



UNIVERSIDAD LAICA ELOY ALFARO DE MANABÍ.

FACULTAD DE CIENCIAS INFORMATICAS.

ESTUDIANTES:

Meza Cedeño Galo Javier.

Luna Bravo Cristhian Alejandro.

ASIGNATURA:

Gestión y calidad del software.

DOCENTE:

Ing. WINTHER ABEL MOLINA LOOR.

CURSO:

Sexto "A".

Ejercicios.

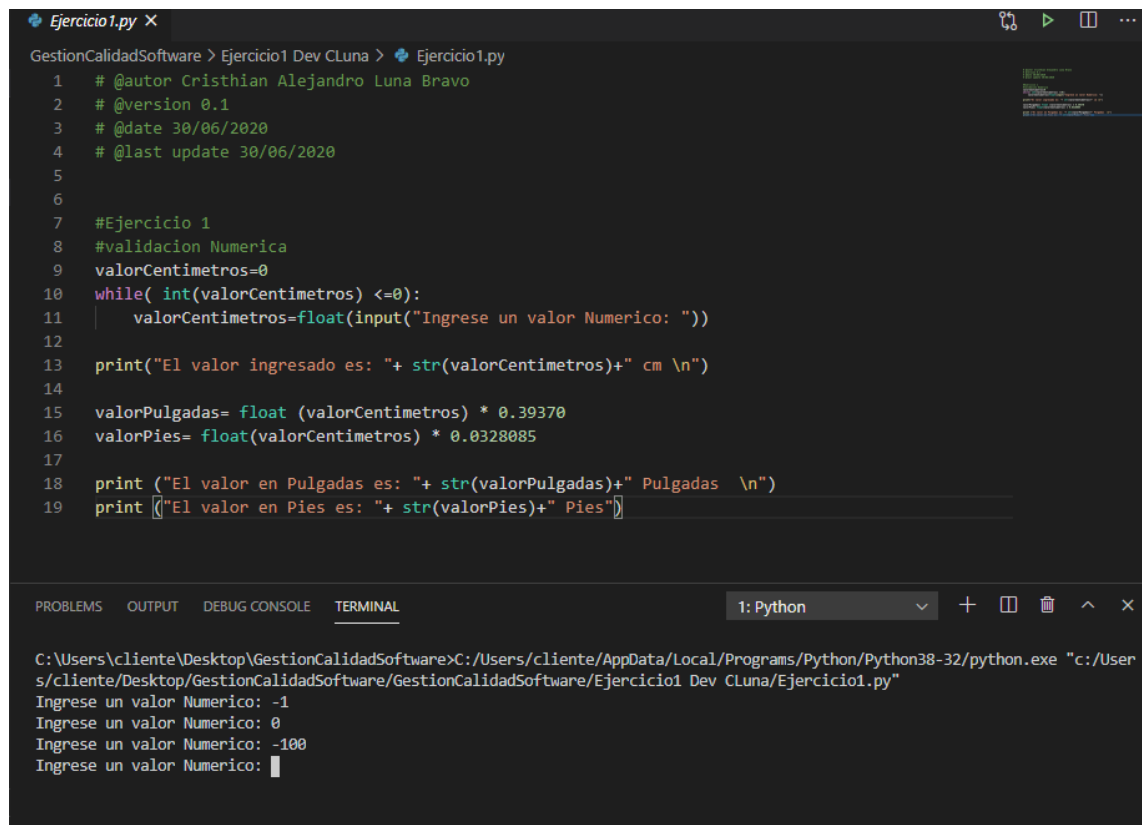
1.-Escriba un programa que dado un valor en centímetros lo convierta a: pulgadas y pies.

Casos

Proceso de Validación
a) Las entradas no pueden ser negativas
b) Las entradas no pueden ser "o"

Casos de Pruebas No Validos

No. Caso de Prueba	N
1	-1
2	0
3	-100



```
Ejercicio1.py X
GestionCalidadSoftware > Ejercicio1 Dev CLuna > Ejercicio1.py
1  # @autor Cristhian Alejandro Luna Bravo
2  # @version 0.1
3  # @date 30/06/2020
4  # @last update 30/06/2020
5
6
7  #Ejercicio 1
8  #validacion Numerica
9  valorCentimetros=0
10 while( int(valorCentimetros) <=0):
11     valorCentimetros=float(input("Ingrese un valor Numerico: "))
12
13 print("El valor ingresado es: "+ str(valorCentimetros)+" cm \n")
14
15 valorPulgadas= float (valorCentimetros) * 0.39370
16 valorPies= float(valorCentimetros) * 0.0328085
17
18 print ("El valor en Pulgadas es: "+ str(valorPulgadas)+" Pulgadas \n")
19 print ("El valor en Pies es: "+ str(valorPies)+" Pies")

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
1: Python
C:\Users\cliente\Desktop\GestionCalidadSoftware>C:/Users/cliente/AppData/Local/Programs/Python/Python38-32/python.exe "c:/User
s/cliente/Desktop/GestionCalidadSoftware/GestionCalidadSoftware/Ejercicio1 Dev CLuna/Ejercicio1.py"
Ingrese un valor Numerico: -1
Ingrese un valor Numerico: 0
Ingrese un valor Numerico: -100
Ingrese un valor Numerico: 
```

Casos de Pruebas Validos

No. Caso de Prueba	N
1	1
2	10
3	100

```

Ejercicio1.py X
GestionCalidadSoftware > Ejercicio1 Dev CLuna > Ejercicio1.py
1 # @autor Cristhian Alejandro Luna Bravo
2 # @version 0.1
3 # @date 30/06/2020
4 # @last update 30/06/2020

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL 1: Python
C:\Users\cliente\Desktop\GestionCalidadSoftware>C:\Users\cliente\AppData\Local\Programs\Python\Python38-32\python.exe "c:/User
s/cliente/Desktop/GestionCalidadSoftware/GestionCalidadSoftware/Ejercicio1 Dev CLuna/Ejercicio1.py"
Ingrese un valor Numerico: 1
El valor ingresado es: 1.0 cm

El valor en Pulgadas es: 0.3937 Pulgadas

El valor en Pies es: 0.0328085 Pies

C:\Users\cliente\Desktop\GestionCalidadSoftware>C:\Users\cliente\AppData\Local\Programs\Python\Python38-32\python.exe "c:/User
s/cliente/Desktop/GestionCalidadSoftware/GestionCalidadSoftware/Ejercicio1 Dev CLuna/Ejercicio1.py"
Ingrese un valor Numerico: 10
El valor ingresado es: 10.0 cm

El valor en Pulgadas es: 3.937 Pulgadas

El valor en Pies es: 0.32808499999999996 Pies

C:\Users\cliente\Desktop\GestionCalidadSoftware>C:\Users\cliente\AppData\Local\Programs\Python\Python38-32\python.exe "c:/User
s/cliente/Desktop/GestionCalidadSoftware/GestionCalidadSoftware/Ejercicio1 Dev CLuna/Ejercicio1.py"
Ingrese un valor Numerico: 100
El valor ingresado es: 100.0 cm

El valor en Pulgadas es: 39.37 Pulgadas

El valor en Pies es: 3.2808499999999996 Pies

C:\Users\cliente\Desktop\GestionCalidadSoftware>
  
```

2.- Diseñar un programa que lea el valor correspondiente a una distancia en millas marinas y las escriba expresadas en: kilómetros, metros, decímetros y centímetros. Sabiendo que 1 milla marina equivale a 1852 metros.

Casos

Proceso de Validación
a) Las opciones al menú no deben ser mayores a 4
b) Los datos del menú no deben ser negativos

```

ejercicio2.py > ...
7
8 # Menu que se muestra para el tipo de conversión a realizar
9 menu = int(input("1. De millas marinas a kilómetros"
10                "\n2. De millas marinas a metros"
11                "\n3. De millas marinas a decímetros"
12                "\n4. De millas marinas a centímetros\n Elija una opción :"))
13
14 # Validamos que los números ingresados sean los correctos
15 while menu!=1 and menu!=2 and menu!=3 and menu!=4:
16     print("\nSelección no válida...")
17     menu = int(input("1. De millas marinas a kilómetros"
18                    "\n2. De millas marinas a metros"
19                    "\n3. De millas marinas a decímetros"
20                    "\n4. De millas marinas a centímetros\n Elija una opción :"))
21
22
23 # if que actua para que el dato ingresado se muestre en kilometros
24 if menu == 1:
25     print("-----")
26     millas_marinas = 0
27     while(int(millas_marinas) <=0):
28         millas_marinas = int(input("Ingrese cantidad de millas marinas a convertir: "))
29
30     kilometros = millas_marinas * 1.852
31     print(millas_marinas, "millas marinas equivale a", kilometros,"kilómetros")
32
33 # if que actua para que el dato ingresado se muestre en metros
34 elif menu == 2:
35     print("-----")
36     millas_marinas = 0
37     while(int(millas_marinas) <=0):
38         millas_marinas = int(input("Ingrese cantidad de millas marinas a convertir: "))
39
40     metros = millas_marinas * 1.852

```

Casos de Pruebas No Validos

No. Caso de Prueba	N
1	5
2	-1

```
Administrador: Símbolo del sistema

C:\Users\Cristhian\Desktop\ejercicios_py>python ejercicio2.py
1. De millas marinas a kilómetros
2. De millas marinas a metros
3. De millas marinas a decímetros
4. De millas marinas a centímetros
Elija una opción :5

Selección no válida...
1. De millas marinas a kilómetros
2. De millas marinas a metros
3. De millas marinas a decímetros
4. De millas marinas a centímetros
Elija una opción :1
-----
Ingrese cantidad de millas marinas a convertir: -1
Ingrese cantidad de millas marinas a convertir: 1
1 millas marinas equivale a 1.852 kilómetros

C:\Users\Cristhian\Desktop\ejercicios_py>python ejercicio2.py
1. De millas marinas a kilómetros
2. De millas marinas a metros
3. De millas marinas a decímetros
4. De millas marinas a centímetros
Elija una opción :2
-----
Ingrese cantidad de millas marinas a convertir: 10
10 millas marinas equivale a 18520 metros

C:\Users\Cristhian\Desktop\ejercicios_py>python ejercicio2.py
1. De millas marinas a kilómetros
2. De millas marinas a metros
3. De millas marinas a decímetros
4. De millas marinas a centímetros
Elija una opción :3
-----
Ingrese cantidad de millas marinas a convertir: 100
100 millas marinas equivale a 1852000 decímetros

C:\Users\Cristhian\Desktop\ejercicios_py>
```

Casos de Pruebas Validos.

