



TSU Servicios en la nube

Práctica. Creación de una calculadora en Python


Guadalupe Moreno Quintanar

Ciudad de México a 16 de febrero de 2024.



Índice

Introducción.....	3
Desarrollo de la práctica.....	4
Conclusión	5
Referencias.....	6



Introducción

Python es un lenguaje de programación ampliamente utilizado en el desarrollo de software, la ciencia de datos y machine learning. Es un lenguaje de programación de código abierto, orientado a objetos, fácil de interpretar y con una sintaxis que permite una lectura simple. Es un lenguaje de programación de alto nivel.

Los primeros pasos para aprender a utilizarlos es entender como realiza las operaciones, que tipo de datos maneja y como lo organiza. Posteriormente es útil conocer como manipular las operaciones a través de sus herramientas de flujo de procesos. La mejor forma de aprender un lenguaje de programación es a través del desarrollo de proyectos. Las herramientas más útiles son los tutoriales, la documentación de los paquetes y los blogs donde se responden las preguntas a los problemas más comunes. Algo que es de mucha utilidad es leer scripts de otros programadores para conocer diferentes formas de resolver los mismos problemas.

Es importante que en el desarrollo de la programación se vaya comentando y documentando los scripts para su posterior consulta, el orden en el código y el uso de nombre de variables simples y descriptivas es la mejor forma de mantener la lectura de los programas.

En esta práctica se desarrollara una calculadora con un programa en Python, con interacción con el usuario y validaciones de datos.

Desarrollo de la práctica

Programación de una calculadora

Se programo en Python una calculadora que realiza las operaciones de suma, resta, multiplicación, división y potencia de números enteros y decimales

Se valida que las entradas sean las esperados, tanto para las cifras como para las operaciones. Los datos se le solicitan al usuario y se obtiene la respuesta de la operación solicitada.

Código

Se adjuntan el código desarrollado que esta comentado para su seguimiento.

```
#Programacion de una calculadora

# Operaciones incluidas
lista = [ '+', '-', '*', '/', '^' ]
# Definiendo la funci'on que realiza las operaciones
def calculadora(a, op ,b):
    if op == '+':
        resultado =(a+b)
    elif op == '-':
        resultado =(a-b)
    elif op == '*':
        resultado =(a*b)
    elif op == '/':
        resultado =(a/b)
    elif op == '^' :
        resultado =(a**b)
    else :
        resultado = ('algo esta mal')
    resultado = str(a)+ ' '+ op + ' '+str(b)+ ' '+'='+' '+ str(resultado)
    return(print(resultado))

print('Calculadora ')
# Solicitando primera cifra con control de errores
while True:
    try:
        a = float(input('Ingresa la primer cantidad:'))
    except ValueError :
        print('No es un número válido')
    else :
```

```

        break

op = '0'
while op not in lista:
    op = str(input('Ingresa el simbolo de la operacion, esta permitido (+, -, *, /, ^):'))
    if op not in lista :
        print('Operacion no valida\n')

#Solicitando segunda cifra con control de errores
while True:
    try:
        b = float(input('Ingresa la segunda cantidad:'))
        if op == '/':
            while b == 0:
                b = float(input('Ingresa una cantidad diferente de cero:'))
    except ValueError :
        print('No es un número válido')
    else :
        break
calculadora(a,op,b)

```

Este programa esta compartido en una carpeta pública en github.

Github: LupitaMoreno/TSU_Nube

https://github.com/LupitaMoreno/TSU_Nube

Resultados

Se adjunta foto como comprobación de su funcionamiento.

```

PS C:\Users\Lupi> & C:/Users/Lupi/AppData/Local/Programs/Python/Python312/python.exe "c:/Users/Lupi/One
acion/calculadora.py"
Calculadora
Ingresa la primer cantidad: 2
Ingresa el símbolo de la operación, se pueden realizar las siguiente operaciones(+, -, *, /, ^): ^
Ingresa la segunda cantidad: 4
2.0 ^ 4.0 = 16.0
PS C:\Users\Lupi>

```

Validaciones

Se tienen varias validaciones en el programas , las solicitadas para el ingreso de los datos y las

operaciones. Y también **se agregó una restricción adicional** para que en el caso de la división el denominador no sea 0.

