**Java Lab Assignment – 2**

**BY :**

**Vaibhav Jaiswal**

**PRN = 17070122071**

**DIV = CS-4**

**Batch = SY CSE ( 2017-2021)**

**PROBLEM STATEMENTS :**

**Lab Assignment #2**

Theory:

Q1. Discuss Arrays & ArrayListin java.

Q2. Explain basic class Array functions in java with suitable examples.

**Lab Assignment 2:**

**Implement the following four programs to understand basic concepts of operators, arrays in Java**

1. W.a.p that declares two arrays named ‘even’ and ‘odd’. Accept numbers from the user and move them to respective arrays depending on whether they are even or odd.
2. Implement a java function that finds 2 neighboring numbers in an array with the smallest distance to each. The function should return the index of the 1st number.
3. Write a Java program to find the second smallest element in an array
4. Write a Java program to convert an array to ArrayList.
5. Write a Java program to convert an ArrayList to an array
6. Write a Java program to find common elements from three sorted (in non-decreasing order) arrays.
7. Write a Java program to get the difference between the largest and smallest values in an array of integers. The length of the array must be 1 and above.
8. W.a.p which performs sorting of group of integer values using bubble sort & Quick Sort .Use Array List.
9. Write a Java program to remove the duplicate elements of a given array and return the new length of the array. Sample array: [20, 20, 30, 40, 50, 50, 50]  
   After removing the duplicate elements the program should return 4 as the new length of the array.
10. **Given an array of integers. Find a peak element in it. An array element is peak if it is NOT smaller than its neighbors. For corner elements, we need to consider only one neighbor. For example, for input array {5, 10, 20, 15}, 20 is the only peak element. For input array {10, 20, 15, 2, 23, 90, 67}, there are two peak elements: 20 and 90. Note that we need to return any one peak element**

**CODE** :

import java.util.Scanner;

import java.util.\*;

import java.lang.\*;

import java.util.ArrayList ;

class HKP{

public void getArray(double a[] , double b[]){

Scanner sc = new Scanner(System.in);

int i = 0 ; int j = 0 ;

double temp ;

System.out.println("Enter Number : (Max Limit -> 20 numbers)\n Enter end to exit");

while (sc.hasNextDouble())

{

temp = sc.nextDouble();

if (temp%2==0)

{

a[i++] = temp;

}

else{

b[j++] = temp;

}

}

}

public double minOfFour(double arr[]){

double min ;

min = (arr[0] < arr[1])?arr[0] : arr[1] ;

min = (arr[2] < min)?arr[2] : min;

min = (arr[3] < min)?arr[3] : min;

return min ;

}

public int smallestDist(double arr[] , int size){

int indx = 0 ;

int k=0 ;

double min = arr[1] - arr[0] ;

for (int i = 1 ; i < size - 1 ; i++)

{

if(min > Math.abs(arr[i+1] - arr[i]))

{

min = arr[i+1] - arr[i];

indx = i;

}

}

return indx + 1;

}

public double secondsmallest(double arr[] , int size)

{

double temp = 0 ;

int minindx = 0 ;

for(int i = 0 ; i < size - 1 ; i++)

{

minindx = i ;

for(int j = i+1 ; j < size ; j++)

{

if (arr[j] < arr[minindx])

minindx = j ;

}

temp = arr[minindx] ;

arr[minindx] = arr[i] ;

arr[i] = temp ;

}

for(int i = 1 ; i < size - 1 ; i++)

{

if(arr[i] != arr[i-1])

return arr[i];

}

return arr[1] ;

}

public void peakElement(double arr[], int siz)

{

double peaks[] = new double[10] ;

int peaksiz= 0 ;

for(int i = 0 ; i < siz ; i++)

{

if(i==0)

{

if(arr[i]>arr[i+1])

{peaks[peaksiz++]= arr[i++] ;}

}

else if (i==siz-1)

