Is Even?

https://tests.mettl.com/authenticateKey/2bd025dc

```
public int isEven(int input1){
  input1=Math.abs(input1);
                if(input1==0 || input1%2==0)
                        return 2;
                else
                        return 1;
        }
}
```

Is Odd?

https://tests.mettl.com/authenticateKey/dbdac2a9

```
  input1=Math.abs(input1);
                if(input1==0 || input1%2==0)
                if(input1%2!=0)
                        return 2;
                else
                        return 1;
        }
```

Return last digit of the given number

https://tests.mettl.com/authenticateKey/454f012b

```
return (Math.abs(input1%10));
```

Return second last digit of given number

https://tests.mettl.com/authenticateKey/9f87004e

```
        public int secondLastDigitOf(int input1){
                // Read only region end
                input1=Math.abs(input1);
                if(String.valueOf(input1).length()==1)
                        return -1;
```

```
                    else
                              return ((input1/10)%10);
          }
```

<u>Last test case failed for the below code</u>

```
class UserMainCode
{
          public int secondLastDigitOf(int input1){
                    // Read only region end
     if(String.valueOf(input1).length()==1)
                              return -1;
                    else
                              return ((Math.abs(input1)/10)%10);
          }
}
```

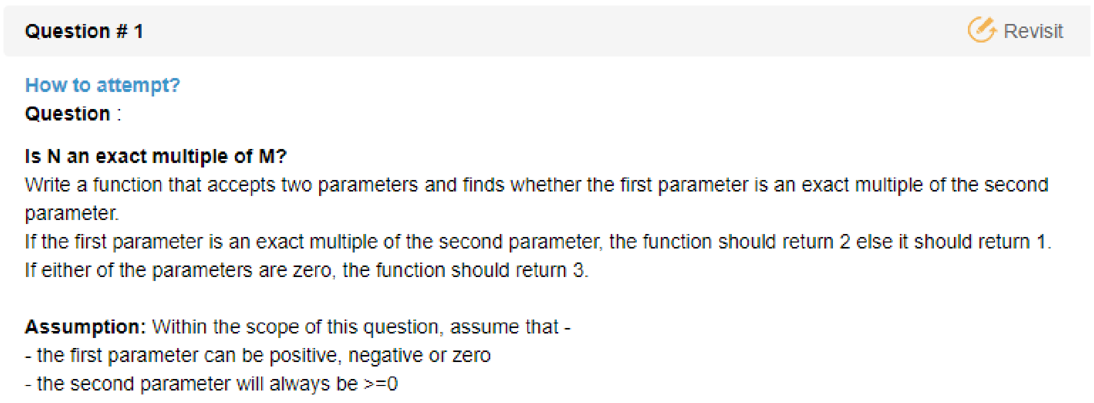
Sum of last digits of two given numbers

https://tests.mettl.com/authenticateKey/783a1fcf

```
          public int addLastDigits(int input1,int input2){
                    input1=Math.abs(input1);
                    input2=Math.abs(input2);
                    return ((input1%10)+(input2%10));
          }
```

Is N an exact multiple of M?

https://tests.mettl.com/authenticateKey/36c4ef58


Question # 1 Revisit

How to attempt?

Question :

Is N an exact multiple of M?

Write a function that accepts two parameters and finds whether the first parameter is an exact multiple of the second parameter.

If the first parameter is an exact multiple of the second parameter, the function should return 2 else it should return 1.

If either of the parameters are zero, the function should return 3.

Assumption: Within the scope of this question, assume that -

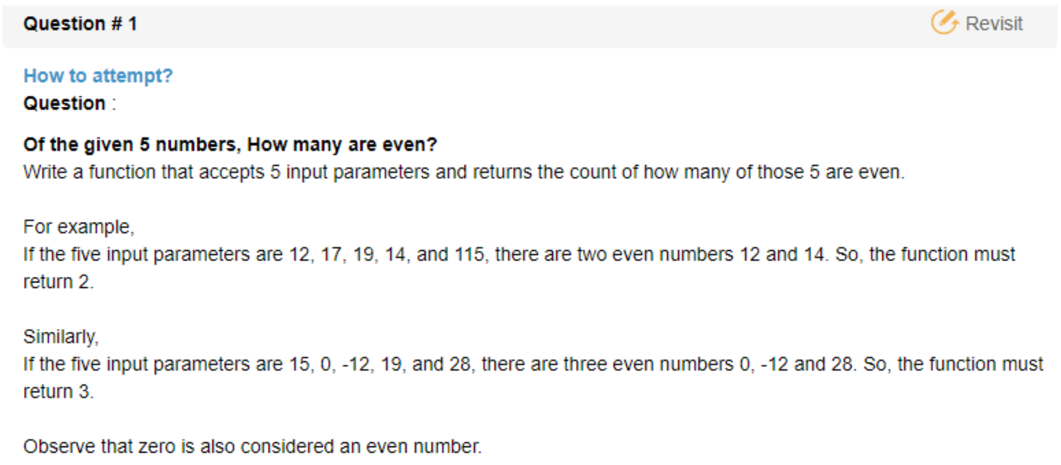
- the first parameter can be positive, negative or zero
- the second parameter will always be >=0

```
public int isMultiple(int input1,int input2){
    if(input1==0 || input2==0)
            return 3;
        else if(Math.abs(input1)%input2==0)
            return 2;
            else
                return 1;
}
```

Of given 5 numbers, how many are even?

https://tests.mettl.com/authenticateKey/8edbe922

```
int count=0;
        if(Math.abs(input1)%2==0 || input1==0)
            count++;
        if(Math.abs(input2)%2==0 || input2==0)
            count++;
        if(Math.abs(input3)%2==0 || input3==0)
            count++;
        if(Math.abs(input4)%2==0 || input4==0)
            count++;
        if(Math.abs(input5)%2==0 || input5==0)
            count++;
        return count;
```



Question # 1 Revisit

How to attempt?

Question :

Of the given 5 numbers, How many are even?

Write a function that accepts 5 input parameters and returns the count of how many of those 5 are even.

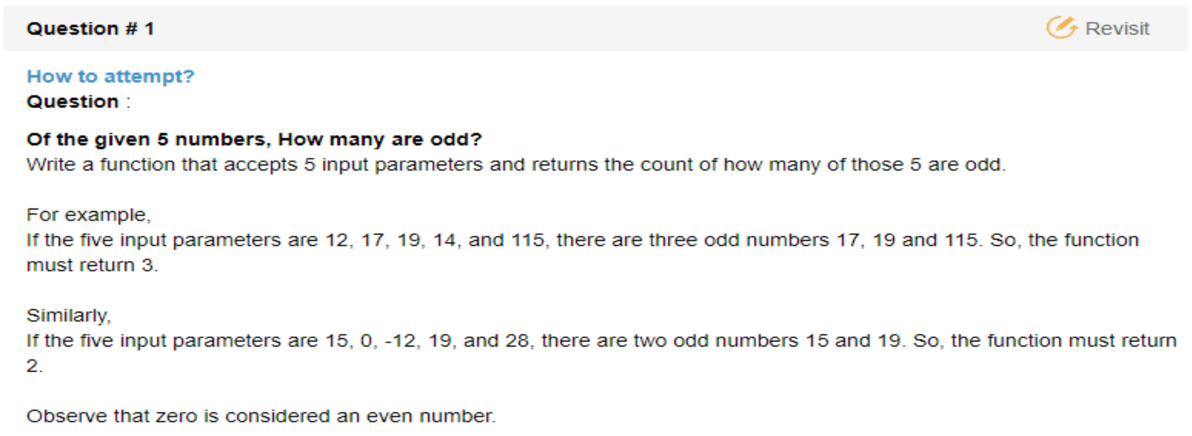
For example,
If the five input parameters are 12, 17, 19, 14, and 115, there are two even numbers 12 and 14. So, the function must return 2.

Similarly,
If the five input parameters are 15, 0, -12, 19, and 28, there are three even numbers 0, -12 and 28. So, the function must return 3.

Observe that zero is also considered an even number.

Of given 5 numbers, how many are odd?

https://tests.mettl.com/authenticateKey/67147bd5



Question # 1 Revisit

How to attempt?

Question :

Of the given 5 numbers, How many are odd?

Write a function that accepts 5 input parameters and returns the count of how many of those 5 are odd.

For example,
If the five input parameters are 12, 17, 19, 14, and 115, there are three odd numbers 17, 19 and 115. So, the function must return 3.

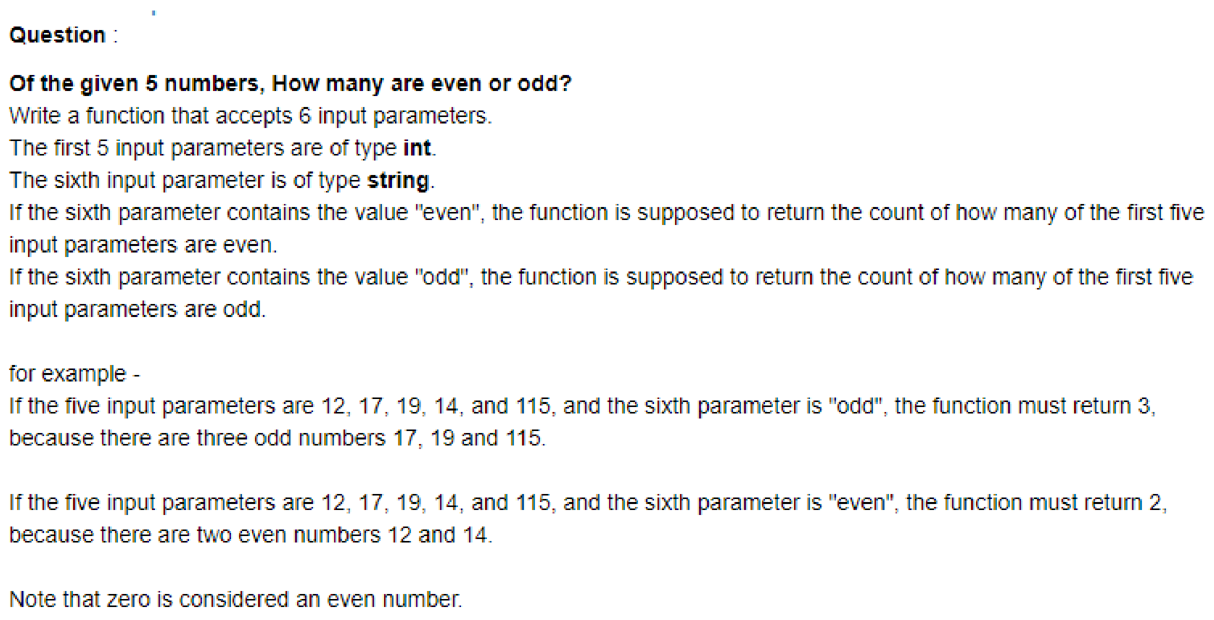
Similarly,
If the five input parameters are 15, 0, -12, 19, and 28, there are two odd numbers 15 and 19. So, the function must return 2.

Observe that zero is considered an even number.

```
    public int countEvens(int input1,int input2,int input3,int input4,int input5){
            // Read only region end
 int count=0;
            if(Math.abs(input1)%2!=0 && input1!=0)
                    count++;
            if(Math.abs(input2)%2!=0 && input2!=0)
                    count++;
            if(Math.abs(input3)%2!=0 && input3!=0)
                    count++;
            if(Math.abs(input4)%2!=0 && input4!=0)
                    count++;
                    if(Math.abs(input5)%2!=0 && input5!=0)

                              count++;

                    return count;

          }
```

Of 5 numbers, how many are even or odd?

https://tests.mettl.com/authenticateKey/607636d7



Question :

Of the given 5 numbers, How many are even or odd?
Write a function that accepts 6 input parameters.
The first 5 input parameters are of type **int**.
The sixth input parameter is of type **string**.
If the sixth parameter contains the value "even", the function is supposed to return the count of how many of the first five input parameters are even.
If the sixth parameter contains the value "odd", the function is supposed to return the count of how many of the first five input parameters are odd.

for example -
If the five input parameters are 12, 17, 19, 14, and 115, and the sixth parameter is "odd", the function must return 3, because there are three odd numbers 17, 19 and 115.

If the five input parameters are 12, 17, 19, 14, and 115, and the sixth parameter is "even", the function must return 2, because there are two even numbers 12 and 14.

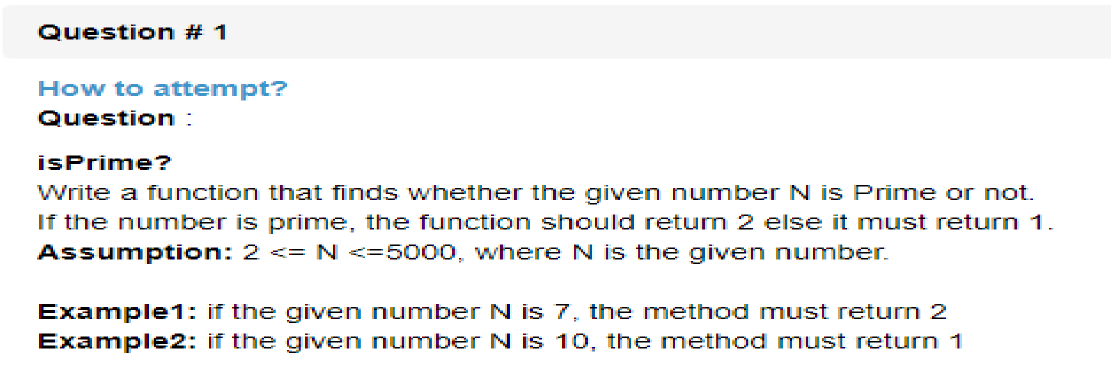
Note that zero is considered an even number.

```
 public int countEvensOdds(int input1,int input2,int input3,int input4,int input5,String input6){

      int count=0;

                    if(input6.equals("odd"))

                    {

                    if(Math.abs(input1)%2!=0 && input1!=0)

                              count++;

                    if(Math.abs(input2)%2!=0 && input2!=0)

                              count++;

                    if(Math.abs(input3)%2!=0 && input3!=0)

                              count++;

                    if(Math.abs(input4)%2!=0 && input4!=0)

                              count++;

                    if(Math.abs(input5)%2!=0 && input5!=0)

                              count++;
```

```
            return count;

            }

            else

            {

        if(Math.abs(input1)%2==0 || input1==0)

                  count++;

            if(Math.abs(input2)%2==0 || input2==0)

                  count++;

            if(Math.abs(input3)%2==0 || input3==0)

                  count++;

            if(Math.abs(input4)%2==0 || input4==0)

                  count++;

            if(Math.abs(input5)%2==0 || input5==0)

                  count++;

            return count;


            }


      }
```

Is Prime?

https://tests.mettl.com/authenticateKey/b1efaa3d



Question # 1

How to attempt?
Question :

isPrime?
Write a function that finds whether the given number N is Prime or not.
If the number is prime, the function should return 2 else it must return 1.
Assumption: 2 <= N <=5000, where N is the given number.

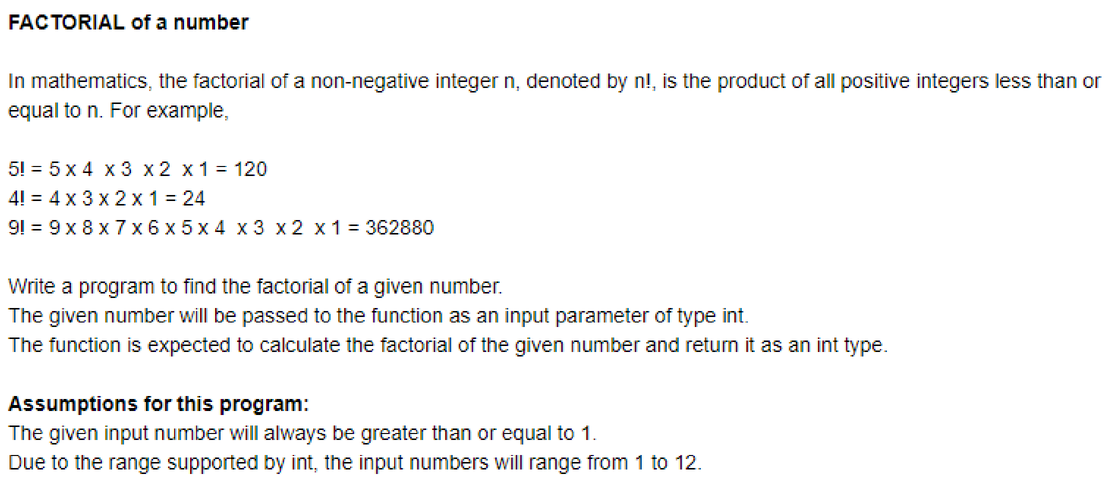
Example1: if the given number N is 7, the method must return 2
Example2: if the given number N is 10, the method must return 1

```
   public int isPrime(int input1){

      boolean flag=true;;
```

```
                for(int i=2;i<input1/2;i++)
                {
                        if(input1%i==0)
                        {
                                flag=false;
                                return 1;
                        }
                }
                if(flag==true)
                        return 2;
                else
                        return 1;
        }
```

Factorial of a number

https://tests.mettl.com/authenticateKey/8c1f2ae



FACTORIAL of a number

In mathematics, the factorial of a non-negative integer n, denoted by n!, is the product of all positive integers less than or equal to n. For example,

5! = 5 x 4 x 3 x 2 x 1 = 120
4! = 4 x 3 x 2 x 1 = 24
9! = 9 x 8 x 7 x 6 x 5 x 4 x 3 x 2 x 1 = 362880

Write a program to find the factorial of a given number.
The given number will be passed to the function as an input parameter of type int.
The function is expected to calculate the factorial of the given number and return it as an int type.

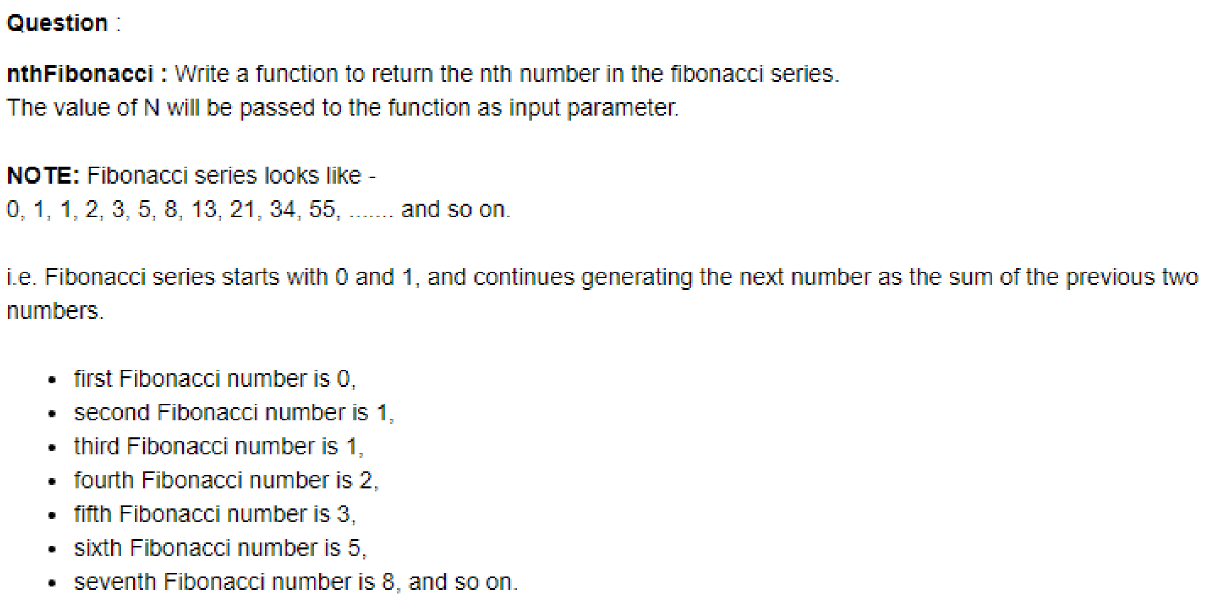
Assumptions for this program:
The given input number will always be greater than or equal to 1.
Due to the range supported by int, the input numbers will range from 1 to 12.

```
public int nFactorial(int input1){
                if(input1==1)
                        return 1;
                else
                        return input1*nFactorial(input1-1);
        }
```

Nth Fibonacci

https://tests.mettl.com/authenticateKey/f390cadf

```
public long nthFibonacci(int input1){
                    int n, a = 0, b = 0, c = 1;
      n = input1;
      for(int i = 1; i <= n; i++)
      {
         a = b;
         b = c;
         c = a + b;
         System.out.print(a+" ");
      }
                    return a;
          }
```



Question :

nthFibonacci : Write a function to return the nth number in the fibonacci series.
The value of N will be passed to the function as input parameter.

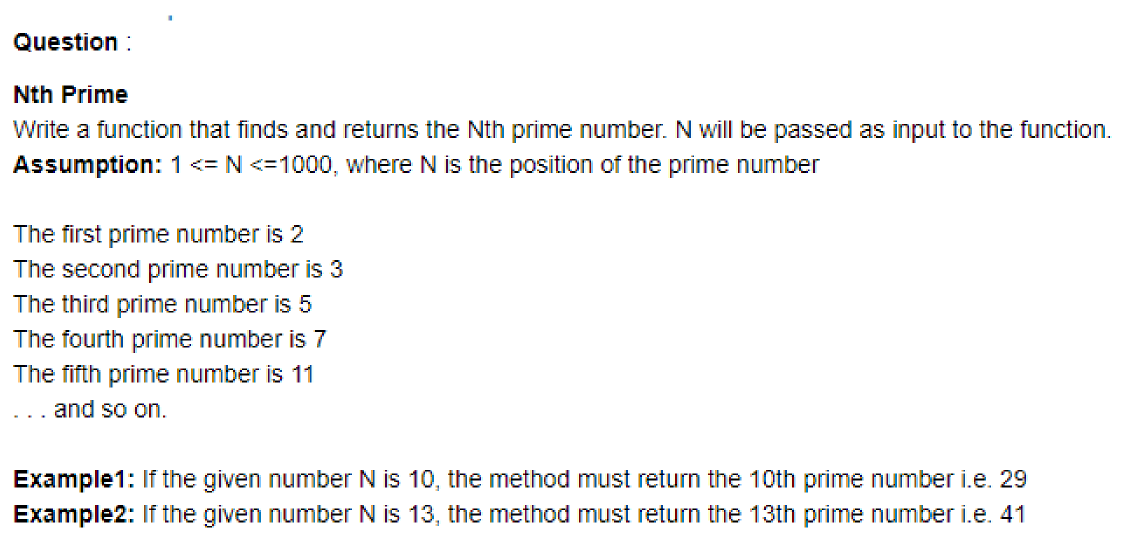
NOTE: Fibonacci series looks like -
0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, ... and so on.

i.e. Fibonacci series starts with 0 and 1, and continues generating the next number as the sum of the previous two numbers.

- first Fibonacci number is 0,
- second Fibonacci number is 1,
- third Fibonacci number is 1,
- fourth Fibonacci number is 2,
- fifth Fibonacci number is 3,
- sixth Fibonacci number is 5,
- seventh Fibonacci number is 8, and so on.

Nth Prime

https://tests.mettl.com/authenticateKey/34fdaa41



Question :

Nth Prime
Write a function that finds and returns the Nth prime number. N will be passed as input to the function.
Assumption: 1 <= N <=1000, where N is the position of the prime number

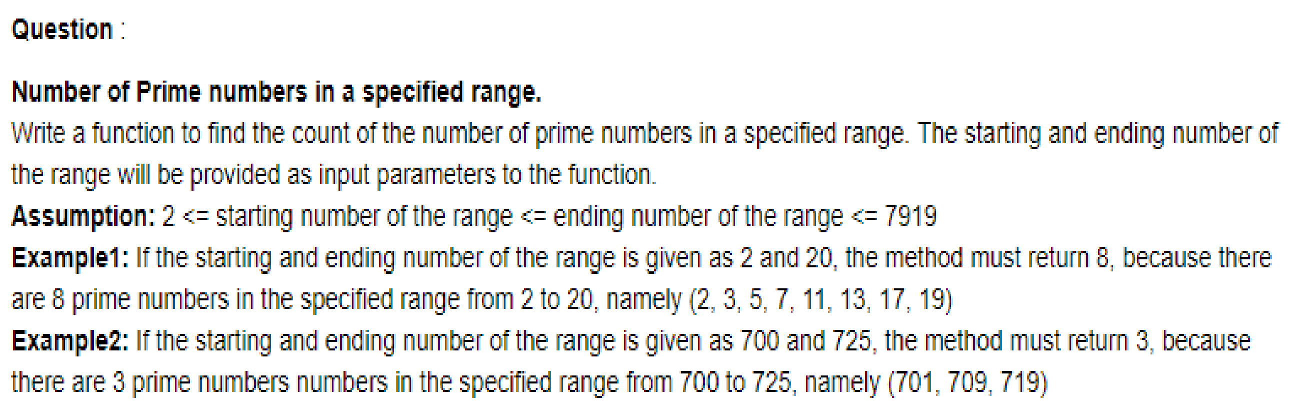
The first prime number is 2
The second prime number is 3
The third prime number is 5
The fourth prime number is 7
The fifth prime number is 11
. . . and so on.

