

# Tarea # 3 - Calculadora en Python

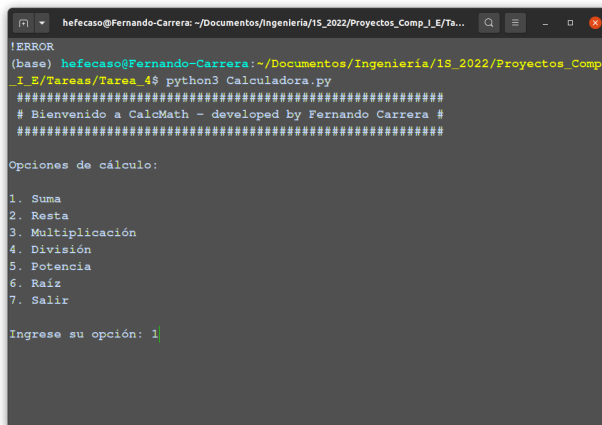
Héctor Fernando Carrera Soto, Carné: 201700923 <sup>\*1</sup>

<sup>1</sup>Universidad de San Carlos de Guatemala, Facultad de ingeniería, Escuela de ingeniería mecánica eléctrica, Ingeniería electrónica.

10 de febrero de 2022

## 1. Procedimiento y resultados

Ejecutando el programa, se observó el menú creado.



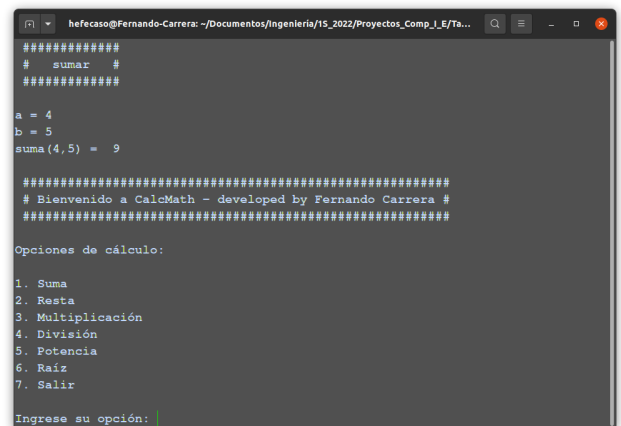
```
hefecaso@Fernando-Carrera: ~/Documentos/Ingenieria/1S_2022/Proyectos_Comp_I/E/Ta...
!ERROR
(base) hefecaso@Fernando-Carrera:~/Documentos/Ingenieria/1S_2022/Proyectos_Comp_I/E/Tareas/Tarea_4$ python3 Calculadora.py
#####
# Bienvenido a CalcMath - developed by Fernando Carrera #
#####

Opciones de cálculo:

1. Suma
2. Resta
3. Multiplicación
4. División
5. Potencia
6. Raíz
7. Salir

Ingrese su opción: 1
```

Figura 1: Menú de la calculadora.



```
hefecaso@Fernando-Carrera: ~/Documentos/Ingenieria/1S_2022/Proyectos_Comp_I/E/Ta...
#####
# sumar #
#####

a = 4
b = 5
suma(4,5) = 9

#####
# Bienvenido a CalcMath - developed by Fernando Carrera #
#####

Opciones de cálculo:

1. Suma
2. Resta
3. Multiplicación
4. División
5. Potencia
6. Raíz
7. Salir

Ingrese su opción: 1
```

Figura 2: Sumando dos números.

Se seleccionó la opción sumar, una vez realizada la operación, permitiendo realizar otra operación o cerrar el programa.

Permitió realizar una operación nueva o finalizar el programa.

<sup>\*</sup>3505043180101@ingenieria.usac.edu.gt

```

hefecaso@Fernando-Carrera: ~/Documentos/Ingenieria/15_2022/Proyectos_Comp_I_E/Ta...
#####
# sumar #
#####
a = 4
b = 5
suma(4,5) = 9

#####
# Bienvenido a CalcMath - developed by Fernando Carrera #
#####

Opciones de cálculo:

1. Suma
2. Resta
3. Multiplicación
4. División
5. Potencia
6. Raíz
7. Salir

Ingrese su opción: 7

```

Figura 3: Opción de realizar otra operación o salir.

así el programa.

```

hefecaso@Fernando-Carrera: ~/Documentos/Ingenieria/15_2022/Proyectos_Comp_I_E/Ta...
(base) hefecaso@Fernando-Carrera:~/Documentos/Ingenieria/15_2022/Proyectos_Comp_I_E/Tareas/Tarea_4$

```

Figura 4: Se seleccionó la opción 7 del menú.

