

Tarea 02

FISI2026-201901

Puede resolver esta tarea trabajando desde el enlace Binder del curso en <https://github.com/ComputoCienciasUniandes/FISI2026-201910>.

Entrega:
Fecha: Jueves 14-feb-2019
Hora: 1:50 pm
Mecanismo único de Entrega: Archivo comprimido en Siciuplus
Puntuación Máxima: 100 puntos.

Lea atentamente toda la tarea antes de empezar a trabajar.

Cree una carpeta que se llame “hw02-” más su usuario de uniandes.
En mi caso se llamaría "hw02-fl.gomez10". Ingrese a ella y trabajaremos todo el tiempo dentro de este directorio.

```
:~$ mkdir hw02-username  
  
:~$ cd hw02-username
```

Trabajo previo a Clase.

Desde la terminal ejecute el intérprete de Python llamando el comando:

```
:~/hw02-username$ python
```

Recuerde que en UNIX se distinguen mayúsculas de minúsculas.
Debería encontrar algo como:

```
jovyan@jupyter-computociencias-2dfisi2026-2d201910-2dnq3vt6qi:~$ mkdir hw02-username  
jovyan@jupyter-computociencias-2dfisi2026-2d201910-2dnq3vt6qi:~$ cd hw02-username/  
jovyan@jupyter-computociencias-2dfisi2026-2d201910-2dnq3vt6qi:~/hw02-username$ ls  
jovyan@jupyter-computociencias-2dfisi2026-2d201910-2dnq3vt6qi:~/hw02-username$ pwd  
/home/jovyan/hw02-username  
jovyan@jupyter-computociencias-2dfisi2026-2d201910-2dnq3vt6qi:~/hw02-username$ python  
Python 3.6.7 | packaged by conda-forge | (default, Nov 21 2018, 03:09:43)  
[GCC 7.3.0] on linux  
Type "help", "copyright", "credits" or "license" for more information.  
>>> █
```

El cursor tienen a su izquierda los símbolos “>>>” que indican que estamos en la consola del intérprete de python. Esto significa que cada orden que escribamos la ejecutará Python, no la terminal de Bash (en la que trabajamos la clase pasada).

Aquí podemos hacer un “Hola mundo!” con la función print.

```
>>> print( "Hola mundo!" )  
Hola mundo!  
>>>
```

podemos crear variables de distintos tipos:

```
>>> a = 3  
>>> b = -2.6  
>>> avogadro = 6.022e23  
>>> texto = "En algún lugar de la Mancha..."
```

imprimirlas:

```
>>> a = 3
>>> b = -2.5
>>> avogadro = 6.022e23
>>> texto = "En algún lugar de la Mancha"
>>>
>>> print(a)
3
>>> print(b)
-2.5
>>> print(avogadro)
6.022e+23
>>> print(texto)
En algún lugar de la Mancha
>>> 
```

Y verificar el tipo de cada variable.

```
>>> type(a)
<class 'int'>
>>> type(b)
<class 'float'>
>>> type(avogadro)
<class 'float'>
>>> type(texto)
<class 'str'>
>>> 
```

Para salir de la consola del intérprete de Python se puede llamar la función “exit()”


```
>>> exit()
```

Volviendo a la terminal podemos crear un archivo “holamundo.py”

```
:~hw02-username/ touch holamundo.py
```

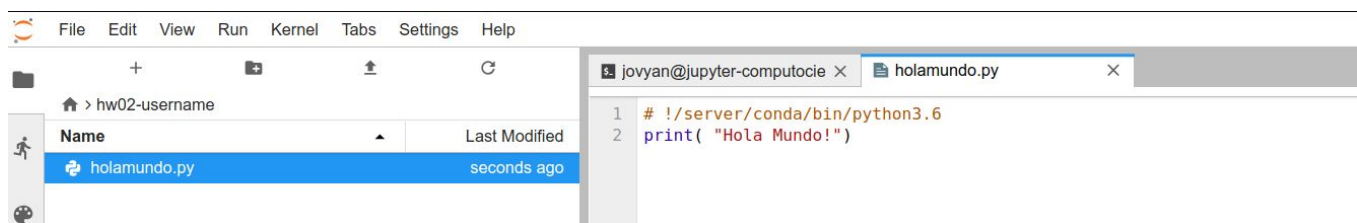
Lo podemos editar desde la consola con “vi” o “nano”, o bien, abriendo el editor de binder haciendo doble click en el nombre del archivo recién creado.