{

if(arr[i-1]<arr[i])

{peaks[peaksiz++]= arr[i] ;}

}

else

{

if(arr[i]>arr[i+1] && arr[i]>arr[i-1])

{peaks[peaksiz++] = arr[i++];}

}

}

System.out.println("Peak Elements are : ");

for(int i = 0 ; i < peaksiz ; i++)

System.out.println(peaks[i]);

}

public int removeDuplicates(double arr[] , int size)

{

for(int i = 0 ; i < size-1 ; i++ )

{

if(arr[i] == arr[i+1])

{

int k = i ;

while (k<size)

{

arr[k] = arr[k+1] ;

k++ ;

}

size-- ;

i-- ;

}

}

return size ;

}

public static void quicksort(ArrayList<Integer> arr,int low,int high){

if(low < high){

Integer pi = HKP.partition(arr,low,high);

quicksort(arr , low , pi.intValue()-1);

quicksort(arr , pi.intValue()+1 , high);

}

}

public static Integer partition(ArrayList<Integer> arr,int low,int high){

Integer pivot = arr.get(high);

int i = low - 1;

int j;

for(j=low ; j<high ; j++){

if(arr.get(j) < pivot){

i++;

Integer temp = arr.get(i);

arr.set(i,arr.get(j));

arr.set(j,temp);

}

}

i++;

Integer temp = arr.get(i);

arr.set(i,arr.get(high));

arr.set(high,temp);

return Integer.valueOf(i);

}

}

class Driver{

public static void main(String args[]) {

Scanner sr = new Scanner(System.in);

HKP mj = new HKP();

/\*

//ODD and EVEN ARRAY

double a[] = new double[20];

double b[] = new double[20];

mj.getArray(a , b);

System.out.println("Even :");

System.out.println(Arrays.toString(a));

System.out.println("ODD :");

System.out.println(Arrays.toString(b));

\*/

/\*

//MIN of FOUR

double arr[] = new double[4];

System.out.println("Enter 4 numbers : ");

for (int i = 0 ; i < arr.length ; i++)

arr[i] = sr.nextDouble();

double min = mj.minOfFour(arr);

System.out.println("Min of 4 values is :" + min);\*/

/\*//Finding pair with smallest difference and getting index of first ele

double num[] = new double[20];

int size = 0 ;

System.out.println("Enter numbers : (Max Limit -> 20 numbers)\n Enter end to exit");

for (size = 0 ; sr.hasNextDouble() ; size++)

{

num[size] = sr.nextDouble();

}

int indexOfele = mj.smallestDist(num , size) ;

System.out.println("Index of 1st element of pair with least distance : " + indexOfele);\*/

/\*

//Finding second smallest element in array

double num[] = new double[20];

int size = 0 ;

System.out.println("Enter numbers : (Max Limit -> 20 numbers)\n Enter end to exit");

for (size = 0 ; sr.hasNextDouble() ; size++)

{

num[size] = sr.nextDouble();

}

double secsmall = mj.secondsmallest(num , size) ;

System.out.println("Second smallest elements : " + secsmall) ;

\*/

/\*

//converting arraylist to array

ArrayList <Double> al = new ArrayList<Double>();

char exit = 'y' ;

System.out.println("Enter elements of ArrayList : ");

while (exit=='y')

{

double temp = sr.nextDouble();

al.add(temp) ;

sr.nextLine();

System.out.println("Enter Another Element : (y/n)") ;

exit = sr.next().charAt(0) ;

}

System.out.println("ArrayList Converted to Array : ");

Object[] arr = al.toArray();

System.out.println(Arrays.toString(arr));\*/

/\*

//Converting array to arraylist

Double[] arr ;

int siz ;

System.out.println("Enter size of arr :");

siz = sr.nextInt();

arr = new Double[siz];

System.out.println("Enter elements of array :");

for(int i = 0 ; i < arr.length ; i++)

arr[i] = sr.nextDouble();

