

Logic Building Assignment : 78

1. Write a java program to find Maximum difference between two elements such that larger element appears after the smaller number

Examples:

If array is [2, 3, 10, 6, 4, 8, 1] then returned value should be 8 (Diff between 10 and 2).

If array is [7, 9, 5, 6, 3, 2] then returned value should be 2 (Diff between 7 and 9)

```
class MaximumDifference
{
    int maxDiff(int arr[], int arr_size)
    {
        // Logic
    }

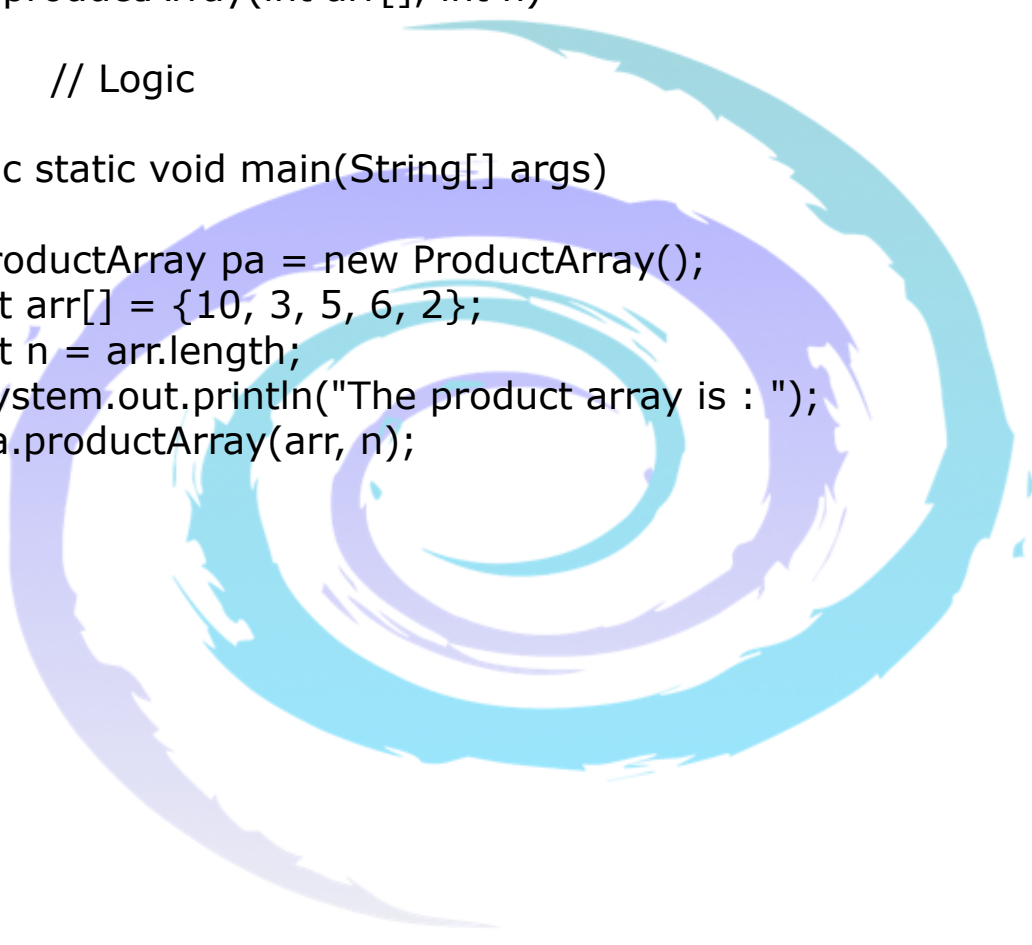
    public static void main(String[] args)
    {
        MaximumDifference maxdif = new MaximumDifference();
        int arr[] = {1, 2, 90, 10, 110};
        System.out.println("Maximum difference is " +maxdif.maxDiff(arr,
5));
    }
}
```

2. Given an array `arr[]` of `n` integers, construct a Product Array `prod[]` (of same size) such that `prod[i]` is equal to the product of all the elements of `arr[]` except `arr[i]`. Solve it without division operator and in $O(n)$.

`arr[] = {10, 3, 5, 6, 2}`

`prod[] = {180, 600, 360, 300, 900}`

```
class ProductArray
{
    void productArray(int arr[], int n)
    {
        // Logic
    }
    public static void main(String[] args)
    {
        ProductArray pa = new ProductArray();
        int arr[] = {10, 3, 5, 6, 2};
        int n = arr.length;
        System.out.println("The product array is : ");
        pa.productArray(arr, n);
    }
}
```



3. Segregate Even and Odd numbers

Given an array A[], write a function that segregates even and odd numbers. The functions should put all even numbers first, and then odd numbers.

Example

Input = {12, 34, 45, 9, 8, 90, 3}

Output = {12, 34, 8, 90, 45, 9, 3}

```
import java.io.*;

class SegregateOddEven
{
    static void segregateEvenOdd(int arr[])
    {
        // Logic
    }

    public static void main (String[] args)
    {
        int arr[] = {12, 34, 45, 9, 8, 90, 3};

        segregateEvenOdd(arr);

        System.out.print("Array after segregation ");
        for (int i = 0; i < arr.length; i++)
            System.out.print(arr[i]+" ");
    }
}
```

4. Equilibrium index of an array

Equilibrium index of an array is an index such that the sum of elements at lower indexes is equal to the sum of elements at higher indexes. For example, in an array A:

$A[0] = -7, A[1] = 1, A[2] = 5, A[3] = 2, A[4] = -4, A[5] = 3, A[6] = 0$

3 is an equilibrium index, because:

$A[0] + A[1] + A[2] = A[4] + A[5] + A[6]$

6 is also an equilibrium index, because sum of zero elements is zero, i.e.,
 $A[0] + A[1] + A[2] + A[3] + A[4] + A[5] = 0$

7 is not an equilibrium index, because it is not a valid index of array A.

Write a function `int equilibrium(int[] arr, int n);` that given a sequence `arr[]` of size `n`, returns an equilibrium index (if any) or -1 if no equilibrium indexes exist.

```
class EquilibriumIndex
{
    int equilibrium(int arr[], int n)
    {
        // Logic
    }

    public static void main(String[] args)
    {
        EquilibriumIndex equi = new EquilibriumIndex();
        int arr[] = {-7, 1, 5, 2, -4, 3, 0};
        int arr_size = arr.length;
        System.out.println("First equilibrium index is " +
                           equi.equilibrium(arr, arr_size));
    }
}
```

5. Pythagorean Triplet in an array

Given an array of integers, write a function that returns true if there is a triplet (a, b, c) that satisfies $a^2 + b^2 = c^2$.

Example:

Input: arr[] = {3, 1, 4, 6, 5}

Output: True

There is a Pythagorean triplet (3, 4, 5).

Input: arr[] = {10, 4, 6, 12, 5}

Output: False

There is no Pythagorean triplet.

```
import java.io.*;

class PythagoreanTriplet
{
    static boolean isTriplet(int ar[], int n)
    {
        // Logic
    }

    public static void main(String[] args)
    {
        int ar[] = {3, 1, 4, 6, 5};
        int ar_size = ar.length;
        if(isTriplet(ar, ar_size) == true)
            System.out.println("Yes");
        else
            System.out.println("No");
    }
}
```