### [**04 - Iteration Control Structures**](https://www.rajalakshmicolleges.net/moodle/course/view.php?id=84#section-4)

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

**Register No.: Name:**

[**Factors of a number**](https://www.rajalakshmicolleges.net/moodle/mod/quiz/view.php?id=5720)

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 |  |

num = int(input())

factors = []

for i in range(1, num + 1):

if num % i == 0:

print(i, end=' ')

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

**Register No.: Name:**

[**Non**](https://www.rajalakshmicolleges.net/moodle/mod/quiz/view.php?id=5717) **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 |

#non repeated digit

n=int(input())

s=int(input())

l=list(set(s))

d=0

for i in range(0,len(l)):

c=0

for j in range(0,len(s)):

if int(l[i])==int(s[j]):

c=c+1

if c==1:

d=d+1

print(d)

**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.

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 |

num = int(input())

for i in range(2, int(num\*\*0.5) + 1):

if num % i == 0:

print(‘1’)

break

else:

print(‘2’)

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

**Register No.: Name:**

**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

#perfect square

n=int(input(())

c=1

while True:

if(c\*c)>n:

print(c\*c)

break

c=c+1

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

**Register No.: Name:**

**Nth Fibonacci**

Write a [program](https://www.rajalakshmicolleges.net/moodle/mod/quiz/view.php?id=3478) to return the nth number in the fibonacci series. The value of N will be passed to the [program](https://www.rajalakshmicolleges.net/moodle/mod/quiz/view.php?id=3478) 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:**

**7**

**Output**

**8**

#Fibanocci

n = int(input())

if n == 1:

print(0)

elif n == 2:

print(1)

else:

a, b = 0, 1

for \_ in range(n - 2):

a, b = b, a + b

print(b)

**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](https://www.rajalakshmicolleges.net/moodle/mod/quiz/view.php?id=3478) 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 |

#diasarium no

n=int(input())

s=len(str(n))

t=n

p=0

while t!=0:

d=t%10

p=p+(d\*\*s)

s=s-1

t=t//10

if p==n:

print(“yes”)

else:

print(“no”)

**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)

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 |

#sos

n=int(input())

a=n

c=1

s=0

while a!=0:

s=s+c

c=(c\*10)+1

a=a-1

print(s)

**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'.

**For example:**

| **Input** | **Result** |
| --- | --- |
| 292 | 2 |
| 1015 | 3 |

#unique

n=int(input())

s=str(n)

c=0

l=list(set(s))

print(len(l))

**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.

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

#single digit

num = int(input())

original\_num = num

possible = True

for i in range(2, 10):

while num % i == 0:

num //= i

if num == 1:

print("Yes")

else:

print("No")

**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.

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 |

#perfect no

n=int(input())

a=n+1

c=0

while True:

if c\*c==a:

print(“yes”)

break

if (c\*c)>a:

print(“no”)

break

c=c+1