ArrayList<Double> ar= new ArrayList<Double>(Arrays.asList(arr));

System.out.println("Array Converted into ArrayList : ") ;

System.out.println(ar) ; \*/

/\*

//Finding common elements from three arrays

int siz1;

System.out.println("Enter the size of array 1 ");

siz1= sr.nextInt();

double arr1[] = new double[siz1] ;

System.out.println("Enter elements of array 1:");

for ( int i = 0 ; i < arr1.length ; i++)

arr1[i] = sr.nextDouble();

int siz2;

System.out.println("Enter the size of array 2 ");

siz2= sr.nextInt();

double arr2[] = new double[siz2] ;

System.out.println("Enter elements of array 2:");

for ( int i = 0 ; i < arr2.length ; i++)

arr2[i] = sr.nextDouble();

int siz3;

System.out.println("Enter the size of array 3 ");

siz3= sr.nextInt();

double arr3[] = new double[siz3] ;

System.out.println("Enter elements of array 3:");

for ( int i = 0 ; i < arr3.length ; i++)

arr3[i] = sr.nextDouble();

Arrays.sort(arr1);

Arrays.sort(arr2);

Arrays.sort(arr3);

ArrayList<Double> comlist = new ArrayList<Double>() ;

for (int i = 0 ; i < arr1.length ; i++)

{

for (int j = 0 ; j < arr2.length ; j++)

{

if (arr1[i] == arr2[j])

comlist.add(arr1[i]);

}

}

boolean isPresent = false ;

for (int i = 0 ; i < comlist.size() ; i++)

{

Double obj = comlist.get(i);

isPresent = false ;

for (int j = 0 ; j < arr3.length ; j++)

{

if(arr3[j] == obj.doubleValue())

{

isPresent = true ;

break ;

}

}

if(isPresent == false)

{

comlist.remove(i) ;

i-- ;

}

}

System.out.println("Common elements in all three are ;");

System.out.println(comlist);

\*/

/\*

//DIFFENCE BTW LARGEST AND SMALLEST NO OF ARRAY

int siz ;

System.out.println("ENter Size of array");

siz = sr.nextInt() ;

double arr[] = new double[siz];

System.out.println("ENter Elements of array");

for (int i = 0 ; i < arr.length ; i++)

{

arr[i] = sr.nextDouble();

}

Arrays.sort(arr) ;

double diff = arr[arr.length-1] - arr[0] ;

System.out.println("difference BTW Smallest and Largest Number is : " + diff) ;

\*/

/\*

// Peak element

double arr[] = new double[20] ;

int siz = 0 ;

System.out.println("ENter Elements of array : (Max 20 elements)");

for (;sr.hasNextDouble();siz++)

{

arr[siz] = sr.nextDouble();

}

mj.peakElement(arr , siz);

\*/

/\*

//Removing Duplicate Elements From Array :

double arr[] = new double[20] ;

int size = 0 ;

System.out.println("Enter Elements of Array : Maxm 20 elements ");

while (sr.hasNextDouble())

{

arr[size++] = sr.nextDouble() ;

}

size = mj.removeDuplicates(arr , size );

System.out.println("After Removing duplicates Array is ; ");

System.out.println(Arrays.toString(arr)) ;

System.out.println("new size is : "+ size) ; \*/

/\*

//Bubble Sort

ArrayList<Double> alist = new ArrayList<Double>() ;

System.out.println("Enter Elements : (enter end to stop ) ") ;

while (sr.hasNextDouble())

{

alist.add(sr.nextDouble()) ;

}

System.out.println("List Before Sorting : (Bubble Sort)");

System.out.println(alist) ;

for(int i = 0 ; i < alist.size()-1 ; i++)

{

for(int j = 0 ; j<alist.size()-i-1 ; j++)

