**LAB NO: 06**

**NAME: NOOR FATIMA**

**ROLL NO:21sw062  
 SECTION: II**

**TASK NO:01**

Task#01Sort the given 05 students in ascending order of their heights

i.Bubble sort //Create method BubbleSort1D(int[] A)

ii.Selection sort //Create method SelectionSort1D(int[] A)

iii.Insertion sort//Create method InsertionSort1D(int[] A)

iv.Quick sort //Create method QuickSort1D(int[] A)

v.Display the execution time of sorting algos and examine which one is the fastestand explain why?

class LAB\_6 {  
 public static void bubbleSort(int[] array){  
 for (int i=0;i<=array.length-2;i++){  
 for (int j=0;j<=array.length-2-i;j++){  
 if (array[j] > array[j+1]){  
 int temp=array[j+1];  
 array[j+1]=array[j];  
 array[j]=temp;}}  
 }  
 System.*out*.print("Bubble sort ");  
 for (int i=0;i<array.length;i++){  
 System.*out*.print( array[i]+" ");  
 }  
 }  
  
  
  
  
 public void selectionSort(int[]arr){  
  
 for (int i = 0; i < arr.length - 1; i++)  
 {  
 int index = i;  
 for (int j = i + 1; j < arr.length; j++){  
 if (arr[j] < arr[index]){  
 index = j;//searching for lowest index  
 }  
 }  
 int smallerNumber = arr[index];  
 arr[index] = arr[i];  
 arr[i] = smallerNumber;  
 }  
 System.*out*.print("Selection sort ");  
 for (int i=0;i<arr.length;i++) {  
 System.*out*.print(arr[i]+" ");  
 }  
  
 }  
  
  
  
  
  
 public void insertionSort(int[] array){  
 for (int i=1;i< array.length;i++){  
 int current=array[i];  
 int j=i-1;  
 while (j>=0 && array[j]>current){  
 array[j+1]=array[j];  
 j--;  
 }  
 array[j+1]=current;  
 }  
 System.*out*.print("insertion sort: ");  
 for (int i=0;i<array.length;i++) {  
 System.*out*.print(array[i]+" ");  
 }}  
  
  
  
  
  
 public static int partition(int[] array, int l,int h){  
 int pivot =array[l];  
 int i=l, j=h;  
 while (i<j){  
 while(array[i]<=pivot) i++;  
 while (array[j]>pivot) j--;  
 if (i<j){  
 int temp=array[i];  
 array[i]=array[j];  
 array[j]=temp;}  
  
 }  
 array[l]=array[j];  
 array[j]=pivot;  
 return j;}  
  
 public void quickSort(int[]array,int l,int h){  
 int present;  
 if (l<h){  
 present=*partition*(array,l,h);  
 quickSort(array,l,present-1);  
 quickSort(array,present+1,h);  
 }  
  
 System.*out*.print("Quick sort: ");  
 for (int i=0;i<array.length;i++){  
 System.*out*.print(array[i]+" ");  
 }  
 }

class Main{  
 public static void main(String[] args) {  
 int[] array={13,400,2,30,5,505};  
 int[][] ar2d={{13,255,30},{400,55,6},{71,809,95}};  
 LAB\_6 lab=new LAB\_6();  
 lab.*bubbleSort*(array);  
 System.*out*.println();  
 lab.selectionSort(array);  
 System.*out*.println();  
 lab.insertionSort(array);  
 System.*out*.println();  
 lab.quickSort(array,0,4);

// Time Execution

long before=System.*nanoTime*();  
lab.*bubbleSort*(array);  
 System.*out*.println();  
long after =System.*nanoTime*();  
 System.*out*.println("Time execution for bubble sorting: "+ (after-before));  
 System.*out*.println();  
 long before1=System.*nanoTime*();  
 lab.selectionSort(array);  
 System.*out*.println();  
 long after1 =System.*nanoTime*();  
 System.*out*.println("Time execution for selection sorting: "+ (after1-before1));  
 System.*out*.println();  
 long before2=System.*nanoTime*();  
 lab.insertionSort(array);  
 System.*out*.println();  
 long after2 =System.*nanoTime*();  
 System.*out*.println("Time execution for insertion sorting: "+ (after2-before2));  
 System.*out*.println();  
 long before3=System.*nanoTime*();  
 lab.quickSort(array,0,4);  
 System.*out*.println();  
 long after3 =System.*nanoTime*();  
 System.*out*.println("Time execution for Quick sorting: "+ (after3-before3));

}}

**OUTPUT:**

"C:\Program Files\Java\jdk-17.0.2\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2022.1\lib\idea\_rt.jar=50078:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2022.1\bin" -Dfile.encoding=UTF-8 -classpath C:\Users\hp\IdeaProjects\DSA\_ALL\_LABS\out\production\DSA\_ALL\_LABS Main  
 Bubble sort 2 5 13 30 400 505  
 Selection sort 2 5 13 30 400 505  
 insertion sort: 2 5 13 30 400 505  
 Quick sort: 2 5 13 30 400 505

Bubble sort 2 5 13 30 400 505

Time execution for bubble sorting: 17056100

Selection sort 2 5 13 30 400 505

Time execution for selection sorting: 1576200

insertion sort: 2 5 13 30 400 505

Time execution for insertion sorting: 457600

Quick sort: 2 5 13 30 400 505

Time execution for Quick sorting: 2329700

Process finished with exit code 0

**TASK NO:02**

Sort the following 2D array using (bubble, selection, insertion, and quick sort)

class LAB\_6 {

public void bubbleSort\_2d(int[][] array){  
 int[] array\_convert=new int[9];  
 int k = 0;  
 for (int i = 0; i < 3; i++) {  
 for (int j = 0; j < 3; j++) {  
 array\_convert[k] = array[i][j];  
 k++;  
 }  
 }  
  