Example1: If the given number N is 10, the method must return the 10th prime number i.e. 29
Example2: If the given number N is 13, the method must return the 13th prime number i.e. 41

```
public int NthPrime(int input1){

                int num=1,count=0,i;

                while(count<input1)

                {

                num=num+1;

                for(i=2;i<=num;i++)

                {

                        if(num%i==0)

                                break;

                }

                if(i==num)

                        count++;

               }

                return num;

               }
```

Number of Primes in a specified range

https://tests.mettl.com/authenticateKey/87c41143



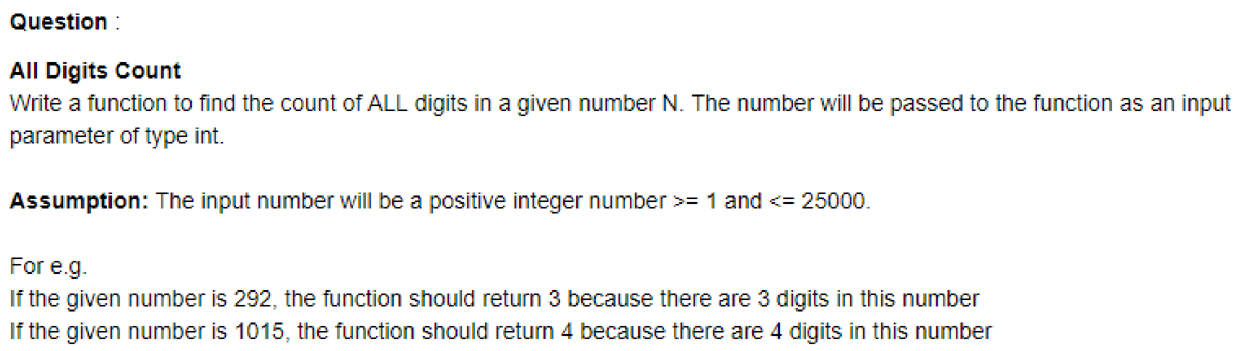
Question :

Number of Prime numbers in a specified range.
Write a function to find the count of the number of prime numbers in a specified range. The starting and ending number of the range will be provided as input parameters to the function.
Assumption: 2 <= starting number of the range <= ending number of the range <= 7919
Example1: If the starting and ending number of the range is given as 2 and 20, the method must return 8, because there are 8 prime numbers in the specified range from 2 to 20, namely (2, 3, 5, 7, 11, 13, 17, 19)
Example2: If the starting and ending number of the range is given as 700 and 725, the method must return 3, because there are 3 prime numbers numbers in the specified range from 700 to 725, namely (701, 709, 719)

```
        public int countPrimesInRange(int input1,int input2){
         int i,num,count=0;
         for(num=input1;num<=input2;num++)
         {
         for(i=2;i<=num;i++)
         {
                 if(num%i==0)
                 break;
         }
         if(i==num)
                 count++;
        }
         return count;
          }
}
```

All Digits Count

https://tests.mettl.com/authenticateKey/ed6b4da



Question :

All Digits Count
Write a function to find the count of ALL digits in a given number N. The number will be passed to the function as an input parameter of type int.

Assumption: The input number will be a positive integer number >= 1 and <= 25000.

For e.g.
If the given number is 292, the function should return 3 because there are 3 digits in this number
If the given number is 1015, the function should return 4 because there are 4 digits in this number

```
public int allDigitsCount(int input1){
                    return (String.valueOf(input1).length());
                  }
}
```

Unique Digits Count

https://tests.mettl.com/authenticateKey/b7aac4a5

Question :

Unique Digits Count
Write a function to find the count of unique digits in a given number N. The number will be passed to the function as an input parameter of type int.
Assumption: The input number will be a positive integer number >= 1 and <= 25000.

For e.g.
If the given number is 292, the function should return 2 because there are only 2 unique digits '2' and '9' in this number
If the given number is 1015, the function should return 3 because there are 3 unique digits in this number, '1', '0', and '5'.

```
public int uniqueDigitsCount(int input1){
                  HashSet<Integer> h=new HashSet<Integer>();
                   while(input1>0)
                   {
                             h.add(input1%10);
                             input1=input1/10;
                   }
                   return h.size();
                  }
```

Non-Repeated Digits' Count

https://tests.mettl.com/authenticateKey/e46500f5

Question :

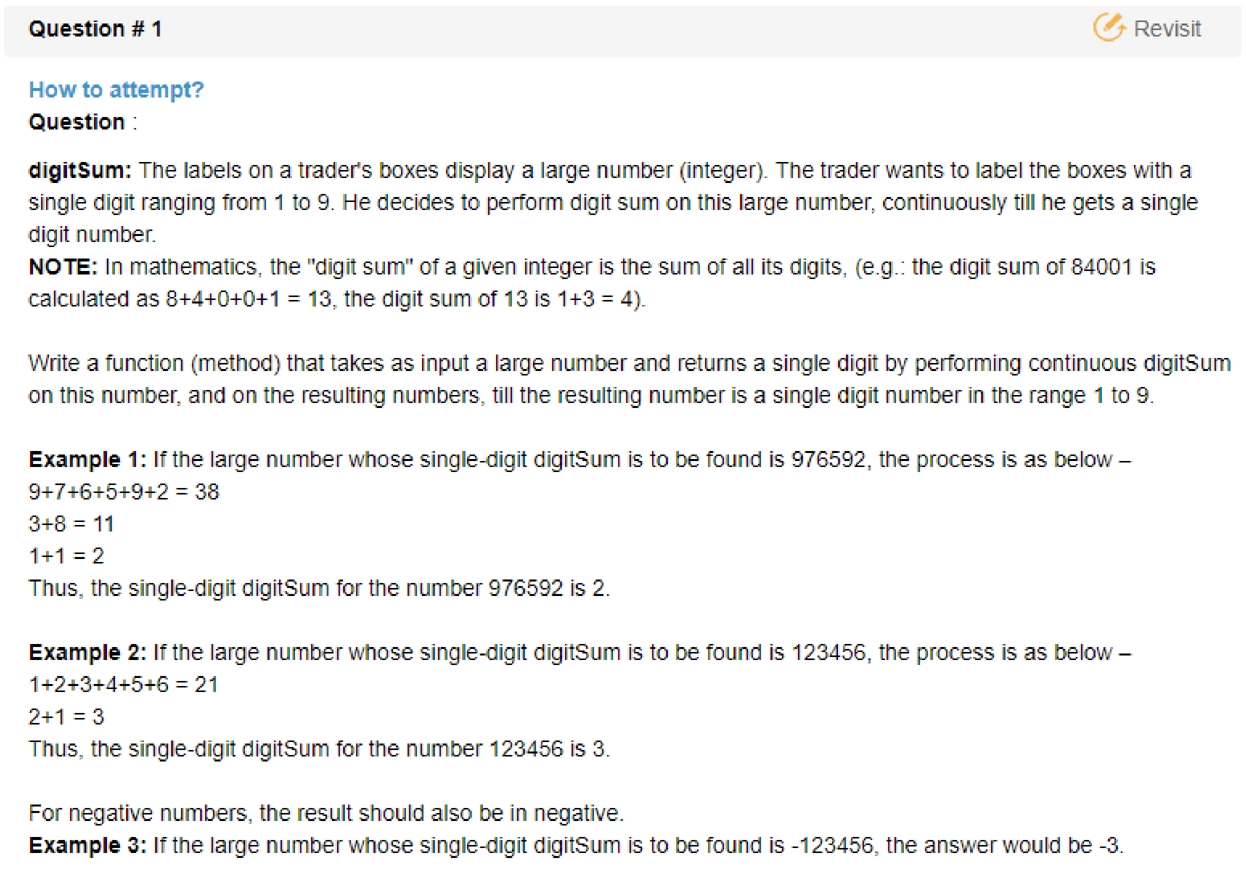
Non-Repeated Digits Count
Write a function to find the count of non-repeated digits in a given number N. The number will be passed to the function as an input parameter of type int.
Assumption: The input number will be a positive integer number >= 1 and <= 25000.

Some examples are as below -
If the given number is 292, the function should return 1 because there is only 1 non-repeated digit '9' in this number
If the given number is 1015, the function should return 2 because there are 2 non-repeated digits in this number, '0', and '5'.
If the given number is 108, the function should return 3 because there are 3 non-repeated digits in this number, '1', '0', and '8'.
If the given number is 22, the function should return 0 because there are NO non-repeated digits in this number.

```
public int nonRepeatDigitsCount(int input1){
String s=""+input1;
 int count=0,count2=0,i;
```

```
                    char digit[]={'0','1','2','3','4','5','6','7','8','9'};

                              for(char c:digit)
                              {
                                        count=0;
                                        for(i=0;i<s.length();i++)
                    {
                                        if(s.charAt(i)==c)
                                        {
                                                  System.out.println("ip ="+s.charAt(i)+" char =:"+c);
                                                  count++;
                                        }
                              }
                              if(count==1)
                              {
                              count2++;
                                        System.out.println("count2="+count2);
                              }

                    }
                              return count2;
}
```

digitSum : sum of all digits in N
https://tests.mettl.com/authenticateKey/ab1d60cc


Question # 1 Revisit

How to attempt?

Question :

digitSum: The labels on a trader's boxes display a large number (integer). The trader wants to label the boxes with a single digit ranging from 1 to 9. He decides to perform digit sum on this large number, continuously till he gets a single digit number.
NOTE: In mathematics, the "digit sum" of a given integer is the sum of all its digits, (e.g.: the digit sum of 84001 is calculated as 8+4+0+0+1 = 13, the digit sum of 13 is 1+3 = 4).

Write a function (method) that takes as input a large number and returns a single digit by performing continuous digitSum on this number, and on the resulting numbers, till the resulting number is a single digit number in the range 1 to 9.

Example 1: If the large number whose single-digit digitSum is to be found is 976592, the process is as below –
9+7+6+5+9+2 = 38
3+8 = 11
1+1 = 2
Thus, the single-digit digitSum for the number 976592 is 2.

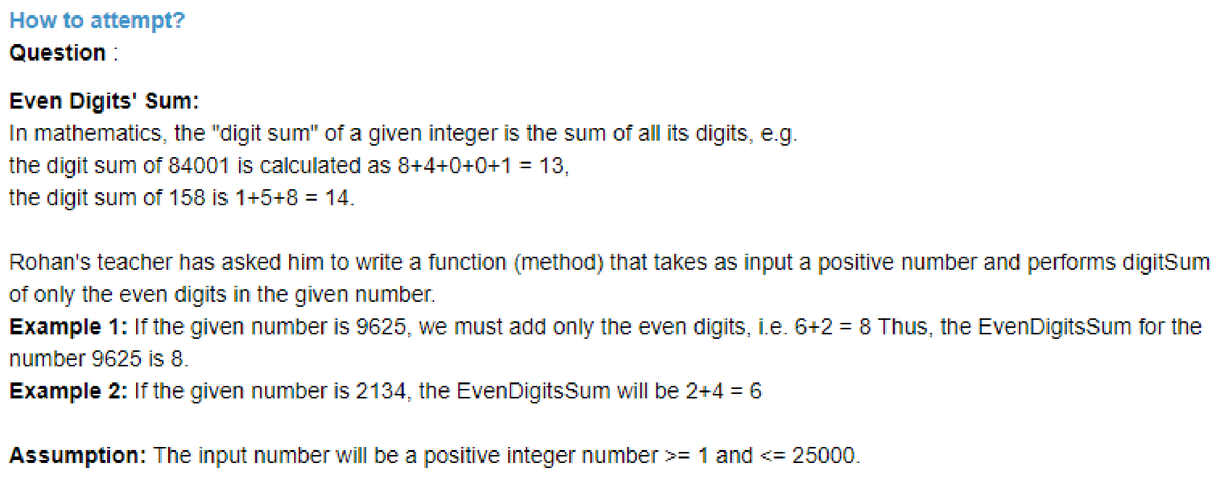
Example 2: If the large number whose single-digit digitSum is to be found is 123456, the process is as below –
1+2+3+4+5+6 = 21
2+1 = 3
Thus, the single-digit digitSum for the number 123456 is 3.

For negative numbers, the result should also be in negative.
Example 3: If the large number whose single-digit digitSum is to be found is -123456, the answer would be -3.

```
public int digitSum(int input1){
                boolean flag=true;
               if(input1<0)
                {
                 flag=false;
                 input1=Math.abs(input1);
                }
                int input2=input1;
                int sum=0;
                while(input1>9)
                {
                         while(input2>0)
                         {
                                sum+=input2%10;
                                input2=input2/10;
                        }
                        input1=sum;
                        input2=input1;
                        sum=0;
                }

                if(flag==false)
                        return -input1;
                else
                return input1;
        }
```

digitSum even: sum of even digits in N

https://tests.mettl.com/authenticateKey/b55d1714



How to attempt?
Question :

Even Digits' Sum:
In mathematics, the "digit sum" of a given integer is the sum of all its digits, e.g.
the digit sum of 84001 is calculated as 8+4+0+0+1 = 13,
the digit sum of 158 is 1+5+8 = 14.

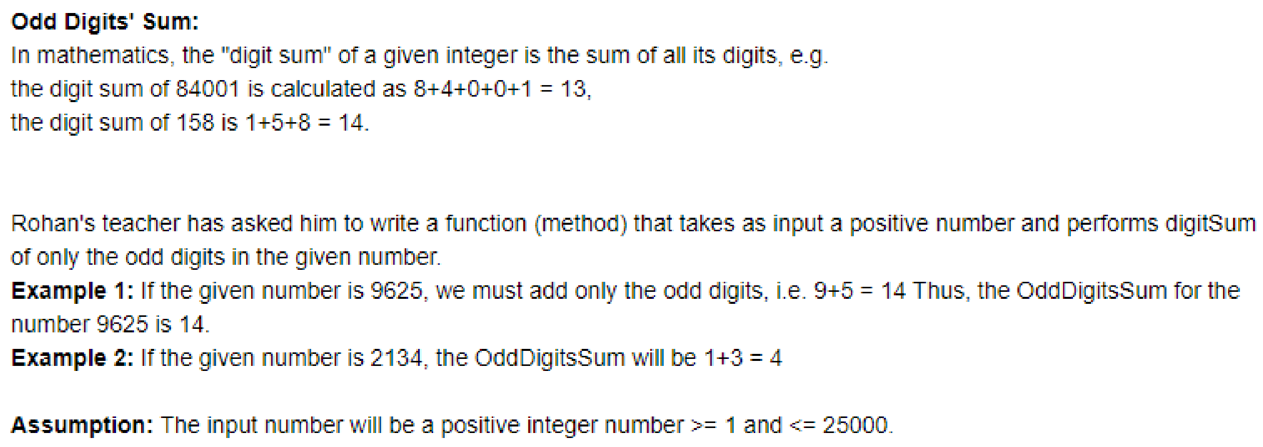
Rohan's teacher has asked him to write a function (method) that takes as input a positive number and performs digitSum of only the even digits in the given number.
Example 1: If the given number is 9625, we must add only the even digits, i.e. 6+2 = 8 Thus, the EvenDigitsSum for the number 9625 is 8.
Example 2: If the given number is 2134, the EvenDigitsSum will be 2+4 = 6

Assumption: The input number will be a positive integer number >= 1 and <= 25000.

```
public int EvenDigitsSum(int input1){
                int sum=0;
                while(input1>0)
                {
    if((input1%10)%2==0)
                        sum+=input1%10;
                input1=input1/10;
                }
                System.out.println("output is:="+sum);
                return sum;
        }
```

digitSum odd: sum of odd digits in N

https://tests.mettl.com/authenticateKey/738fdee0



Odd Digits' Sum:
In mathematics, the "digit sum" of a given integer is the sum of all its digits, e.g.
the digit sum of 84001 is calculated as 8+4+0+0+1 = 13,
the digit sum of 158 is 1+5+8 = 14.

Rohan's teacher has asked him to write a function (method) that takes as input a positive number and performs digitSum of only the odd digits in the given number.
Example 1: If the given number is 9625, we must add only the odd digits, i.e. 9+5 = 14 Thus, the OddDigitsSum for the number 9625 is 14.
Example 2: If the given number is 2134, the OddDigitsSum will be 1+3 = 4

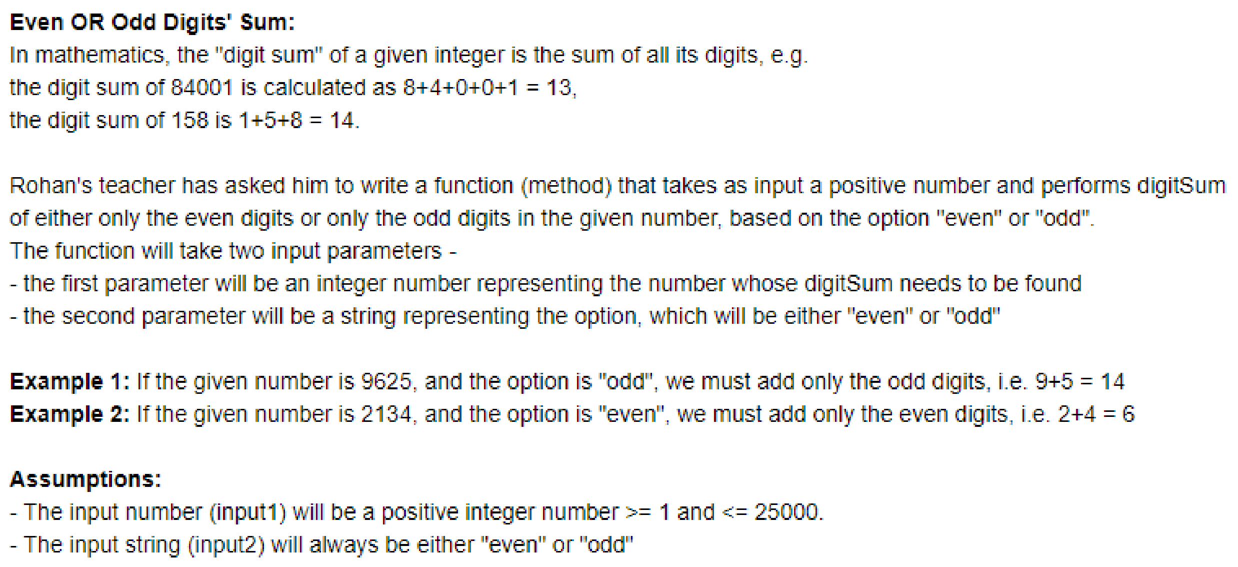
Assumption: The input number will be a positive integer number >= 1 and <= 25000.

```
        public int OddDigitsSum(int input1){
         int sum=0;
```

```
                 while(input1>0)
                 {
         if((input1%10)%2!=0)
                         sum+=input1%10;
                 input1=input1/10;
                 }
                 System.out.println("output is:="+sum);
                 return sum;


                }
      }
```

digitSum opt: sum of even or odd digits

https://tests.mettl.com/authenticateKey/a05abbcf



Even OR Odd Digits' Sum:

In mathematics, the "digit sum" of a given integer is the sum of all its digits, e.g.
the digit sum of 84001 is calculated as 8+4+0+0+1 = 13,
the digit sum of 158 is 1+5+8 = 14.

Rohan's teacher has asked him to write a function (method) that takes as input a positive number and performs digitSum of either only the even digits or only the odd digits in the given number, based on the option "even" or "odd".
The function will take two input parameters -
- the first parameter will be an integer number representing the number whose digitSum needs to be found
- the second parameter will be a string representing the option, which will be either "even" or "odd"

Example 1: If the given number is 9625, and the option is "odd", we must add only the odd digits, i.e. 9+5 = 14
Example 2: If the given number is 2134, and the option is "even", we must add only the even digits, i.e. 2+4 = 6

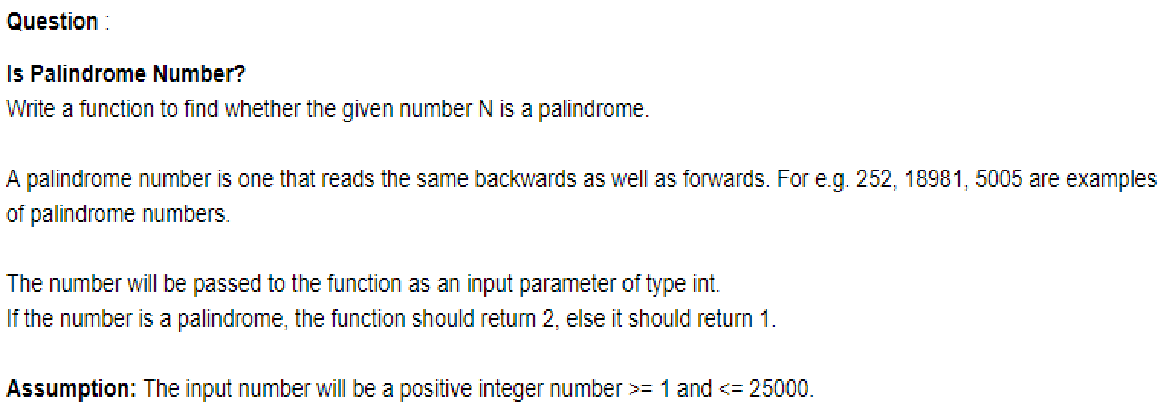
Assumptions:
- The input number (input1) will be a positive integer number >= 1 and <= 25000.
- The input string (input2) will always be either "even" or "odd"

```
public int EvenOddDigitsSum(int input1,String input2){
                if(input2.equals("odd"))
                 {
                         int sum=0;
                 while(input1>0)
                 {
         if((input1%10)%2!=0)
                         sum+=input1%10;
                 input1=input1/10;
                 }
                 System.out.println("output is:="+sum);
                 return sum;

                 }
                 else if(input2.equals("even"))
                 {
                         int sum=0;
```

```
                    while(input1>0)
                    {
              if((input1%10)%2==0)
                         sum+=input1%10;
                    input1=input1/10;
                    }
                    System.out.println("output is:="+sum);
                    return sum;

                    }
                    return 0;
                   }
```

Is Palindrome Number?

https://tests.mettl.com/authenticateKey/28c41d9d



Question :

Is Palindrome Number?
Write a function to find whether the given number N is a palindrome.

A palindrome number is one that reads the same backwards as well as forwards. For e.g. 252, 18981, 5005 are examples of palindrome numbers.

The number will be passed to the function as an input parameter of type int.
If the number is a palindrome, the function should return 2, else it should return 1.

Assumption: The input number will be a positive integer number >= 1 and <= 25000.

```
public int isPalinNum(int input1){
                   String s=""+input1;
                    if(s.length()%2!=0)
                    {
                       String s1=(s.substring(0,s.length()/2));
                        StringBuffer sb=new StringBuffer();
                        sb.append(s.substring(s.length()/2+1));
                        sb.reverse();
                        String rev=String.valueOf((sb.reverse()).toString());
                          if(s1.equals(rev))
                              return 2;
                          else
                              return 1;
                    }
                    else
                    {
                      String s1=(s.substring(0,s.length()/2));
                        StringBuffer sb=new StringBuffer();
                        sb.append(s.substring(s.length()/2));
                        sb.reverse();
                        String rev=String.valueOf((sb.reverse()).toString());
                        if(s1.equals(rev))
                                return 2;
                        else
                        return 1;
                    }
                }
```

Is Palindrome Possible?

