1. Write a program to check whether a given number is a perfect number or not.

A perfect number is a positive number which sum of all positive divisors excluding that number is equal to that number.

For example, 6 is a perfect number since the divisors of 6 are 1, 2, and 3. Sum of its divisors is 1 + 2 + 3 = 6.

**Sample Test Cases**

**Test Case 1**

**Input**  
6

**Output**  
YES

**Test Case 2**

**Input**  
45

**Output**  
NO

# perfect number

n = int(input())

c = 0

for i in range(1, (n // 2) + 1):

if n % i == 0:

c += i

if c == n:

print("YES")

else:

print("NO")

2. Write a program that reads a positive integer, n, from the user and then displays the sum of all of the integers from 1 to n.

**Sample Input**

10

**Sample Output**

The sum of the first 10 positive integers is 55.0

**For example:**

| **Input** | **Result** |
| --- | --- |
| 10 | The sum of the first 10 positive integers is 55.0 |

# sum

def sum(n):

if n == 1:

return 1

return n + sum(n - 1)

n = int(input())

c = sum(n)

print("The sum of the first ", n, " positive integers is ", c, '.0', sep='')

3. A strong number is a special number whose sum of factorial of digits is equal to the original number.

For example, 145 is a strong number. Since, 1! + 4! + 5! = 145.

Write a program to find whether the given number is a Strong Number or not.

**Input Format:**

The input consists of a single integer n.

**Output Format:**

Output consists of a single word 'Yes' or 'No'.

**Sample Input 1:**

145

**Sample Output 1:**

Yes

# strong number

import math

def strong(n):

if n == 0:

return 0

return math.factorial(n % 10) + strong(n // 10)

n = int(input())

c = strong(n)

if c == n:

print("Yes")

else:

print("No")

4. Rakesh loves playing with numbers. He took the Fibonacci series and wants to find the sum of squares of the series until a given value. Write a code that implements his task.

**Input Format:**

Single Integer N

**Output Format:**

Display the sum of squares of the Fibonacci series until the Nth term.

**Example Input:**

9

**Output:**

1870

**Explanation:**

The numbers are: 1 1 2 3 5 8 13 21 34  
Sum of their squares is: 1 + 1 + 4 + 9 + 25 + 64 + 169 + 441 + 1156 = 1870

**For example:**

| **Input** | **Result** |
| --- | --- |
| 9 | 1870 |

# fibonacci

def fiba(n):

if n == 1:

return 0

if n == 2:

return 1

return fiba(n - 1) + fiba(n - 2)

n = int(input())

d = 0

for i in range(1, n + 2):

c = fiba(i)

d = d + (c \* c)

print(d)

5. A number is stable if each digit occurs the same number of times, i.e., the frequency of each digit in the number is the same. For example, 2277, 4004, 11, 23, 538835, 1010 are examples of stable numbers. Similarly, a number is unstable if the frequency of each digit in the number is NOT the same.

**Sample Input:**

2277

**Sample Output:**

Stable Number

**Sample Input 2:**

121

**Sample Output 2:**

Unstable Number

**For example:**

| **Input** | **Result** |
| --- | --- |
| 2277 | Stable Number |

# stable

n = int(input())

s = str(n)

l = len(list(set(s)))

f = list()

for i in set(s):

f.append(s.count(i))

if f.count(f[0]) == l:

print("Stable Number")

else:

print("Unstable Number")