Assignment - 3

Objective.

Task list:-

1. Selection sort
2. Array Reduction
3. Merging two sorted arrays
4. Check reverse
5. Finding first repeated elements in an array
6. Print duplicates in a list
7. Find the missing number in an array
8. Given an array of integers, find the element pair with minimum/maximum difference.
9. Given a list of n numbers, find the element which appears max^m no. of times
10. Find sum of distinct elements of the array. If there is some value repeated then they should be added once.

Q.1. Selection Sort

Program:

```
import java.util.*;
public class Program_1 {
    public static void SelectionSort(int[] arr) {
        for(int i=0; i< arr.length-1; i++) {
            int minIdx = i;
```

```
for(int j= i+1 ; j< arr.length ;j++) {
    if( arr[j] < arr[minIdx])
        minIdx =j ;
}
int temp = arr[minIdx] ;
arr[minIdx ] = arr[i] ;
arr[i] = temp ;
}
System.out.println("Sorted Array: " + Arrays.toString(arr)) ;
}

public static void main(String [] args) {
    Scanner sc = new Scanner(System.in);
    System.out.println("Enter the size of array: ");
    int n = sc.nextInt();
    int a[]= new int[n];
    System.out.println("Enter the elements: ");
    for(int i=0; i<n; i++) {
        a[i] = sc.nextInt();
    }
    SelectionSort(a);
  }
}
```

Testing :-

Enter the size of array :
5
Enter the elements :
-1
8
9
2
6
Sorted Array : [-1, 2, 6, 8, 9]

Testing:-

Enter the size of array:

5

Enter the elements:

-1
8
9
2
6

Sorted Array: [-1, 2, 6, 8, 9]

P.2. Array Reduction

Problem: Element left after reductions, Given an array of positive elements. You need to perform reduction operation. In each reduction operation smallest positive element value is picked and all the elements are subtracted by that value. You need to print the number of elements left after each reduction process.

Program:-

```
import java.util.*;
public class Program_2 {
    public static void main (String[] args) {
        int[] a = {5, 1, 1, 1, 2, 3, 5};
        int n = a.length;
        while(true) {
            int min = find_smallest_positive (a);
            if(min == 0) {
                System.out.println("0 corresponds to [0] after reduction");
            } break;
```

Name:_____ Regd. Number:_____

```
for(int i=0 ; i<n; i++) {
  if(a[i] >0)
    a[i] -= min ;
  }
  int count = count_positive(a) ;
  System.out.println (count + " corresponds to [ ") ;
  print_positive_array (a) ;
  System.out.println ( "] after subtraction") ;
 }
}

public static int find_smallest_positive (int [] a) {
   int min = Integer. MAX_VALUE ;
   boolean f = false ;
   for ( int i : a) {
      if ( i >0 && i < min) {
         min = i ;
         f = true ;
      }
   }
  return f ? min : 0 ;
}
```

```
    return count;
}

public static void print_positive_array(int[] a) {
    boolean f=true;
    for(int i : a) {
        if(i>0) {
            if(!f)
                System.out.print(", ");
            System.out.print(i);
            f= false;
        }
    }
}
```

Testing :-

4 corresponds to [4,1,2,4] after subtraction
3 corresponds to [3,1,3] after [illegible]
2 corresponds to [2,2] after subtraction
0 corresponds to [0] after reduction

Q.3 Merging two sorted Arrays

Given two sorted arrays. Sort the elements of these arrays so that first half of sorted elements will lie in first array and second half lies in second array.

Program :-

```
import java.util.Arrays ;
public class Program_3 {
    static void merge_sorted_array (int[] A, int[] B){
        int[] merged = new int[A.length + B.length];
        int i=0, j=0, k=0 ;
        while( i< A.length && j < B.length) {
            if (A[i] < B[i])
                merged[k++] = A[i++];
            else
                merged[k++]= B[j++];
        }
        while (i< A.length)
            merged[k++]= A[i++];
        while (j< B.length)
            merged[k++] = B[j++];
        System.out.println("Merged Array: "+Arrays.toString(merged));
    }
```

Name:_____ Regd. Number:_____

```
public static void main(String [] args) {
    int [] A = {5, 7, 11, 19, 23};
    int [] B = {6, 13, 20};
    System.out.println("Merge two sorted Arrays: ");
    merge_sorted_array(A, B);
    }
}
```

Testing

Merge two sorted Arrays:

Merged Array : [5, 6, 7, 11, 13, 19, 20, 23]

Q.4 Check Reverse

Problem : Given an array of integers, find if reversing a sub-array makes the array sorted.

Example

Input : A = [1,2,6,5,4,7]

Output : True

Explanation : If we reverse sub array [6,5,4] the whole array become sorted.