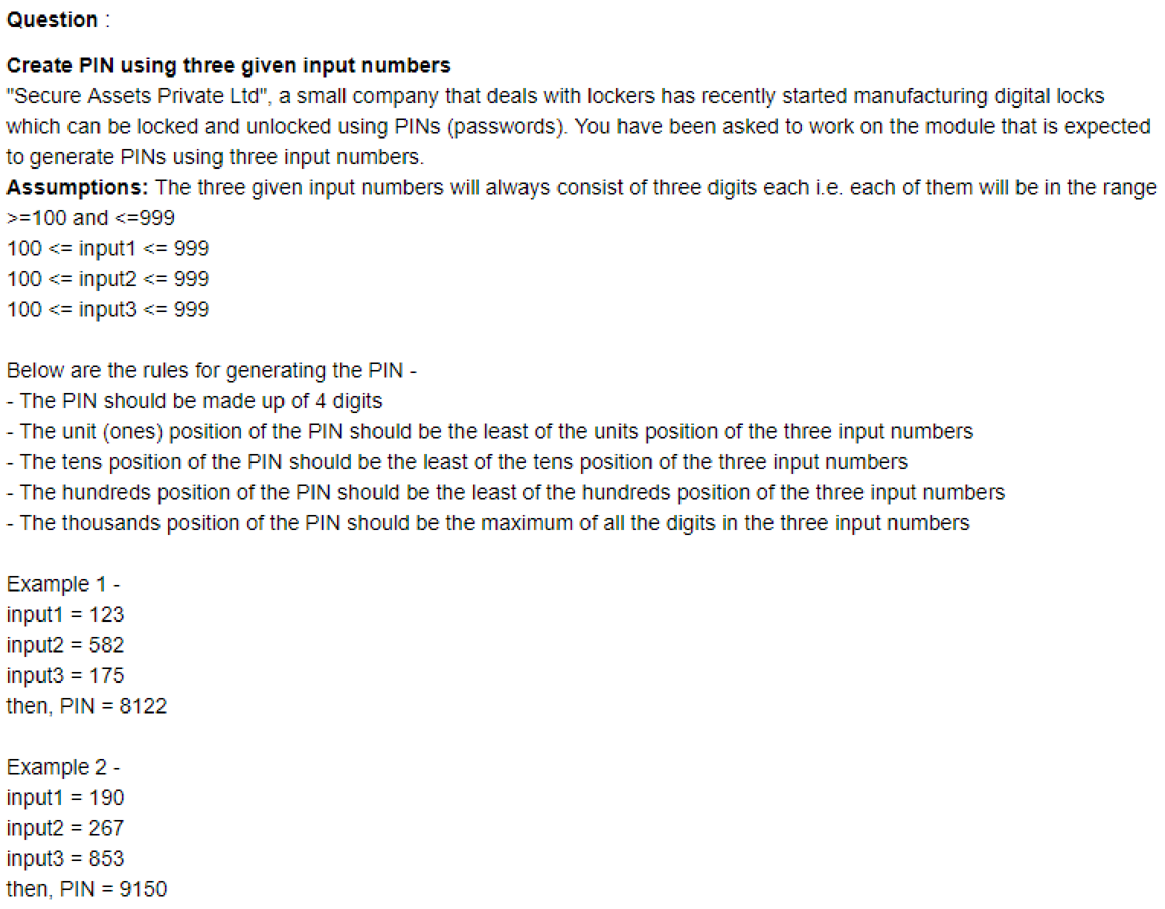
https://tests.mettl.com/authenticateKey/f4fdb02

```
import java.io.*;
import  java.util.*;

// Read only region start
class UserMainCode
{

                public int isPalinNumPossible(int input1){
                    // Read only region end
                    int ip=input1;
                    if(input1<10)
                    {
                            return 2;
                    }
                    else
                    {
                    String s =""+input1;
                    List<Integer> l= new ArrayList<Integer>();
                    int i=0;
                    while(ip>0){
                    l.add(ip%10);
                    ip=ip/10;
                    l.get(i);
                    i++;
                    }
                    int j=0,count=0;
                    for(int k :l)
                    {
                    if(Collections.frequency(l, l.get(j))%2==0)
                            count++;
                    j++;
                    }
                    //System.err.println("count "+count);
                    if(count>s.length()/2)
                            return 2;
                            //System.err.println("palindrome");
                    else
                            return 1;
```

```
                                //System.err.println("not palindrome");
                    }

                    }
}
```

Create PIN using alpha, beta, gamma

https://tests.mettl.com/authenticateKey/be582d9f



Question :

Create PIN using three given input numbers
"Secure Assets Private Ltd", a small company that deals with lockers has recently started manufacturing digital locks which can be locked and unlocked using PINs (passwords). You have been asked to work on the module that is expected to generate PINs using three input numbers.
Assumptions: The three given input numbers will always consist of three digits each i.e. each of them will be in the range >=100 and <=999
100 <= input1 <= 999
100 <= input2 <= 999
100 <= input3 <= 999

Below are the rules for generating the PIN -
- The PIN should be made up of 4 digits
- The unit (ones) position of the PIN should be the least of the units position of the three input numbers
- The tens position of the PIN should be the least of the tens position of the three input numbers
- The hundreds position of the PIN should be the least of the hundreds position of the three input numbers
- The thousands position of the PIN should be the maximum of all the digits in the three input numbers

Example 1 -
input1 = 123
input2 = 582
input3 = 175
then, PIN = 8122

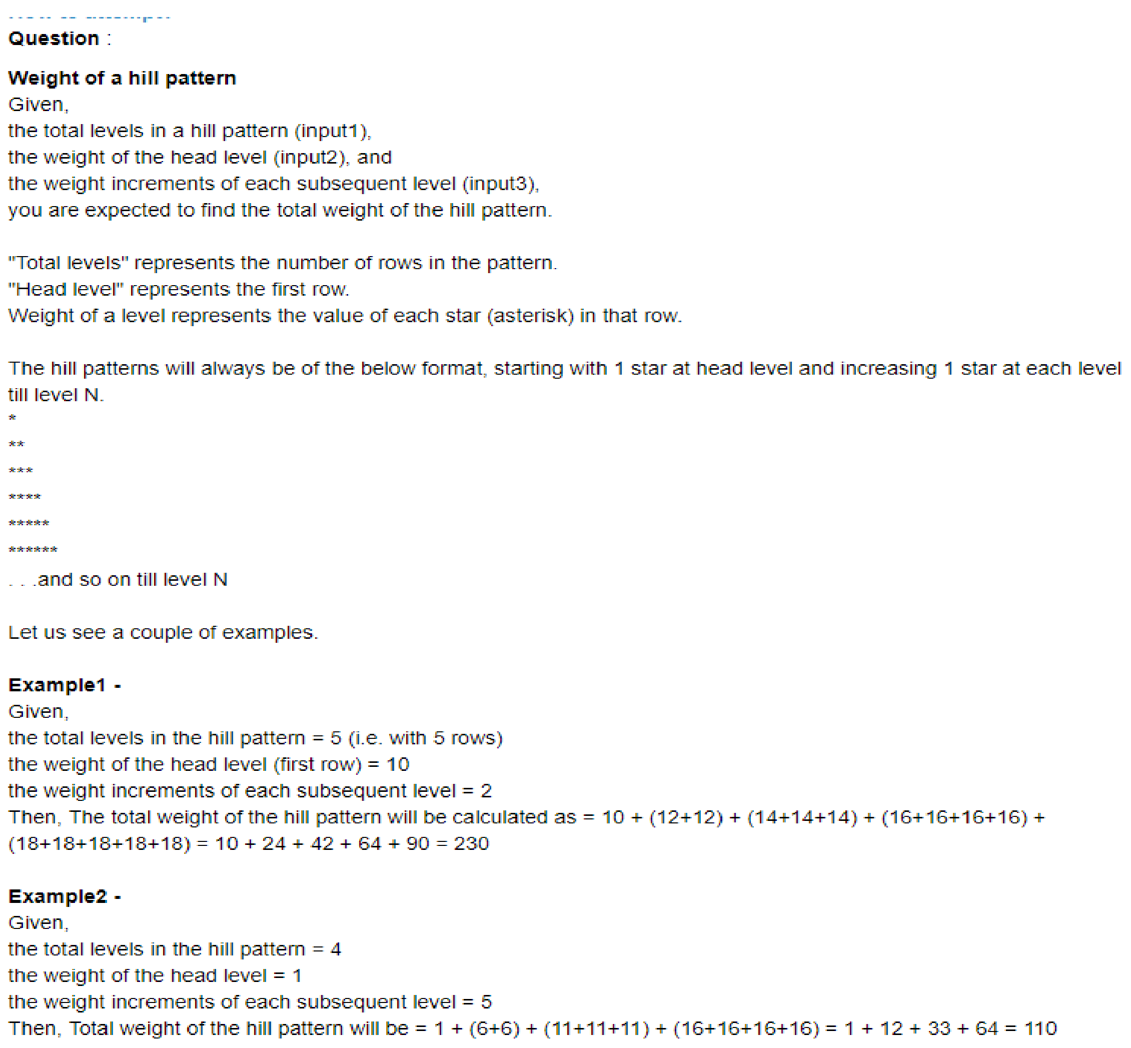
Example 2 -
input1 = 190
input2 = 267
input3 = 853
then, PIN = 9150

```
public int createPIN(int input1,int input2,int input3){
                    List<Integer> max=new ArrayList<Integer>();
                    List<Integer> ones=new ArrayList<Integer>();
                    List<Integer> tens=new ArrayList<Integer>();
                    List<Integer> hundreds=new ArrayList<Integer>();
                    ones.add(input1%10);
                    ones.add(input2%10);
                    ones.add(input3%10);

                    tens.add(input1/10%10);
                    tens.add(input2/10%10);
                    tens.add(input3/10%10);

                    hundreds.add(input1/10/10%10);
                    hundreds.add(input2/10/10%10);
                    hundreds.add(input3/10/10%10);
```

```
              while(input1>0)
              {
                     max.add(input1%10);
                     input1=input1/10;
              }
              while(input2>0)
              {
                     max.add(input2%10);
                     input2=input2/10;
              }
              while(input3>0)
              {
                     max.add(input3%10);
                     input3=input3/10;
              }
             return
 (Collections.max(max)*1000+Collections.min(hundreds)*100+Collections.min(tens)*10+Collections.
 min(ones));
             }
```

Weight of a hill pattern

https://tests.mettl.com/authenticateKey/d612c0e6



Question :

Weight of a hill pattern
Given,
the total levels in a hill pattern (input1),
the weight of the head level (input2), and
the weight increments of each subsequent level (input3),
you are expected to find the total weight of the hill pattern.

"Total levels" represents the number of rows in the pattern.
"Head level" represents the first row.
Weight of a level represents the value of each star (asterisk) in that row.

The hill patterns will always be of the below format, starting with 1 star at head level and increasing 1 star at each level till level N.

```
*
**
***
****
*****
******
```

. . .and so on till level N

Let us see a couple of examples.

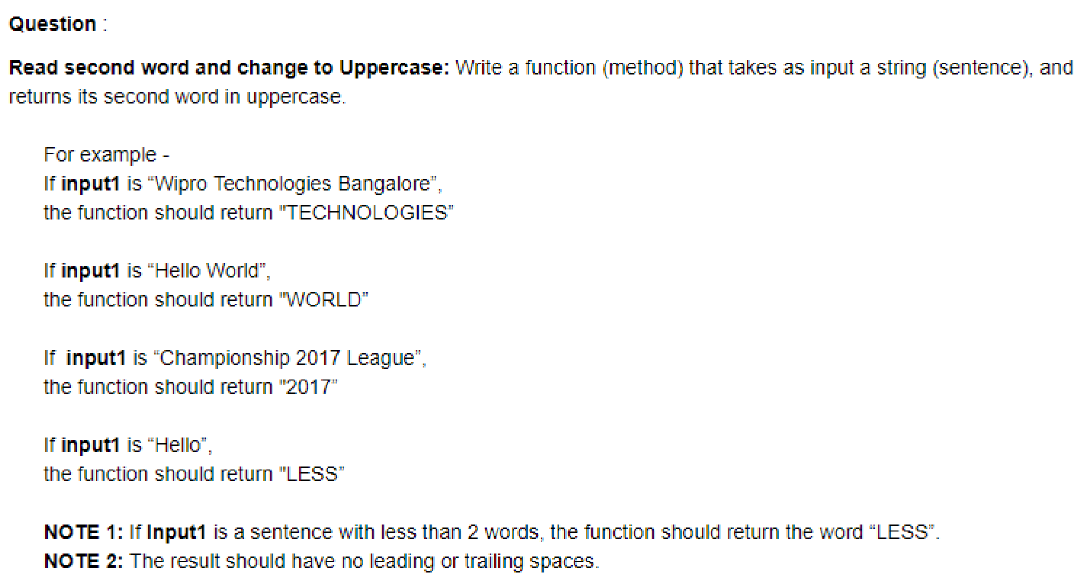
Example1 -
Given,
the total levels in the hill pattern = 5 (i.e. with 5 rows)
the weight of the head level (first row) = 10
the weight increments of each subsequent level = 2
Then, The total weight of the hill pattern will be calculated as = 10 + (12+12) + (14+14+14) + (16+16+16+16) + (18+18+18+18+18) = 10 + 24 + 42 + 64 + 90 = 230

Example2 -
Given,
the total levels in the hill pattern = 4
the weight of the head level = 1
the weight increments of each subsequent level = 5
Then, Total weight of the hill pattern will be = 1 + (6+6) + (11+11+11) + (16+16+16+16) = 1 + 12 + 33 + 64 = 110

```
public int totalHillWeight(int input1,int input2,int input3){
            int k=1,sum=0;
            while(input1>0)
            {

                  for(int j=1;j<=k;j++)
                  {
                        sum+=input2;
                  }
                  System.out.println("\n");
                  k++;
                  input2=input2+input3;
                  input1--;
            }
            return sum;
           }
```

Return second word in Uppercase

https://tests.mettl.com/authenticateKey/4a72723f



Question :

Read second word and change to Uppercase: Write a function (method) that takes as input a string (sentence), and returns its second word in uppercase.

For example -
If **input1** is “Wipro Technologies Bangalore”,
the function should return "TECHNOLOGIES"

If **input1** is “Hello World”,
the function should return "WORLD"

If **input1** is “Championship 2017 League”,
the function should return "2017"

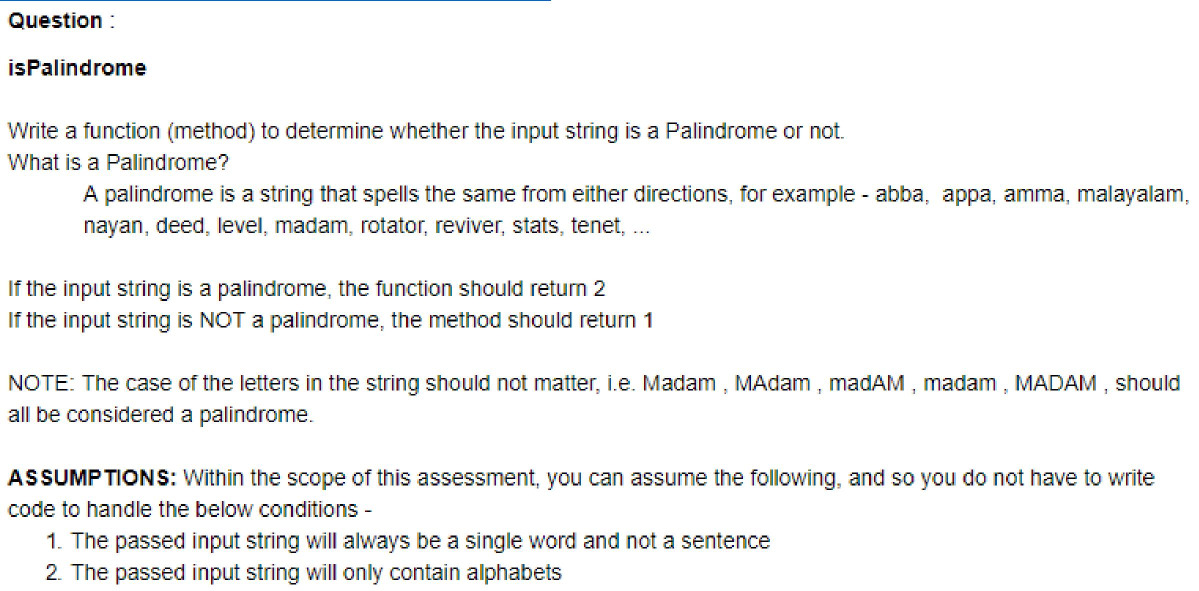
If **input1** is “Hello”,
the function should return "LESS"

NOTE 1: If **Input1** is a sentence with less than 2 words, the function should return the word “LESS”.
NOTE 2: The result should have no leading or trailing spaces.

```
public String secondWordUpperCase(String input1){
                    // Read only region end
                    String s[]=input1.split(" ");
                    if(s.length==1)
                              return "LESS";
                    else
                              return s[1].toUpperCase();

                  }
```

is Palindrome (string)

https://tests.mettl.com/authenticateKey/ffe8042



Question :

isPalindrome

Write a function (method) to determine whether the input string is a Palindrome or not.
What is a Palindrome?
A palindrome is a string that spells the same from either directions, for example - abba, appa, amma, malayalam, nayan, deed, level, madam, rotator, reviver, stats, tenet, ...

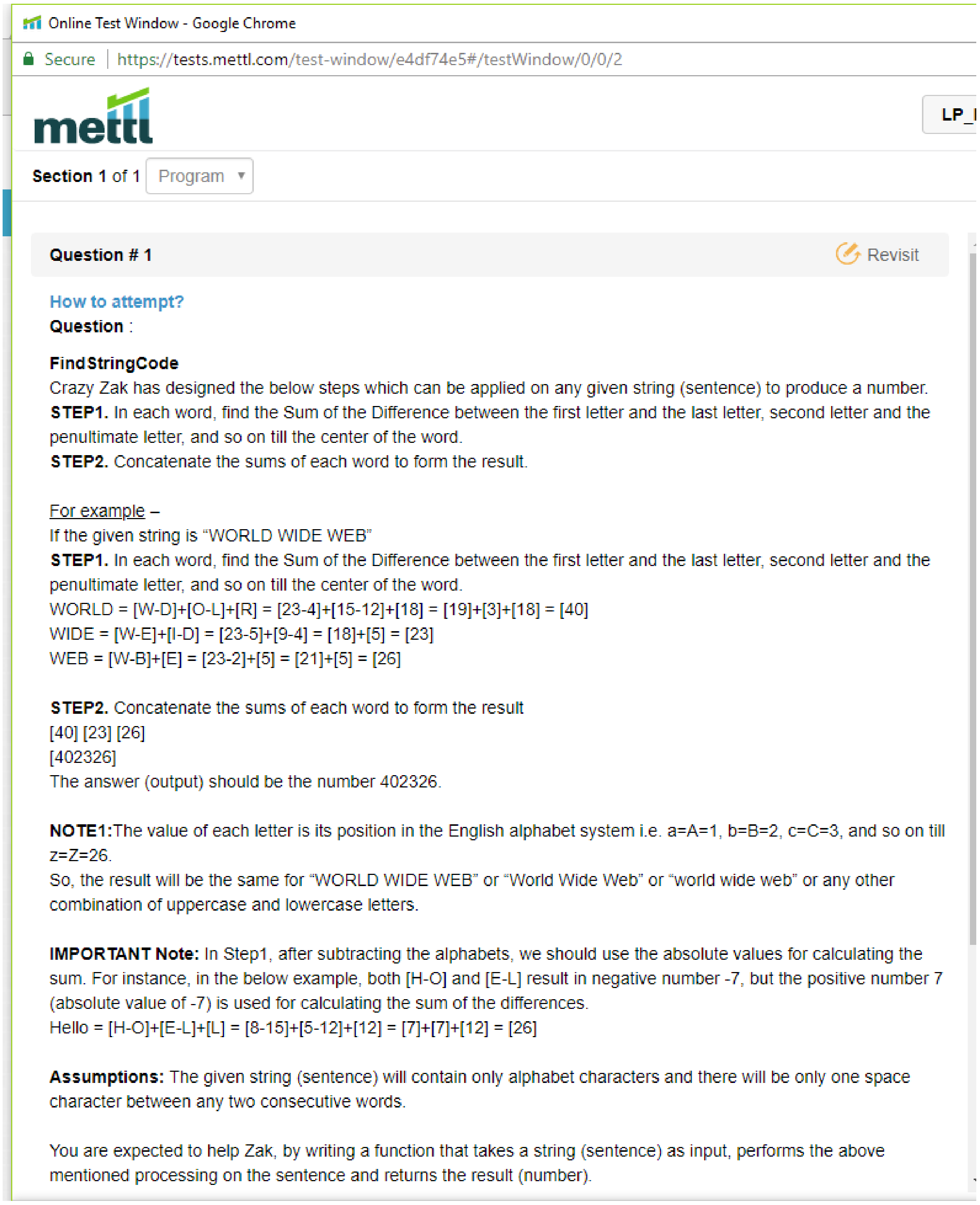
If the input string is a palindrome, the function should return 2
If the input string is NOT a palindrome, the method should return 1

NOTE: The case of the letters in the string should not matter, i.e. Madam , MAdam , madAM , madam , MADAM , should all be considered a palindrome.

ASSUMPTIONS: Within the scope of this assessment, you can assume the following, and so you do not have to write code to handle the below conditions -
1. The passed input string will always be a single word and not a sentence
2. The passed input string will only contain alphabets

```
public int isPalindrome(String input1){
                  String s=input1.toUpperCase();
                    if(s.length()%2!=0)
                    {
```

```
      String s1=(s.substring(0,s.length()/2));
       StringBuffer sb=new StringBuffer();
       sb.append(s.substring(s.length()/2+1));
       sb.reverse();
       String rev=String.valueOf(sb);
       if(s1.equals(rev))
            return 2;
        else
            return 1;
  }
  else
  {
     String s1=(s.substring(0,s.length()/2));
       StringBuffer sb=new StringBuffer();
       sb.append(s.substring(s.length()/2));
       sb.reverse();
       String rev=String.valueOf(sb);
       if(s1.equals(rev))
              return 2;
       else
     return 1;
  }

}
```


Online Test Window - Google Chrome
Secure | https://tests.mettl.com/test-window/e4df74e5#/testWindow/0/0/2
mettl
LP_
Section 1 of 1 Program
Revisit

Question # 1

How to attempt?

Question :

FindStringCode

Crazy Zak has designed the below steps which can be applied on any given string (sentence) to produce a number.

STEP1. In each word, find the Sum of the Difference between the first letter and the last letter, second letter and the penultimate letter, and so on till the center of the word.

STEP2. Concatenate the sums of each word to form the result.

For example –

If the given string is "WORLD WIDE WEB"

STEP1. In each word, find the Sum of the Difference between the first letter and the last letter, second letter and the penultimate letter, and so on till the center of the word.

WORLD = [W-D]+[O-L]+[R] = [23-4]+[15-12]+[18] = [19]+[3]+[18] = [40]

WIDE = [W-E]+[I-D] = [23-5]+[9-4] = [18]+[5] = [23]

WEB = [W-B]+[E] = [23-2]+[5] = [21]+[5] = [26]

STEP2. Concatenate the sums of each word to form the result

[40] [23] [26]

[402326]

The answer (output) should be the number 402326.

NOTE1: The value of each letter is its position in the English alphabet system i.e. a=A=1, b=B=2, c=C=3, and so on till z=Z=26.

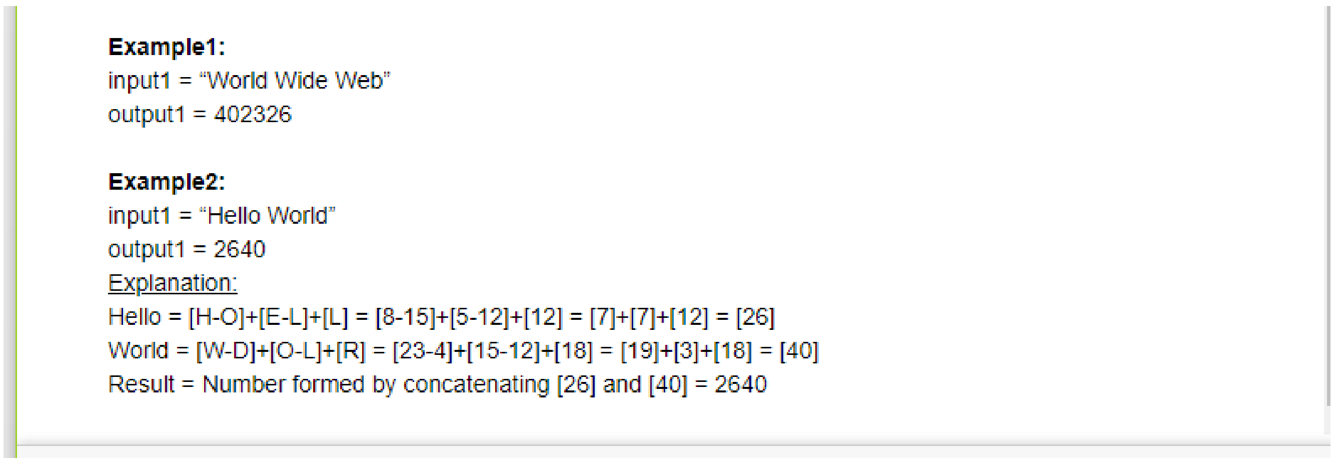
So, the result will be the same for "WORLD WIDE WEB" or "World Wide Web" or "world wide web" or any other combination of uppercase and lowercase letters.

