

ALGRITMOS Y PROGRAMACION I

Francisco Nahuel Tapia Maturana

Catedra: Essaya

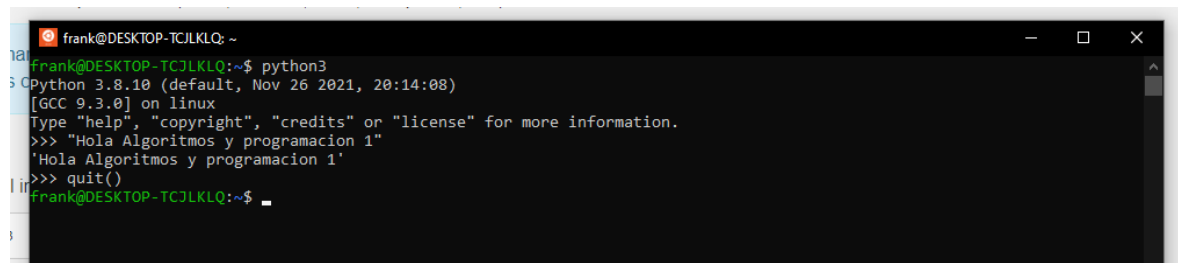
Practica: Bárbara

Padrón: 107128

Ayudante a Cargo: Julián Crespo

PARTE 1

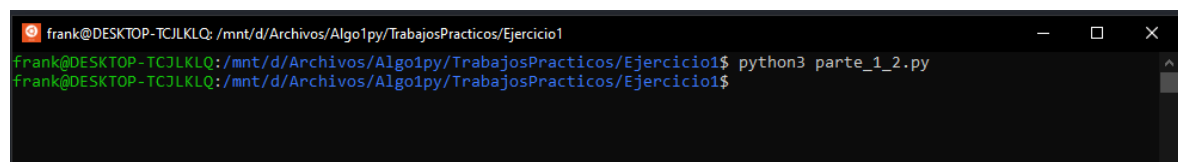
1.1:



```
frank@DESKTOP-TCJLKLQ: ~  
frank@DESKTOP-TCJLKLQ:~$ python3  
Python 3.8.10 (default, Nov 26 2021, 20:14:08)  
[GCC 9.3.0] on linux  
Type "help", "copyright", "credits" or "license" for more information.  
>>> "Hola Algoritmos y programacion 1"  
'Hola Algoritmos y programacion 1'  
>>> quit()  
frank@DESKTOP-TCJLKLQ:~$
```

Interprete de pyhton en la terminal.

1.2:

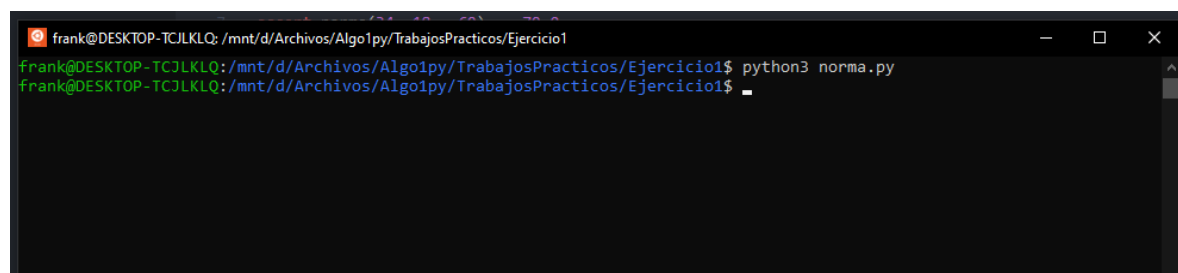


```
frank@DESKTOP-TCJLKLQ: /mnt/d/Archivos/Algo1py/TrabajosPracticos/Ejercicio1  
frank@DESKTOP-TCJLKLQ: /mnt/d/Archivos/Algo1py/TrabajosPracticos/Ejercicio1$ python3 parte_1_2.py  
frank@DESKTOP-TCJLKLQ: /mnt/d/Archivos/Algo1py/TrabajosPracticos/Ejercicio1$
```

Programa ejecutado en la terminal.