No. Caso de Prueba	N
1	1
2	10
3	100

3.-Haga un programa que permita calcular y mostrar el máximo común divisor de dos valores previamente ingresados.

Casos

Proceso de Validación
a) Las entradas no pueden ser negativas
b) Las entradas no pueden ser "o"

Casos de Pruebas No Validos

No. Caso de Prueba	N
1	-1
2	0
3	-100
4	-200

```

Ejercicio3.py X
GestionCalidadSoftware > Ejercicio3 Dev Cluna > Ejercicio3.py > ...
1  # @autor Cristhian Alejandro Luna Bravo
2  # @version 0.1
3  # @date 30/06/2020
4  # @last update 30/06/2020
5
6  #Ejercicio 3
7  #Se importa la libreria math
8  import math
9
10 #Contadores
11 valor1=0
12 valor2=0
13
14 #validacion
15 while( int(valor1) <=0):
16     valor1=int(input("Ingrese primer valor numerico: "))
17
18 while( int(valor2) <=0):
19     valor2=int(input("Ingrese segundo valor numerico: "))
20
21
22
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
(c) 2018 Microsoft Corporation. Todos los derechos reservados.
C:\Users\cliente\Desktop\GestionCalidadSoftware>C:/Users/cliente/AppData/Local/
s/cliente/Desktop/GestionCalidadSoftware/GestionCalidadSoftware/Ejercicio3 Dev Cluna/Ejercicio3.py"
Ingrese primer valor numerico: -1
Ingrese primer valor numerico: -0
Ingrese primer valor numerico: -100
Ingrese primer valor numerico: -200
Ingrese primer valor numerico:
  
```

Casos de Pruebas Validos

No. Caso de Prueba	N
1	1 10
2	10 10
3	100 100
4	200 150

```

Ejercicio3.py x
GestionCalidadSoftware > Ejercicio3 Dev Cluna > Ejercicio3.py > ...
o
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL 1: Python
C:\Users\cliente\Desktop\GestionCalidadSoftware>C:/Users/cliente/AppData/Local/Programs/Python/Python38-32/python.exe "c:/User
s/cliente/Desktop/GestionCalidadSoftware/GestionCalidadSoftware/Ejercicio3 Dev Cluna/Ejercicio3.py"
Ingrese primer valor numerico: 1
Ingrese segundo valor numerico: 10
El Maximo Comun Divisor entre 1 y 10 es: 1

C:\Users\cliente\Desktop\GestionCalidadSoftware>C:/Users/cliente/AppData/Local/Programs/Python/Python38-32/python.exe "c:/User
s/cliente/Desktop/GestionCalidadSoftware/GestionCalidadSoftware/Ejercicio3 Dev Cluna/Ejercicio3.py"
Ingrese primer valor numerico: 10
Ingrese segundo valor numerico: 10
El Maximo Comun Divisor entre 10 y 10 es: 10

C:\Users\cliente\Desktop\GestionCalidadSoftware>C:/Users/cliente/AppData/Local/Programs/Python/Python38-32/python.exe "c:/User
s/cliente/Desktop/GestionCalidadSoftware/GestionCalidadSoftware/Ejercicio3 Dev Cluna/Ejercicio3.py"
Ingrese primer valor numerico: 100
Ingrese segundo valor numerico: 100
El Maximo Comun Divisor entre 100 y 100 es: 100

C:\Users\cliente\Desktop\GestionCalidadSoftware>C:/Users/cliente/AppData/Local/Programs/Python/Python38-32/python.exe "c:/User
s/cliente/Desktop/GestionCalidadSoftware/GestionCalidadSoftware/Ejercicio3 Dev Cluna/Ejercicio3.py"
Ingrese primer valor numerico: 200
Ingrese segundo valor numerico: 150
El Maximo Comun Divisor entre 200 y 150 es: 50

```

4.-Algoritmo que permita ingresar los lados de un triángulo (validar que se forme un triángulo), luego pida al usuario (ingrese) cual sería el área de ese triángulo. Muestre en cuanto porcentaje se equivocó el usuario, si es menos del 5% imprima "usted es una calculadora!!!", si es entre 5% y 20%, imprima "Bien, bien, no muy bien pero bien", si supera el 25% imprima "sería bueno hacer más cálculos mentales".

Casos.

Proceso de Validación
a) Las entradas no pueden ser negativas
b) Las entradas no pueden ser "o"

```

1  # @autor Meza cedeño Gald
2  # Ejercicio 4
3
4  # Definimos las variables
5  lado_a=0
6  lado_b=0
7  lado_c=0
8
9  # Validación de los datos a ingresar
10 while(float(lado_a) <=0):
11     lado_a = float(input("Ingrese el PRIMER lado del triángulo (QUE SEA POSITIVO): "))
12
13 while(float(lado_b) <=0):
14     lado_b = float(input("\nIngrese el SEGUNDO lado del triángulo (QUE SEA POSITIVO): "))
15
16 while(float(lado_c) <=0):
17     lado_c = float(input("\nIngrese el TERCER lado del triángulo (QUE SEA POSITIVO):"))
18
19 #condicional que actua para que los datos ingresados sean iguales
20 #imprima "los lados corresponden a un triángulo"
21 if(lado_a+lado_b)>lado_c and (lado_a+lado_c)>lado_b and (lado_b+lado_c)>lado_a:
22     print("\nlos lados corresponden a un triángulo")
23
24 # Se calcula el area
25

```

Casos de prueba no válidos.

No. Caso de Prueba	N
1	0
2	-1


```
C:\Users\Cristhian\Desktop\ejercicios_py>python ejercicio4.py
Ingrese el PRIMER lado del triángulo (QUE SEA POSITIVO): 0
Ingrese el PRIMER lado del triángulo (QUE SEA POSITIVO): -1
Ingrese el PRIMER lado del triángulo (QUE SEA POSITIVO): 5

Ingrese el SEGUNDO lado del triángulo (QUE SEA POSITIVO): 5

Ingrese el TERCER lado del triángulo (QUE SEA POSITIVO):5

los lados corresponden a un triángulo
El area del triangulo es  10.825317547305483
Usted es una calculadora

C:\Users\Cristhian\Desktop\ejercicios_py>
```

Casos de prueba válidos.

No. Caso de Prueba	N
1	1
2	3
3	5

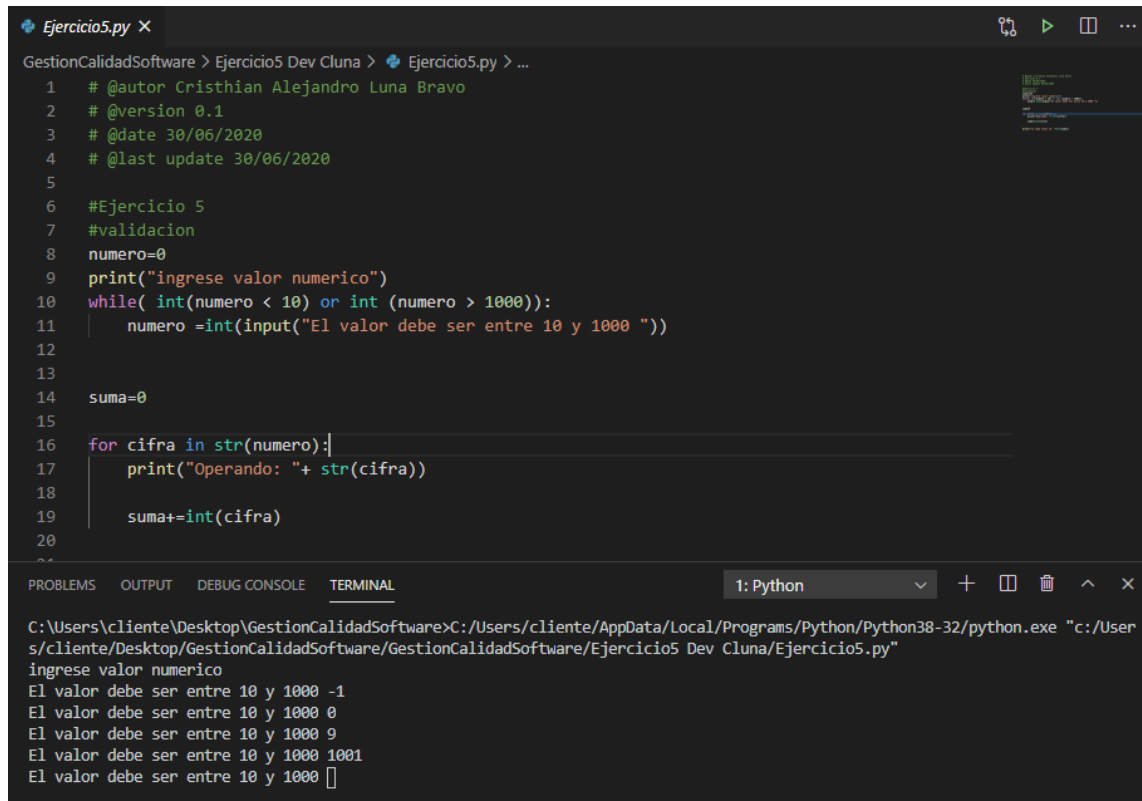
5.-Algoritmo que dado un número (>10 y < 1000) calcula y muestra la suma de sus dígitos.