IMPORTANT Note: In Step1, after subtracting the alphabets, we should use the absolute values for calculating the sum. For instance, in the below example, both [H-O] and [E-L] result in negative number -7, but the positive number 7 (absolute value of -7) is used for calculating the sum of the differences.

Hello = [H-O]+[E-L]+[L] = [8-15]+[5-12]+[12] = [7]+[7]+[12] = [26]

Assumptions: The given string (sentence) will contain only alphabet characters and there will be only one space character between any two consecutive words.

You are expected to help Zak, by writing a function that takes a string (sentence) as input, performs the above mentioned processing on the sentence and returns the result (number).



Example1:
input1 = “World Wide Web”
output1 = 402326

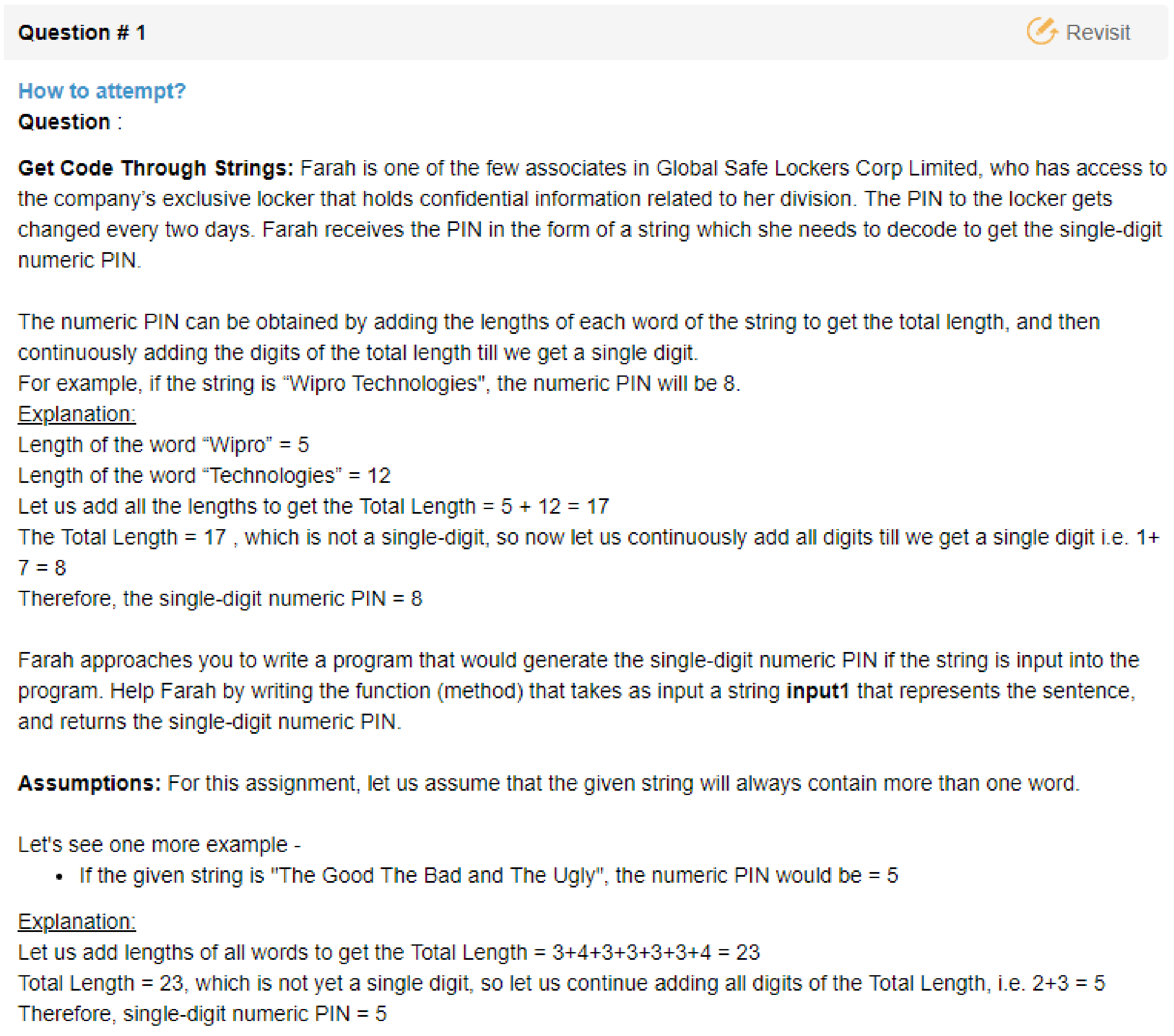
Example2:
input1 = “Hello World”
output1 = 2640
Explanation:
Hello = [H-O]+[E-L]+[L] = [8-15]+[5-12]+[12] = [7]+[7]+[12] = [26]
World = [W-D]+[O-L]+[R] = [23-4]+[15-12]+[18] = [19]+[3]+[18] = [40]
Result = Number formed by concatenating [26] and [40] = 2640

```
public int findStringCode(String input1){
            // Read only region end
            input1=input1.toUpperCase();
            String[] s=input1.split(" ");
            int length=0;
            int length2=s.length-1;
            int sum=0;
            List<Integer> l=new ArrayList<Integer>();
            int left,right;
            while(length<=length2)
            {
                  right=s[length].length()-1;
                  left=0;
                  while(left<right)
                  {
                        System.out.println(s[length].charAt(left));
                        System.out.println(s[length].charAt(right));
                        System.out.println(s[length].charAt(left)-s[length].charAt(right));
                        sum+=Math.abs((s[length].charAt(left)-s[length].charAt(right)));
                        left=left+1;
                        right=right-1;
                  }
                  if(s[length].length()%2!=0)
```

```
                {
                        sum+=Math.abs(s[length].charAt(left)-64);
                        System.out.println("odd char="+s[length].charAt(left));
                        //l.add(s[length].charAt(left)-64);
                }
                l.add(sum);
                sum=0;
                length=length+1;
        }
        StringBuffer s2=new StringBuffer();
        for(int i:l)
        {
                s2.append(i);
        }
        String s3=s2.toString();
        sum=Integer.parseInt(s3);
        return sum;


}
```

<u>Code 2</u>

```
import java.io.*;
import  java.util.*;

// Read only region start
class UserMainCode
{

        public int findStringCode(String input1){
                // Read only region end
                input1=input1.toUpperCase();
                String[] ip=input1.split(" ");
                int mid=0,sum=0,sum2=0;
                StringBuffer sb=new StringBuffer();
                for(String temp:ip)
                {
                        int j=temp.length()-1;
                        for(int i=0;i<=j;i++)
                        {
                                sum2=0;
                                System.out.println(temp.charAt(i)+"-"+temp.charAt(j));
                                sum2+=Math.abs((temp.charAt(i)-64)-(temp.charAt(j)-64));
                                System.out.println("temp sum= "+sum2);
                                sum+=sum2;
                                j--;
                        }
                        if(temp.length()%2!=0)
                        {
                                System.out.println(temp.charAt(temp.length()/2));
                                sum+=temp.charAt(temp.length()/2)-64;
                                System.out.println("odd "+sum);
                        }
                        else
                        System.out.println("even "+sum);
                        sb.append(sum);
                        sum=0;

                }
                System.out.println(sb.toString());
            return Integer.parseInt(sb.toString());
        }
}
```



Question # 1 Revisit

How to attempt?

Question :

Get Code Through Strings: Farah is one of the few associates in Global Safe Lockers Corp Limited, who has access to the company's exclusive locker that holds confidential information related to her division. The PIN to the locker gets changed every two days. Farah receives the PIN in the form of a string which she needs to decode to get the single-digit numeric PIN.

The numeric PIN can be obtained by adding the lengths of each word of the string to get the total length, and then continuously adding the digits of the total length till we get a single digit.
For example, if the string is "Wipro Technologies", the numeric PIN will be 8.
Explanation:
Length of the word "Wipro" = 5
Length of the word "Technologies" = 12
Let us add all the lengths to get the Total Length = 5 + 12 = 17
The Total Length = 17 , which is not a single-digit, so now let us continuously add all digits till we get a single digit i.e. 1+ 7 = 8
Therefore, the single-digit numeric PIN = 8

Farah approaches you to write a program that would generate the single-digit numeric PIN if the string is input into the program. Help Farah by writing the function (method) that takes as input a string **input1** that represents the sentence, and returns the single-digit numeric PIN.

Assumptions: For this assignment, let us assume that the given string will always contain more than one word.

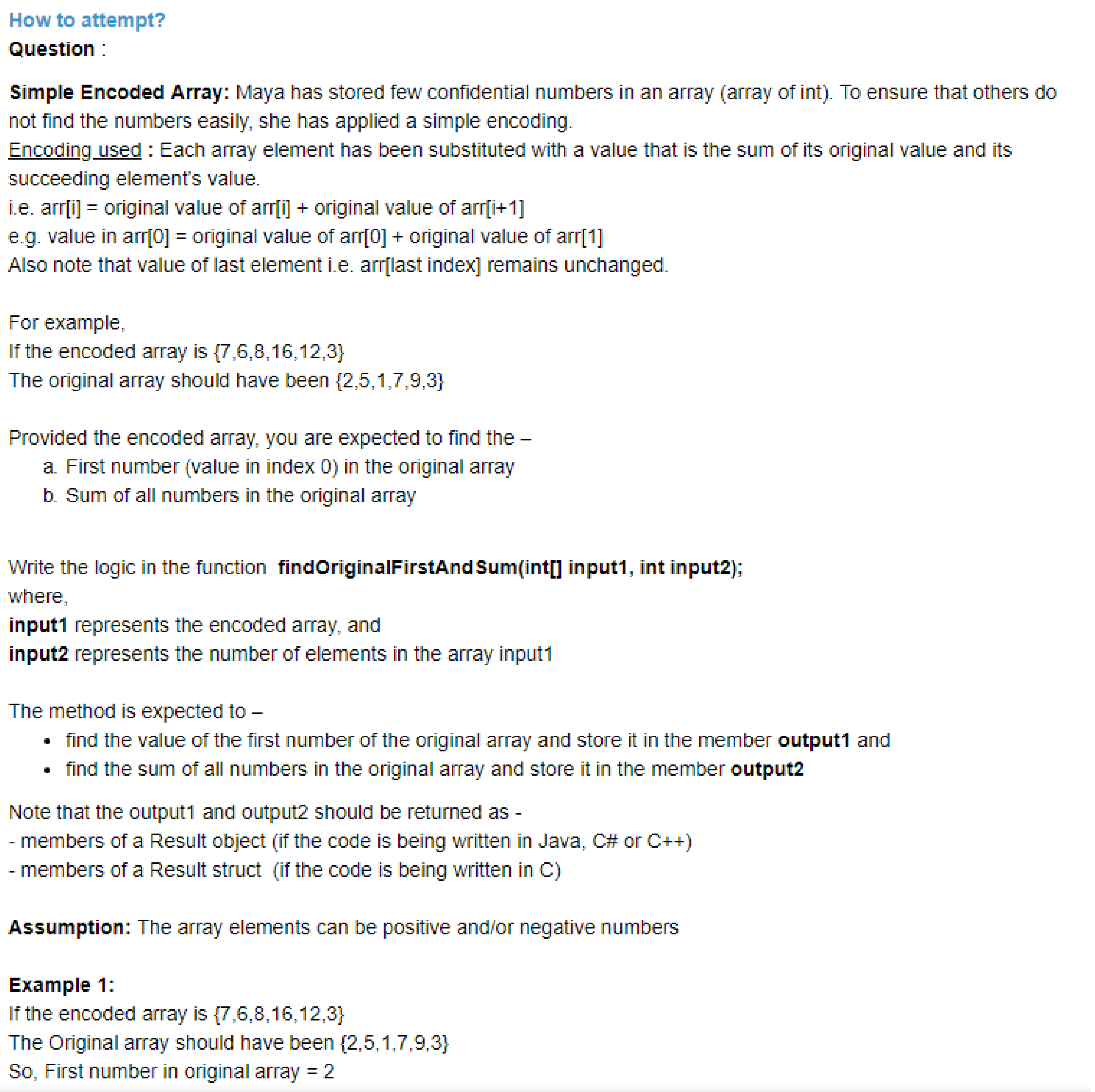
Let's see one more example -

- If the given string is "The Good The Bad and The Ugly", the numeric PIN would be = 5

Explanation:
Let us add lengths of all words to get the Total Length = 3+4+3+3+3+3+4 = 23
Total Length = 23, which is not yet a single digit, so let us continue adding all digits of the Total Length, i.e. 2+3 = 5
Therefore, single-digit numeric PIN = 5

```
public int getCodeThroughStrings(String input1){

            // Read only region end

            input1=input1.toUpperCase();

            String[] s=input1.split(" ");

            int length=s.length-1;

            int sum=0;

            int sum2=0;

            while(length>=0)
```

```
        {
                sum+=s[length].length();
                System.out.println(sum);
                while(sum>9)
                {
                        while(sum>0)
                        {
                        sum2+=sum%10;
                        sum=sum/10;
                        }
                        sum=sum2;
                        sum2=0;
                }
        System.out.println(sum);
                length=length-1;
        }
        System.out.println("final sum is"+sum);
        return sum;

}
```



How to attempt?

Question :

Simple Encoded Array: Maya has stored few confidential numbers in an array (array of int). To ensure that others do not find the numbers easily, she has applied a simple encoding.
Encoding used **:** Each array element has been substituted with a value that is the sum of its original value and its succeeding element's value.
i.e. arr[i] = original value of arr[i] + original value of arr[i+1]
e.g. value in arr[0] = original value of arr[0] + original value of arr[1]
Also note that value of last element i.e. arr[last index] remains unchanged.

For example,
If the encoded array is {7,6,8,16,12,3}
The original array should have been {2,5,1,7,9,3}

Provided the encoded array, you are expected to find the –

a. First number (value in index 0) in the original array
b. Sum of all numbers in the original array

Write the logic in the function **findOriginalFirstAndSum(int[] input1, int input2);**
where,
input1 represents the encoded array, and
input2 represents the number of elements in the array input1

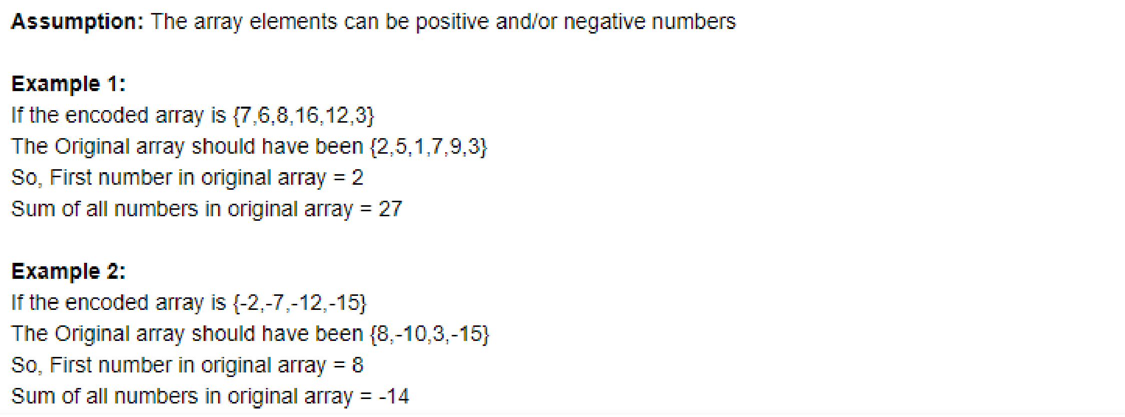
The method is expected to –

- find the value of the first number of the original array and store it in the member **output1** and
- find the sum of all numbers in the original array and store it in the member **output2**

Note that the output1 and output2 should be returned as -
- members of a Result object (if the code is being written in Java, C# or C++)
- members of a Result struct (if the code is being written in C)

Assumption: The array elements can be positive and/or negative numbers

Example 1:
If the encoded array is {7,6,8,16,12,3}
The Original array should have been {2,5,1,7,9,3}
So, First number in original array = 2



Assumption: The array elements can be positive and/or negative numbers

Example 1:
If the encoded array is {7,6,8,16,12,3}
The Original array should have been {2,5,1,7,9,3}
So, First number in original array = 2
Sum of all numbers in original array = 27

Example 2:
If the encoded array is {-2,-7,-12,-15}
The Original array should have been {8,-10,3,-15}
So, First number in original array = 8
Sum of all numbers in original array = -14

```
public class Result{
                public final int output1;
                public final int output2;

                public Result(int out1, int out2){
                        output1 = out1;
                        output2 = out2;
                }
               }

   public Result findOriginalFirstAndSum(int[] input1,int input2){
               // Read only region end
     int sum=input1[input2-1];
                for(int i=input2-2;i>=0;i--)
                {
                        input1[i]=input1[i]-input1[i+1];
                        sum+=input1[i];
                        System.out.println(input1[i]);
                }
                Result r=new Result(input1[0],sum);
                return r;
   }
```

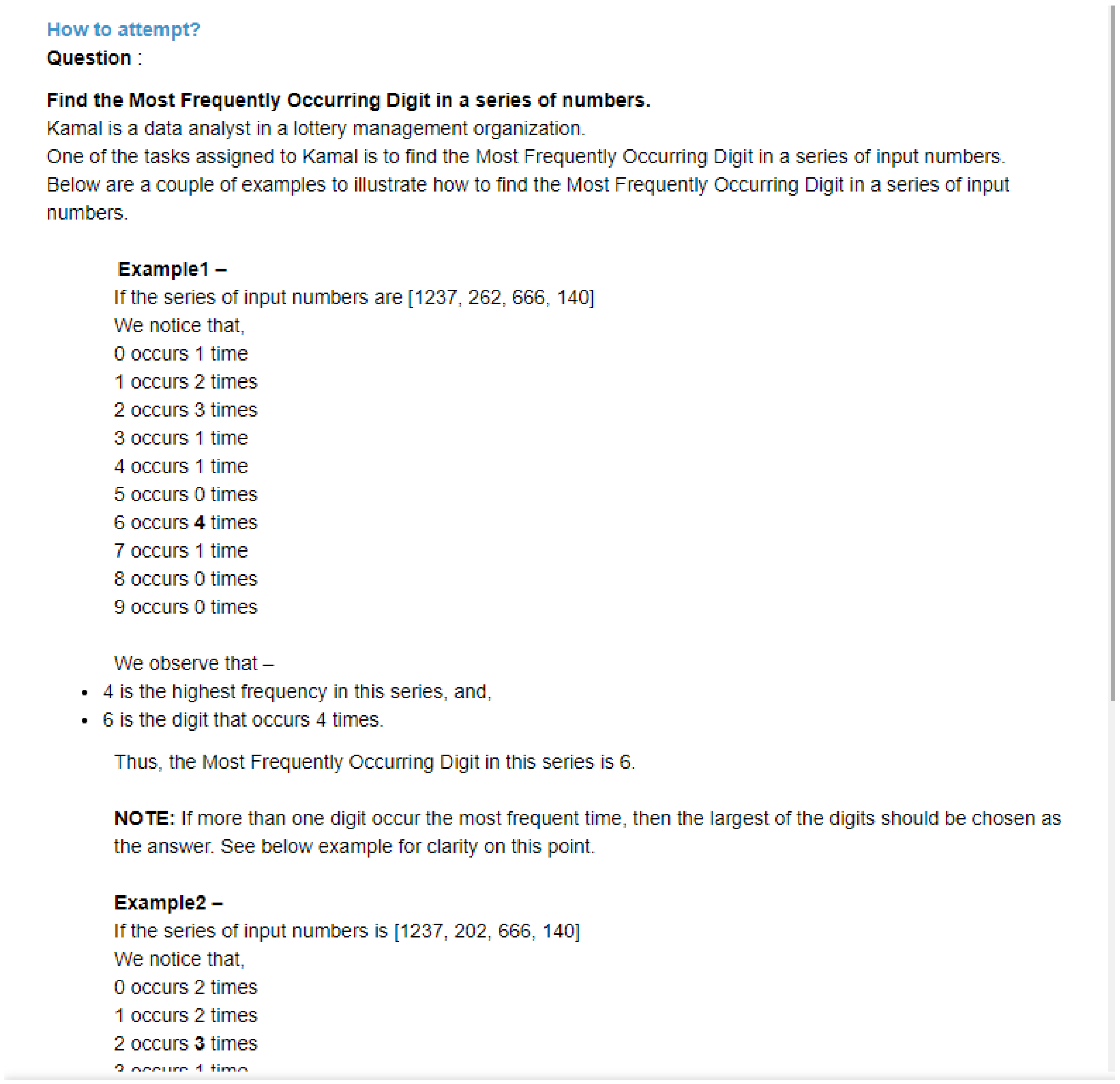
Decreasing sequence

```
import java.io.*;
import  java.util.*;

// Read only region start
class UserMainCode
{

               public class Result{
                public final int output1;
                public final int output2;

                public Result(int out1, int out2){
                        output1 = out1;
                        output2 = out2;
                }
               }

   public Result decreasingSeq(int[] input1,int input2){
               // Read only region end
     int max=0,seq=0,num=0;;
                int count=0;
                int count2=0;
                if(input2<=1)
                {
                        Result r= new Result(0,0);
                        return r;
                }
                else
                {
                        for(int i=0;i<input2-1;i++)
                        {
                                if(input1[i]>input1[i+1])
                                {
                                        count++;
                                        count2++;
                                System.out.println("num1="+input1[i]);
                                System.out.println("num2="+input1[i+1]);
                                }

                                if(max<count)
                                {
                                  num++;
                                        max=count;
                                        seq=count2;
                                        System.out.println(max);
                                        System.out.println(seq);
                             System.out.println("count="+num);
                                        //seq=0;
           count=0;
                                        i++;
                                }
                        }

                        Result r= new Result(num,seq);
                        return r;
                }

   }
}
```

Code2

```
import java.io.*;
import  java.util.*;

// Read only region start
class UserMainCode
{

               public class Result{
                 public final int output1;
                 public final int output2;

                 public Result(int out1, int out2){
                         output1 = out1;
                         output2 = out2;
                 }
```

```
                    }

    public Result decreasingSeq(int[] input1,int input2){
                    // Read only region end
        int output1=1,output2=1,seq=0,max=0,flag=0;
                    for(int i=0;i<input2-1;i++)
                    {

                            if(input1[i]>input1[i+1])
                            {
                                    System.out.println(input1[i]+"compared with"+input1[i+1]);
                                    output1++;
                                    flag=1;
                                    if(max<output1)
                                    {
                                            max=output1;
                                            System.out.println("max ="+max);
                                    }
                                    if(i==input2-2){
                                            seq++;
                                            System.out.println("final seq ="+seq);
                                    }
                            }
                            else
                            {
                                    System.out.println("Entered else");

                                    if(output1>1)
                                    {
                                            System.out.println("Entered seq");
                                            seq++;
                                            System.out.println("seq ="+seq);
                                            flag=0;
                                    }
                                    i++;
                                    output1=1;
                            }

                    }
                    Result res;
                    if(max==0)
                    res= new Result(0,0);
                    else
                    res= new Result(seq,max);
                    return res;
    }
}
```



How to attempt?

Question :

Find the Most Frequently Occurring Digit in a series of numbers.

Kamal is a data analyst in a lottery management organization.
One of the tasks assigned to Kamal is to find the Most Frequently Occurring Digit in a series of input numbers.
Below are a couple of examples to illustrate how to find the Most Frequently Occurring Digit in a series of input numbers.

