int z = 0;
        while (i1<= mid && i2<= end)
        {
            if (arr[i1]<=arr[i2])
                merge[z++] = arr[i1++];
            else
                merge[z++]=arr[i2++];
        }
        while (i1<=mid){
            merge[z++]=arr[i1++];
        }
        while (i2<=end){
            merge[z++] = arr[i2++];
        }
        for (int i=0,j=start;i<merge.length;i++,j++)
            arr[j] = merge[i];
    }
    1 usage
    static void mergesort(int[] arr)
    {
        int start = 0;
        int end = arr.length -1;
        divide(arr,start,end);
    }
```



```java
static void divide (int[] arr, int start, int end)
{
    if (start>=end)
        return;
    int mid = start + (end-start)/2;
    divide(arr,start,mid);
    divide(arr, start: mid+1,end);
    combine(arr,start,mid,end);
}
public static void main(String[] args)
{
    int[] arr = {12,64,27,99,34,98,56,39,11,20};
    System.out.println("Initial Array:");
    for (int i = 0; i<arr.length;i++)
        System.out.print(arr[i]+ " ");

    System.out.println();
    mergesort(arr);
    System.out.println("Array after being Merge Sorted:");
    for (int j =0; j<arr.length;j++){
        System.out.print(arr[j] + " ");
    }
    System.out.println();
}
```

Output:



```
"C:\Program Files\Java\jdk-19\bin\
Initial Array:
12 64 27 99 34 98 56 39 11 20
Array after being Merge Sorted:
11 12 20 27 34 39 56 64 98 99
```

Name: Harshita Pasupuleti

Reg No: 21BCE8421