Problem :-

```
public class Program_4 {
    static boolean check_Reverse(int[] a) {
        int n = a.length;
        int i = 0, j = n-1;
        while(i < n-1 && a[i] < a[i+1])
            i++;
        if(i == n-1)
            return true;
        while(j > 0 && a[j] > a[j-1])
            j--;
        reverse(a, i, j);
        boolean sorted = is_sorted(a);
        reverse(a, i, j);
        return sorted;
    }
    static void reverse(int[] a, int l, int r) {
        while(l < r) {
            int temp = a[l];
            a[l++] = a[r];
            a[r--] = temp;
        }
    }
}
```

Name:_____ Regd. Number:_____

```
static boolean is_sorted(int[] a){
  for(int i= 0; i<a.length-1 ;i++)
    if (a[i] > a[i+1])
      return false;
  return true;
}

public static void main(String[] args){
  int[] a = {1,2,6,5,4,7};
  System.out.println("Check reverse: ");
  System.out.println(check_reverse(a));
}
}
```

Testing -

Check Reverse :

true

Q.5. Finding first repeated elements in an array.

Program :-

```
public class Program_5{
  public static int first_repeated(int[]a){
    for(int i= 0; i<a.length ;i++){
      for(int j= i+1 ; j<a.length; j++){
        if(a[i]==a[j]){
          return a[i];
        }}
```

```
    }
    return Integer.MIN_VALUE;
}

public static void main(String[] args) {
    int[] a = {2,5,1,2,3,5,1};
    int r = first_repeated(a);
    if(r == Integer.MIN_VALUE){
        System.out.println("No repeated elements found.");
    }else{
        System.out.println("First repeated element: "+res);
    }
}
}
```

Testing :-

First repeated element : 2

Q.6 Print duplicates in a list

Program:-

```
public class Program_6 {
    public static void main(String[] args){
        int[] a= {3,5,2,3,8,5,9,2,2,7};
        print_duplicates(a);
    }

    public static void print_duplicates(int[] a) {
        int n = a.length;
        boolean[] printed= new boolean[n];
        for(int i=0; i<n; i++) {
            if (printed[i]) {
                continue;
            }
            int val = a[i];
            boolean isduplicate= false;
            for (int j= i+1; j<n; j++) {
                if(a[j]==val) {
                    isduplicate= true;
                    printed[j]=true;
                }
            }
            if(isduplicate) {
                System.out.println(val);
            }
        }
    }
```

```
    }
}
```

Testing :- 3
5
2

Q.7. Find the missing number in an array.
Problem : In a given list of n-1 elements, which are in the range of 1 to n. There are no duplicates in the array. One of the integers is missing. Find the missing element.

Program :-

```
public class Program_7 {
    static void find_missing (int[] a, int n) {
        int total = n * (n+1)/2 ;
        int sum = 0 ;
        for(int i : a)
            sum += i ;
        System.out.println("Missing number : " +(total- sum));
    }

public static void main(String[] args) {
    int[] a= { 1, 2,4,5} ;
    System.out.println ("Find Missing number : ");
    find_missing(a, 5);
    }
}
```

Testing :-

Find Missing number :

Missing number : 3

Q.8. Given an array of integers, find the element pair with minimum/ maximum difference.

Program :-

```
import java.util.*;
public class Program_8{
    static void difference(int[] a){
        Arrays.sort(a);
        int min_diff = Integer.MAX_VALUE, max_diff = a[a.length-1]-a[0];
        for(int i=1; i<a.length; i++)
            min_diff = Math.min(min_diff, a[i]-a[i-1]);
        System.out.println("Minimum difference: "+min_diff);
        System.out.println("Maximum difference: "+max_diff);
    }
    public static void main(String[] args){
        int[] a = {10, 3, 6, 8, 15};
        System.out.println("Min/max difference: ");
        difference(a);
    }
}
```

Testing -

Min/max difference :

Minimum difference : 2

Maximum difference : 12

Q.9. Given a list n numbers, find the element which appears maximum number of times.

Program:-

```
import java.util.Scanner;
public class Program_9 {
    public static void main(String [] args) {
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter size: ");
        int n = sc.nextInt();
        int [] a = new int [n];
        System.out.println("Enter elements: ");
        for(int i=0; i< a.length ; i++) {
            a[i]= sc.next Int();
        }
        int count = 0;
        int num = 0;
        for(int i = 0; i< a.length ; i++) {
            int count_1 = 0;
            for(int j = i ; j< a.length ; j++) {
                if( a[i] == a[j]) {
                    count_1++;
                }
            }
            if( count_1 > count) {
                count= count_1 ;
                num=a[i];
        } }
```

Name:_____ Regd. Number:_____

```
System.out.println(num + " maximum appearance is : " + count);
    }
}
```

Testing

Enter size : 3
Enter elements :
1
1
1

1 maximum appearance is : 3

Q.10. Given an array of size N. The elements in the array may be repeated. You need to find the sum of distinct elements of the array. If there is some value repeated then they should be added once.

Program :-

```
import java.util.Scanner;
public class Program_10 {
    public static void main(String[] args) {
        int [] a = { 1,2,3,1,1,1,5,6,5};
        int [] a1 = new int [a.length];
        int sum = 0;
        for (int i=0; i<a.length; i++) {
            if (a[i] != -1) {
                for (int j = i+1; j< a.length; j++) {
                    a[i] = a[i];
                    if (a[i] == a[j])
                        a[j] = -1;
```

```
        }
    }
}

System.out.println();
for(int i= 0; i< a1.length ; i++){
    System.out.println (a1[i]);
     if ( a1[i]==0)
        continue;
        sum+= a[i];
    }

    System.out.println("The sum is: "+sum);
    }
}
```

Testing-

The sum is: 21