Casos

Proceso de Validación
c) Las entradas no pueden ser negativas
d) Las entradas no pueden ser “o”
e) Las entradas no pueden ser menores 10 ni mayores a 1000

Casos de Pruebas No Validos

No. Caso de Prueba	N
1	-1
2	0
3	9
4	1001



```
Ejercicio5.py X
GestionCalidadSoftware > Ejercicio5 Dev Cluna > Ejercicio5.py > ...
1  # @autor Cristhian Alejandro Luna Bravo
2  # @version 0.1
3  # @date 30/06/2020
4  # @last update 30/06/2020
5
6  #Ejercicio 5
7  #validacion
8  numero=0
9  print("ingrese valor numerico")
10 while( int(numero < 10) or int (numero > 1000)):
11     numero =int(input("El valor debe ser entre 10 y 1000 "))
12
13
14 suma=0
15
16 for cifra in str(numero):
17     print("Operando: "+ str(cifra))
18
19     suma+=int(cifra)
20
21
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL
1: Python
C:\Users\cliente\Desktop\GestionCalidadSoftware>C:\Users\cliente\AppData\Local\Programs\Python\Python38-32\python.exe "c:/User
s/cliente/Desktop/GestionCalidadSoftware/GestionCalidadSoftware/Ejercicio5 Dev Cluna/Ejercicio5.py"
ingrese valor numerico
El valor debe ser entre 10 y 1000 -1
El valor debe ser entre 10 y 1000 0
El valor debe ser entre 10 y 1000 9
El valor debe ser entre 10 y 1000 1001
El valor debe ser entre 10 y 1000
```

Casos de Pruebas Validos

No. Caso de Prueba	N
1	11
2	10
3	1000
4	900

```

Ejercicio5.py X
GestionCalidadSoftware > Ejercicio5 Dev Cluna > Ejercicio5.py > ...
1 # @autor: Cristhian Alejandro Luna Bravo

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL 1: Python
C:\Users\cliente\Desktop\GestionCalidadSoftware>C:/Users/cliente/AppData/Local/Programs/Python/Python38-32/python.exe "c:/User
s/cliente/Desktop/GestionCalidadSoftware/GestionCalidadSoftware/Ejercicio5 Dev Cluna/Ejercicio5.py"
ingrese valor numerico
El valor debe ser entre 10 y 1000 11
Operando: 1
Operando: 1
La suma final es: 2

C:\Users\cliente\Desktop\GestionCalidadSoftware>C:/Users/cliente/AppData/Local/Programs/Python/Python38-32/python.exe "c:/User
s/cliente/Desktop/GestionCalidadSoftware/GestionCalidadSoftware/Ejercicio5 Dev Cluna/Ejercicio5.py"
ingrese valor numerico
El valor debe ser entre 10 y 1000 1000
Operando: 1
Operando: 0
Operando: 0
Operando: 0
La suma final es: 1

C:\Users\cliente\Desktop\GestionCalidadSoftware>C:/Users/cliente/AppData/Local/Programs/Python/Python38-32/python.exe "c:/User
s/cliente/Desktop/GestionCalidadSoftware/GestionCalidadSoftware/Ejercicio5 Dev Cluna/Ejercicio5.py"
ingrese valor numerico
El valor debe ser entre 10 y 1000 900
Operando: 9
Operando: 0
Operando: 0
La suma final es: 9

C:\Users\cliente\Desktop\GestionCalidadSoftware>C:/Users/cliente/AppData/Local/Programs/Python/Python38-32/python.exe "c:/User
s/cliente/Desktop/GestionCalidadSoftware/GestionCalidadSoftware/Ejercicio5 Dev Cluna/Ejercicio5.py"
ingrese valor numerico
El valor debe ser entre 10 y 1000 10
Operando: 1
Operando: 0
La suma final es: 1

```

6.-Algoritmo que determine los números automórficos menores que 1000. Un número se dice automórfico si su cuadrado termina en los mismos dígitos que el número original, por ejemplo $762^2 = 5776$.

Casos

Proceso de Validación
a) Las entradas no pueden ser negativas
b) Las entradas no pueden ser “0”
c) Las entradas no pueden ser “1001”

```

2  # Ejercicio 6
3
4  #programa que identifique numeros naturales Automórficos.
5  print("-----numeros Automórficos-----")
6
7  numero=0
8  while(int(numero <=0) or int(numero > 1000)):
9      numero = int(input("Ingrese un número del 1 al 1000: "))
10

```

Casos de Pruebas No Validos

No. Caso de Prueba	N
1	-1
2	0
3	1001

```

C:\Users\Cristhian\Desktop\ejercicios_py>python ejercicio6.py
-----numeros Automórficos-----
Ingrese un número del 1 al 1000: -1
Ingrese un número del 1 al 1000: 0
Ingrese un número del 1 al 1000: 1001
Ingrese un número del 1 al 1000: 1
El numero ingresado SI es Automórfico.

C:\Users\Cristhian\Desktop\ejercicios_py>python ejercicio6.py
-----numeros Automórficos-----
Ingrese un número del 1 al 1000: 350
El numero natural NO es Automórfico.

C:\Users\Cristhian\Desktop\ejercicios_py>

```

Casos de Pruebas Validos.

No. Caso de Prueba	N
1	1
2	10
3	1000

7.-Construir un programa que permita determinar si tres valores ingresados son o no un “trío pitagórico”. Un trío pitagórico se define como un conjunto de tres números, a, b y c que cumplen con la relación.

Casos

Proceso de Validación
d) Las entradas no pueden ser negativas
e) Las entradas no pueden ser “o”

Casos de Pruebas No Validos

No. Caso de Prueba	N
1	-1
2	0
3	-100

```

ejercicio7.py X
GestionCalidadSoftware > Ejercicio7 Dev Cluna > ejercicio7.py > ...
1 # @autor Cristhian Alejandro Luna Bravo
2 # @version 0.1
3 # @date 30/06/2020
4 # @last update 30/06/2020
5
6 #Ejercicio 7
7
8 primerValor=0
9 segundoValor=0
10 tercerValor=0
11
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
1: Python
C:\Users\cliente\Desktop\GestionCalidadSoftware>C:/Users/cliente/AppData/Local/Programs/Python/Python38-32/python.exe "c:/User
s/cliente/Desktop/GestionCalidadSoftware/GestionCalidadSoftware/Ejercicio7 Dev Cluna/ejercicio7.py"
Ingrese primer valor: -1
Ingrese primer valor: 0
Ingrese primer valor: -100
Ingrese primer valor:

```

Casos de Pruebas Validos

No. Caso de Prueba	N
1	23
2	22
3	11

```

ejercicio7.py X
GestionCalidadSoftware > Ejercicio7 Dev Cluna > ejercicio7.py > ...
1 # @autor Cristhian Alejandro Luna Bravo
2 # @version 0.1
3 # @date 30/06/2020
4 # @last update 30/06/2020
5
6 #Ejercicio 7
7
8 primerValor=0
9 segundoValor=0
10 tercerValor=0
11
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
1: Python
Focus folder in explorer (ctrl + click)
C:\Users\cliente\Desktop\GestionCalidadSoftware>C:/Users/cliente/AppData/Local/Programs/Python/Python38-32/python.exe "c:/User
s/cliente/Desktop/GestionCalidadSoftware/GestionCalidadSoftware/Ejercicio7 Dev Cluna/ejercicio7.py"
Ingrese primer valor: 23
Ingrese segundo valor: 22
Ingrese tercer valor: 11
EL PRIMER VALOR: 23 es un numero primo
EL SEGUNDO VALOR no es primo
22 veces 1 es 22
EL TERCER VALOR: 11 es un numero primo
El PRIMER valor 23 , el SEGUNDO valor 22 y el TERCER valor 11 no son TRIO PITAGORICO
C:\Users\cliente\Desktop\GestionCalidadSoftware>

```

8.-Escriba un programa que ingrese un entero de tres dígitos, y entregue el número con los dígitos en orden inverso.

Casos

Proceso de Validación
a) Las entradas no pueden ser negativas
b) Las entradas no pueden ser "0"
c) Las entradas no pueden ser mayores a 999

```

ejercicio8.py > ...
1  # @autor Meza cedeño Galo
2  # Ejercicio 8
3
4  # Definimos la lista que va a contener los datos ingresados
5  # variable con un input tipo entero para
6  # el ingreso de datos
7  lista = []
8  numero = int(input("Ingrese tres dígitos a invertir: "))
9
10 # Bucle while restringe el valor del número ingresado
11 while numero > 1000 or numero < 100 or numero == 1000:
12     print(" \nDeben ser tres dígitos... ")
13     numero = int(input("Ingrese tres digitos a invertir: "))
14

```

Casos de Pruebas No Validos.

No. Caso de Prueba	N
1	-1
2	0
3	1000

```

C:\Users\Cristhian\Desktop\ejercicios_py>python ejercicio8.py
Ingrese tres dígitos a invertir: -111

Deben ser tres dígitos menores a 1000...
Ingrese tres digitos a invertir: 000

Deben ser tres dígitos menores a 1000...
Ingrese tres digitos a invertir: 1000

Deben ser tres dígitos menores a 1000...
Ingrese tres digitos a invertir: 123

Número ingresado: 123

Número invertido: 321

C:\Users\Cristhian\Desktop\ejercicios_py>

```

Casos de Pruebas Validos.