Comprobación de manejo de errores.

```
PS C:\Users\Lupi> & C:/Users/Lupi/AppData/Local/Programs/Python/Python312/python.exe "c:/Users/Lupi/OneDrive/
acion/calculadora.py"
Calculadora
Ingresa la primer cantidad: diez
No es un número válido
Ingresa la primer cantidad: 3r
No es un número válido
Ingresa la primer cantidad: +
No es un número válido
Ingresa la primer cantidad: █
```

Comprobación de manejo de errores en el ingreso la operación solicitada.

```
PS C:\Users\Lupi> & C:/Users/Lupi/AppData/Local/Programs/Python/Python312/python.exe "c:/Users/Lupi/OneDrive/
acion/calculadora.py"
Calculadora
Ingresa la primer cantidad: 2
Ingresa el símbolo de la operación, se pueden realizar las siguiente operaciones(+, -, *, /,^): =
Operación no válida

Ingresa el símbolo de la operación, se pueden realizar las siguiente operaciones(+, -, *, /,^): $
Operación no válida

Ingresa el símbolo de la operación, se pueden realizar las siguiente operaciones(+, -, *, /,^): -
Ingresa la segunda cantidad: █
```

Uso de git para el versionado

Se estuvo realizando el versionado del programa para poder revisar después el avance del desarrollo. Las primeras pruebas se realizaban en jupyter notebook y después se juntaba todo el código para hacer las pruebas necesarias.

Se está usando git vinculado al Visual Studio Code para este fin.

The screenshot shows the Visual Studio Code editor with a Python file named `calculadora.py` open. The code implements a calculator with error handling for invalid inputs. The terminal window at the bottom shows the program's execution, including prompts for the first number, the operation, and the second number, as well as error messages for invalid inputs.

```

28     except ValueError :
29         print('No es un número válido')
30     else :
31         break
32
33     op = '0'
34     while op not in lista:
35         op = str(input('Ingresa el símbolo de la operación, se pueden realizar las siguiente operas
36         if op not in lista :
37             print('Operación no válida\n')
38
39
40     #Solicitando segunda cifra con control de errores
41     while True:
42         try:
43             b = float(input('Ingresa la segunda cantidad: '))
44             if op == '/':

```

Terminal Output:

```

Calculadora
Ingresa la primer cantidad: 2
Ingresa el símbolo de la operación, se pueden realizar las siguiente operaciones(+, -, *, /,^): =
Operación no válida
Ingresa el símbolo de la operación, se pueden realizar las siguiente operaciones(+, -, *, /,^): $
Operación no válida
Ingresa el símbolo de la operación, se pueden realizar las siguiente operaciones(+, -, *, /,^): -

```

Puntos extras

Realicé el puntos extra y el puntos extras 2. El programa completo se puede revisar en Github con el nombre de archivo `calculadora2.py`

```

solicitud = str(input('Ingresa la operacion que deseas, ej. 3+2 /n'))

#solicitud = '1+2+3*4/5^6'
#Creando una lista con la secuencia de operaciones
ops = []
for op in solicitud:
    if op in lista :
        ops.append(op)
ops

sec = []
pos = 0
for ele in ops :
    pos += 1
    #print(ele)
    sec.append(solicitud.split(ele,1)[0])
    sec.append(ele)
    #print(sec)

```

```
solicitud = solicitud.split(ele,1)[1]
print(solicitud)
if pos == len(ops):
    sec.append(solicitud)
```

Conclusión

Aprender a trabajar con los tipos de datos y variables es muy importante para la generación de soluciones a problemas y también para encontrar los caminos más sencillos.

El proceso de programación es un proceso creativo, generalmente inicias con una idea de cómo lo harás, que herramientas utilizaras, pero nada sale como se espera y entonces realizas ajustes al plan inicial e investigas nuevas herramientas para resolver los problemas o salidas no esperadas.

Durante la captura de datos puedes encontrar muchos errores, por lo que hay que ser muy claros en las indicaciones a los usuarios y sobre todo que el programa tenga las validaciones necesarias. El manejo de errores en el ingreso de datos es fundamental para que la calculadora de los resultados esperados.

Si pude resolver los errores que encontré, lo que no pude es realizar todos los ejercicios de los puntos extras.



Referencias

Deitel, P. & Deitel, H. (2019). Intro to Python for Computer Science and Data Science: Learning to Program with AI, Big Data and The Cloud. City: Pearson.