Análisis de Datos y Toma de Decisiones en Computación
Facultad de Ingeniería de Sistemas Computacionales
Grupo 1IL-131
1er Semestre 2024

Laboratorio #1 – Python Inicial

Asignada: 27 de marzo de 2024
Fecha de Entrega: 3 de abril de 2024 (11:59pm).

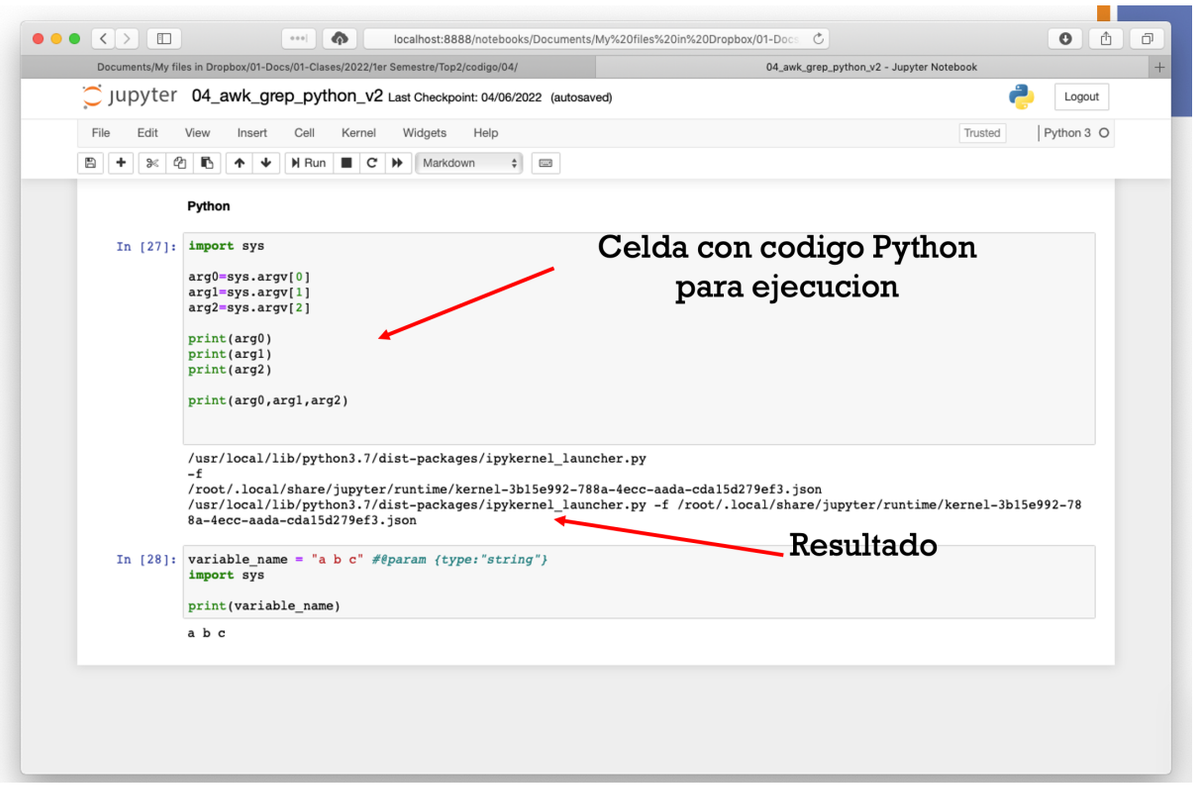
Nota: La tarea debe ser enviada de manera **individual**, a través de la plataforma TEAMS (en un archivo .zip si son múltiples archivos). Provea sus respuestas en formato digital (formato Word o en código Python, o en un Python notebook, formato .ipynb). En cuanto sea posible, sea claro con sus respuestas, e identifique su código en la celda inicial con su nombre y cedula.

Primeros pasos con la Línea de Comandos de Python

Parte esencial para la familiarización y la comprensión en el uso de Python comienza con tener acceso a ella. En esta sección vamos a completar la instalación de la consola de Python.