{

if(alist.get(j) > alist.get(j+1)){

Double temp = alist.get(j) ;

alist.set(j , alist.get(j+1));

alist.set(j+1 , temp);

}

}

}

System.out.println("List After Sorting : (Bubble Sort)");

System.out.println(alist);\*/

/\*

// Quick Sort

ArrayList<Integer> alist = new ArrayList<Integer>() ;

System.out.println("Enter Elements : (enter end to stop ) ") ;

while (sr.hasNextDouble())

{

alist.add(sr.nextInt()) ;

}

System.out.println("List Before Sorting : (QuickSort Sort)");

System.out.println(alist) ;

HKP.quicksort(alist,0,alist.size()-1);

System.out.println("List After Sorting : (QuickSort Sort)");

System.out.println(alist) ;

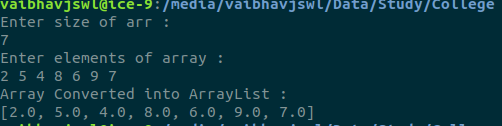
\*/

}

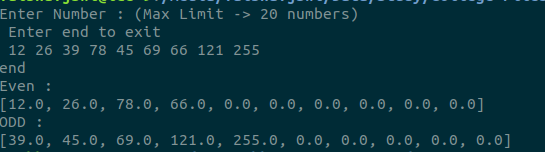
}

**OUTPUTS :**

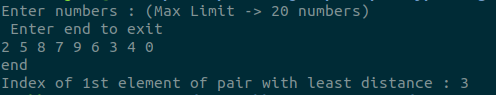
**Array to ArrayList :**

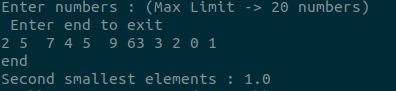
****

**Dividing Array into Even or Odd :**

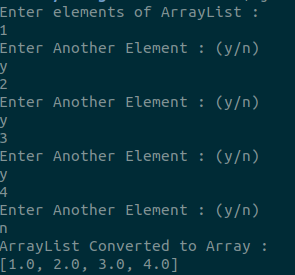
****

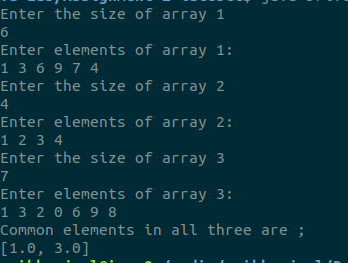
**Pair With Least Distance :**

****

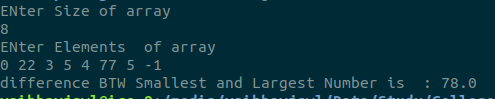
**Finding Second Smallest Element :**

**Converting ArrayList TO Array :**

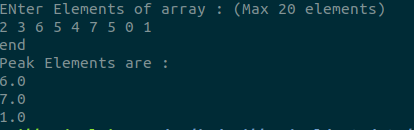
****

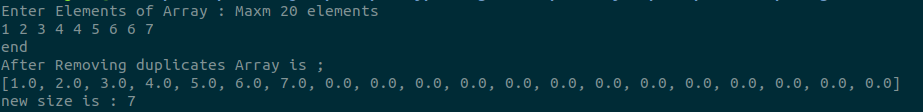
****

**Common Elements from 3 arrays :**

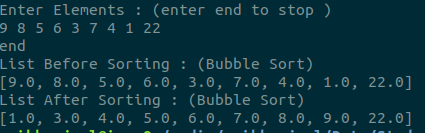
**Difference Btw Smallest and Largest Elements of array :**

**Peaks Elements in An Array :**

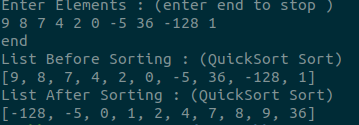
****

**Removing Duplicate Elements in Array :**

**Bubble Sort On ArrayList :**

****

**Quicksort on Array List :**

****