Task: 3 Array Manipulation and Analysis Programs

(A) Return the max sliding window.

Aim:

To write a Java program to find the maximum value in each sliding window of size K as the window moves from left to right across a given integer array.

Algorithm:

```
1. start
2. Read the array nums and window size K.
3. Create an output array of size n-K+1.
4. For each window starting from index i=0 to n-K:
     * Assume the first element of the window is the maximum
     * Compare all K elements in the current window.
     * update the maximum value.
     * store the maximum in the result array.
5. print the result.
```

Program:

```
import java.util.Scanner;
class sliding window Maximum{
  Public static void main(string[]args){
    Scanner sc= new Scanner(system.in);

   System.out.print("Enter array size:");
     int n = sc.nextInt();
   int[]nums= new int[n];

   system.out.println("Enter"+n+"array elements:");
    for(int i=0; i<n;i++){
      nums[i]= sc.nextInt();
    }
    System.out.print("Enter window size:");
     int K= sc.nextInt();
     int[] result = new int[n-K+1];

     for(int i=0;i<=n-K;i++){
       int max= nums[i];
      for(int j=i;j<i+k;j++){
       if(nums[j]>max){
         max= nums[j];
        }
      }
      result[i] = max;
    }
```

Output:

```
Enter array size: 8
Enter 8 array elements:
  1 3 -1 -3 5 3 6 7
  Enter window size: 3
  Sliding window maximum: 3 3 5 5 6 7
```

[illegible]

```
System.out.print("sliding window maximum:");
 for (int i=0; i< result.length; i++){
     System.out.print(result[i]+" ");
  }
 }
}
```

VEL TECH-CSE	
EX NO	
PERFORMANCE (5)	
RESULT AND ANALYSIS (5)	
VIVA VOCE (5)	
RECORD (5)	
TOTAL (20)	
SIGN WITH DATE	

Result:

The Java Program successfully finds the maximum element in each sliding window of size K. The output correctly represents the maximum values for all window positions as the window moves from left to right.

(B) ADJUSTED SUM

Aim: To write a Java program that calculates the adjusted sum of an integer array by adding all even numbers and subtracting all odd numbers.

Algorithm:

```
1. Start the program
2. Read the value of N (size of the array)
3. Read N integer elements into the array.
4. Initialise a variable sum to 0.
5. Traverse each element of the array:
      * If the element is even, add it to sum
      * If the element is odd, subtract it from sum.
6. Display the final adjusted sum.
7. Stop the program.
```

Program:

```
import java.util.Scanner;
  class AdjustedSum {
   Public static void main(String []args){
      Scanner sc = new Scanner(System.in);
     System.out.print("Enter the size of the array:");
     int N=sc.nextInt();
    int []arr = new int[N];
    int sum=0;
   System.out.println("Enter the array elements:");
    for(int i=0; i<N;i++){
       arr[i]=sc.nextInt();
       if(arr[i]%2==0){
         sum= sum+ arr[i];
      }else{
        sum= sum-arr[i];
        }
     }
     System.out.println("Adjusted sum:" + sum);
     }
  }
```

Output:

```
Enter the size of the array: 5
Enter the array elements:
 10 15 20 25 30
  Adjusted sum: 20
```

c) Rotate array and find max difference

Aim: To write a Java program that rotates an array to the right by k positions and finds the maximum absolute difference between any two adjacent elements after rotation.

Algorithm:

1. Start the program.
2. Read the size of the array N.
3. Read N elements into the array.
4. Read the value of K.
5. Rotate the array to the right by k positions using the formula:
   rotated[(i+k)%N] = arr[i].
6. Traverse the rotated array and calculate the absolute diffrence between adjacent elements using Math.abs().
7. Store the maximum difference found.
8. Display the rotated array and the maximum difference.
9. Stop the program.

Program:

```
import java.util.Scanner;
  class RotateMaxDifference{
    Public static void main (string[]args){
      Scanner sc = new scanner (system.in);
     System.out.print ("Enter the size of the array:");
     int N = sc.nextInt();
    int []arr=new int[N];
    int []rotated = new int [N];
    System.out.println ("Enter the array elements:");
      for (int i= 0; i<N; i++){
         arr[i]= sc.nextInt();
      }
       System.out.print ("Enter k value:");
       int k = sc.nextInt();
      K-k%N;
      for (int i=0; i<N; i++){
          rotated [(i+k)% N] = arr [i];
      }
```

Output:

```
Enter the size of the array: 5
Enter the array elements:
 3 8 9 7 6
Enter k value: 1
 Rotated Array:
   6 3 8 9 7
  Maximum difference = 5
```

```
int maxDiff = 0;
for (int i=0; i<N-1; i++){
  int diff = Math.abs (rotated[i] - rotated [i+1]);
    if (diff > maxDiff){
       maxDiff = diff;
    }
}
System.out.println ("Rotated Array:");
for (int num: rotated){
     System.out. print(num + " ");
  }
 System.out.println("\n Maximum difference = " + maxDiff);
  }
}
```

VEL TECH-CSE	
EX NO.	3
PERFORMANCE (5)	5
RESULT AND ANALYSIS (5)	5
VIVA VOCE (5)	5
RECORD (5)	5
TOTAL (20)	20
SIGN WITH DATE	[illegible] 10/02/26.

Result: Thus, the Java program successfully rotates the array to the right by k positions and finds the maximum absolute difference between adjacent elements.