Similar a la línea de comandos de la Terminal de Windows o de Linux, nosotros podemos introducir comandos directamente en la línea de comandos de Python. El compilador del lenguaje Python viene con su propia su interface de llamada **IDLE** (Integrated Development and Learning Environment). Existen otras opciones como lo son: **Anaconda**[1], **PyCharm**[2]**,** o utilizarlo desde **VisualCode**[3]. En general se usan IDEs vienen pre-instalados con paquetes importantes como lo son: Numpy, Matplotlib y con opciones para instalar otros paquetes con un manejador de paquetes o a través del comando **PIP** y/o **Conda.**

Por defecto en esta clase usare notebooks en **Anaconda Jupyter o Google Colab.** Su interface grafica inicial es como se muestra en la figura de abajo. Por defecto se puede utilizar tanto como **Celdas para la ejecución de Código de Python.**



[1] https://www.anaconda.com/download/
[2] https://www.jetbrains.com/pycharm/
[3] https://code.visualstudio.com/docs/languages/python

Problema 1. Comandos básicos en Python. Una vez tenga instalado su terminal, proceda a introducir los siguientes comandos:

a) print()
b) print
c) print("hola mundo")
d) Print("hola mundo")
e) print()
f) a=5
g) print(a)
h) print('a')
i) a+5
j) print(a)+%5
k) 'import sys; print(sys.version)'
l) import sys; print(sys.version)

Salve la ejecución de estos comandos en su consola personal a través de capturas de pantalla.

Problema 2. Módulos de Python. El lenguaje Python proporciona módulos y librerías dentro de su librería estándar (https://docs.python.org/3/tutorial/stdlib.html) destinadas a permitir a que fácilmente se creen scripts o programas de diversas complejidades.

Escriba un script llamado que realice las siguientes tareas:

1. Obtenga la versión de Python que esté utilizando para realizar esta tarea. ***Consejo:*** verifique la pagina de documentación de Python y en especifico el modulo de sistemas (*sys*) https://docs.python.org/3/library/sys.html

2. Obtenga e imprima la fecha y hora del sistema. ***Consejo:*** importe el modulo de manejo de tiempos (*time*), https://docs.python.org/3/library/time.html

3. Calcule el seno y el coseno de la constante pi. ***Consejo:*** importe el modulo de operaciones matemáticas (*math*), https://docs.python.org/3/library/math.html

4. Como vimos en clase el *os.name* es una cadena que contiene el nombre del sistema operativo en que se está ejecutando el compilador de Python.

Si usted es que se ejecuta en una máquina Unix (o basada en UNIX), dará como resultado "POSIX ". Su script debe verificar el tipo de sistema operativo en donde ejecuta su compilador. ***Consejo***: importe el modulo *os*, https://docs.python.org/3/library/os.html

5. Un punto importante para la simulación de sistemas en computo científico es la generación de números aleatorios. Sele pide que genere un numero aleatorio entre 0.0 y 1.0. ***Consejo***: importe el modulo *random*, https://docs.python.org/3/library/random.html

Problema 3. Lectura de argumentos desde la línea de comandos y su captura en Python. Dentro de las funciones del sistema (*sys*) existe una en especifico que ayuda a manejar las variables u opciones que se le pasan a un script como argumentos, esta es, sys.argv(). Utilizando esta función resuelva las siguientes tareas:

1. Escriba un script llamado que tome dos números como argumentos del usuario e imprima su suma.

Entrada1= 2
Entrada2 = 22

Su programa imprimirá en pantalla: *El total de la suma es 24*

2. Escriba un script llamado que toma dos frases como argumentos del usuario e imprima la concatenación de sus argumentos. Por ejemplo:

Entrada1= “Hello”
Entrada2 = “World”

Su programa imprimirá en pantalla la frase: *HelloWorld*

3. Escriba un script llamado que multiplique dos números flotantes)(con al menos 4 posiciones decimales después del punto),obtenidos como argumentos del usuario e imprima su multiplicación con un formato de flotante de 2 posiciones.

Entrada1= 49.29080
Entrada2 = 12.3456

Su programa imprimirá en pantalla: *El total de la multiplicación es 608.52*

Nota: debe verificar que los números en realidad son flotantes, sino transfórmalos. Además, utilice el formato visto en clase con “{ }” para visualizar el numero requerido de cifras significativas.

Problema 4. Calcular el tiempo de ejecución de una operación. Dentro de las funciones del modulo (*time*) existe una que nos da el tiempo local en el instante justo en que se ejecuta time.time(), que se puede usar para calcular el tiempo que dura una operación, como se muestra en el Script #1.

Script #1

```
import time as t
t1 = t.time()

#Inserte el código aquí!

t2 = t.time()
print('Tiempo de Ejecucion: ',t2 - t1, 'segundos')
```

Utilice el script #1 como base para probar cada uno de los códigos abajo descritos

Código #1	**Código #2**	**Código #3**

Print(“Esto es un mensaje”)	A=5
	B=10
	Print(“total”, a+b)
		def fibonacci(x):
		a, b = 0, 1
		for item in range(x):
		a, b = b, a + b
		return a
		
		x = int(input())
		Print(fibonacci(x))

Reporte los tiempos de ejecución de cada uno de los códigos #1-#3.

Nota: también se puede usar el modulo *datetime.now().* https://docs.python.org/2/library/datetime.html