No. Caso de Prueba	N
1	123
2	246
3	999

9.-Diseñar un programa que ingrese el total de kilómetros recorridos, el precio de la gasolina (por litro), el dinero de gasolina gastado en el viaje y el tiempo que se ha tardado (en horas y minutos) y que calcule y muestre:

- Consumo de gasolina (en litros y dólares) por cada 100 km.
- Consumo de gasolina (en litros y dólares) por cada km.
- Velocidad media (en km/h y m/s).

Casos

Proceso de Validación
d) Las entradas no pueden ser negativas
e) Las entradas no pueden ser "0"

Casos de Pruebas No Validos

No. Caso de Prueba	N
1	-1
2	0
3	-100

The screenshot shows a Python IDE with a file named `Ejercicio9.py`. The code is a program for calculating travel costs based on kilometers, gasoline price, and money. It includes comments in Spanish and uses `while` loops for input validation. The terminal at the bottom shows the program's execution with inputs `-1`, `0`, and `-100` for kilometers, demonstrating the program's behavior with invalid data.

```
1  # @autor Cristhian Alejandro Luna Bravo
2  # @version 0.1
3  # @date 30/06/2020
4  # @last update 30/06/2020
5
6  #Ejercicio 9
7
8  #InicIALIZACION de Variables
9  valorKilometros=0
10 valorGasolina= 0
11 valorDinero= 0
12 valorTiempo=0
13
14 #Validacion Numerica
15 while( float(valorKilometros) <=0):
16     valorKilometros=float(input("Ingrese valor total de Km recorridos: "))
17     print()
18
19 while( float(valorGasolina) <=0):
20     valorGasolina=float(input("Ingrese valor $ de gasolina por Litros: "))
21     print()
22
23 while( float(valorDinero) <=0):
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

1: Python

Ingrese valor total de Km recorridos: -1
Ingrese valor total de Km recorridos: 0
Ingrese valor total de Km recorridos: -100
Ingrese valor total de Km recorridos:

Casos de Pruebas Validos

No. Caso de Prueba	N
1	1
2	100
3	100
4	52

The screenshot shows a Python IDE with a file named `Ejercicio9.py` open. The code is a Python script that calculates the cost of a trip based on distance, fuel consumption, and time. It includes several `while` loops for input validation and `print` statements for output. The terminal output shows the results of the program execution for specific inputs.

```

12 valorTiempo=0
13
14 #Validacion Numerica
15 while( float(valorKilometros) <=0):
16     valorKilometros=float(input("Ingrese valor total de Km recorridos: "))
17     print()
18
19 while( float(valorGasolina) <=0):
20     valorGasolina=float(input("Ingrese valor $ de gasolina por Litros: "))
21     print()
22
23 while( float(valorDinero) <=0):
24     valorDinero=float(input("Ingrese valor $ gastado en el viaje: "))
25     print()
26
27 while( float(valorTiempo) <=0):
28     valorTiempo=int(input("Ingrese valor de Horas: "))
29     print()
30

```

The terminal output shows the following results:

```

El consumo de gasolina por 100Km Dinero: 10000.0 USD
El consumo de gasolina por Litro: 1.0 l
El consumo de gasolina por Dinero: 100.0 USD

El valor de velocidad (Km/s) : 0.019230769230769232
El valor de velocidad (m/s) : 0.0011538461538461537

C:\Users\cliente\Desktop\GestionCalidadSoftware>

```

10.- Crear un programa que calcule el valor a pagar por un vehículo al circular por una pista. El vehículo puede ser una bicicleta, una moto, un carro o un camión. El valor se calcula según los siguientes datos:

- a. Un valor fijo de 0.50 centavos para las bicicletas.
- b. Las motos y los carros pagarán 0.30 centavos por km.
- c. Los camiones pagaran 0.50 centavos por km más 0.10 centavos por Tm (toneladas métricas).

Al final muestre el resultado solicitado.

Casos

Proceso de Validación
a) Las entradas no pueden ser negativas
b) Las entradas no pueden ser "0"
c) Las entradas del menú deben ser mayores a cero hasta el 4.

```

1  # @autor Meza cedeño Galo
2  # Ejercicio 10
3
4
5  # Menu que se muestra el medio de transporte a elegir
6  print("-----Elija el medio de transporte----- ")
7  menu = int(input("\n1. Bicicleta."
8                  "\n2. Moto."
9                  "\n3. Auto."
10                 "\n4. Camión.\nIngrese opción: "))
11
12 # Validamos que los números de seleccion sean solo del 1 al 4
13 while menu!=1 and menu!=2 and menu!=3 and menu!=4:
14     print("\nSelección no válida...")
15     menu = int(input("1. Bicicleta."
16                     "\n2. Moto."
17                     "\n3. Auto."
18                     "\n4. Camión\n Elija una opción :"))
19
20 # Como el valor de circulación de una bicicleta es fijo lo imprimimos irectamente
21 if menu == 1:
22     print("El valor de circulación de las bicicletas es fijo = 0.50 ctv")
23
24
25 # Selección 2 del menú
26 elif menu == 2:
27     print("-----")
28     valor_moto = 0
29
30     # while hace que no se puedan ingresar valores negativos
31     while(float(valor_moto) <=0):
32         valor_moto = float(input("Ingrese los km recorridos: "))
33     a pagar = valor_moto * 0.30

```

Casos de Pruebas No Validos

No. Caso de Prueba	N
1	-1
2	0
3	5

```

C:\Users\Cristhian\Desktop\ejercicios_py>ejercicio10.py
-----Elija el medio de transporte-----

1. Bicicleta.
2. Moto.
3. Auto.
4. Camión.
Ingrese opción: -1

Selección no válida...
1. Bicicleta.
2. Moto.
3. Auto.
4. Camión
Elija una opción :1
El valor de circulación de las bicicletas es fijo = 0.50 ctv

C:\Users\Cristhian\Desktop\ejercicios_py>ejercicio10.py
-----Elija el medio de transporte-----

1. Bicicleta.
2. Moto.
3. Auto.
4. Camión.
Ingrese opción: 4
-----
Ingrese los km recorridos: 2
Ingrese el valor de toneladas métricas del camión: 3
El valor a pagar es: 1.3 dólares

C:\Users\Cristhian\Desktop\ejercicios_py>

```

Casos de Pruebas válidos.

No. Caso de Prueba	N
1	1
2	10
3	1.5

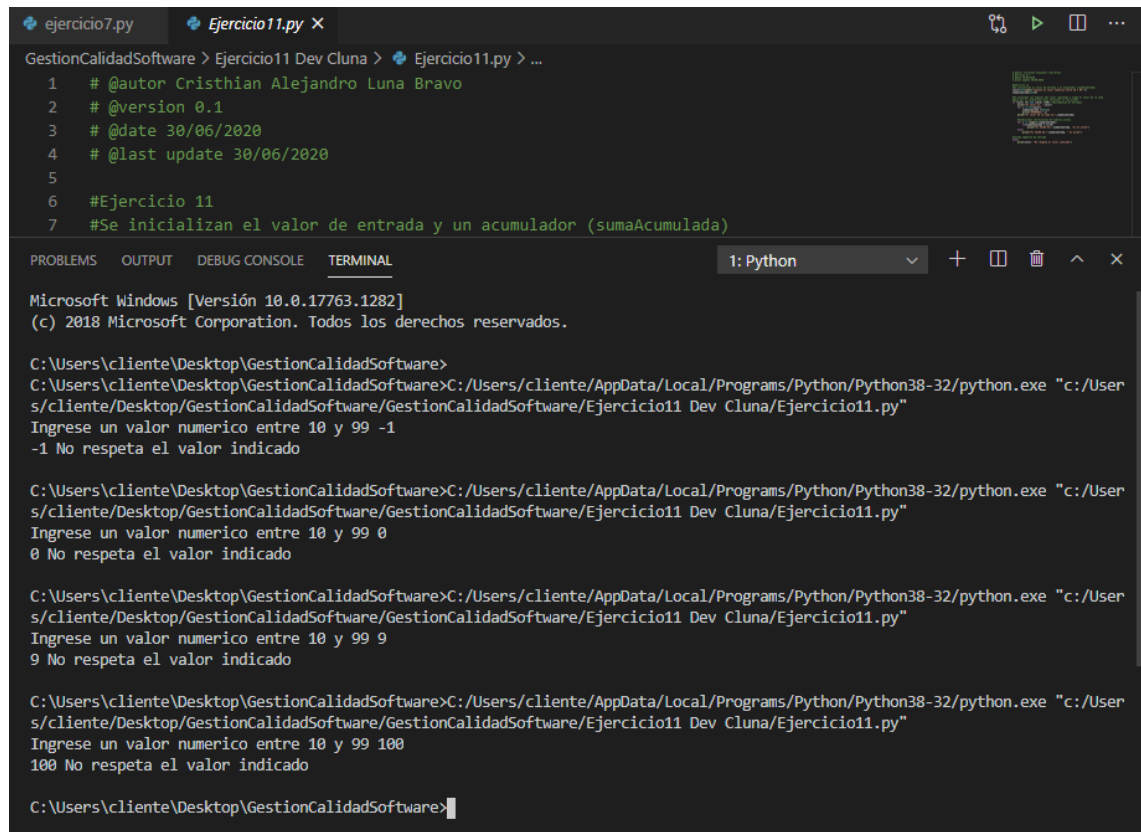
11.- Dado un número entero (entre 10 y 99 ambos inclusive) determinar si la suma de sus dígitos corresponde o no a un número primo.

Casos

Proceso de Validación
d) Las entradas no pueden ser negativas
e) Las entradas no pueden ser “0”
f) Las entradas no pueden ser menores a 10 ni superiores a 99

Casos de Pruebas No Validos

No. Caso de Prueba	N
1	-1
2	0
3	9
4	100



```

ejercicio7.py Ejercicio11.py X
GestionCalidadSoftware > Ejercicio11 Dev Cluna > Ejercicio11.py > ...
1 # @autor Cristhian Alejandro Luna Bravo
2 # @version 0.1
3 # @date 30/06/2020
4 # @last update 30/06/2020
5
6 #Ejercicio 11
7 #Se inicializan el valor de entrada y un acumulador (sumaAcumulada)

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
1: Python
Microsoft Windows [Versión 10.0.17763.1282]
(c) 2018 Microsoft Corporation. Todos los derechos reservados.