Example1 –

If the series of input numbers are [1237, 262, 666, 140]
We notice that,
0 occurs 1 time
1 occurs 2 times
2 occurs 3 times
3 occurs 1 time
4 occurs 1 time
5 occurs 0 times
6 occurs **4** times
7 occurs 1 time
8 occurs 0 times
9 occurs 0 times

We observe that –

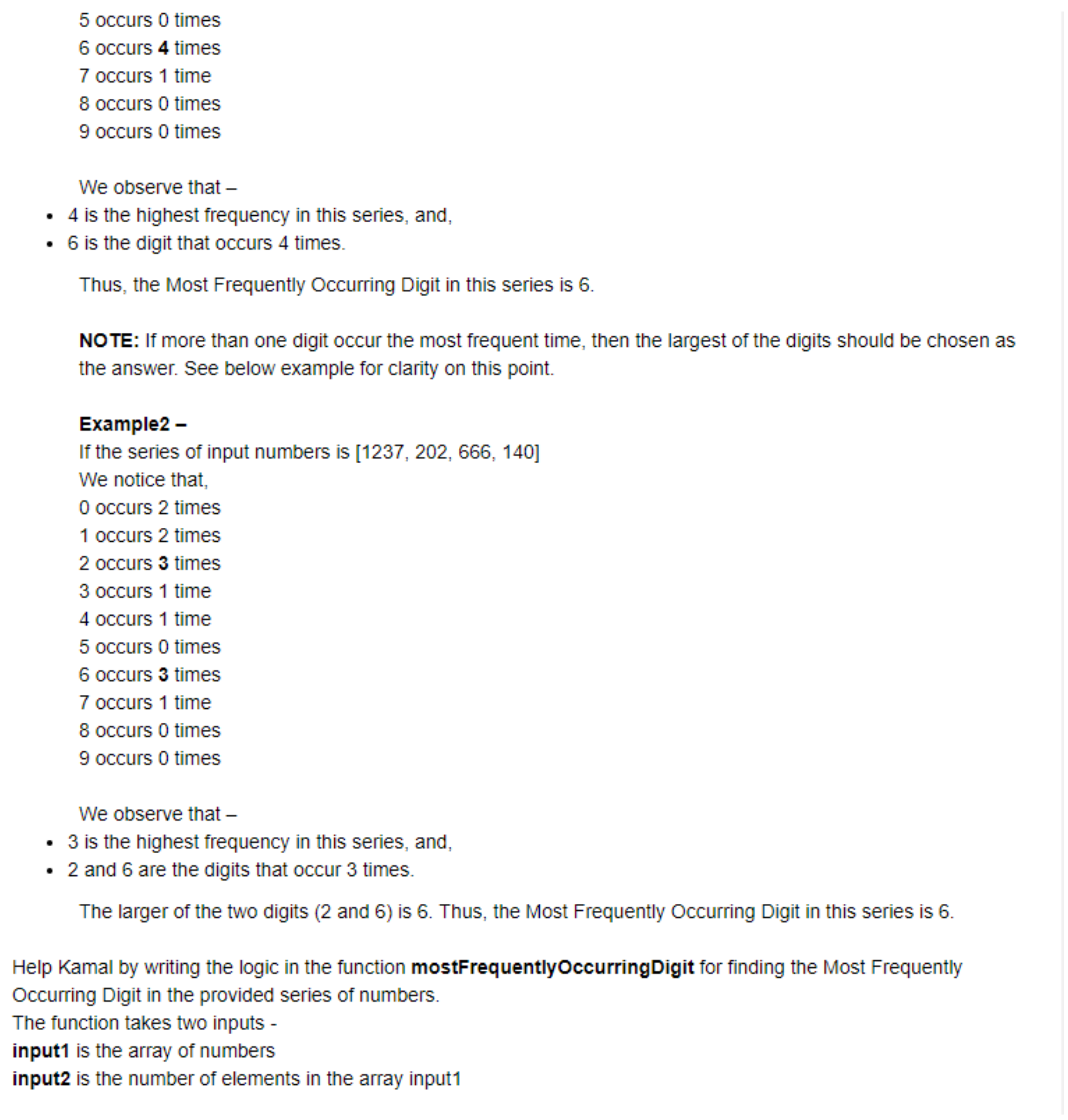
- 4 is the highest frequency in this series, and,
- 6 is the digit that occurs 4 times.

Thus, the Most Frequently Occurring Digit in this series is 6.

NOTE: If more than one digit occur the most frequent time, then the largest of the digits should be chosen as the answer. See below example for clarity on this point.

Example2 –

If the series of input numbers is [1237, 202, 666, 140]
We notice that,
0 occurs 2 times
1 occurs 2 times
2 occurs **3** times
3 occurs 1 time


5 occurs 0 times
6 occurs **4** times
7 occurs 1 time
8 occurs 0 times
9 occurs 0 times

We observe that –

- 4 is the highest frequency in this series, and,
- 6 is the digit that occurs 4 times.

Thus, the Most Frequently Occurring Digit in this series is 6.

NOTE: If more than one digit occur the most frequent time, then the largest of the digits should be chosen as the answer. See below example for clarity on this point.

Example2 –
If the series of input numbers is [1237, 202, 666, 140]
We notice that,
0 occurs 2 times
1 occurs 2 times
2 occurs **3** times
3 occurs 1 time
4 occurs 1 time
5 occurs 0 times
6 occurs **3** times
7 occurs 1 time
8 occurs 0 times
9 occurs 0 times

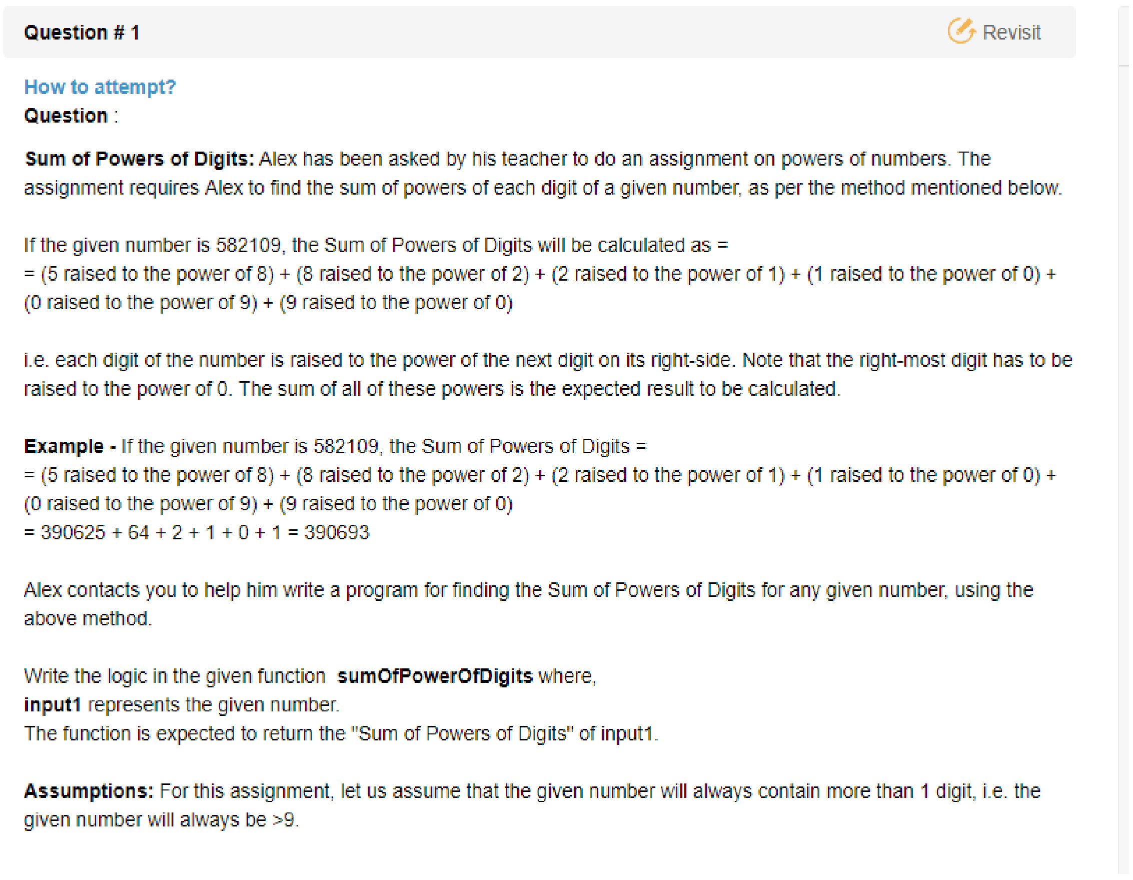
We observe that –

- 3 is the highest frequency in this series, and,
- 2 and 6 are the digits that occur 3 times.

The larger of the two digits (2 and 6) is 6. Thus, the Most Frequently Occurring Digit in this series is 6.

Help Kamal by writing the logic in the function **mostFrequentlyOccurringDigit** for finding the Most Frequently Occurring Digit in the provided series of numbers.
The function takes two inputs -
input1 is the array of numbers
input2 is the number of elements in the array input1

```
import java.io.*;
import  java.util.*;

// Read only region start
class UserMainCode
{

                public int mostFrequentlyOccurringDigit(int[] input1,int input2){
                 // Read only region end
                 List<Integer> l=new ArrayList<Integer>();
                 int freq=0,digit=input1[0]%10;
                 for(int i=0;i<input2;i++)
                 {
                          while(input1[i]>0)
                          {
                                   l.add(input1[i]%10);
                                   if(freq<Collections.frequency(l,input1[i]%10))
                                   {
                                           freq=Collections.frequency(l,input1[i]%10);
                                           digit=input1[i]%10;
                                           System.out.println("frequncy ="+freq);
                                           System.out.println("Digit ="+digit);
                                   }
                                   else if(freq==Collections.frequency(l,input1[i]%10))
                                   {
                                           freq=Collections.frequency(l,input1[i]%10);
                                           if(digit<input1[i]%10)
                                                   digit=input1[i]%10;
                                           System.out.println("frequncy ="+freq);
                                           System.out.println("Digit ="+digit);
                                   }
                                   System.out.println("Frequency of
"+input1[i]%10+"is"+Collections.frequency(l,input1[i]%10));
                                   input1[i]=input1[i]/10;
                          }

                 }
                 return digit;

                }
}
```



Question # 1

Revisit

How to attempt?

Question :

Sum of Powers of Digits: Alex has been asked by his teacher to do an assignment on powers of numbers. The assignment requires Alex to find the sum of powers of each digit of a given number, as per the method mentioned below.

If the given number is 582109, the Sum of Powers of Digits will be calculated as =
= (5 raised to the power of 8) + (8 raised to the power of 2) + (2 raised to the power of 1) + (1 raised to the power of 0) + (0 raised to the power of 9) + (9 raised to the power of 0)

i.e. each digit of the number is raised to the power of the next digit on its right-side. Note that the right-most digit has to be raised to the power of 0. The sum of all of these powers is the expected result to be calculated.

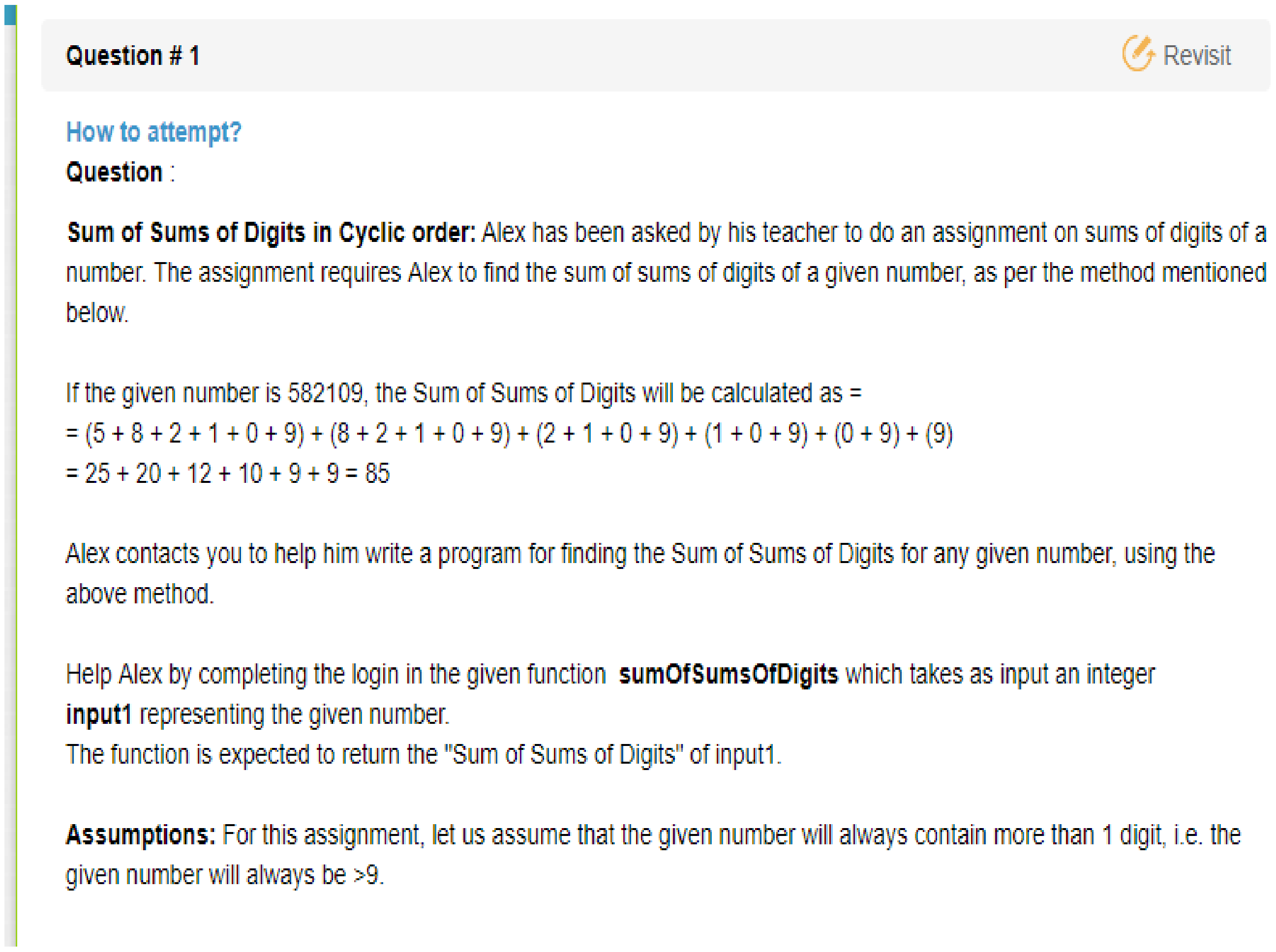
Example - If the given number is 582109, the Sum of Powers of Digits =
= (5 raised to the power of 8) + (8 raised to the power of 2) + (2 raised to the power of 1) + (1 raised to the power of 0) + (0 raised to the power of 9) + (9 raised to the power of 0)
= 390625 + 64 + 2 + 1 + 0 + 1 = 390693

Alex contacts you to help him write a program for finding the Sum of Powers of Digits for any given number, using the above method.

Write the logic in the given function **sumOfPowerOfDigits** where,
input1 represents the given number.
The function is expected to return the "Sum of Powers of Digits" of input1.

Assumptions: For this assignment, let us assume that the given number will always contain more than 1 digit, i.e. the given number will always be >9.

```
import java.io.*;
import  java.util.*;

// Read only region start
class UserMainCode
{

            public int sumOfPowerOfDigits(int input1){
             // Read only region end
             int sum=0;
             StringBuffer s=new StringBuffer();
             s.append(input1);
             s.reverse();
             String s2=s.toString();
             input1=Integer.parseInt(s2);
             while(input1>0)
             {
                   sum+=Math.pow(input1%10,input1/10%10);
                   input1=input1/10;
             }
             return sum;
            }
}
```



Question # 1

Revisit

How to attempt?

Question :

Sum of Sums of Digits in Cyclic order: Alex has been asked by his teacher to do an assignment on sums of digits of a number. The assignment requires Alex to find the sum of sums of digits of a given number, as per the method mentioned below.

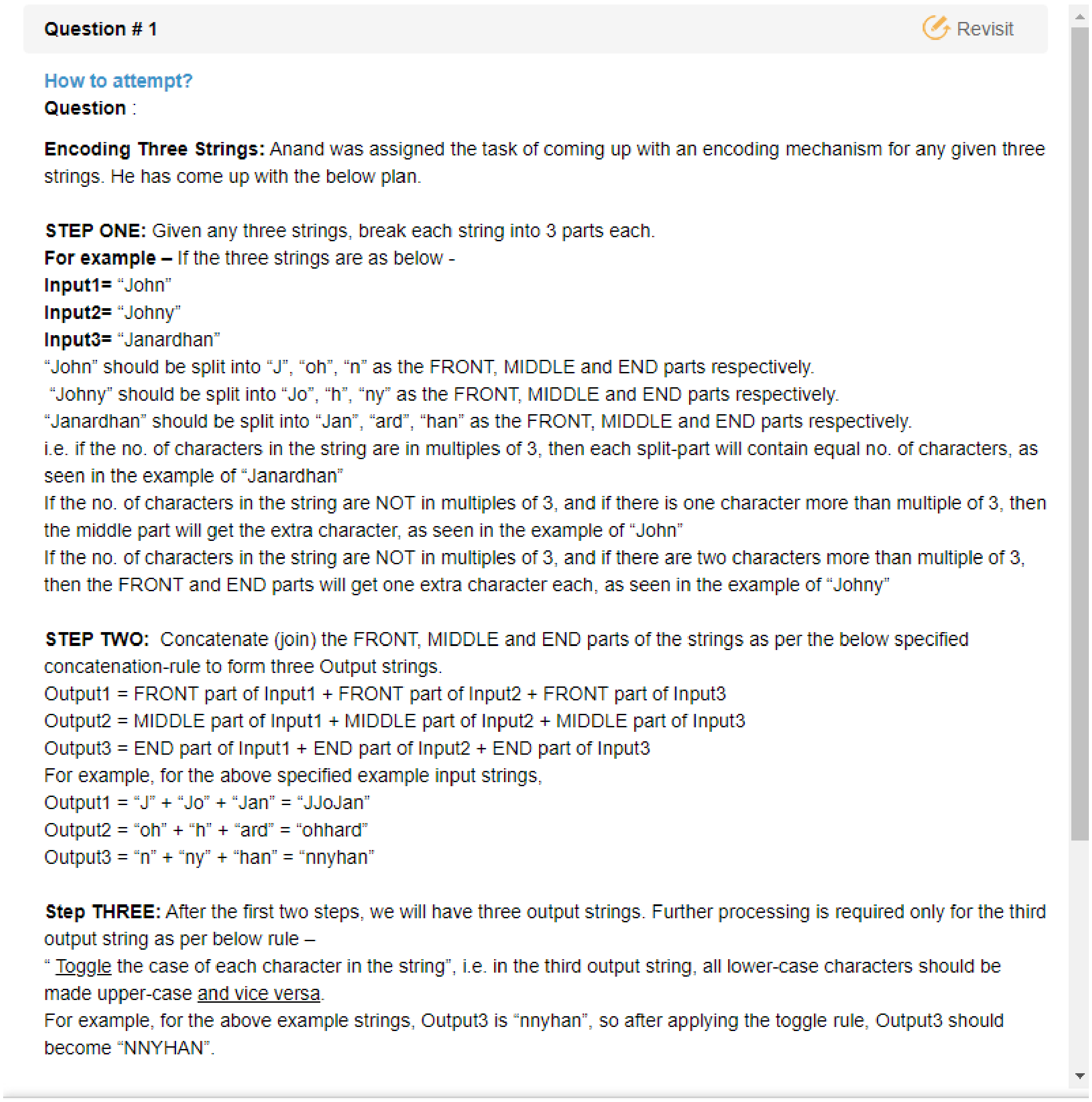
If the given number is 582109, the Sum of Sums of Digits will be calculated as =
= (5 + 8 + 2 + 1 + 0 + 9) + (8 + 2 + 1 + 0 + 9) + (2 + 1 + 0 + 9) + (1 + 0 + 9) + (0 + 9) + (9)
= 25 + 20 + 12 + 10 + 9 + 9 = 85

Alex contacts you to help him write a program for finding the Sum of Sums of Digits for any given number, using the above method.

Help Alex by completing the login in the given function **sumOfSumsOfDigits** which takes as input an integer **input1** representing the given number.
The function is expected to return the "Sum of Sums of Digits" of input1.

Assumptions: For this assignment, let us assume that the given number will always contain more than 1 digit, i.e. the given number will always be >9.

```
import java.io.*;
import  java.util.*;

// Read only region start
class UserMainCode
{

                    public int sumOfSumsOfDigits(int input1){
                     // Read only region end
                     StringBuffer sb=new StringBuffer();
                     sb.append(input1);
                     sb.reverse();
                     String tmp=sb.toString();
                     input1=Integer.parseInt(tmp);
                     String s=input1+"";
                     int length=s.length();
                     int temp=input1;
                     int sum=0;
                     for(int i=0;i<length;i++)
                     {

                                for(int j=i;j<length;j++)
                                {
                                   sum+=temp%10;
                                          temp=temp/10;
                                          System.out.println("sum="+sum);
                                          System.out.println("temp="+temp);
                                }
                                temp=input1/10;
                                input1=input1/10;
                                System.out.println("num="+temp);
                     }
                     return sum;
                    }
}
```



Question # 1

Revisit

How to attempt?

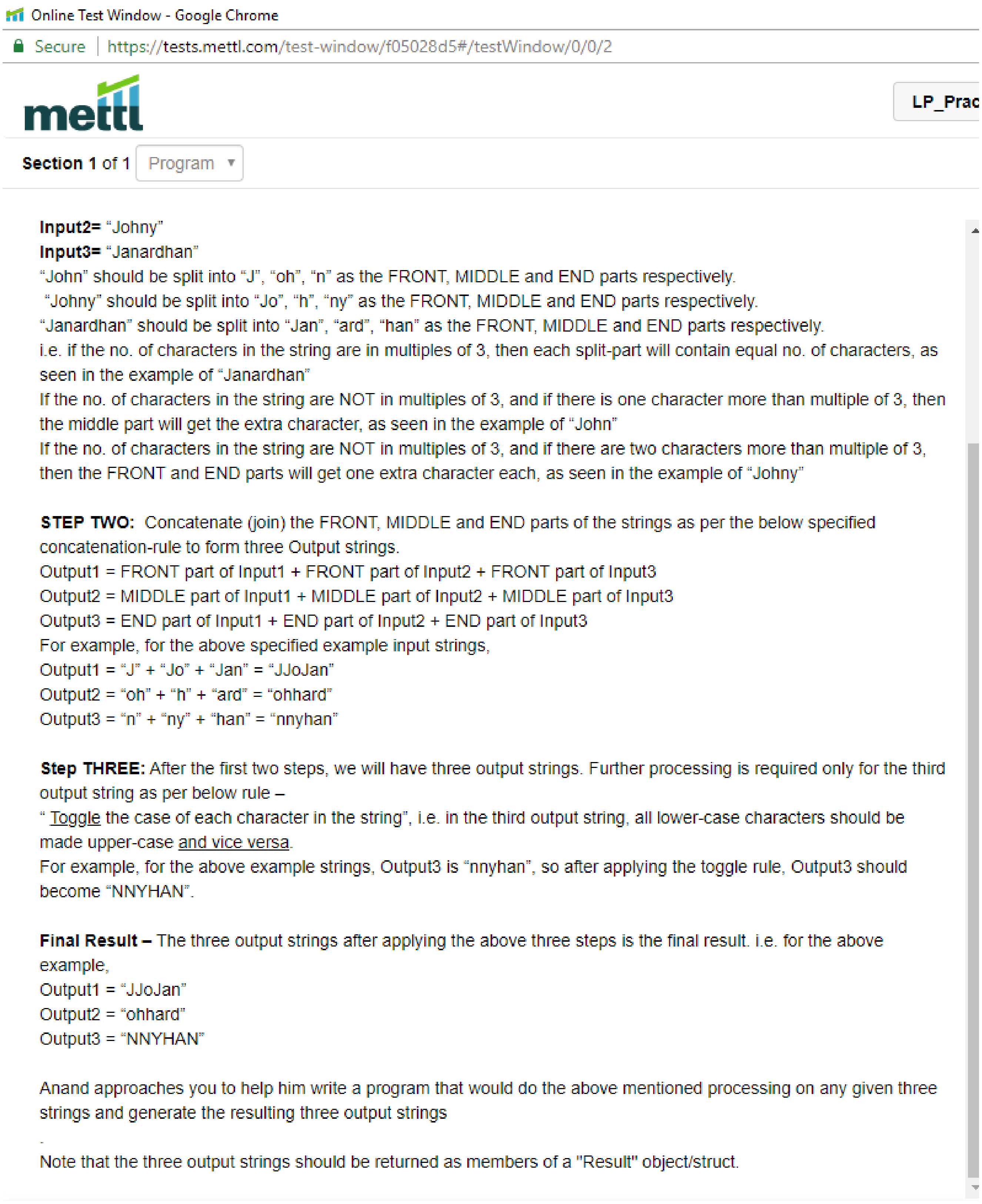
Question :

Encoding Three Strings: Anand was assigned the task of coming up with an encoding mechanism for any given three strings. He has come up with the below plan.

