04 - Iteration Control Structures

`

<u>Nth Fibonacci</u>

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

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	Result
1	0
4	2
7	8

Program:

```
n=int(input())
f0=0
f1=1
i=0
if(n==1):
    print(f0)
else:
    while(i<n-2):
        f2=f0+f1
        f0=f1
        f1=f2
        i=i+1
    print(f2)
```

	Input	Expected	Got	
✔	1	0	0	✔
✔	4	2	2	✔
✔	7	8	8	✔

.

Factors of a number

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

For example:

Input	Result
20	1 2 4 5 10 20

Program:

```
n=int(input())
list1=[]
for i in range(1,n+1):
    if(n%i==0):
        print(i,end=' ')
```

	Input	Expected	Got	
✔	20	1 2 4 5 10 20	1 2 4 5 10 20	✔
✔	5	1 5	1 5	✔
✔	13	1 13	1 13	✔

.

Product of single digit

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

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

Program:

```python
n=int(input())
flag=False
for i in range(2,10):
    if(n%i==0):
        flag=True
        break
if(flag):
    print("Yes")
else:
```

	Input	Expected	Got	
✔	14	Yes	Yes	✔
✔	13	No	No	✔

print("N_____o")

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'.

For example:

Input	Result
292	2
1015	3

Program:

n=int(input())

list1=[]

list2=[]

while(n>0):

   rem=n%10

   if(rem in list1):

     pass

   else:

     list1.append(rem)

   n=n//10

print(len (list1))

	Input	Expected	Got	
✔	292	2	2	✔
✔	1015	3	3	✔
✔	123	3	3	✔

.

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.

For example:

Input	Result
292	1
1015	2
108	3
22	0

Program:

```
n=int(input())

list1=[]

list2=[]

while(n>0):

    rem=n%10

    n=n//10

    list1.append(rem)

for i in list1:

    c1=list1.count(i)

    if(c1<2):

        list2.append(i)

print(len(list2))
```

	Input	Expected	Got	
✔	292	1	1	✔
✔	1015	2	2	✔
✔	108	3	3	✔
✔	22	0	0	✔

.

Next Perfect Square

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

Input Format:

Integer input from stdin.

Output Format:

Perfect square greater than N.

Example Input:

10

Output:

16

Program:

```
import math

n=int(input())

n1=math.floor(math.sqrt(n))

n1=n1+1

print(pow(n1,2))
```

	Input	Expected	Got	
✔	10	16	16	✔

`

<u>Sum of Series</u>

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)

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

Program:

n=int(input())

sum1=0

t=1

while(n):

   sum1=sum1+t

   t=t*10 +1

   n=n-1

print(sum1)

	Input	Expected	Got	
✔	4	1234	1234	✔
✔	6	123456	123456	✔

.

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.

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

Program:

```
n=int(input())
flag=0
for i in range(2,n):
    if(n%i==0):
        flag=0
        break
    else:
        flag=1
        break
if(flag==1):
    print(2)
```

---

elif(flag==0):

  print(1)

	Input	Expected	Got	
✔	7	2	2	✔
✔	10	1	1	✔

.

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.

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

Program:

```
import math
n=int(input())
i=len(str(n))
temp=n
sum1=0
while(n>0):
    rem=n%10
    sum1=sum1+pow(rem,i)
    i=i-1
    n=n//10
if(sum1==temp):
    print("Yes")
else:
    print("No")
```

	Input	Expected	Got	
✔	175	Yes	Yes	✔
✔	123	No	No	✔

'

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.

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

Program:

import math

n=int(input())

n=n+1

if(math.sqrt(n)==math.floor(math.sqrt(n))):

   print("Yes")

else:

   print("No")

	Input	Expected	Got	
✔	24	Yes	Yes	✔
✔	26	No	No	✔