C:\Users\cliente\Desktop\GestionCalidadSoftware>
C:\Users\cliente\Desktop\GestionCalidadSoftware>C:/Users/cliente/AppData/Local/Programs/Python/Python38-32/python.exe "c:/User
s/cliente/Desktop/GestionCalidadSoftware/GestionCalidadSoftware/Ejercicio11 Dev Cluna/Ejercicio11.py"
Ingresa un valor numerico entre 10 y 99 -1
-1 No respeta el valor indicado

C:\Users\cliente\Desktop\GestionCalidadSoftware>C:/Users/cliente/AppData/Local/Programs/Python/Python38-32/python.exe "c:/User
s/cliente/Desktop/GestionCalidadSoftware/GestionCalidadSoftware/Ejercicio11 Dev Cluna/Ejercicio11.py"
Ingresa un valor numerico entre 10 y 99 0
0 No respeta el valor indicado

C:\Users\cliente\Desktop\GestionCalidadSoftware>C:/Users/cliente/AppData/Local/Programs/Python/Python38-32/python.exe "c:/User
s/cliente/Desktop/GestionCalidadSoftware/GestionCalidadSoftware/Ejercicio11 Dev Cluna/Ejercicio11.py"
Ingresa un valor numerico entre 10 y 99 9
9 No respeta el valor indicado

C:\Users\cliente\Desktop\GestionCalidadSoftware>C:/Users/cliente/AppData/Local/Programs/Python/Python38-32/python.exe "c:/User
s/cliente/Desktop/GestionCalidadSoftware/GestionCalidadSoftware/Ejercicio11 Dev Cluna/Ejercicio11.py"
Ingresa un valor numerico entre 10 y 99 100
100 No respeta el valor indicado

C:\Users\cliente\Desktop\GestionCalidadSoftware>

```

Casos de Pruebas Validos

No. Caso de Prueba	N
1	10
2	15
3	60
4	70

12.-Pida por teclado cuantos billetes de 100, 20, 10, 5 y 1 tiene la persona en el bolsillo. Ahora pida por teclado cuánto vale un artículo, el programa debe decir si tiene el dinero suficiente para comprarlo y cuanto sería el vuelto, si le hace falta debe salir un aviso diciendo: "te falta dinero" y debe decir cuánto le falta.

Casos

Proceso de Validación
a) Las entradas no pueden ser negativas
b) Las entradas en el precio del artículo no debe ser menor a cero

```

1  # @autor Cristhian Alejandro Luna Bravo
2  # @version 0.1
3  # @date 30/06/2020
4  # @last update 01/07/2020
5
6  print("---Si no tiene el billete de alguna denominacion ingrese el valor 0---")
7  billetes100 = int(input("Ingrese la cantidad de billetes de $100 que tiene: "))
8
9  # Bucles que valida las el ingreso de la cantidad de billetes
10 # en cada uno de los campos de ingreso
11 while (billetes100 < 0):
12     billetes100= int(input("Ingrese la cantidad de billetes de $100 que tiene: "))
13
14 billetes20 = int(input("Ingrese la cantidad de billetes de $20 que tiene: "))
15
16 while (billetes20 < 0):
17     billetes20= int(input("Ingrese la cantidad de billetes de $20 que tiene: "))
18
19 billetes10 = int(input("Ingrese la cantidad de billetes de $10 que tiene: "))
20 while (billetes10 < 0):
21     billetes10= int(input("Ingrese la cantidad de billetes de $10 que tiene: "))
22
23 billetes5 = int(input("Ingrese la cantidad de billetes de $5 que tiene: "))
24
25 while (billetes5 < 0):
26     billetes5= int(input("Ingrese la cantidad de billetes de $5 que tiene: "))
27
28
29 billetes1 = int(input("Ingrese la cantidad de billetes de $1 que tiene: "))
30
31 while (billetes1 < 0):
32     billetes1= int(input("Ingrese la cantidad de billetes de $1 que tiene: "))
33

```

Casos de Pruebas no Validos.

No. Caso de Prueba	N
1	-1
2	-0
3	-10

```

C:\Users\Cristhian\Desktop\ejercicios_py>ejercicio12.py
---Si no tiene el billete de alguna denominacion ingrese el valor 0---
Ingrese la cantidad de billetes de $100 que tiene: -1
Ingrese la cantidad de billetes de $100 que tiene: 0
Ingrese la cantidad de billetes de $20 que tiene: 3
Ingrese la cantidad de billetes de $10 que tiene: 7
Ingrese la cantidad de billetes de $5 que tiene: 4
Ingrese la cantidad de billetes de $1 que tiene: 8
cuenta con un total de: $ 142.0 USD
Ingrese valor del articulo: -3050
Ingrese valor del articulo: 3050
El dinero faltante es: 2908.0

C:\Users\Cristhian\Desktop\ejercicios_py>ejercicio12.py
---Si no tiene el billete de alguna denominacion ingrese el valor 0---
Ingrese la cantidad de billetes de $100 que tiene: 1
Ingrese la cantidad de billetes de $20 que tiene: 2
Ingrese la cantidad de billetes de $10 que tiene: 3
Ingrese la cantidad de billetes de $5 que tiene: 4
Ingrese la cantidad de billetes de $1 que tiene: 5
cuenta con un total de: $ 179.0 USD
Ingrese valor del articulo: 38
El suelto es: 141.0

C:\Users\Cristhian\Desktop\ejercicios_py>

```

Casos de Pruebas Validos

No. Caso de Prueba	N
1	0
2	1
3	10
4	100

13.-Escriba un programa que determine si un carácter ingresado es letra, número, o ninguno de los dos. En caso que sea letra, determine si es mayúscula o minúscula. (Consultar tabla ASCII).

Casos

Proceso de Validación
c) Las entradas no pueden ser negativas
d) Las entradas no pueden ser 2 caracteres

Casos de Pruebas No Validos

No. Caso de Prueba	N
1	11
2	Lg
3	9c

```

ejercicio7.py Ejercicio 13.py X
GestionCalidadSoftware > Ejercicio13 Dev Cluna > Ejercicio 13.py > ...
1 # @autor Cristhian Alejandro Luna Bravo
2 # @version 0.1
3 # @date 30/06/2020
4 # @last update 30/06/2020
5
6 #Ejercicio 13
7 valor=""
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL 1: Python
Microsoft Windows [Versión 10.0.17763.1282]
(c) 2018 Microsoft Corporation. Todos los derechos reservados.

C:\Users\cliente\Desktop\GestionCalidadSoftware>C:/Users/cliente/AppData/Local/Programs/Python/Python38-32/python.exe "c:/User
s/cliente/Desktop/GestionCalidadSoftware/GestionCalidadSoftware/Ejercicio13 Dev Cluna/Ejercicio 13.py"
Ingreso un solo caracter
Ingreso Valor: 11

Ingreso un solo caracter
Ingreso Valor: Lg

Ingreso un solo caracter
Ingreso Valor: 9c

Ingreso un solo caracter
Ingreso Valor: 
  
```

Casos de Pruebas Validos

No. Caso de Prueba	N
1	H
2	5
3	K
4	9

```

ejercicio7.py Ejercicio 13.py X
GestionCalidadSoftware > Ejercicio13 Dev Cluna > Ejercicio 13.py > ...
6 #Ejercicio 13
7 valor=""
8
9 while len(valor)!= 1:

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL 1: Python
Ingrese un solo caracter
Ingrese Valor: H

El valor es: H
El valor " H " representa una LETRA MAYUSCULA el codigo ASCII: 72

C:\Users\cliente\Desktop\GestionCalidadSoftware>C:/Users/cliente/AppData/Local/Programs/Python/Python38-32/python.exe "c:/User
s/cliente/Desktop/GestionCalidadSoftware/GestionCalidadSoftware/Ejercicio13 Dev Cluna/Ejercicio 13.py"
Ingrese un solo caracter
Ingrese Valor: 5

El valor es: 5
El valor " 5 " representa un NUMERO el codigo ASCII: 53

C:\Users\cliente\Desktop\GestionCalidadSoftware>C:/Users/cliente/AppData/Local/Programs/Python/Python38-32/python.exe "c:/User
s/cliente/Desktop/GestionCalidadSoftware/GestionCalidadSoftware/Ejercicio13 Dev Cluna/Ejercicio 13.py"
Ingrese un solo caracter
Ingrese Valor: K

El valor es: K
El valor " K " representa una LETRA MAYUSCULA el codigo ASCII: 75

C:\Users\cliente\Desktop\GestionCalidadSoftware>C:/Users/cliente/AppData/Local/Programs/Python/Python38-32/python.exe "c:/User
s/cliente/Desktop/GestionCalidadSoftware/GestionCalidadSoftware/Ejercicio13 Dev Cluna/Ejercicio 13.py"
Ingrese un solo caracter
Ingrese Valor: 9

El valor es: 9
El valor " 9 " representa un NUMERO el codigo ASCII: 57

```

14.-Un señor adquiere un vehículo en un valor X, dicha persona desea saber la depreciación que sufrirá dicho vehículo en N años. Previo análisis desarrolle un diagrama que permita mostrar el año y la depreciación que sufre el vehículo en cada año. Para calcular la depreciación usará el método de la “suma de dígitos”.

Ejemplo: Si el vehículo vale X cantidad en N años, deberá considerar:

$1+2+3+.....+N$ (Sumatoria)

Depreciación primer año: $N / \text{sumatoria} * X$

Depreciación segundo año: $N-1 / \text{sumatoria} * X$

Depreciación tercer año: $N-2 / \text{sumatoria} * X$

Así sucesivamente.....

Casos

Proceso de Validación
a) Las entradas no pueden ser negativas
b) Las entradas no pueden ser cero.