STEP ONE: Given any three strings, break each string into 3 parts each.
For example – If the three strings are as below -
Input1= “John”
Input2= “Johny”
Input3= “Janardhan”
“John” should be split into “J”, “oh”, “n” as the FRONT, MIDDLE and END parts respectively.
“Johny” should be split into “Jo”, “h”, “ny” as the FRONT, MIDDLE and END parts respectively.
“Janardhan” should be split into “Jan”, “ard”, “han” as the FRONT, MIDDLE and END parts respectively.
i.e. if the no. of characters in the string are in multiples of 3, then each split-part will contain equal no. of characters, as seen in the example of “Janardhan”
If the no. of characters in the string are NOT in multiples of 3, and if there is one character more than multiple of 3, then the middle part will get the extra character, as seen in the example of “John”
If the no. of characters in the string are NOT in multiples of 3, and if there are two characters more than multiple of 3, then the FRONT and END parts will get one extra character each, as seen in the example of “Johny”

STEP TWO: Concatenate (join) the FRONT, MIDDLE and END parts of the strings as per the below specified concatenation-rule to form three Output strings.
Output1 = FRONT part of Input1 + FRONT part of Input2 + FRONT part of Input3
Output2 = MIDDLE part of Input1 + MIDDLE part of Input2 + MIDDLE part of Input3
Output3 = END part of Input1 + END part of Input2 + END part of Input3
For example, for the above specified example input strings,
Output1 = “J” + “Jo” + “Jan” = “JJoJan”
Output2 = “oh” + “h” + “ard” = “ohhard”
Output3 = “n” + “ny” + “han” = “nnyhan”

Step THREE: After the first two steps, we will have three output strings. Further processing is required only for the third output string as per below rule –
“ Toggle the case of each character in the string”, i.e. in the third output string, all lower-case characters should be made upper-case and vice versa.
For example, for the above example strings, Output3 is “nnyhan”, so after applying the toggle rule, Output3 should become “NNYHAN”.


Online Test Window - Google Chrome
Secure | https://tests.mettl.com/test-window/f05028d5#/testWindow/0/0/2
mettl
LP_Prac
Section 1 of 1
Program

Input2= "Johny"
Input3= "Janardhan"
"John" should be split into "J", "oh", "n" as the FRONT, MIDDLE and END parts respectively.
"Johny" should be split into "Jo", "h", "ny" as the FRONT, MIDDLE and END parts respectively.
"Janardhan" should be split into "Jan", "ard", "han" as the FRONT, MIDDLE and END parts respectively.
i.e. if the no. of characters in the string are in multiples of 3, then each split-part will contain equal no. of characters, as seen in the example of "Janardhan"
If the no. of characters in the string are NOT in multiples of 3, and if there is one character more than multiple of 3, then the middle part will get the extra character, as seen in the example of "John"
If the no. of characters in the string are NOT in multiples of 3, and if there are two characters more than multiple of 3, then the FRONT and END parts will get one extra character each, as seen in the example of "Johny"

STEP TWO: Concatenate (join) the FRONT, MIDDLE and END parts of the strings as per the below specified concatenation-rule to form three Output strings.
Output1 = FRONT part of Input1 + FRONT part of Input2 + FRONT part of Input3
Output2 = MIDDLE part of Input1 + MIDDLE part of Input2 + MIDDLE part of Input3
Output3 = END part of Input1 + END part of Input2 + END part of Input3
For example, for the above specified example input strings,
Output1 = "J" + "Jo" + "Jan" = "JJoJan"
Output2 = "oh" + "h" + "ard" = "ohhard"
Output3 = "n" + "ny" + "han" = "nnyhan"

Step THREE: After the first two steps, we will have three output strings. Further processing is required only for the third output string as per below rule –
" Toggle the case of each character in the string", i.e. in the third output string, all lower-case characters should be made upper-case and vice versa.
For example, for the above example strings, Output3 is "nnyhan", so after applying the toggle rule, Output3 should become "NNYHAN".

Final Result – The three output strings after applying the above three steps is the final result. i.e. for the above example,
Output1 = "JJoJan"
Output2 = "ohhard"
Output3 = "NNYHAN"

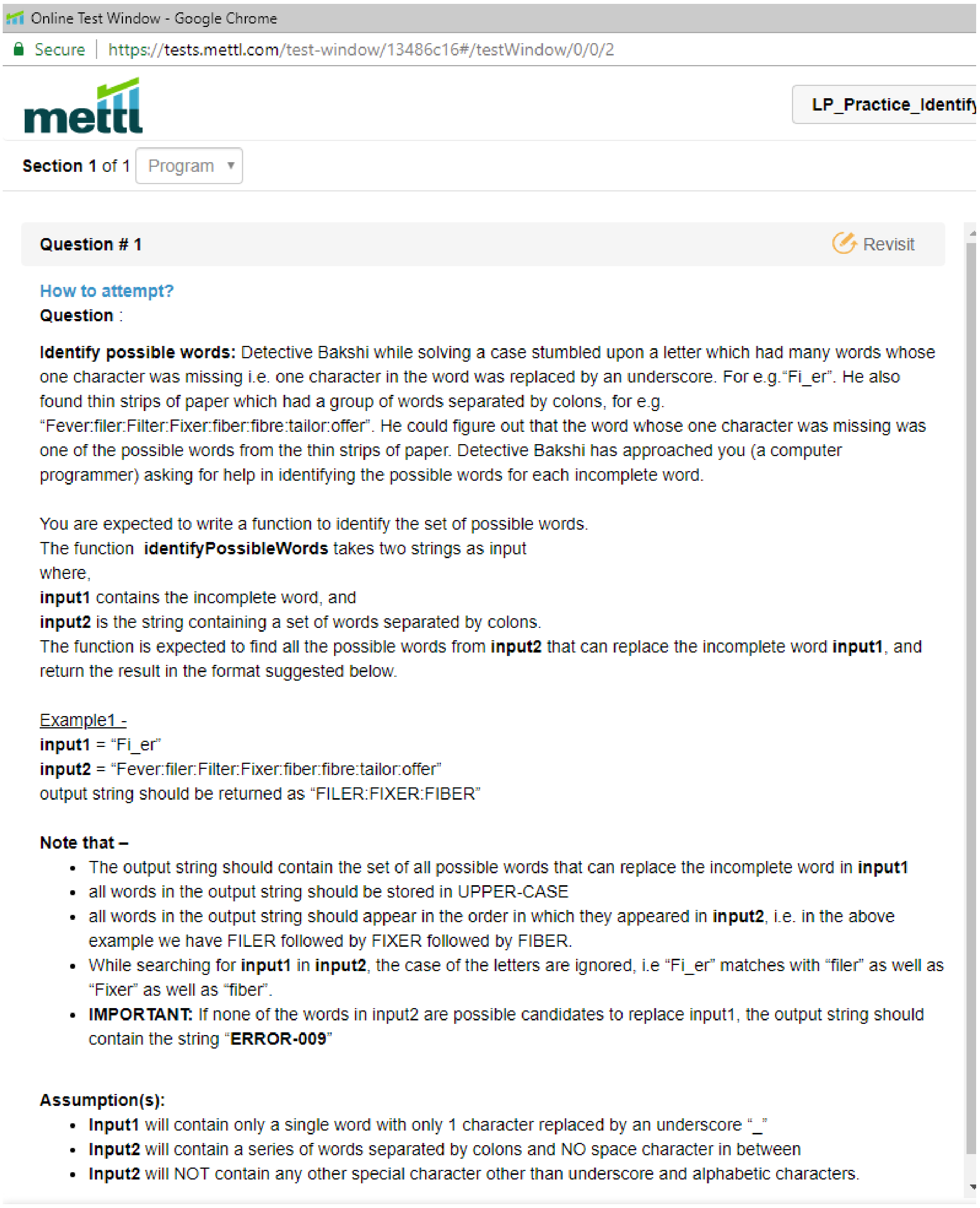
Anand approaches you to help him write a program that would do the above mentioned processing on any given three strings and generate the resulting three output strings

Note that the three output strings should be returned as members of a "Result" object/struct.

Code

```
import java.io.*;
import  java.util.*;
```

```
// Read only region start
class UserMainCode
{

public class Result{
public final String output1;
public final String output2;
public final String output3;

public Result(String out1, String out2, String out3){
output1 = out1;
output2 = out2;
output3 = out3;
}
}
    public Result encodeThreeStrings(String input1,String input2,String input3){
     // Read only region end
       int length=input1.length();
//int length2=input2.length();
//int length3=input3.length();
ArrayList<String> a=new ArrayList<String>();
a.add(input1);
a.add(input2);
a.add(input3);
int quo,rem;
String first,second,third;
String op1="",op2="",op3="";
quo=length/3;
rem=length%3;
System.out.println("Length="+length);

for(String temp:a)
{
    length=temp.length();
System.out.println("Length"+length);
quo=length/3;
    rem=length%3;
if(length%3==0)
{
System.out.println("Entered if");
first=(temp.substring(0,quo));
second=(temp.substring(quo,quo+quo));
third=(temp.substring(quo+quo));
op1+=first;
op2+=second;
op3+=third;
System.out.println(first+" "+second+" "+third);
}
else if(length%3==1)
{
System.out.println("Entered else if1 ");
first=(temp.substring(0,quo));
second=(temp.substring(quo,quo+quo+1));
third=(temp.substring(quo+quo+1));
System.out.println(first+" "+second+" "+third);
op1+=first;
op2+=second;
op3+=third;
}
else if(length%3==2)
{
System.out.println("Entered  else if2");
first=(temp.substring(0,quo+1));
second=(temp.substring(quo+1,quo+quo+1));
third=(temp.substring(quo+quo+1));
System.out.println(first+" "+second+" "+third);
op1+=first;
op2+=second;
op3+=third;
}
System.out.println("Exit");
}
System.out.println(op1);
System.out.println(op2);
System.out.println(op3);
StringBuilder tmp=new StringBuilder(op3);
for (int index = 0; index < tmp.length(); index++) {
  char c = tmp.charAt(index);
  if (Character.isLowerCase(c)) {
        tmp.setCharAt(index, Character.toUpperCase(c));
    } else {
        tmp.setCharAt(index, Character.toLowerCase(c));
    }
}
op3=tmp.toString();
     Result r=new Result(op1,op2,op3);
return r;

    }
}
```


Online Test Window - Google Chrome
Secure | https://tests.mettl.com/test-window/13486c16#/testWindow/0/0/2

LP_Practice_Identify

Section 1 of 1 Program

Question # 1

Revisit

How to attempt?

Question :

Identify possible words: Detective Bakshi while solving a case stumbled upon a letter which had many words whose one character was missing i.e. one character in the word was replaced by an underscore. For e.g."Fi_er". He also found thin strips of paper which had a group of words separated by colons, for e.g. "Fever:filer:Filter:Fixer:fiber:fibre:tailor:offer". He could figure out that the word whose one character was missing was one of the possible words from the thin strips of paper. Detective Bakshi has approached you (a computer programmer) asking for help in identifying the possible words for each incomplete word.

You are expected to write a function to identify the set of possible words.
The function **identifyPossibleWords** takes two strings as input
where,
input1 contains the incomplete word, and
input2 is the string containing a set of words separated by colons.
The function is expected to find all the possible words from **input2** that can replace the incomplete word **input1**, and return the result in the format suggested below.

Example1 -
input1 = "Fi_er"
input2 = "Fever:filer:Filter:Fixer:fiber:fibre:tailor:offer"
output string should be returned as "FILER:FIXER:FIBER"

Note that –

- The output string should contain the set of all possible words that can replace the incomplete word in **input1**
- all words in the output string should be stored in UPPER-CASE
- all words in the output string should appear in the order in which they appeared in **input2**, i.e. in the above example we have FILER followed by FIXER followed by FIBER.
- While searching for **input1** in **input2**, the case of the letters are ignored, i.e "Fi_er" matches with "filer" as well as "Fixer" as well as "fiber".
- **IMPORTANT:** If none of the words in input2 are possible candidates to replace input1, the output string should contain the string "**ERROR-009**"

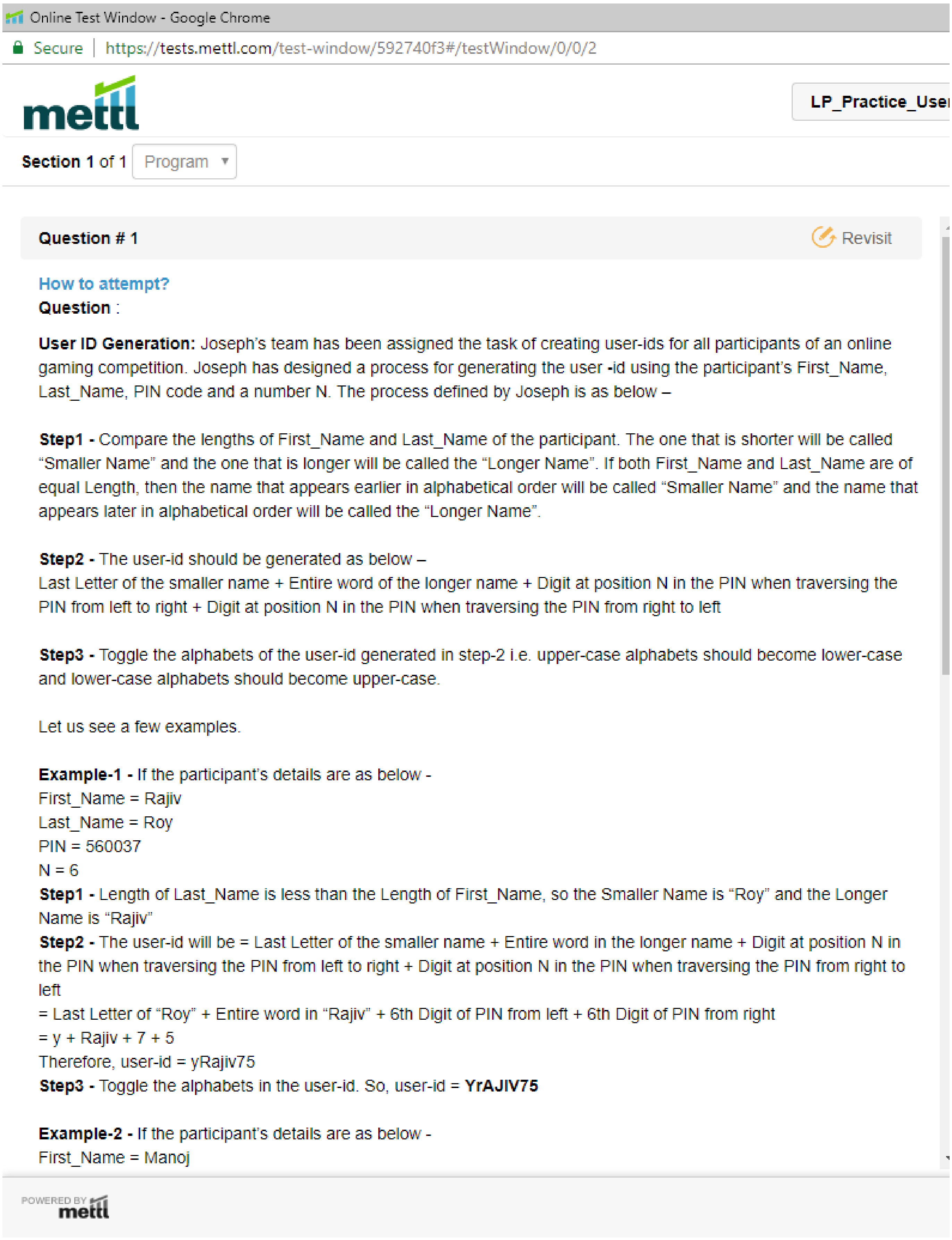
Assumption(s):

- **Input1** will contain only a single word with only 1 character replaced by an underscore "_"
- **Input2** will contain a series of words separated by colons and NO space character in between
- **Input2** will NOT contain any other special character other than underscore and alphabetic characters.

Code

```
import java.io.*;
import  java.util.*;
import java.util.regex.*;

// Read only region start
```

```
class UserMainCode
{

                    public String identifyPossibleWords(String input1,String input2){
                      // Read only region end
                    input1=input1.replaceAll("_",".").toUpperCase();
                    String[] ip2=input2.split(":");
                    String op="";
                    for(String s:ip2)
                    {
                      if(Pattern.matches(input1, s.toUpperCase())==true)
                                op+=s.toUpperCase()+" ";
                    }
   op=op.trim();
                      if(op.length()==0)
                                return "ERROR-009";
                      else
                      {
                        op=op.replaceAll(" ",":");
                      return op;
                      }
                    }
}
```


Online Test Window - Google Chrome

Secure | https://tests.mettl.com/test-window/592740f3#/testWindow/0/0/2

LP_Practice_User

Section 1 of 1 Program

Question # 1 Revisit

How to attempt?

Question :

User ID Generation: Joseph's team has been assigned the task of creating user-ids for all participants of an online gaming competition. Joseph has designed a process for generating the user -id using the participant's First_Name, Last_Name, PIN code and a number N. The process defined by Joseph is as below –

Step1 - Compare the lengths of First_Name and Last_Name of the participant. The one that is shorter will be called "Smaller Name" and the one that is longer will be called the "Longer Name". If both First_Name and Last_Name are of equal Length, then the name that appears earlier in alphabetical order will be called "Smaller Name" and the name that appears later in alphabetical order will be called the "Longer Name".

Step2 - The user-id should be generated as below –
Last Letter of the smaller name + Entire word of the longer name + Digit at position N in the PIN when traversing the PIN from left to right + Digit at position N in the PIN when traversing the PIN from right to left

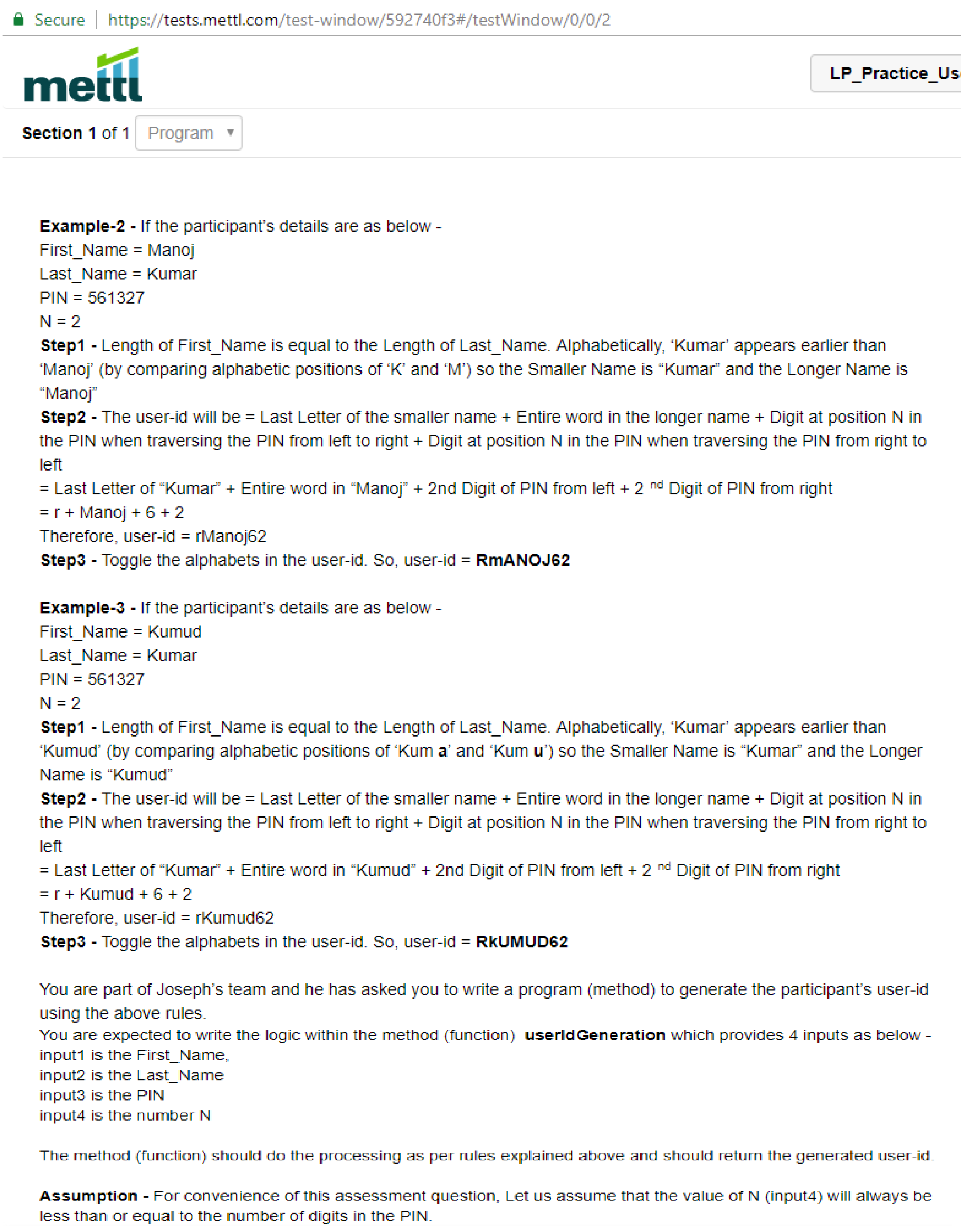
Step3 - Toggle the alphabets of the user-id generated in step-2 i.e. upper-case alphabets should become lower-case and lower-case alphabets should become upper-case.

Let us see a few examples.

Example-1 - If the participant's details are as below -
First_Name = Rajiv
Last_Name = Roy
PIN = 560037
N = 6
Step1 - Length of Last_Name is less than the Length of First_Name, so the Smaller Name is "Roy" and the Longer Name is "Rajiv"
Step2 - The user-id will be = Last Letter of the smaller name + Entire word in the longer name + Digit at position N in the PIN when traversing the PIN from left to right + Digit at position N in the PIN when traversing the PIN from right to left
= Last Letter of "Roy" + Entire word in "Rajiv" + 6th Digit of PIN from left + 6th Digit of PIN from right
= y + Rajiv + 7 + 5
Therefore, user-id = yRajiv75
Step3 - Toggle the alphabets in the user-id. So, user-id = **YrAJIV75**

Example-2 - If the participant's details are as below -
First_Name = Manoj

POWERED BY mettl


Secure | https://tests.mettl.com/test-window/592740f3#/testWindow/0/0/2
LP_Practice_Us
Section 1 of 1 Program

Example-2 - If the participant's details are as below -
First_Name = Manoj
Last_Name = Kumar
PIN = 561327
N = 2
Step1 - Length of First_Name is equal to the Length of Last_Name. Alphabetically, 'Kumar' appears earlier than 'Manoj' (by comparing alphabetic positions of 'K' and 'M') so the Smaller Name is "Kumar" and the Longer Name is "Manoj"
Step2 - The user-id will be = Last Letter of the smaller name + Entire word in the longer name + Digit at position N in the PIN when traversing the PIN from left to right + Digit at position N in the PIN when traversing the PIN from right to left
= Last Letter of "Kumar" + Entire word in "Manoj" + 2nd Digit of PIN from left + 2 nd Digit of PIN from right
= r + Manoj + 6 + 2
Therefore, user-id = rManoj62
Step3 - Toggle the alphabets in the user-id. So, user-id = **RmANOJ62**

Example-3 - If the participant's details are as below -
First_Name = Kumud
Last_Name = Kumar
PIN = 561327
N = 2
Step1 - Length of First_Name is equal to the Length of Last_Name. Alphabetically, 'Kumar' appears earlier than 'Kumud' (by comparing alphabetic positions of 'Kum **a**' and 'Kum **u**') so the Smaller Name is "Kumar" and the Longer Name is "Kumud"
Step2 - The user-id will be = Last Letter of the smaller name + Entire word in the longer name + Digit at position N in the PIN when traversing the PIN from left to right + Digit at position N in the PIN when traversing the PIN from right to left
= Last Letter of "Kumar" + Entire word in "Kumud" + 2nd Digit of PIN from left + 2 nd Digit of PIN from right
= r + Kumud + 6 + 2
Therefore, user-id = rKumud62
Step3 - Toggle the alphabets in the user-id. So, user-id = **RkUMUD62**

You are part of Joseph's team and he has asked you to write a program (method) to generate the participant's user-id using the above rules.
You are expected to write the logic within the method (function) **userIdGeneration** which provides 4 inputs as below -
input1 is the First_Name,
input2 is the Last_Name
input3 is the PIN
input4 is the number N

The method (function) should do the processing as per rules explained above and should return the generated user-id.

