

Mohammad Ali Jinnah University

Chartered by Government of Sindh - Recognized by HEC

**Lab Task 5**

**Name:** Muhamad Fahad

**Id:** FA19-BSSE-0014

**Subject:** Data Structures and Algorithms Lab (CS 2511)

**Lab Title:** Merge & Quick Sort

**Section:** AM

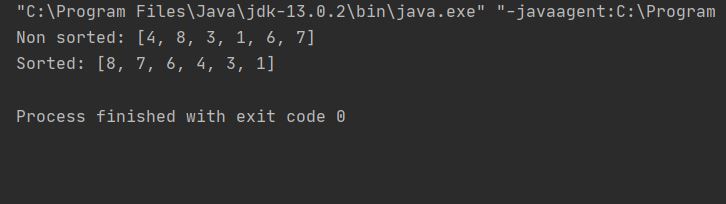
**Teacher:** MUHAMMAD MUBASHIR KHAN

**Date:** Sunday, November 15, 2020

1. **Customize the given code of Merge Sort for descending order sorting. Code:**

import java.util.Arrays;  
  
public class mergesort {  
 static void merge(int A[ ] , int start, int mid, int end) {  
 int low = start, middle = mid+1, count=0;  
 int temp[] = new int[(end-start)+1];  
  
 while (low<=mid&&middle<=end){  
 if (A[low] > A[middle])  
 temp[count] = A[low++];  
  
 else  
 temp[count] = A[middle++];  
  
 count++;  
 }  
  
 while(low<=mid){  
 temp[count] = A[low];  
 count++;  
 low++;  
 }  
  
 while(middle<=end){  
 temp[count] = A[middle];  
 count++;  
 middle++;  
 }  
  
 for (int i = 0; i<count; i++,start++) {  
 A[start] = temp[i];  
 }  
 }  
  
 static void sort(int arr[],int low,int high){  
 int mid;  
 if(low < high){  
 mid=(low+high)/2;  
  
 *sort*(arr,low,mid);  
 *sort*(arr,mid+1,high);  
  
 *merge*(arr,low,mid,high);  
 }  
 }  
  
 public static void main(String[] args) {  
 int[] arr={4,8,3,1,6,7};  
 int low = 0;  
 int high = arr.length-1;  
 System.*out*.println("Non sorted: "+Arrays.*toString*(arr));  
  
 *sort*(arr,low,high);  
   
 System.*out*.println("Sorted: "+Arrays.*toString*(arr));  
 }  
}

**Output:**

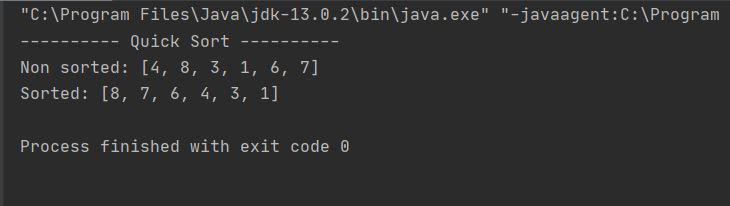
****

1. **Customize the given code of Quick Sort for descending order sorting.**

**Code:**

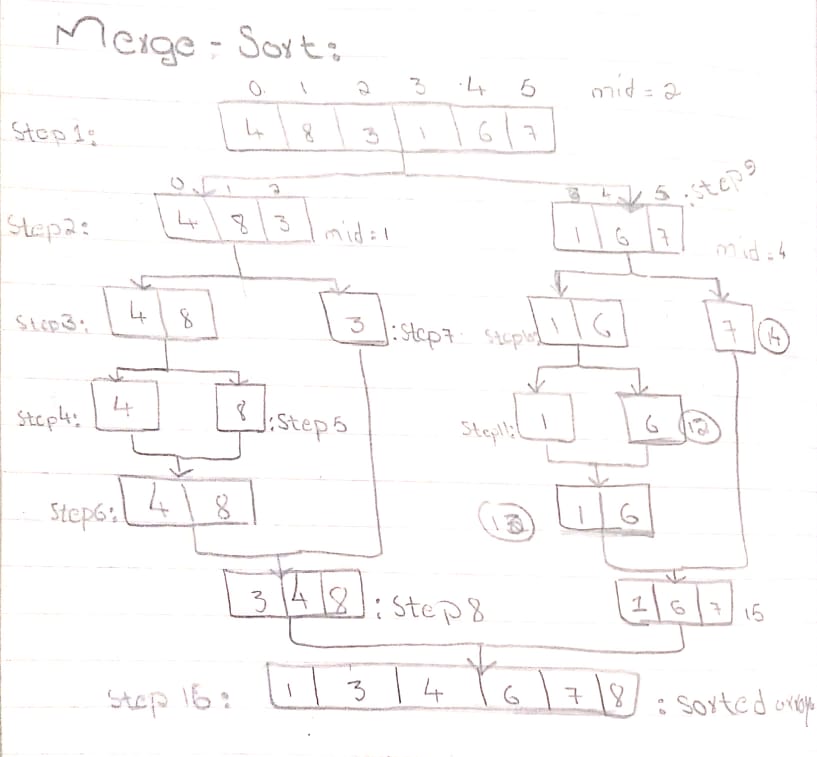
package com.company.Sorting;  
  
import java.util.Arrays;  
  
public class QuickSort {  
 public static void main(String[] args) {  
 int[] arr={4,8,3,1,6,7};  
 int low = 0;  
 int high = arr.length-1;  
  