```

ejercicio14.py ● ejercicio16.py ejercicio2.py ejercicio8.py ▶
ejercicio14.py > calcularDenominador

# Ejercicio 14

#declaramos variables
print('Calculo de la depreciación anual de un valor')
numAnos = 0
valorInicial = 0

while(int(numAnos) <=0):
    numAnos = int(input('Introduce el número de años: '))

while(float(valorInicial) <=0):
    valorInicial = float(input('Introduce el valor del vehiculo:

float(input('Introduce el valor del vehiculo: '))

```

Casos de Pruebas no Validos.

No. Caso de Prueba	N
1	0
2	-1

```

C:\Users\Cristhian\Desktop\ejercicios_py>ejercicio14.py
Calculo de la depreciación anual de un valor
Introduce el número de años: 3
Introduce el valor del vehiculo: 3650
Final del año: 1
Depreciacion: 1825.0
Valor actual: 1825.0
Final del año: 1
Depreciacion: 1216.6666666666665
Valor actual: 2433.3333333333335
Final del año: 1
Depreciacion: 608.3333333333333
Valor actual: 3041.6666666666667

C:\Users\Cristhian\Desktop\ejercicios_py>

```

Casos de Pruebas Validos

No. Caso de Prueba	N
1	3000
2	5

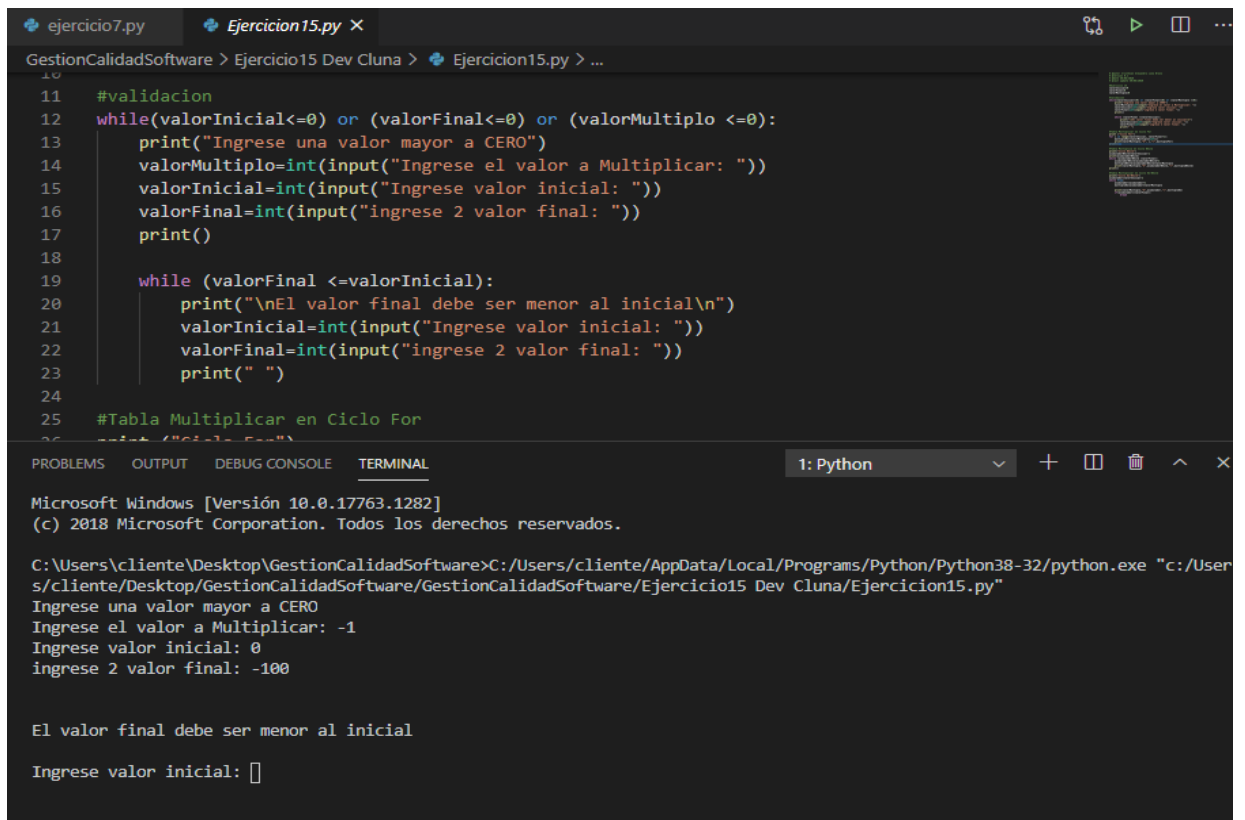
15.- Programa que permita generar una tabla de multiplicar “X” desde un valor “ini” hasta un valor “fin”. La tabla se deberá realizar utilizando las estructuras repetitivas: For... While... Repeat (ver en el lenguaje utilizado sus equivalencias).

Casos

Proceso de Validación
a) Las entradas no pueden ser negativas
b) Las entradas no pueden ser “o”
c) El valor final no puede ser menor al inicial

Casos de Pruebas No Validos

No. Caso de Prueba	N
1	-1
2	0
3	-100



The image shows a Python IDE with two tabs: 'ejercicio7.py' and 'Ejercicio15.py'. The active tab is 'Ejercicio15.py', which contains the following code:

```
10
11 #validacion
12 while(valorInicial<=0) or (valorFinal<=0) or (valorMultiplo <=0):
13     print("Ingrese una valor mayor a CERO")
14     valorMultiplo=int(input("Ingrese el valor a Multiplicar: "))
15     valorInicial=int(input("Ingrese valor inicial: "))
16     valorFinal=int(input("ingrese 2 valor final: "))
17     print()
18
19     while (valorFinal <=valorInicial):
20         print("\nEl valor final debe ser menor al inicial\n")
21         valorInicial=int(input("Ingrese valor inicial: "))
22         valorFinal=int(input("ingrese 2 valor final: "))
23         print(" ")
24
25 #Tabla Multiplicar en Ciclo For
26 print("Tabla For")
```

The terminal window shows the execution of the program. It displays the Windows version (10.0.17763.1282) and the path to the Python executable. The user input is as follows:

```
C:\Users\cliente\Desktop\GestionCalidadSoftware>C:/Users/cliente/AppData/Local/Programs/Python/Python38-32/python.exe "c:/Users/cliente/Desktop/GestionCalidadSoftware/GestionCalidadSoftware/Ejercicio15 Dev Cluna/Ejercicio15.py"
Ingrese una valor mayor a CERO
Ingrese el valor a Multiplicar: -1
Ingrese valor inicial: 0
ingrese 2 valor final: -100

El valor final debe ser menor al inicial

Ingrese valor inicial: 
```

Casos de Pruebas Validos

No. Caso de Prueba	N
1	10 1 6
2	5 1 9
3	9 2 6

```

ejercicio7.py Ejercicio15.py X
GestionCalidadSoftware > Ejercicio15 Dev Cluna > Ejercicio15.py > ...
10
11 #validacion
12 while(valorInicial<=0) or (valorFinal<=0) or (valorMultiplo <=0):
13     print("Ingrese una valor mayor a CERO")
14     valorMultiplo=int(input("Ingrese el valor a Multiplicar: "))

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL 1: Python
Ingrese el valor a Multiplicar: 10
Ingrese valor inicial: 1
ingrese 2 valor final: 6

Ciclo For
10 X 1 = 10
10 X 2 = 20
10 X 3 = 30
10 X 4 = 40
10 X 5 = 50
10 X 6 = 60

Ciclo While
0
10 X 1 = 10
10 X 2 = 20
10 X 3 = 30
10 X 4 = 40
10 X 5 = 50
10 X 6 = 60

Ciclo Do-While
10 X 1 = 10
10 X 2 = 20
10 X 3 = 30
10 X 4 = 40
10 X 5 = 50
10 X 6 = 60
  
```

```

ejercicio7.py Ejercicio15.py X
GestionCalidadSoftware > Ejercicio15 Dev Cluna > Ejercicio15.py > ...
10
11 #validacion
12 while(valorInicial<=0) or (valorFinal<=0) or (valorMultiplo <=0):
13     print("Ingrese una valor mayor a CERO")
14     valorMultiplo=int(input("Ingrese el valor a Multiplicar: "))

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL 1: Python
Ingrese el valor a Multiplicar: 5
Ingrese valor inicial: 1
ingrese 2 valor final: 9

Ciclo For
5 X 1 = 5
5 X 2 = 10
5 X 3 = 15
5 X 4 = 20
5 X 5 = 25
5 X 6 = 30
5 X 7 = 35
5 X 8 = 40
5 X 9 = 45

Ciclo While
0
5 X 1 = 5
5 X 2 = 10
5 X 3 = 15
5 X 4 = 20
5 X 5 = 25
5 X 6 = 30
5 X 7 = 35
5 X 8 = 40
5 X 9 = 45

Ciclo Do-While
5 X 1 = 5
  
```

```
ejercicio7.py Ejercicio15.py X
GestionCalidadSoftware > Ejercicio15 Dev Cluna > Ejercicio15.py > ...