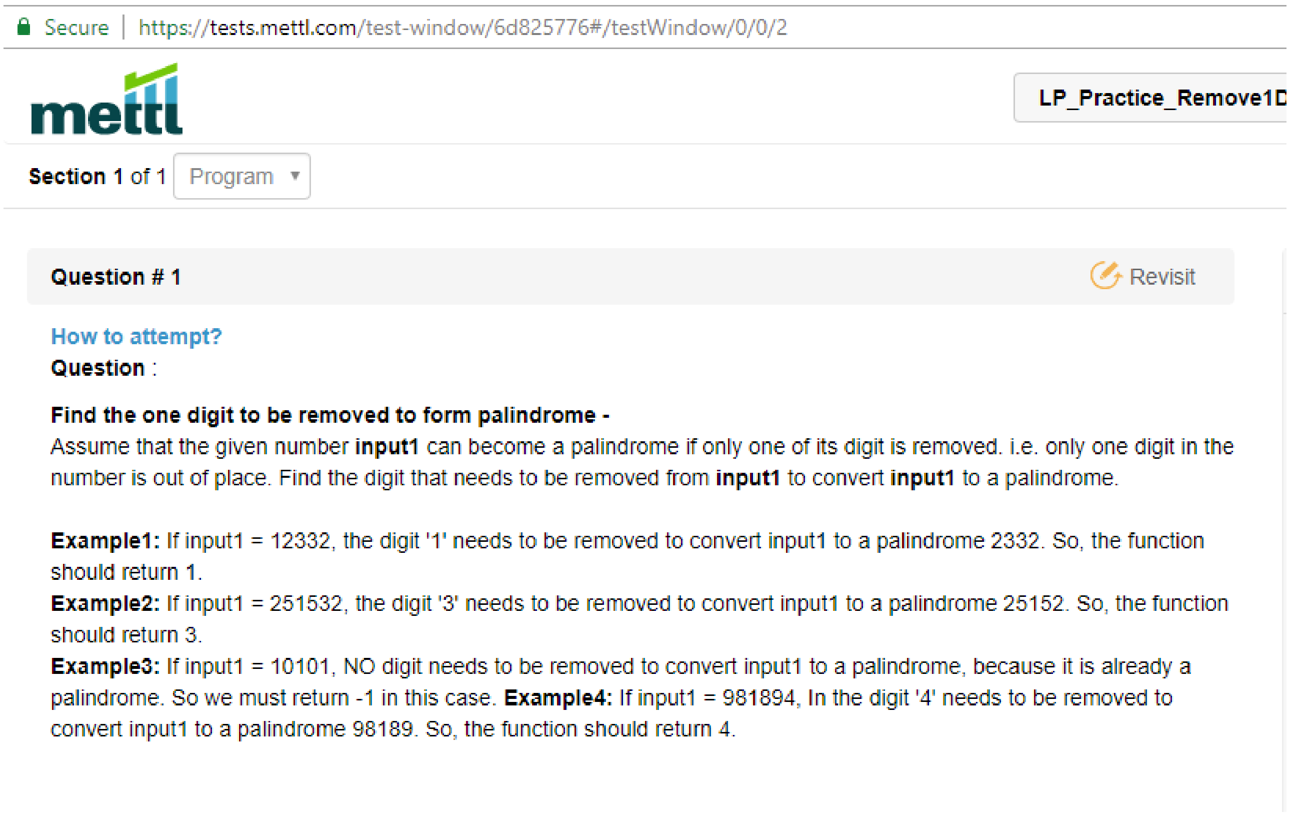
Assumption - For convenience of this assessment question, Let us assume that the value of N (input4) will always be less than or equal to the number of digits in the PIN.

USER ID GENERATION

```
import java.io.*;
import  java.util.*;
```

```
// Read only region start
class UserMainCode
{

            public String userIdGeneration(String input1,String input2,int input3,int input4){
             // Read only region end
             String last="",longer="",pin="",temp=input3+"";
             if(input1.length()<input2.length())
             {
                    System.out.println("first string is smaller");
                    last=input1.charAt(input1.length()-1)+"";
                    longer=input2;
                    pin=temp.charAt((input4)-1)+""+temp.charAt(temp.length()-input4);
                    System.out.println(pin);

             }
             else if(input1.length()>input2.length())
             {
                    System.out.println("second string is smaller");
                    last=input2.charAt(input2.length()-1)+"";
                    longer=input1;
                    pin=temp.charAt((input4)-1)+""+temp.charAt(temp.length()-input4);
                    System.out.println(pin);
             }
             else if(input1.length()==input2.length())
             {
                    if(input1.compareTo(input2)<0){
                    System.out.println("first string is smaller but both are equal");
                    last=input1.charAt(input1.length()-1)+"";
                    longer=input2;
                    }
                    else if(input1.compareTo(input2)>0){
                    System.out.println("second string is smaller but both are equal");
                    last=input2.charAt(input2.length()-1)+"";
                    longer=input1;
                    }
                    else
                    {
                    System.out.println("first string is smaller and both are equal");
                    last=input1.charAt(input1.length()-1)+"";
                    longer=input2;
                    }

                    pin=temp.charAt((input4)-1)+""+temp.charAt(temp.length()-input4);
                    System.out.println(pin);
             }
             StringBuffer sb=new StringBuffer(longer);
```

```
                    for(int i=0;i<sb.length();i++)
                    {
                            if(Character.isLowerCase(sb.charAt(i)))
                            {
                                    sb.setCharAt(i,Character.toUpperCase(sb.charAt(i)));
                                    System.out.println(sb.charAt(i));
                            }
                            else
                            {
                                    sb.setCharAt(i,Character.toLowerCase(sb.charAt(i)));
                                    System.out.println(sb.charAt(i));
                            }

                    }
            longer=sb.toString();
            System.out.println(last+""+longer+""+pin);
            return (last.toUpperCase()+""+longer+""+pin);
        }
}
```



Secure | https://tests.mettl.com/test-window/6d825776#/testWindow/0/0/2

LP_Practice_Remove1D

Section 1 of 1 Program

Question # 1

How to attempt?

Question :

Find the one digit to be removed to form palindrome -

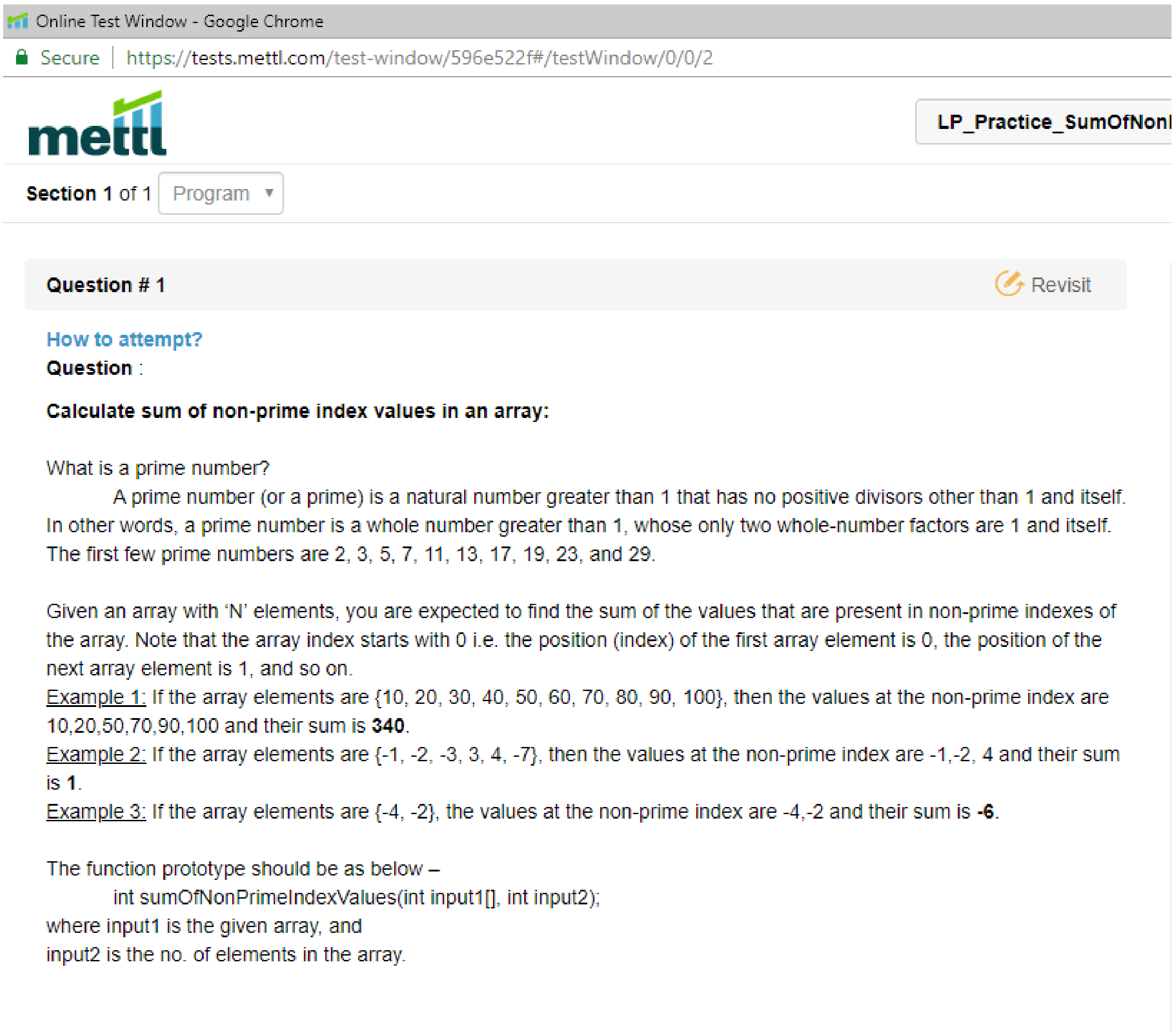
Assume that the given number **input1** can become a palindrome if only one of its digit is removed. i.e. only one digit in the number is out of place. Find the digit that needs to be removed from **input1** to convert **input1** to a palindrome.

Example1: If input1 = 12332, the digit '1' needs to be removed to convert input1 to a palindrome 2332. So, the function should return 1.
Example2: If input1 = 251532, the digit '3' needs to be removed to convert input1 to a palindrome 25152. So, the function should return 3.
Example3: If input1 = 10101, NO digit needs to be removed to convert input1 to a palindrome, because it is already a palindrome. So we must return -1 in this case. **Example4:** If input1 = 981894, In the digit '4' needs to be removed to convert input1 to a palindrome 98189. So, the function should return 4.

CODE

```
import java.io.*;
import  java.util.*;

// Read only region start
```

```
class UserMainCode
{

            public int digitRemove_Palin(int input1){
             // Read only region end
             StringBuffer sb=new StringBuffer(input1+"");
             if((input1+"").equals(sb.reverse().toString()))
                    return -1;

             String ip=input1+"";
             char[] cc=ip.toCharArray();
             ArrayList<Character> c= new ArrayList<Character>();
             for(char temp:cc)
                    c.add(temp);
             String op="";
             for(char temp:cc){
              System.out.println("frquency of"+temp+"is"+Collections.frequency(c,temp));
                    if(Collections.frequency(c,temp)==1)
                    {
                           op+=temp+"";
                    }
             }
             System.out.println(op);
             if(op.length()>1)
                    return Integer.parseInt(op.charAt(op.length()-1)+"");
             else
                    return Integer.parseInt(op.charAt(0)+"");
            }
}
```



Online Test Window - Google Chrome

Secure | https://tests.mettl.com/test-window/596e522f#/testWindow/0/0/2

Question # 1

How to attempt?

Question :

Calculate sum of non-prime index values in an array:

What is a prime number?

A prime number (or a prime) is a natural number greater than 1 that has no positive divisors other than 1 and itself. In other words, a prime number is a whole number greater than 1, whose only two whole-number factors are 1 and itself. The first few prime numbers are 2, 3, 5, 7, 11, 13, 17, 19, 23, and 29.

Given an array with 'N' elements, you are expected to find the sum of the values that are present in non-prime indexes of the array. Note that the array index starts with 0 i.e. the position (index) of the first array element is 0, the position of the next array element is 1, and so on.

Example 1: If the array elements are {10, 20, 30, 40, 50, 60, 70, 80, 90, 100}, then the values at the non-prime index are 10,20,50,70,90,100 and their sum is **340**.

Example 2: If the array elements are {-1, -2, -3, 3, 4, -7}, then the values at the non-prime index are -1,-2, 4 and their sum is **1**.

Example 3: If the array elements are {-4, -2}, the values at the non-prime index are -4,-2 and their sum is **-6**.

The function prototype should be as below –

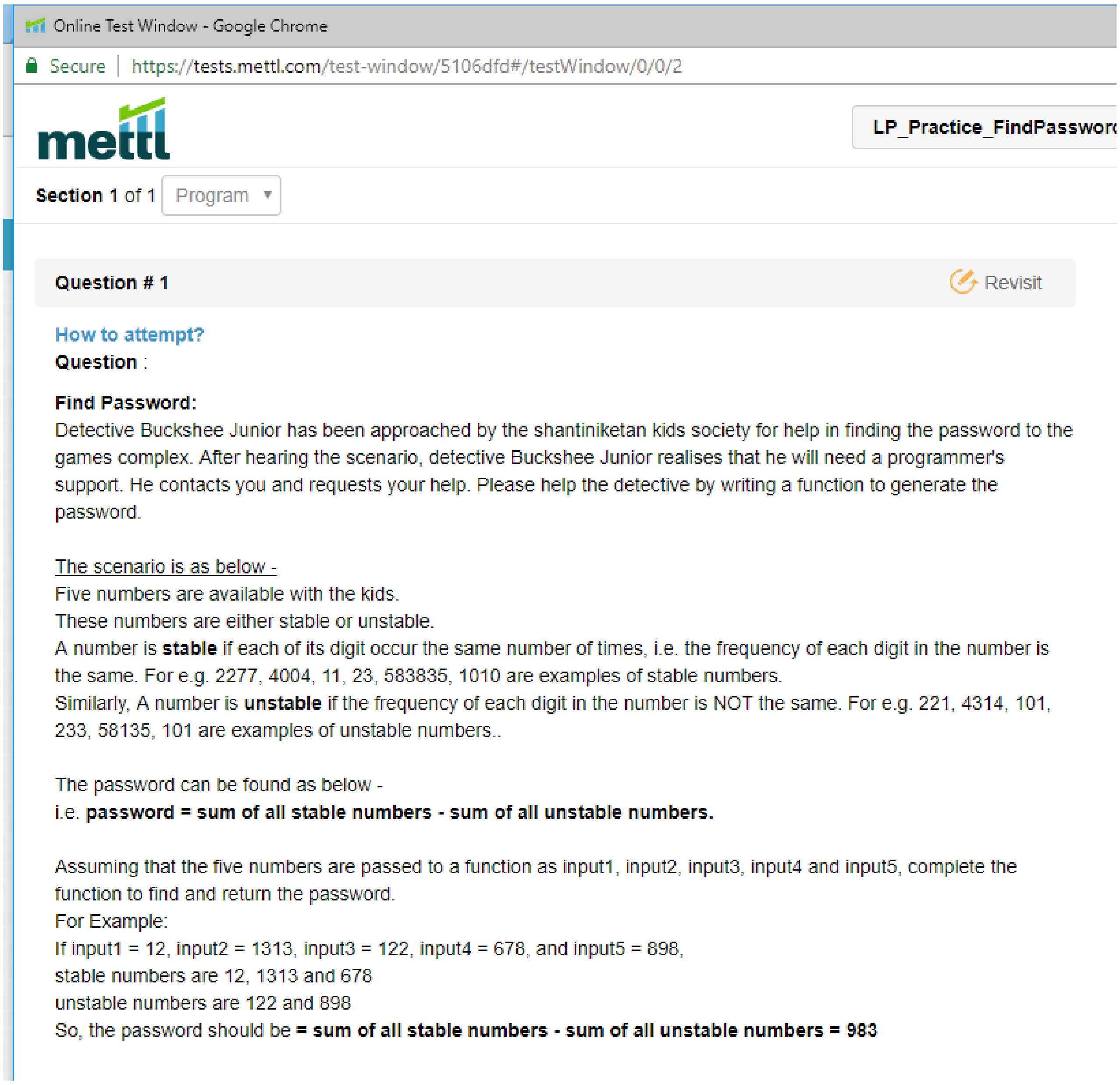
int sumOfNonPrimeIndexValues(int input1[], int input2);

where input1 is the given array, and

input2 is the no. of elements in the array.

```
import java.io.*;
import  java.util.*;

// Read only region start
class UserMainCode
{

                        public int sumOfNonPrimeIndexValues(int[] input1,int input2){
                          // Read only region end
                          int sum=0;
                          for(int i=0;i<input1.length;i++)
                          {
                                      if(i==0||i==1)
                                      {
                                                  System.out.println(input1[i]);
                                                  sum+=input1[i];
                                      }
```

```
                else
                {
                        if(isprime(i)==1)
                        {
                                System.out.println(input1[i]);
                                sum+=input1[i];
                        }
                }
         }
         System.out.println(sum);
         return sum;
        }
        public int isprime(int n)
        {
         for(int i=2;i<n;i++)
         {
                if(n%i==0)
                        return 1;
         }
         return 0;
        }
}
```


Online Test Window - Google Chrome

Secure | https://tests.mettl.com/test-window/5106dfd#/testWindow/0/0/2

LP_Practice_FindPasswor

Section 1 of 1 Program

Question # 1 Revisit

How to attempt?

Question :

Find Password:

Detective Buckshee Junior has been approached by the shantiniketan kids society for help in finding the password to the games complex. After hearing the scenario, detective Buckshee Junior realises that he will need a programmer's support. He contacts you and requests your help. Please help the detective by writing a function to generate the password.

The scenario is as below -

Five numbers are available with the kids.

These numbers are either stable or unstable.

A number is **stable** if each of its digit occur the same number of times, i.e. the frequency of each digit in the number is the same. For e.g. 2277, 4004, 11, 23, 583835, 1010 are examples of stable numbers.

Similarly, A number is **unstable** if the frequency of each digit in the number is NOT the same. For e.g. 221, 4314, 101, 233, 58135, 101 are examples of unstable numbers..

The password can be found as below -

i.e. **password = sum of all stable numbers - sum of all unstable numbers.**

Assuming that the five numbers are passed to a function as input1, input2, input3, input4 and input5, complete the function to find and return the password.

For Example:

If input1 = 12, input2 = 1313, input3 = 122, input4 = 678, and input5 = 898,

stable numbers are 12, 1313 and 678

unstable numbers are 122 and 898

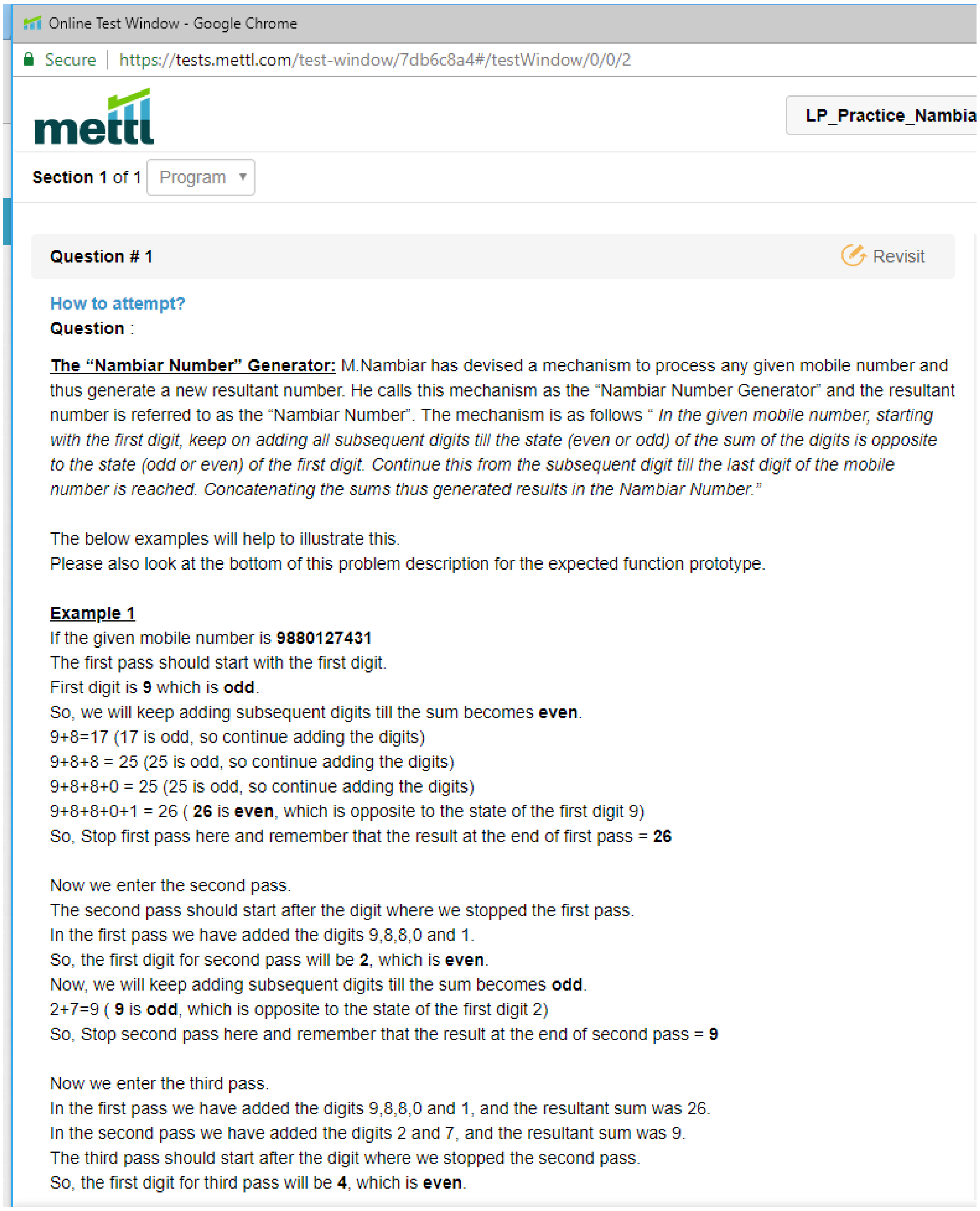
So, the password should be **= sum of all stable numbers - sum of all unstable numbers = 983**

Code

```
import java.io.*;
import  java.util.*;

// Read only region start
class UserMainCode
{

            public int findPassword(int input1,int input2,int input3,int input4,int input5){
              ArrayList<Integer> ar=new ArrayList<Integer>();
              ArrayList<Integer> tar=new ArrayList<Integer>();
              int stable=0,unstable=0,flag=0;
              int freq=0;
              ar.add(input1);ar.add(input2);ar.add(input3);ar.add(input4);ar.add(input5);
              for(int temp:ar)
```

```
                    {
                            flag=0;
                            tar.clear();
                            int temp3=temp;
                            while(temp3>0)
                            {
                                    tar.add(temp3%10);
                                    temp3=temp3/10;
                            }
                            freq=Collections.frequency(tar,tar.get(0));

                            for(int temp2:tar)
                                    if(Collections.frequency(tar,temp2)!=freq)
                                            flag=1;

                            if(flag==1)
                                    unstable+=temp;
                            else
                                    stable+=temp;
                    }
                    return stable-unstable;
                   }
}
```

Nambiar Number


Online Test Window - Google Chrome
Secure | https://tests.mettl.com/test-window/7db6c8a4#/testWindow/0/0/2
mettl
LP_Practice_Nambia
Section 1 of 1
Program
Revisit

Question # 1

How to attempt?