 //sorting  
 for (int i=1;i< array\_convert.length;i++){  
 int current=array\_convert[i];  
 int j=i-1;  
 while (j>=0 && array\_convert[j]>current){  
 array\_convert[j+1]=array\_convert[j];  
 j--;  
 }  
 array\_convert[j+1]=current;  
 }  
int count1=0;  
 for (int i=0;i<3;i++){  
 for (int j=0;j<3;j++){  
 array[i][j]=array\_convert[count1];  
 count1++;;  
 }  
 }  
 for (int i=0;i<3;i++){  
 for (int j=0;j<3;j++){  
 System.*out*.print(array[i][j]+" ");  
 }  
 System.*out*.println();  
 }  
  
 }  
  
  
public void selection\_sort2d(int arr[][]){  
 int[] array\_convert=new int[9];  
 int k = 0;  
 for (int i = 0; i < 3; i++) {  
 for (int j = 0; j < 3; j++) {  
 array\_convert[k] = arr[i][j];  
 k++;  
 }  
 }  
 // sorting  
 for (int i = 0; i < array\_convert.length - 1; i++)  
 {  
 int index = i;  
 for (int j = i + 1; j < array\_convert.length; j++){  
 if (array\_convert[j] < array\_convert[index]){  
 index = j;//searching for lowest index  
 }  
 }  
 int smallerNumber = array\_convert[index];  
 array\_convert[index] = array\_convert[i];  
 array\_convert[i] = smallerNumber;  
 }  
// convert  
 int count1=0;  
 for (int i=0;i<3;i++){  
 for (int j=0;j<3;j++){  
 arr[i][j]=array\_convert[count1];  
 count1++;;  
 }  
 }  
 for (int i=0;i<3;i++){  
 for (int j=0;j<3;j++){  
 System.*out*.print(arr[i][j]+" ");  
 }  
 System.*out*.println();  
 }}  
  
 public void insertionSort2d(int[][] array){  
 int[] array\_convert=new int[9];  
 int k = 0;  
 for (int i = 0; i < 3; i++) {  
 for (int j = 0; j < 3; j++) {  
 array\_convert[k] = array[i][j];  
 k++;  
 }  
 }  
 // SORTING  
 for (int i=1;i< array\_convert.length;i++){  
 int current=array\_convert[i];  
 int j=i-1;  
 while (j>=0 && array\_convert[j]>current){  
 array\_convert[j+1]=array\_convert[j];  
 j--;  
 }  
 array\_convert[j+1]=current;  
 }  
//convert  
  
 int count1=0;  
 for (int i=0;i<3;i++){  
 for (int j=0;j<3;j++){  
 array[i][j]=array\_convert[count1];  
 count1++;;  
 }  
 }  
  
 for (int i=0;i<3;i++){  
 for (int j=0;j<3;j++){  
 System.*out*.print(array[i][j]+" ");  
 }  
 System.*out*.println();}  
 }}

class ArraySort2d  
{ public static int[] DoubletoSing(int[][] arr){  
 int [] arr1 = new int[arr.length\*arr[0].length];  
 int k = 0;  
 for(int i=0;i<arr.length;i++){  
 for(int j=0;j<arr[i].length;j++){  
 arr1[k] = arr[i][j];  
 k++;  
 }  
 }  
 return arr1;  
 }  
 public static void quickSort2d(int [][] arr1){  
 int [] arr = *DoubletoSing*(arr1);  
 // sorting array using quicksort algorithm  
 *quickSort*(arr, 0,arr.length-1);  
  
 // printing sorted array  
 System.*out*.println(“Qick sort\n”+Arrays.*toString*(arr));  
 }  
 public static void quickSort(int[] arr, int low, int high) {  
 if (low < high) {  
 // partitioning index  
 int pi = *partition*(arr, low, high);  
  
 // sort elements before partition and after partition  
 *quickSort*(arr, low, pi - 1);  
 *quickSort*(arr, pi + 1, high);  
 }  
 }  
  
 public static int partition(int[] arr, int low, int high) {  
 // pivot element  
 int pivot = arr[high];  
  
 // index of smaller element  
 int i = low - 1;  
 for (int j = low; j < high; j++) {  
 // if current element is smaller than pivot  
 if (arr[j] < pivot) {  
 i++;  
  
 // swap arr[i] and arr[j]  
 int temp = arr[i];  
 arr[i] = arr[j];  
 arr[j] = temp;  
 }  
 }  
  
 // swap arr[i+1] and arr[high] (or pivot)  
 int temp = arr[i + 1];  
 arr[i + 1] = arr[high];  
 arr[high] = temp;  
  
 return i + 1;  
 }  
}

**OUTPUT:**

"C:\Program Files\Java\jdk-17.0.2\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2022.1\lib\idea\_rt.jar=50165:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2022.1\bin" -Dfile.encoding=UTF-8 -classpath C:\Users\hp\IdeaProjects\DSA\_ALL\_LABS\out\production\DSA\_ALL\_LABS Main  
  
 Bubble sort 2d  
  
 1 2 3  
 4 5 6  
 9 10 15  
 selection sort 2d  
  
 1 2 3  
 4 5 6  
 9 10 15  
 insertion sort 2d  
  
 1 2 3  
 4 5 6  
 9 10 15

Quick sort 2d  
 [1, 2, 3, 4, 5, 6, 9, 10, 15]  
  
 Process finished with exit code 0