10
11 #validacion
12 while(valorInicial<=0) or (valorFinal<=0) or (valorMultiplo <=0):
13     print("Ingrese una valor mayor a CERO")
14     valorMultiplo=int(input("Ingrese el valor a Multiplicar: "))

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL 1: Python + - X

5 X 6 = 30
5 X 7 = 35
5 X 8 = 40
5 X 9 = 45

Ciclo While
0
5 X 1 = 5
5 X 2 = 10
5 X 3 = 15
5 X 4 = 20
5 X 5 = 25
5 X 6 = 30
5 X 7 = 35
5 X 8 = 40
5 X 9 = 45

Ciclo Do-While
5 X 1 = 5
5 X 2 = 10
5 X 3 = 15
5 X 4 = 20
5 X 5 = 25
5 X 6 = 30
5 X 7 = 35
5 X 8 = 40
5 X 9 = 45

C:\Users\cliente\Desktop\GestionCalidadSoftware\
```

```
ejercicio7.py Ejercicio15.py X
GestionCalidadSoftware > Ejercicio15 Dev Cluna > Ejercicio15.py > ...

10
11 #validacion
12 while(valorInicial<=0) or (valorFinal<=0) or (valorMultiplo <=0):
13     print("Ingrese una valor mayor a CERO")
14     valorMultiplo=int(input("Ingrese el valor a Multiplicar: "))

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL 1: Python + - X

s/cliente/Desktop/GestionCalidadSoftware/GestionCalidadSoftware/Ejercicio15 Dev Cluna/Ejercicio15.py"
Ingrese una valor mayor a CERO
Ingrese el valor a Multiplicar: 9
Ingrese valor inicial: 2
ingrese 2 valor final: 6

Ciclo For
9 X 2 = 18
9 X 3 = 27
9 X 4 = 36
9 X 5 = 45
9 X 6 = 54

Ciclo While
1
9 X 2 = 18
9 X 3 = 27
9 X 4 = 36
9 X 5 = 45
9 X 6 = 54

Ciclo Do-While
9 X 2 = 18
9 X 3 = 27
9 X 4 = 36
9 X 5 = 45
9 X 6 = 54
```

16.- Programa que permita ingresar N cantidades positivas, encuentre y muestre la segunda mayor cantidad ingresada. (Asuma que todos los valores ingresados son diferentes).

Casos

Proceso de Validación
a. Las entradas no pueden ser negativas
b. Las entradas no pueden ser "o"

```

1  # @autor Meza cedeño Galo
2  # Ejercicio 16
3
4  # Declaramos como variables los números a ingresar
5  numero1 = 0
6  numero2 = 0
7  numero3 = 0
8
9  # Ciclo que valida que la entrada se mayor a 0
10 while(int(numero1) <=0):
11     numero1 = int(input("Ingrese el PRIMER número positivo mayor que cero: "))
12
13 while(int(numero2) <=0):
14     numero2 = int(input("Ingrese el SEGUNDO número positivo mayor que cero: "))
15
16 while(int(numero3) <=0):
17     numero3 = int(input("Ingrese el TERCER número positivo mayor que cero: "))
18

```

Casos de Pruebas No Validos

No. Caso de Prueba	N
1	-1
2	0


```

C:\Users\Cristhian\Desktop\ejercicios_py>python ejercicio16.py
Ingrese el PRIMER número positivo mayor que cero: 0
Ingrese el PRIMER número positivo mayor que cero: -1
Ingrese el PRIMER número positivo mayor que cero: 1
Ingrese el SEGUNDO número positivo mayor que cero: 2
Ingrese el TERCER número positivo mayor que cero: 3
El segundo mayor ingresado es: 2

C:\Users\Cristhian\Desktop\ejercicios_py>
C:\Users\Cristhian\Desktop\ejercicios_py>python ejercicio16.py
Ingrese el PRIMER número positivo mayor que cero: 3
Ingrese el SEGUNDO número positivo mayor que cero: 2
Ingrese el TERCER número positivo mayor que cero: 1
El segundo mayor ingresado es: 2

C:\Users\Cristhian\Desktop\ejercicios_py>python ejercicio16.py
Ingrese el PRIMER número positivo mayor que cero: 2
Ingrese el SEGUNDO número positivo mayor que cero: 3
Ingrese el TERCER número positivo mayor que cero: 1
El segundo mayor ingresado es: 2

C:\Users\Cristhian\Desktop\ejercicios_py>

```

Casos de Pruebas Validos

No. Caso de Prueba	N
1	123
2	321
3	231

17.- A través de un Programa desarrolle y muestre el resultado de la siguiente fórmula:

$$S = (1/2) 1^2 + (2/4) 2^2 + (3/6) 3^2 + \dots + (N/(N*2)) N^2$$

Casos

Proceso de Validación
c. Las entradas no pueden ser negativas
d. Las entradas no pueden ser "0"

Casos de Pruebas No Validos

No. Caso de Prueba	N
1	-1
2	0
3	-1
4	f

```

1 # @autor Cristhian Alejandro Luna Bravo
2 # @version 0.1
3 # @date 30/06/2020

Microsoft Windows [Versión 10.0.17763.1282]
(c) 2018 Microsoft Corporation. Todos los derechos reservados.

C:\Users\cliente\Desktop\GestionCalidadSoftware>C:\Users\cliente\AppData\Local\Programs\Python\Python38-32\python.exe "c:/User
s/cliente/Desktop/GestionCalidadSoftware/GestionCalidadSoftware/ejercicio17 Dev Cluna/Ejercicio17.py"
Ingrese un número límite para calcular el valor de la expresión matemática
s=(1/0!)...(1/n!): -1
Ha ingresado un numero negativo. Introduzca un Valor mayor a 0
Ingrese un número límite para calcular el valor de la expresión matemática
s=(1/0!)...(1/n!): 0

s = 1/0! +
s = 1.0
EL PROGRAMA HA FINALIZADO

C:\Users\cliente\Desktop\GestionCalidadSoftware>C:\Users\cliente\AppData\Local\Programs\Python\Python38-32\python.exe "c:/User
s/cliente/Desktop/GestionCalidadSoftware/GestionCalidadSoftware/ejercicio17 Dev Cluna/Ejercicio17.py"
Ingrese un número límite para calcular el valor de la expresión matemática
s=(1/0!)...(1/n!): -100
Ha ingresado un numero negativo. Introduzca un Valor mayor a 0
Ingrese un número límite para calcular el valor de la expresión matemática
s=(1/0!)...(1/n!): f
los caracteres no son validos, Intente nuevamente
invalid literal for int() with base 10: 'f'
Ingrese un número límite para calcular el valor de la expresión matemática
s=(1/0!)...(1/n!):

```

Casos de Pruebas Validos

No. Caso de Prueba	N
1	1
2	8
3	12
4	10

```

ejercicio7.py  Ejercicio17.py X  Ejercicio19.py  Ejercicio1.py
GestionCalidadSoftware > ejercicio17 Dev Cluna > Ejercicio17.py > ...
1 # Autor: Cristhian Alejandro Luna Bravo

PROBLEMAS  OUTPUT  DEBUG CONSOLE  TERMINAL
1: Python

Ingrese un número límite para calcular el valor de la expresión matemática
s=(1/0!)+...(1/n!): 1

s = 1/0! + 1/1! +
s = 2.0
EL PROGRAMA HA FINALIZADO

C:\Users\cliente\AppData\Local\Programs\Python\Python38-32\python.exe "c:/User
s/cliente/Desktop/GestionCalidadSoftware/ejercicio17 Dev Cluna/Ejercicio17.py"
Ingrese un número límite para calcular el valor de la expresión matemática
s=(1/0!)+...(1/n!): 8

s = 1/0! + 1/1! + 1/2! + 1/3! + 1/4! + 1/5! + 1/6! + 1/7! + 1/8! +
s = 3.7182539682539684
EL PROGRAMA HA FINALIZADO

C:\Users\cliente\Desktop\GestionCalidadSoftware>C:\Users\cliente\AppData\Local\Programs\Python\Python38-32\python.exe "c:/User
s/cliente/Desktop/GestionCalidadSoftware/GestionCalidadSoftware/ejercicio17 Dev Cluna/Ejercicio17.py"
Ingrese un número límite para calcular el valor de la expresión matemática
s=(1/0!)+...(1/n!): 12

s = 1/0! + 1/1! + 1/2! + 1/3! + 1/4! + 1/5! + 1/6! + 1/7! + 1/8! + 1/9! + 1/10! + 1/11! + 1/12! +
s = 3.718281826198493
EL PROGRAMA HA FINALIZADO

C:\Users\cliente\Desktop\GestionCalidadSoftware>C:\Users\cliente\AppData\Local\Programs\Python\Python38-32\python.exe "c:/User
s/cliente/Desktop/GestionCalidadSoftware/GestionCalidadSoftware/ejercicio17 Dev Cluna/Ejercicio17.py"
Ingrese un número límite para calcular el valor de la expresión matemática
s=(1/0!)+...(1/n!): 10

s = 1/0! + 1/1! + 1/2! + 1/3! + 1/4! + 1/5! + 1/6! + 1/7! + 1/8! + 1/9! + 1/10! +
s = 3.7182815255731922
EL PROGRAMA HA FINALIZADO

```

18.- Un par de números m y n son llamados amistosos (o se conocen como un par amigable), si la suma de todos los divisores de m (excluyendo a m) es igual al número n, y la suma de todos los divisores del número n (excluyendo a n) es igual a m (con $m \neq n$).

Por ejemplo, los números 220 y 284 son un par amigable porque los únicos números que dividen de forma exacta 220 son 1, 2, 4, 5, 10, 11, 20, 22, 44, 55 y 110, y $1 + 2 + 4 + 5 + 10 + 11 + 20 + 22 + 44 + 55 + 110 = 284$

Por lo tanto, 220 es un número amigoso. Los únicos números que dividen exactamente 284 son 1, 2, 4, 71 y 142 y $1 + 2 + 4 + 71 + 142 = 220$

Por lo tanto, 284 es un número amigoso. Haga un programa y determine si m y n son o no amistosos.

Casos

Proceso de Validación
a. Las entradas no pueden ser negativas
b. Las entradas no pueden ser "o"