Question :

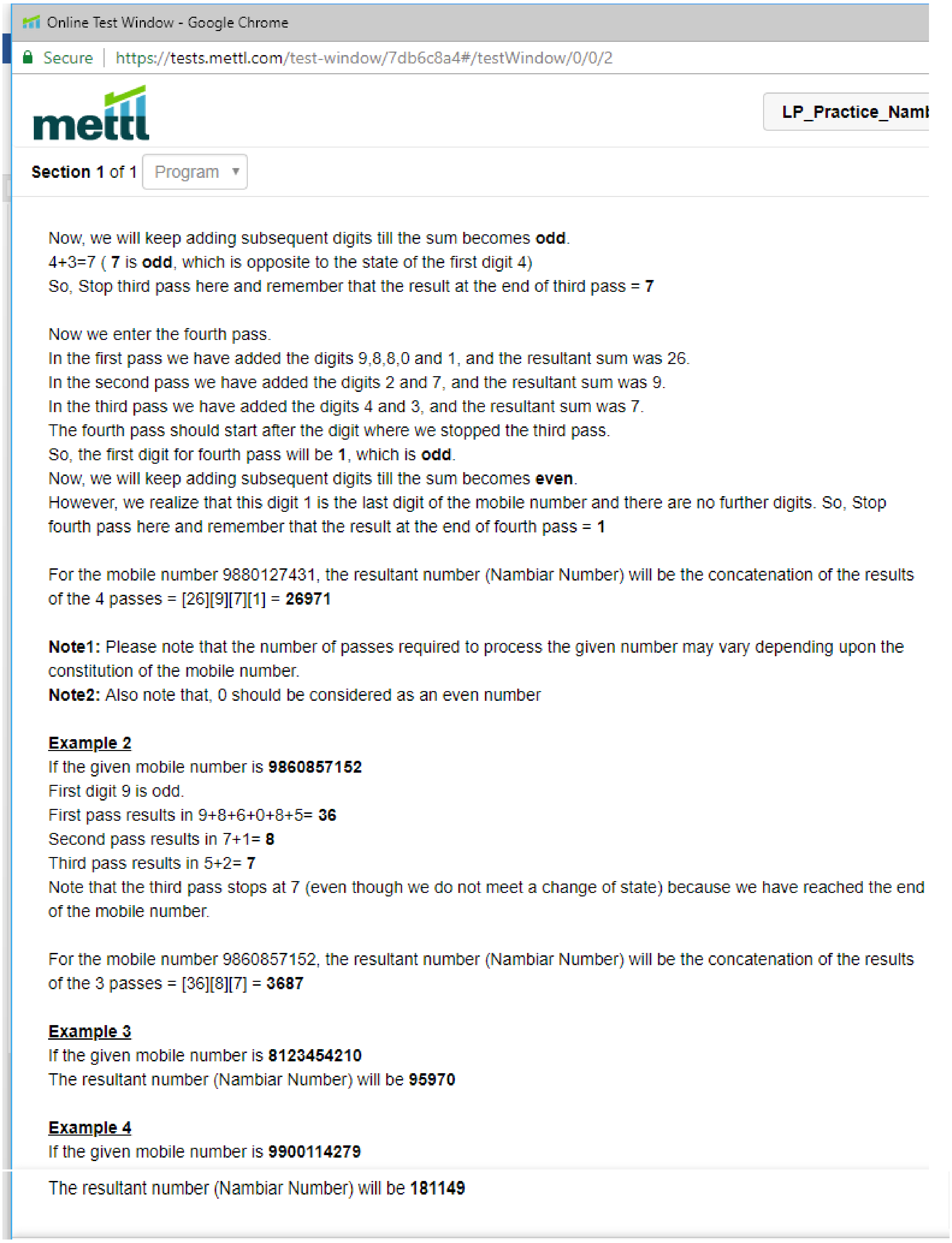
The "Nambiar Number" Generator: M.Nambiar has devised a mechanism to process any given mobile number and thus generate a new resultant number. He calls this mechanism as the "Nambiar Number Generator" and the resultant number is referred to as the "Nambiar Number". The mechanism is as follows " *In the given mobile number, starting with the first digit, keep on adding all subsequent digits till the state (even or odd) of the sum of the digits is opposite to the state (odd or even) of the first digit. Continue this from the subsequent digit till the last digit of the mobile number is reached. Concatenating the sums thus generated results in the Nambiar Number."*

The below examples will help to illustrate this.
Please also look at the bottom of this problem description for the expected function prototype.

Example 1
If the given mobile number is **9880127431**
The first pass should start with the first digit.
First digit is **9** which is **odd**.
So, we will keep adding subsequent digits till the sum becomes **even**.
9+8=17 (17 is odd, so continue adding the digits)
9+8+8 = 25 (25 is odd, so continue adding the digits)
9+8+8+0 = 25 (25 is odd, so continue adding the digits)
9+8+8+0+1 = 26 (**26** is **even**, which is opposite to the state of the first digit 9)
So, Stop first pass here and remember that the result at the end of first pass = **26**

Now we enter the second pass.
The second pass should start after the digit where we stopped the first pass.
In the first pass we have added the digits 9,8,8,0 and 1.
So, the first digit for second pass will be **2**, which is **even**.
Now, we will keep adding subsequent digits till the sum becomes **odd**.
2+7=9 (**9** is **odd**, which is opposite to the state of the first digit 2)
So, Stop second pass here and remember that the result at the end of second pass = **9**

Now we enter the third pass.
In the first pass we have added the digits 9,8,8,0 and 1, and the resultant sum was 26.
In the second pass we have added the digits 2 and 7, and the resultant sum was 9.
The third pass should start after the digit where we stopped the second pass.
So, the first digit for third pass will be **4**, which is **even**.


Online Test Window - Google Chrome
Secure
https://tests.mettl.com/test-window/7db6c8a4#/testWindow/0/0/2
mettl
LP_Practice_Namb
Section 1 of 1
Program

Now, we will keep adding subsequent digits till the sum becomes **odd**.
4+3=7 (**7** is **odd**, which is opposite to the state of the first digit 4)
So, Stop third pass here and remember that the result at the end of third pass = **7**

Now we enter the fourth pass.
In the first pass we have added the digits 9,8,8,0 and 1, and the resultant sum was 26.
In the second pass we have added the digits 2 and 7, and the resultant sum was 9.
In the third pass we have added the digits 4 and 3, and the resultant sum was 7.
The fourth pass should start after the digit where we stopped the third pass.
So, the first digit for fourth pass will be **1**, which is **odd**.
Now, we will keep adding subsequent digits till the sum becomes **even**.
However, we realize that this digit 1 is the last digit of the mobile number and there are no further digits. So, Stop fourth pass here and remember that the result at the end of fourth pass = **1**

For the mobile number 9880127431, the resultant number (Nambiar Number) will be the concatenation of the results of the 4 passes = [26][9][7][1] = **26971**

Note1: Please note that the number of passes required to process the given number may vary depending upon the constitution of the mobile number.
Note2: Also note that, 0 should be considered as an even number

Example 2
If the given mobile number is **9860857152**
First digit 9 is odd.
First pass results in 9+8+6+0+8+5= **36**
Second pass results in 7+1= **8**
Third pass results in 5+2= **7**
Note that the third pass stops at 7 (even though we do not meet a change of state) because we have reached the end of the mobile number.

For the mobile number 9860857152, the resultant number (Nambiar Number) will be the concatenation of the results of the 3 passes = [36][8][7] = **3687**

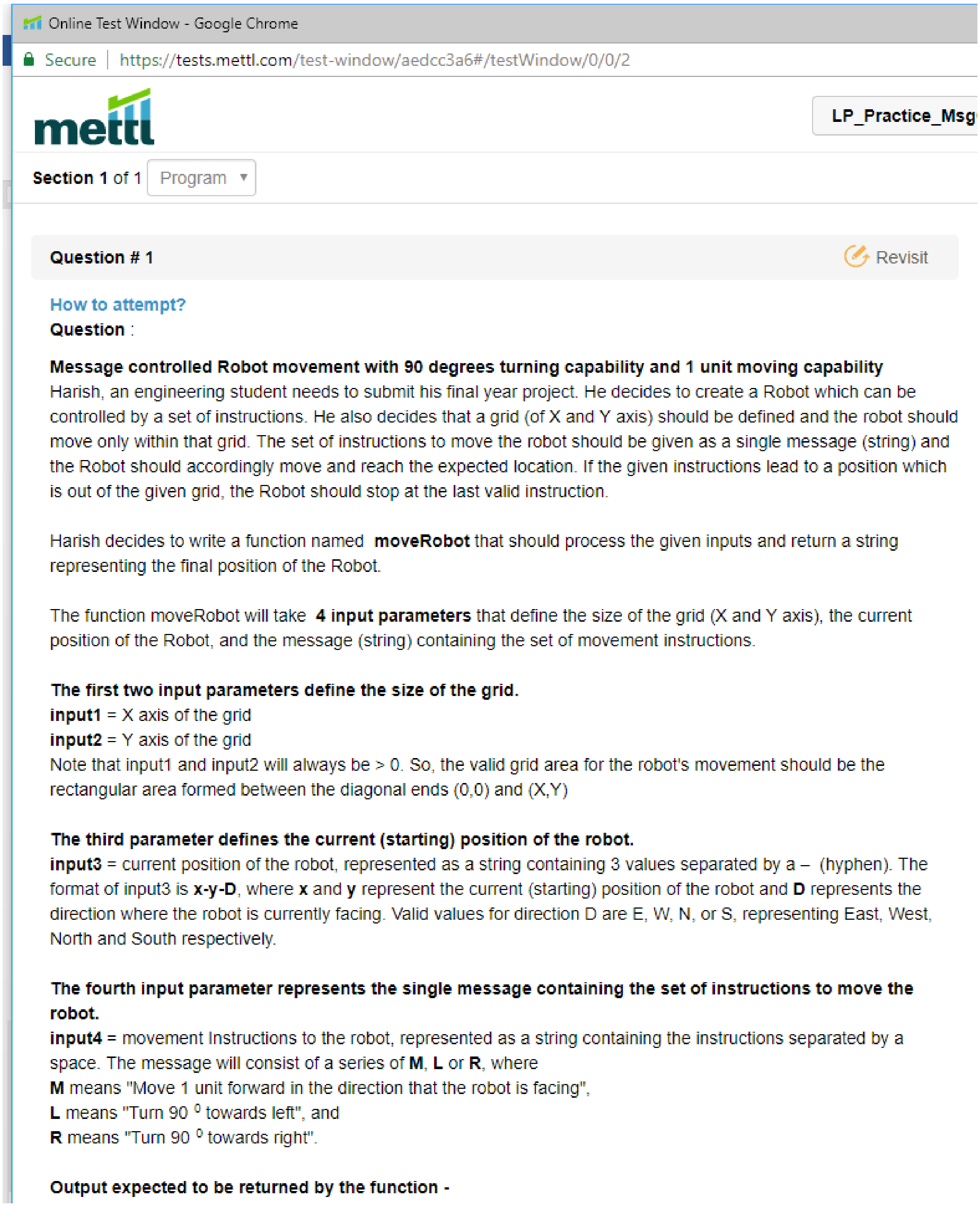
Example 3
If the given mobile number is **8123454210**
The resultant number (Nambiar Number) will be **95970**

Example 4
If the given mobile number is **9900114279**
The resultant number (Nambiar Number) will be **181149**

Code

```
import java.io.*;
import  java.util.*;

// Read only region start
class UserMainCode
{

                        public int nnGenerator(String input1){
                          // Read only region end
                          int len=input1.length();
                        int sum=0;
                        int i,j;
                        int f=0;
                        String res="";
                        int []a=new int[len];
                        for(int k=len-1;k>=0;k--)
                        {
                        a[k]=input1.charAt(k)-48;

                        }
                        for(i=0;i<a.length;)
                        {
                          if(a[i]%2==0)
                                f=0;
                          else
                                f=1;
                          sum=a[i++];
                          if(f==0)
                          {
                          while(i<a.length&&sum%2==0)
                          {
                                sum+=a[i++];
                          }
                        }
                          else
                          {
                                while(i<a.length&&sum%2!=0)
                                {
                                        sum+=a[i++];
                                }
                          }
                          res+=sum;
                        }
                          return (Integer.parseInt(res));

                        }
}
```

Robot



Online Test Window - Google Chrome
Secure | https://tests.mettl.com/test-window/aedcc3a6#/testWindow/0/0/2
mettl
LP_Practice_Msg
Section 1 of 1 Program
Question # 1
Revisit
How to attempt?

Question :

Message controlled Robot movement with 90 degrees turning capability and 1 unit moving capability
Harish, an engineering student needs to submit his final year project. He decides to create a Robot which can be controlled by a set of instructions. He also decides that a grid (of X and Y axis) should be defined and the robot should move only within that grid. The set of instructions to move the robot should be given as a single message (string) and the Robot should accordingly move and reach the expected location. If the given instructions lead to a position which is out of the given grid, the Robot should stop at the last valid instruction.

Harish decides to write a function named **moveRobot** that should process the given inputs and return a string representing the final position of the Robot.

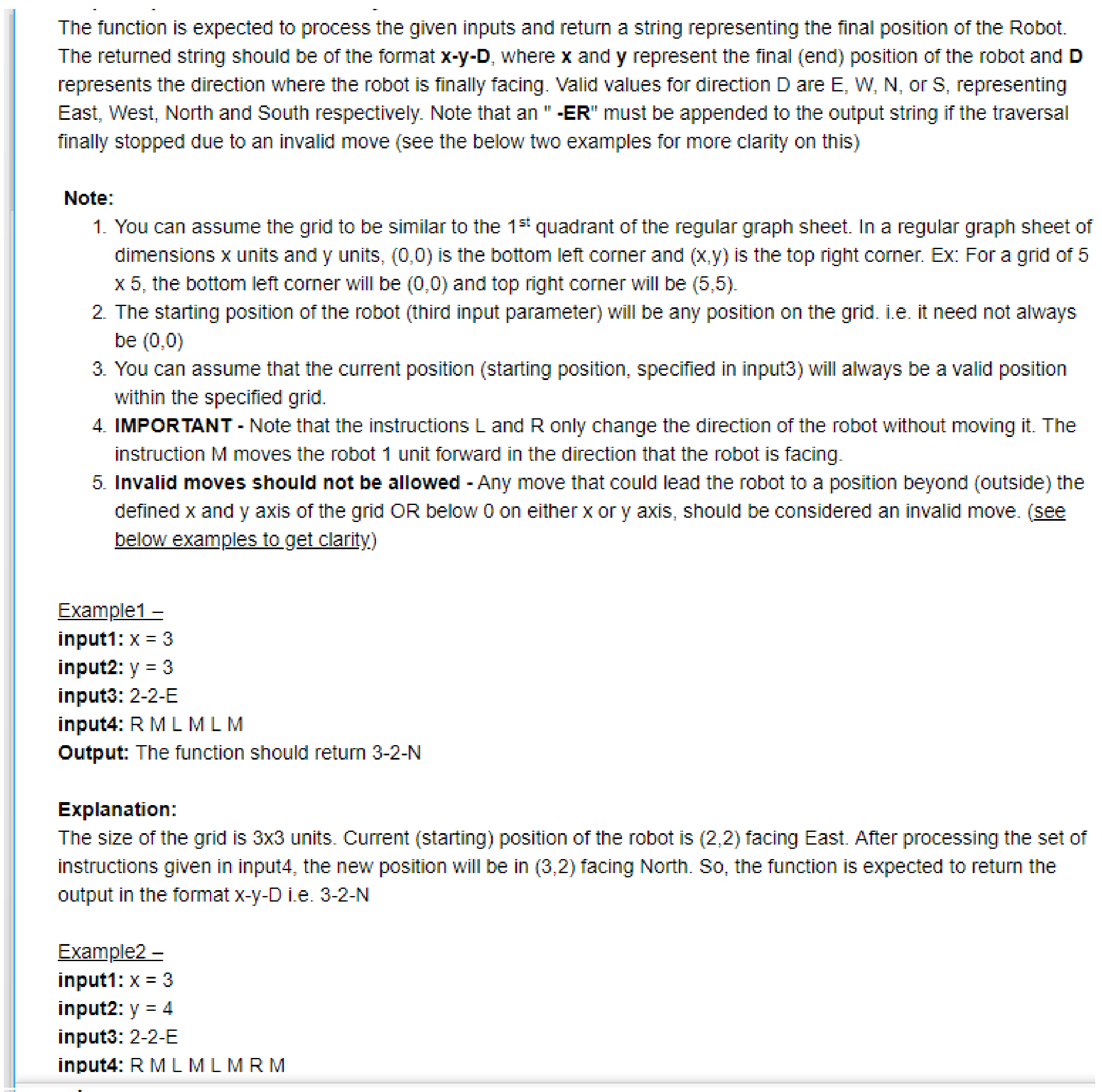
The function moveRobot will take **4 input parameters** that define the size of the grid (X and Y axis), the current position of the Robot, and the message (string) containing the set of movement instructions.

The first two input parameters define the size of the grid.
input1 = X axis of the grid
input2 = Y axis of the grid
Note that input1 and input2 will always be > 0. So, the valid grid area for the robot's movement should be the rectangular area formed between the diagonal ends (0,0) and (X,Y)

The third parameter defines the current (starting) position of the robot.
input3 = current position of the robot, represented as a string containing 3 values separated by a – (hyphen). The format of input3 is **x-y-D**, where **x** and **y** represent the current (starting) position of the robot and **D** represents the direction where the robot is currently facing. Valid values for direction D are E, W, N, or S, representing East, West, North and South respectively.

The fourth input parameter represents the single message containing the set of instructions to move the robot.
input4 = movement Instructions to the robot, represented as a string containing the instructions separated by a space. The message will consist of a series of **M**, **L** or **R**, where
M means "Move 1 unit forward in the direction that the robot is facing",
L means "Turn 90 ^{0} towards left", and
R means "Turn 90 ^{0} towards right".

Output expected to be returned by the function -


The function is expected to process the given inputs and return a string representing the final position of the Robot. The returned string should be of the format **x-y-D**, where **x** and **y** represent the final (end) position of the robot and **D** represents the direction where the robot is finally facing. Valid values for direction D are E, W, N, or S, representing East, West, North and South respectively. Note that an " **-ER**" must be appended to the output string if the traversal finally stopped due to an invalid move (see the below two examples for more clarity on this)

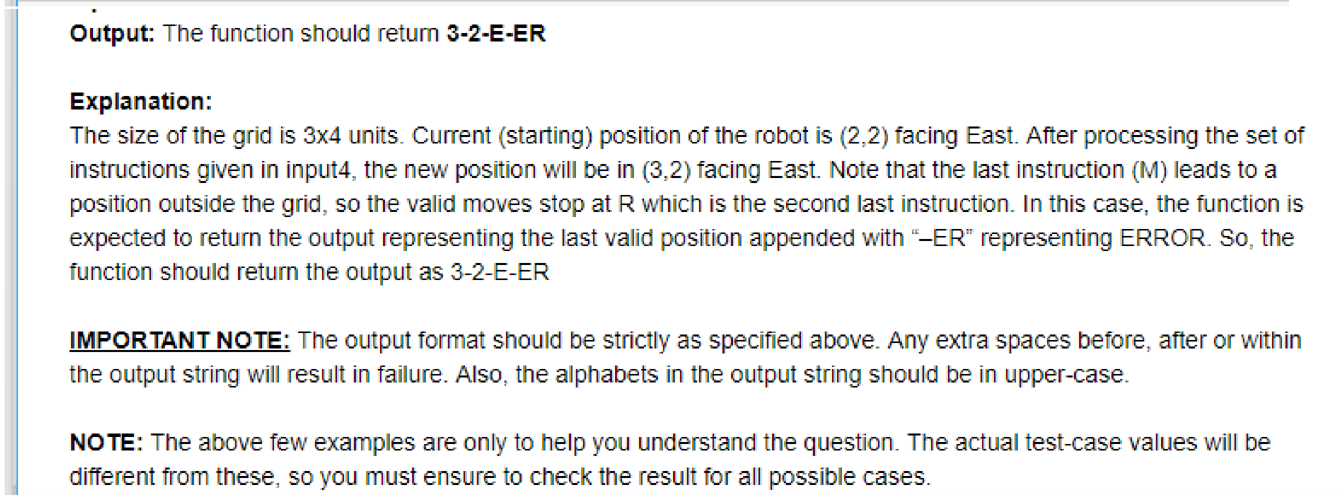
Note:

1. You can assume the grid to be similar to the 1st quadrant of the regular graph sheet. In a regular graph sheet of dimensions x units and y units, (0,0) is the bottom left corner and (x,y) is the top right corner. Ex: For a grid of 5 x 5, the bottom left corner will be (0,0) and top right corner will be (5,5).
2. The starting position of the robot (third input parameter) will be any position on the grid. i.e. it need not always be (0,0)
3. You can assume that the current position (starting position, specified in input3) will always be a valid position within the specified grid.
4. **IMPORTANT -** Note that the instructions L and R only change the direction of the robot without moving it. The instruction M moves the robot 1 unit forward in the direction that the robot is facing.
5. **Invalid moves should not be allowed -** Any move that could lead the robot to a position beyond (outside) the defined x and y axis of the grid OR below 0 on either x or y axis, should be considered an invalid move. (see below examples to get clarity)

Example1 –
input1: x = 3
input2: y = 3
input3: 2-2-E
input4: R M L M L M
Output: The function should return 3-2-N

Explanation:
The size of the grid is 3x3 units. Current (starting) position of the robot is (2,2) facing East. After processing the set of instructions given in input4, the new position will be in (3,2) facing North. So, the function is expected to return the output in the format x-y-D i.e. 3-2-N

Example2 –
input1: x = 3
input2: y = 4
input3: 2-2-E
input4: R M L M L M R M


Output: The function should return **3-2-E-ER**

Explanation:
The size of the grid is 3x4 units. Current (starting) position of the robot is (2,2) facing East. After processing the set of instructions given in input4, the new position will be in (3,2) facing East. Note that the last instruction (M) leads to a position outside the grid, so the valid moves stop at R which is the second last instruction. In this case, the function is expected to return the output representing the last valid position appended with "–ER" representing ERROR. So, the function should return the output as 3-2-E-ER

IMPORTANT NOTE: The output format should be strictly as specified above. Any extra spaces before, after or within the output string will result in failure. Also, the alphabets in the output string should be in upper-case.

NOTE: The above few examples are only to help you understand the question. The actual test-case values will be different from these, so you must ensure to check the result for all possible cases.

```
import java.io.*;
import  java.util.*;

// Read only region start
class UserMainCode
{

                        public String moveRobot(int input1,int input2,String input3,String input4){
                          // Read only region end
                          int row=input1,col=input2;
                        String[] s2=input3.split("-");
                        int r=Integer.parseInt(s2[0]);
                        int c=Integer.parseInt(s2[1]);
                          if(r>input1||c>input2)
                                                return (input3+"-ER");
                        String[] s=input4.split(" ");
                        //System.out.println(s2[2]);
                        for(String ip:s)
                        {
                          System.out.println("ip"+ip);
                        switch(ip)
                        {
                        case "L":
                          System.out.println("Left");
                          if(s2[2].equals("E")){
                                        s2[2]="N";
                                        System.out.println(s2[2]);break;}
                          else if(s2[2].equals("N")){
                                        s2[2]="W";
                                        System.out.println(s2[2]);break;}
                          else if(s2[2].equals("W")){
                                        s2[2]="S";
                                        System.out.println(s2[2]);break;}
                          else if(s2[2].equals("S")){
                                        s2[2]="E";
                                        System.out.println(s2[2]);break;}

                        case "R":
                          System.out.println("Right");
                          if(s2[2].equals("E")){
                                        s2[2]="S";System.out.println(s2[2]);break;}
                          else if(s2[2].equals("S")){
                                        s2[2]="W";System.out.println(s2[2]);break;}
                          else if(s2[2].equals("W")){
                                        s2[2]="N";System.out.println(s2[2]);break;}
                          else if(s2[2].equals("N")){
                                        s2[2]="E";System.out.println(s2[2]);break;}

                        case "M":
```

```
                  System.out.println("row="+r+"col="+c);
                  if(s2[2].equals("E"))
                          if((r+1)<=row)
                                  r=r+1;
                          else
                                  return (r+"-"+c+"-"+s2[2]+"-ER");
                  else if(s2[2].equals("N"))
                          if((c+1)<=col)
                                  c=c+1;
                          else
                                  return (r+"-"+c+"-"+s2[2]+"-ER");
                  else if(s2[2].equals("W"))
                          if((r-1)<=row && (r-1)>=0)
                                  r=r-1;
                          else
                                  return (r+"-"+c+"-"+s2[2]+"-ER");
                  else if(s2[2].equals("S"))
                          if((c-1)<=col && (c-1)>=0)
                                  c=c-1;
                          else
                                  return (r+"-"+c+"-"+s2[2]+"-ER");
                  else
                          return (r+"-"+c+"-"+s2[2]+"-ER");
                System.out.println("Move");
                break;
}
                System.out.println(r+"-"+c+"-"+s2[2]);
                }
                  return (r+"-"+c+"-"+s2[2]);
                }
}
```