- Para conseguir el mismo resultado que en el ejercicio anterior hay que usar print antes de la leyenda que quieres que salga en pantalla.
- En la parte 1.1 vemos el resultado sin necesidad de print ya que abrimos el intérprete de Python en modo interactivo en la terminal y este efectúa la acción al saltar de línea con enter.

PARTE 2



```
frank@DESKTOP-TCJLKLQ: /mnt/d/Archivos/Algo1py/TrabajosPracticos/Ejercicio1  
frank@DESKTOP-TCJLKLQ: /mnt/d/Archivos/Algo1py/TrabajosPracticos/Ejercicio1$ python3 norma.py  
frank@DESKTOP-TCJLKLQ: /mnt/d/Archivos/Algo1py/TrabajosPracticos/Ejercicio1$
```

Programa norma.py con el comentario

```
frank@DESKTOP-TCJLKLQ: /mnt/d/Archivos/Algo1py/TrabajosPracticos/Ejercicio1
frank@DESKTOP-TCJLKLQ: /mnt/d/Archivos/Algo1py/TrabajosPracticos/Ejercicio1$ python3 norma.py
frank@DESKTOP-TCJLKLQ: /mnt/d/Archivos/Algo1py/TrabajosPracticos/Ejercicio1$ python3 norma.py
Traceback (most recent call last):
  File "norma.py", line 17, in <module>
    assert norma(-70, 14, z) == 111.0
AssertionError
frank@DESKTOP-TCJLKLQ: /mnt/d/Archivos/Algo1py/TrabajosPracticos/Ejercicio1$
```

Programa norma.py descomentado

- La salida del programa es un mensaje de assertion error.
- Si podemos saber en qué línea se generó el error ya que el mismo Python nos lo dice en la terminal.
- La instrucción assert lo que hace es verificar que sea correcta la condición que nosotros le indicamos, al no serlo cuando ejecutamos en la terminal sale un mensaje de assertion error.
- Para que no de error "z" tiene que ser (+85) o (-85).

PARTE 3

```
frank@DESKTOP-TCJLKLQ: /mnt/d/Archivos/Algo1py/TrabajosPracticos/Ejercicio1
frank@DESKTOP-TCJLKLQ: /mnt/d/Archivos/Algo1py/TrabajosPracticos/Ejercicio1$ python3 diferencia.py
Traceback (most recent call last):
  File "diferencia.py", line 10, in <module>
    assert diferencia(1, 2, 3, 1, 2, 3) == (0, 0, 0)
  File "diferencia.py", line 6, in diferencia
    return dif_x, dif_y, diff_z
NameError: name 'diff_z' is not defined
frank@DESKTOP-TCJLKLQ: /mnt/d/Archivos/Algo1py/TrabajosPracticos/Ejercicio1$ _
```

Programa diferencia mostrando el error en diff_z

- Si se detectó un error.
- El error fue que en return se escribió mal el nombre con el que anteriormente se defino a z el cual era "dif_z" y en cambio en return se puso "diff_z".
- La línea que estaba fallando era la 6.

PARTE 4

- Muestra un error de tipo AssertionError en la línea 10.
- Se usan nombres representativos para entender mejor la función y ver rápidamente a que pertenece cada variable.
- Si se puede escribir en una sola línea poniendo directamente las operaciones a realizar en el return.

```
1 def mi_funcion(x1, y1, z1, x2, y2, z2):  
2     """Recibe las coordenadas de dos vectores en R3 y devuelve el producto vectorial"""  
3     return y1*z2 - z1*y2, z1*x2 - x1*z2, x1*y2 - y1*x2
```

PARTE 5

- Es muy útil reutilizar funciones para ahorrar tiempo y trabajo.