```

ejercicio18.py > ...
1  # @autor Meza cedeño Galo
2  # EJERCICIO 18
3
4  # Definimos el ciclo for y el rango desde dónde
5  # se empieza a incrementar
6  def suma(N,S):
7      for i in range(2,N):
8          if(N % i==0):
9              S=S+i
10         return S
11 suma1,suma2=1,1
12
13 #Ingresamos las variables de los numeros a ingresa
14 numero1 = 0
15 numero2 = 0
16
17 # Bucle while que valida a entrada de datos
18 # que no sean cero ni negativos
19 while(numero1 <=0):
20     numero1=int(input("ingrese primer número mayor a cero\n"))
21
22 while(numero2 <=0):
23     numero2=int(input("ingrese segundo número mayor a cero\n"))
24

```

Casos de Pruebas No Validos

No. Caso de Prueba	N
1	-1
2	0
3	-123

```

C:\Users\Cristhian\Desktop\ejercicios_py>ejercicio18.py
ingrese PRIMER número mayor a cero:
0
ingrese PRIMER número mayor a cero:
-1
ingrese PRIMER número mayor a cero:
220
ingrese SEGUNDO número mayor a cero:
284
los numeros 220 y 284 Si son numeros amigos

C:\Users\Cristhian\Desktop\ejercicios_py>ejercicio18.py
ingrese PRIMER número mayor a cero:
123
ingrese SEGUNDO número mayor a cero:
456
los numeros 123 y 456 No son numeros amigos

C:\Users\Cristhian\Desktop\ejercicios_py>

```

Casos de Pruebas Validos

No. Caso de Prueba	N
1	1
2	220
3	284

19.- Los números romanos aún son utilizados para algunos propósitos. Los símbolos básicos y sus equivalencias decimales son:

M	1000
D	500
C	100
L	50
X	10
V	5
I	1

Los enteros romanos se escriben de acuerdo a las siguientes reglas:

a) Si una letra está seguida inmediatamente por una de igual o menor valor, su valor se suma al total acumulado. Así, XX = 20, XV = 15 y VI = 6.

b) Si una letra está seguida inmediatamente por una de mayor valor, su valor se sustrae del total acumulado. Así, IV = 4, XL = 40 y CM = 900.

Escriba un programa que reciba un string con un número en notación romana, y entregue el entero equivalente a arábigo. Ejemplos:

Romano MCMXIV Arabigo: 1914

Romano XIV Arabigo: 14

Romano X Arabigo: 10

Romano IV Arabigo: 4

Casos

Proceso de Validación
a. Las entradas no pueden ser negativas
b. Las entradas no pueden ser numeros enteros

Casos de Prueba No Validos.

No. Caso de Prueba	N
1	
2	
3	

```

ejercicio7.py Ejercicio17.py Ejercicio19.py X
GestionCalidadSoftware > ejercicio19 Dev Cluna > Ejercicio19.py > ...
1 # @autor: Cristhian Alejandro Luna Bravo
2 # @version 0.1
3 # @date 30/06/2020
4 # @last uodate 30/06/2020

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
1: Python
C:\Users\cliente\Desktop\GestionCalidadSoftware>C:\Users\cliente\AppData\Local\Programs\Python\Python38-32\python.exe "c:/User
s/cliente/Desktop/GestionCalidadSoftware/GestionCalidadSoftware/ejercicio19 Dev Cluna/Ejercicio19.py"
Los numeros Romanos son: 'I': 1, 'V': 5, 'X': 10, 'L': 50, 'C': 100, 'D': 500, 'M': 100
Ingrese Numero Romano: n
Traceback (most recent call last):
  File "c:/Users/cliente/Desktop/GestionCalidadSoftware/GestionCalidadSoftware/ejercicio19 Dev Cluna/Ejercicio19.py", line 23,
in <module>
    print("El numero escrito es: ", romano_a_entero(numeroReal))
  File "c:/Users/cliente/Desktop/GestionCalidadSoftware/GestionCalidadSoftware/ejercicio19 Dev Cluna/Ejercicio19.py", line 19,
in romano_a_entero
    entero += romanos[romano[i]]
KeyError: 'n'

C:\Users\cliente\Desktop\GestionCalidadSoftware>C:\Users\cliente\AppData\Local\Programs\Python\Python38-32\python.exe "c:/User
s/cliente/Desktop/GestionCalidadSoftware/GestionCalidadSoftware/ejercicio19 Dev Cluna/Ejercicio19.py"
Los numeros Romanos son: 'I': 1, 'V': 5, 'X': 10, 'L': 50, 'C': 100, 'D': 500, 'M': 100
Ingrese Numero Romano: x
Traceback (most recent call last):
  File "c:/Users/cliente/Desktop/GestionCalidadSoftware/GestionCalidadSoftware/ejercicio19 Dev Cluna/Ejercicio19.py", line 23,
in <module>
    print("El numero escrito es: ", romano_a_entero(numeroReal))
  File "c:/Users/cliente/Desktop/GestionCalidadSoftware/GestionCalidadSoftware/ejercicio19 Dev Cluna/Ejercicio19.py", line 19,
in romano_a_entero
    entero += romanos[romano[i]]
KeyError: 'x'

C:\Users\cliente\Desktop\GestionCalidadSoftware>C:\Users\cliente\AppData\Local\Programs\Python\Python38-32\python.exe "c:/User
s/cliente/Desktop/GestionCalidadSoftware/GestionCalidadSoftware/ejercicio19 Dev Cluna/Ejercicio19.py"
Los numeros Romanos son: 'I': 1, 'V': 5, 'X': 10, 'L': 50, 'C': 100, 'D': 500, 'M': 100
Ingrese Numero Romano: f

```

Casos de Pruebas Validos

No. Caso de Prueba	N
1	
2	
3	
4	

```

ejercicio7.py  Ejercicio17.py  Ejercicio19.py x  Ejercicio1.py
GestionCalidadSoftware > ejercicio19 Dev Cluna > Ejercicio19.py > ...
1  # @autor Cristhian Alejandro Luna Bravo
2  # @version 0.1
3  # @date 30/06/2020

PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL
1: Python

Microsoft Windows [Versión 10.0.17763.1282]
(c) 2018 Microsoft Corporation. Todos los derechos reservados.

C:\Users\cliente\Desktop\GestionCalidadSoftware>C:\Users\cliente\AppData\Local\Programs\Python\Python38-32\python.exe "c:/User
s/cliente/Desktop/GestionCalidadSoftware/GestionCalidadSoftware/ejercicio19 Dev Cluna/Ejercicio19.py"
Los numeros Romanos son: 'I': 1, 'V': 5, 'X': 10, 'L': 50, 'C': 100, 'D': 500, 'M': 100
Ingrese Numero Romano: I
El numero escrito es: 1

C:\Users\cliente\Desktop\GestionCalidadSoftware>C:\Users\cliente\AppData\Local\Programs\Python\Python38-32\python.exe "c:/User
s/cliente/Desktop/GestionCalidadSoftware/GestionCalidadSoftware/ejercicio19 Dev Cluna/Ejercicio19.py"
Los numeros Romanos son: 'I': 1, 'V': 5, 'X': 10, 'L': 50, 'C': 100, 'D': 500, 'M': 100
Ingrese Numero Romano: X
El numero escrito es: 10

C:\Users\cliente\Desktop\GestionCalidadSoftware>C:\Users\cliente\AppData\Local\Programs\Python\Python38-32\python.exe "c:/User
s/cliente/Desktop/GestionCalidadSoftware/GestionCalidadSoftware/ejercicio19 Dev Cluna/Ejercicio19.py"
Los numeros Romanos son: 'I': 1, 'V': 5, 'X': 10, 'L': 50, 'C': 100, 'D': 500, 'M': 100
Ingrese Numero Romano: VI
El numero escrito es: 6

C:\Users\cliente\Desktop\GestionCalidadSoftware>C:\Users\cliente\AppData\Local\Programs\Python\Python38-32\python.exe "c:/User
s/cliente/Desktop/GestionCalidadSoftware/GestionCalidadSoftware/ejercicio19 Dev Cluna/Ejercicio19.py"
Los numeros Romanos son: 'I': 1, 'V': 5, 'X': 10, 'L': 50, 'C': 100, 'D': 500, 'M': 100
Ingrese Numero Romano: IX
El numero escrito es: 9

```

20.- Diseñar un programa que permita calcular los N primeros números perfectos (un número es perfecto, cuando la suma de sus divisores, sin incluirlo al número es exactamente el mismo número).

Casos.

Proceso de Validación
a. No debe calcular los números no perfectos.
b. No debe mostrar negativos.

```

# @autor Meza cedeño Gald
# Ejercicio 20

# Por ciclo for y rango que hará que se muestren los primeros
# números perfectos
for i in range(2,9000):
    numero = 0

    # Variable controlador tomará valores des 1
    for controlador in range(1, (i // 2) +1):

        #identifica el residuo de una división
        if((i % controlador) == 0):
            numero = numero + controlador

    # Si la vanumero es igual a controlador imprimimos el mensaje
    if(numero == i):
        print("El número " + str(i) + " es perfecto")


```

Casos de Prueba No Validos.

No. Caso de Prueba	N
1	-6
2	-28
3	-496

Casos de Pruebas Validos

No. Caso de Prueba	N
1	6
2	28
3	498

 Administrador: Símbolo del sistema

```
C:\Users\Cristhian\Desktop\ejercicios_py>python ejercicio20.py
El número 6 es perfecto
El número 28 es perfecto
El número 496 es perfecto
El número 8128 es perfecto

C:\Users\Cristhian\Desktop\ejercicios_py>
```