04 - Iteration Control Structures

For example:

Input	Result
20	1 2 4 5 10 20

Ex. No. : 4.1 **Date:**

Register No.: **Name:**

Factors of a number

Determine the factors of a number (i.e., all positive integer values that evenly divide into a number).

```
N=int(input())

    For I in range(1,n+1):

        If n%i==0:

        Print(I,end=“ ”)
```

For example:

Input	Result
292	1
1015	2
108	3
22	0

Ex. No. : 4.2 **Date:**

Register No.: **Name:**

Non Repeated Digit Count

Write a program to find the count of non-repeated digits in a given number N. The number will be passed to the program as an input 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 program should return 1 because there is only 1 non--repeated digit '9' in this number
If the given number is 1015, the program should return 2 because there are 2 non-repeated digits in this number, '0', and '5'.
If the given number is 108, the program 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.

```
N=input()
        M={}
        For I in n :
            If I in m :
                M[i]+=1
        P=0
        For I in m.values():
            If i==1:
```

P=p+1

Print(p)

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

For example:

Input	Result
7	2
10	1

Ex. No. : 4.3 **Date:**

Register No.: **Name:**

Prime Checking

Write a program that finds whether the given number N is Prime or not. If the number is prime, the program should return 2 else it must return 1.

Assumption: 2 <= N <=5000, where N is the given number.

```
N=int(input())
        Temp= 2
        If n >= 2 and n <= 5000 :
            For I in range (2, n):
                If n% * I ==0 :
                    Temp= 1
                    Break
        If temp=-1:
                    Print (1)
        Else:
                    Print(2)
```

Input Format:

Integer input from stdin.

Output Format:

Perfect square greater than N.

Example Input:

10

Output:

16

Ex. No. : 4.4 **Date:**

Register No.: **Name:**

Next Perfect Square

Given a number N, find the next perfect square greater than N.

```
N=int(input())
M=0
For I in range (1, n) :
    If (I * i>n) :
        M =i* I
        Break
Print(m)
```

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.

For example:

Input:

7

Output

8

Ex. No. : 4.5 **Date:**

Register No.: **Name:**

Nth Fibonacci

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

```
N=int(input())
        Temp= 0
        A = 1
        S = 0
        For I in range (0, n) :
            S =temp+a
            A=temp
            Temp= s
         Print(a)
```

Input Format:

Single Integer Input from stdin.

Output Format:

Yes or No.

Example Input:

175

Output:

Yes

Explanation

1^1 + 7^2 +5^3 = 175

Example Input:

123

Output:

No

For example:

Input	Result
175	Yes
123	No

Ex. No. : 4.6 **Date:**

Register No.: **Name:**

Disarium Number

A Number is said to be Disarium number when the sum of its digit raised to the power of their respective positions becomes equal to the number itself. Write a program to print number is Disarium or not.

```
N =int(input())

        Num = len(str(n))

        P = num

        Q = n

        R = 0

        Sum=0

       For I in range(0,num):

            R = n% * 10

            Sum = sum + r **p

            P = p – 1

            N =n//10

       If um ==q :
```

```
        Print("Yes")
    Else:
        Print("No")
```

Sample Test Cases

Test Case 1

Input

4

Output

1234

Explanation:

as input is 4, have to take 4 terms.

1 + 11 + 111 + 1111

Test Case 2

Input

6

Output

123456

For example:

Input	Result
3	123

Ex. No. : 4.7 **Date:**

Register No.: **Name:**

Sum of Series

Write a program to find the sum of the series 1 +11 + 111 + 1111 + . . . + n terms (n will be given as input from the user and sum will be the output)

```
N=int(input())
        Sum = 1
        C = 1
        For I in range (1, n) :
            Sum =sum * 10+1
            C = C + sum
        Print(c)
```

For example:

Input	Result
292	2
1015	3

Ex. No. : 4.8 **Date:**

Register No.: **Name:**

Unique Digit Count

Write a program to find the count of unique digits in a given number N. The number will be passed to the program as an input 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 program should return 2 because there are only 2 unique digits '2' and '9' in this number
If the given number is 1015, the program should return 3 because there are 3 unique digits in this number, '1', '0', and '5'.

```
N=int(input())
        A =[ ]
        While n >0:
            If n%10 not in a:
                a.append(n%10)
            n =n//10
        print(len(a))
```

Input Format:

Single Integer input.

Output Format:

Output displays Yes if condition satisfies else prints No.

Example Input:

14

Output:

Yes

Example Input:

13

Output:

No

Ex. No. : 4.9 **Date:**

Register No.: **Name:**

Product of single digit

Given a positive integer N, check whether it can be represented as a product of single digit numbers.

```
N =int(input())

        If( n%2 ==0 or n%3 ==0 or n%5 ==0 or n%7
==0) :

            Print (" Yes ")

        Else:

            Print (" No ")
```

Input Format:

Single integer input.

Output Format:

Yes or No.

Example Input:

24

Output:

Yes

Example Input:

26

Output:

No

For example:

Input	Result
24	Yes

Ex. No. : 4.10 **Date:**

Register No.: **Name:**

Perfect Square After adding One

Given an integer N, check whether N the given number can be made a perfect square after adding 1 to it.

```
N=int(input())+1
        A = 0
        If(n==0 or n ==1) :
            A = 1
        For I in range ( 2 ,(n//2)) :
            If (n==i*i) :
                A = 1
                Break
        If a ==1 :
            Print("Yes")
        Else:
            Print (" No ")
```