 System.*out*.println("---------- Quick Sort ----------");  
 System.*out*.println("Non sorted: "+ Arrays.*toString*(arr));  
  
 *sort*(arr,low,high);  
  
 System.*out*.println("Sorted: "+Arrays.*toString*(arr));  
 }  
  
 static void sort(int arr[], int low, int high){  
 if (low < high){  
 int pi = *partition*(arr, low, high);  
  
 *sort*(arr, low, pi-1);  
 *sort*(arr, pi+1, high);  
 }  
 }  
  
 static int partition(int arr[], int low, int high){  
 int pivot = arr[high];  
 int i = (low-1); // index of smaller element  
  
 for (int j=low; j<high; j++)  
 if (arr[j] > pivot)  
 i += *swap*(arr,i+1,j);  
  
 return (i+*swap*(arr,i+1,high));  
 }  
  
 public static int swap(int arr[], int i, int j) {  
 int temp = arr[i];  
 arr[i] = arr[j];  
 arr[j] = temp;  
  
 return 1;  
 }  
}

**Output:**

****

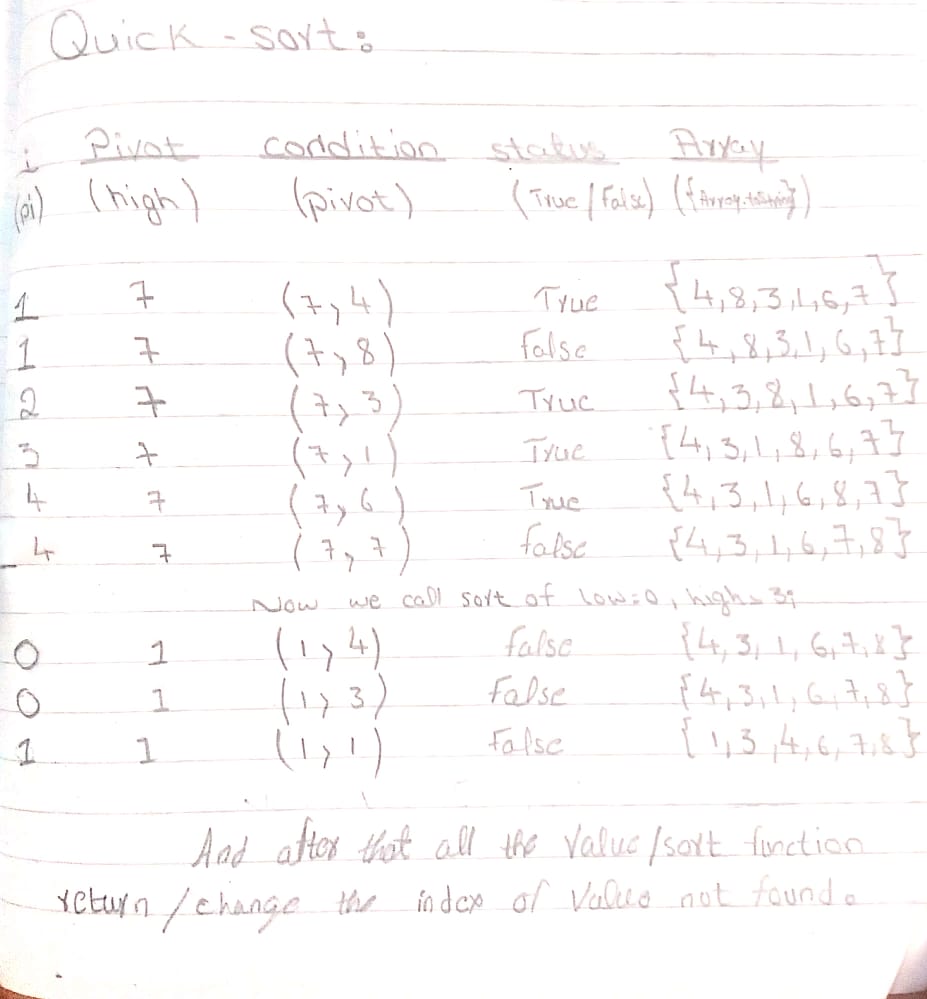
1. **Perform dry run of Merge sort on the following array: [4, 8, 3, 1, 6, 7].**

**Output:**

****

1. **Perform dry run of Quick sort on the following array: [4, 8, 3, 1, 6, 7 ].**

**Output:**

****