## 2. Contenido del archivo Calculadora.py

```

#####
# Programa creado por Héctor Fernando Carrera Soto #
# 201700923 #
#####

```

```

import numpy as np
'''

```

Numpy es una librería en la que se define un tipo de dato que representa matrices multidimensionales, equivalentes a las matrices del R. Además incluye algunas funcionalidades básicas para trabajar con ellas.

```

'''

```

```

import os

```

```

'''

```

El módulo os nos permite acceder a funcionalidades dependientes del Sistema Operativo. Sobre todo, aquellas que nos refieren información sobre el entorno del mismo y nos permiten manipular la estructura de directorios (para leer y escribir archivos).

```

'''

```

```

#####
# Funciones matemáticas #
#####

```

```

class fun_math:
    def sumar (a,b):
        x = a + b

```

```

        return x

def restar(a,b):
    x = a - b
    return x

def multiplicar(a,b):
    x = a * b
    return x

def dividir(a,b):
    x = a / b
    return x

def potencia(a,b):
    x = np.power(a,b)
    return x

def raiz(a,b):
    x = np.power(a,1/b)
    return x #Definimos una clase con las operaciones

#####
#  Métodos de ejecución      #
#####

class exec: #Definimos la clase exec
    def menu(): #Definimos método menú
        print(' #####')
        print(' # Bienvenido a CalcMath - developed by Fernando Carrera #')
        print(' ##### \n')

        print('Opciones de cálculo: \n')
        print('1. Suma')
        print('2. Resta')
        print('3. Multiplicación')
        print('4. División')
        print('5. Potencia')
        print('6. Raíz')
        print('7. Salir \n')

    def sum(): #Definimos método suma
        print(' #####')
        print(' #   sumar   #')
        print(' #####\n')

        a = int(input('a = '))
        b = int(input('b = '))
        x = fun_math.sumar(a,b)

```

```

print(f'suma({a},{b}) = ', x)

def res(): #Definimos método resta
    print(' #####')
    print(' #   restar   #')
    print(' #####\n')

    a = int(input('a = '))
    b = int(input('b = '))
    x = fun_math.restar(a,b)

    print(f'restar({a},{b}) = ', x)

def mult(): #Definimos método multiplicación
    print(' #####')
    print(' #   multiplicar   #')
    print(' #####\n')

    a = int(input('a = '))
    b = int(input('b = '))
    x = fun_math.multiplicar(a,b)

    print(f'multiplicar({a},{b}) = ', x)

def div(): #Definimos método dividir
    print(' #####')
    print(' #   dividir   #')
    print(' #####\n')

    a = int(input('a = '))
    b = int(input('b = '))
    x = fun_math.dividir(a,b)

    print(f'multiplicar({a},{b}) = ', x)

def pot(): #Definimos método potencia
    print(' #####')
    print(' #   potencia   #')
    print(' #####\n')

    a = int(input('a = '))
    b = int(input('b = '))
    x = fun_math.potencia(a,b)

    print(f'multiplicar({a},{b}) = ', x)

def raiz(): #Definimos método raíz
    print(' #####')

```

```

print(' #   raiz   #')
print(' #####\n')

a = int(input('a = '))
b = int(input('b = '))
x = fun_math.raiz(a,b)

print(f'multiplicar({a},{b}) = ', x)

#####
#   Iniciando ciclos   #
#####

while True:
    '''
    while True significa bucle para siempre. La while declaración
    toma una expresión y ejecuta el cuerpo del bucle mientras que la
    expresión se evalúa como (booleana) "verdadera". True siempre se
    evalúa como booleano "verdadero" y, por lo tanto, ejecuta el cuerpo
    del bucle de forma indefinida.
    '''
    try:
        '''
        Dentro del bloque try se ubica todo el código que pueda llegar
        a levantar una excepción, se utiliza el término levantar para
        referirse a la acción de generar una excepción.
        '''
        exec.menu()

        opc1 = int(input('Ingrese su opción: '))
        os.system ("clear")

        if opc1 == 1:
            exec.sum()
            print('')

        elif opc1 == 2:
            exec.res()
            print('')

        elif opc1 == 3:
            exec.mult()
            print('')

        elif opc1 == 4:
            exec.div()
            print('')

        elif opc1 == 5:

```

```

        exec.pot()
        print('')

    elif opc1 == 6:
        exec.raiz()
        print('')

    elif opc1 == 7:
        break
        '''
        La instrucción break le proporciona la oportunidad de cerrar
        un bucle cuando se activa una condición externa.
        '''

except: #En caso de error:
    '''
    Durante la ejecución de un programa pueden aparecer errores o
    excepciones. Cuando eso sucede, el programa se detendrá como
    consecuencia. Para evitar esta situación, existen las sentencias
    try y except en Python. Las mismas nos permitirán «atrapar»
    excepciones y como resultado, responder sin que el programa falle.
    '''

    os.system("clear")
    print ("!ERROR")
    op = '?'
    break

```