Ex. No. : 4.1 Date: 08/04/2024

Register No.: 230401023 Name: CAROLINE

Factors of a number

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

```
def print_factors (x):
    Print ("The factors of ",x," are :")
    for i in range (1, x+1):
        if x % i = =0 :
        print(i)

num =20
print_factors (num)
```

Ex. No. : 4.2 Date: 03/04/2024

Register No.: 230401023 Name: CAROLINE

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.

```
def countuniquedigits (N):
    res =0
    cnt = [0]*10
    while (N>0):
        rem = N%10
    cnt [rem]+ =1
    N =N//10
for i in range (10):
    if (cnt [i]) ==1:
        res+ =1
    return res
N = int (input())
Print (countuniquedigits(N))
```

Ex. No. : 4.3 Date: 03/04/2024

Register No.: 230401023 Name: CAROLINE

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())
if n>1:
    for i in range (2,n):
        if (n%1) ==0 :
            print ("1")
            break
else:
    print ("2")
```

Ex. No. : 4.4 Date: 03/04/2024

Register No.: 230401023 Name: CAROLINE

Next Perfect Square

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

```
N = int(input())
root = int(N**0.5)+1
result = root * root
print(result)
```

Ex. No. : 4.5 Date: 03/04/2024

Register No.: 230401023 Name: CAROLINE

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.

```
def fib_num(n):
    if n<=0
        print ("fibnonacci cant be compared")
    elif n==1:
        return 0
    elif n==2
        return 1
    else
        return fib_num(n-1) + fib_num(n-2)
n = int(input())
print (fib_num (n))
```

Ex. No. : 4.6 Date: 03/04/2024

Register No.: 230401023 Name: CAROLINE

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.

```
num = int(input())
power-sum = 0
count = len(str(num))
temp = num
while temp > 0:
    digit = temp % 10
    powersum += digit ** count
    temp // =10
    count -= 1
if num == power-sum:
    print("Yes")
else:
    print("No")
```

Ex. No. : 4.7 Date: 03/04/2024

Register No.: 230401023 Name: CAROLINE

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_series = 0
for i in range(1,n+1):
    term = int('1'*i)
    sum_series += term
print(sum_series)
```

Ex. No. : 4.8 Date: 03/04/2024

Register No.: 230401023 Name: CAROLINE

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'.

```
num = input()
unique_digits = set(num)
print len(unique_digits)
```

Ex. No. : 4.9 Date: 03/04/2024

Register No.: 230401023 Name: CAROLINE

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())
can_be_represented = False
for i in range (1, 10):
    if N % i == 0 and N//i < 10:
        can_be_represented = True
if can_be_represent:
    print ("Yes")
else:
    print ("No")
```

Ex. No. : 4.10 Date: 03/04/2024

Register No.: 230401023 Name: CAROLINE

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())
import math as mt
def is_perfect_square(x):
    sr = mt.sqrt(x)
    return ((sr= mt.floor(sr)) == 0)
def is_any_num(n):
    if (is_perfect_square(n+1)):
        return True
    return False
if (is_any_num(n)) :
    print("Yes")
else:
    print("No")
```