

Algoritmos y Programación 1

Ejercicio 1

Cátedra: Essaya

Práctica: Grace/Juani

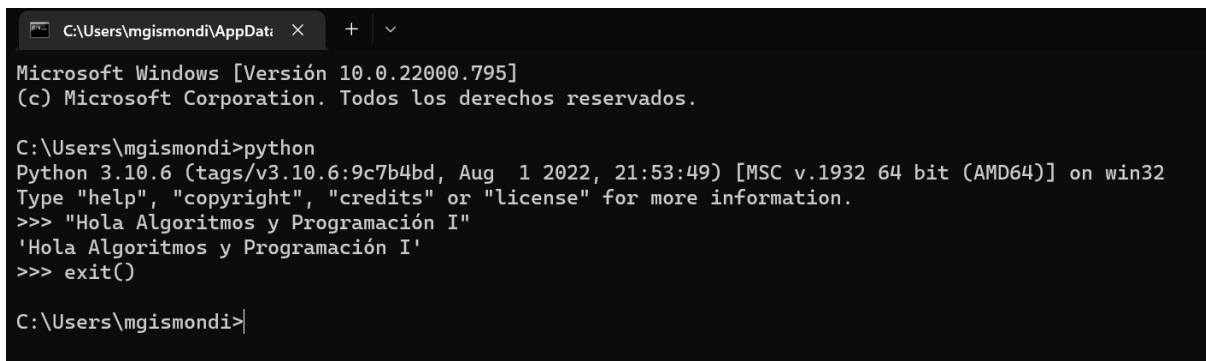
Nombre del Alumno: Máximo Gismondi

Padrón: 110119

Ayudante a cargo: Cristóbal Sabella Rosa

Parte 1

1.1

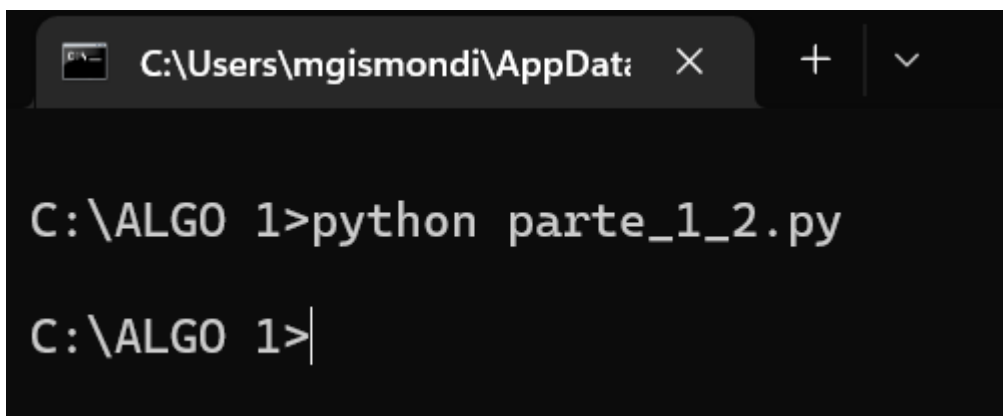


```
C:\Users\mgismondi\AppData: X + v
Microsoft Windows [Versión 10.0.22000.795]
(c) Microsoft Corporation. Todos los derechos reservados.

C:\Users\mgismondi>python
Python 3.10.6 (tags/v3.10.6:9c7b4bd, Aug 1 2022, 21:53:49) [MSC v.1932 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> "Hola Algoritmos y Programación I"
'Hola Algoritmos y Programación I'
>>> exit()

C:\Users\mgismondi>
```

1.2



```
C:\Users\mgismondi\AppData: X + v

C:\ALGO 1>python parte_1_2.py

C:\ALGO 1>
```

Deberíamos usar la función print para ver el mismo resultado.

En el primer caso vemos el resultado porque cuando escribimos una expresión en el modo interactivo, nos devuelve el resultado de la expresión, en este caso la expresión "Hola Algoritmos y Programación I" da como resultado "Hola Algoritmos y Programación I".

Parte 2

```
C:\Users\mgismondi\AppData\Local\Programs\Python\Python310\Scripts>python norma.py

C:\ALGO 1>
```

```
C:\Users\mgismondi\AppData\Local\Programs\Python\Python310\Scripts>python norma.py
Traceback (most recent call last):
  File "C:\ALGO 1\norma.py", line 18, in <module>
    assert norma(-70, 14, z) == 111.0
AssertionError

C:\ALGO 1>
```

La salida es una excepción, un AssertionError más específicamente.

El error, se generó en la línea 18, tal como dice la ubicación del error "line 18".

Según internet, assert es una instrucción donde establecemos una condición que debe cumplirse siempre para que el programa se ejecute sin excepciones. Es decir que nos permite definir una condición donde si el resultado es verdadero (True), el programa no devuelve nada, afirmandonos que pasó esa prueba con éxito. Caso contrario, si el resultado es falso (False) nos saltará una excepción en la consola, detallando la ubicación y el motivo del error.

```
C:\ALGO 1>python
Python 3.10.6 (tags/v3.10.6:9c7b4bd, Aug 1 2022, 21:53:49) [MSC v.1932 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> z = (111 ** 2 - 70 ** 2 - 14 ** 2) ** 0.5
>>> z
85.0
>>> |
```

z = 85, para que no de error

```
C:\ALGO 1>python norma.py
```

```
C:\ALGO 1>|
```

Parte 3

```
C:\ALGO 1>python diferencia.py
Traceback (most recent call last):
  File "C:\ALGO 1\diferencia.py", line 17, in <module>
    assert diferencia(Ax1, Ay1, Az1, Bx1, By1, Bz1) == (-39, -162, -21)
  File "C:\ALGO 1\diferencia.py", line 6, in diferencia
    return dif_x, dif_y, diff_z
NameError: name 'diff_z' is not defined. Did you mean: 'dif_z'?

C:\ALGO 1>|
```

Se detectó un error, un NameError en la línea 6 (la 17 es de donde se llama a la función), al llamar a la variable “diff_z” ya que no está definida. La que si esta y es correcta como indica el error es “dif_z”.

Parte 4

```
C:\ALGO 1>python prodvectorial.py
Traceback (most recent call last):
  File "C:\ALGO 1\prodvectorial.py", line 11, in <module>
    assert mi_funcion(726, 434, 110, 488, 962, 820) == (250060, -541640, 486620)
AssertionError
```

Hay un AssertionError, la prueba de la línea 11 falló.

Es importante renombrar las variables y funciones con nombres declarativos para cuando en un futuro queramos leer la función, sea más fácil comprenderla. Igualmente si queremos compartirla con otras personas, les será más fácil modificarla y adaptarla a su código.

Si se puede escribir el cuerpo en una línea, escribiendo el mismo entre comillas en la primera línea de la función.

Parte 5

Lo importante de reutilizar funciones es que nos permite ahorrar tiempo a la hora de escribir nuestros programas/funciones ya que no debemos repetir algo que en su día ya programamos.