🏠 > hw02-username

Name	Last Modified
 holamundo.py	a minute ago

```
>>> print( "Hola mundo!" )
Hola mundo!
>>>
>>> a = 3
>>> b = -2.5
>>> avogadro = 6.022e23
>>> texto = "En algún lugar de la Mancha"
>>>
>>> print(a)
3
>>> print(b)
-2.5
>>> print(avogadro)
6.022e+23
>>> print(texto)
En algún lugar de la Mancha
>>>
>>> type(a)
<class 'int'>
>>> type(b)
<class 'float'>
>>> type(avogadro)
<class 'float'>
>>> type(texto)
<class 'str'>
jovyan@jupyter-computociencias-2dfisi2026-2d201910-2dnq3vt6qi:~/hw02-username$
jovyan@jupyter-computociencias-2dfisi2026-2d201910-2dnq3vt6qi:~/hw02-username$ tou
h holamundo.py
jovyan@jupyter-computociencias-2dfisi2026-2d201910-2dnq3vt6qi:~/hw02-username$ 
```

El tener el editor y la consola en ventanas separadas puede agilizar las cosas.



No olvide guardar con “ctr + s”.

Puede cambiar rápidamente de pestaña con los atajos del teclado

“ ctrl + shift + [”

para ir a la siguiente pestaña, para ir a la anterior se usa:

“ ctrl + shift + { ”

Al volver a la terminal se ejecuta el script de python “holamundo.py” invocando previamente el intérprete de python. En este caso el intérprete ejecuta el script y al terminar la consola queda libre. No es necesario salir de python.

```
jovyan@jupyter-computociencias-2dfisi2026-2d201910-2dnq3vt6qi:~/hw02-username$ python holamundo.py
Hola Mundo!
jovyan@jupyter-computociencias-2dfisi2026-2d201910-2dnq3vt6qi:~/hw02-username$
```

Python 3.6 vs Python 2.7

Grosso modo, la operación de división cambia, junto con la función “print”.

Nombres de Variables

Los nombres de variables pueden ser combinaciones de letras y números. No pueden empezar por un número. Hay ciertos caracteres reservados que no pueden ser parte del nombre de una variable

(. , \ ^ : + - * / % & |)

Válido d = 37.5

Válido D = 37.5

Válido: Data_03_Oct_2019 = 37.5

No Válido: 2019_03_Oct_Data = 37.5

No válido: 2019-03-Oct_Data = 37.5

Python lo asume como “2019 menos 3 menos Oct_Data igual a 37.5” y arroja error.

Una sintaxis clara y bonita:

Dejar un espacio antes y después de cada operador matemático y lógico (+ - * / % & |)

En ciclos for, while y en definición de funciones dejar una sangría, o indentar con cuatro espacios.

¡4 espacios!

¡4 espacios!

¡4 espacios!

Esto ahorrará muchos dolores de cabeza.

Ejercicio 1

(20 puntos)

Cree un script de python llamado “ejercicio01.py” que realice las siguientes operaciones:

```
>>> a = 13
>>> b = 5
>>> c = True
>>> d = False
>>> a / b
>>> # comentario sobre la operación.
>>> a // b
>>> # comentario sobre la operación.
>>> a % 5
>>> # comentario sobre la operación.
>>> c = a + b
>>> # comentario sobre la operación.
>>> c += b
>>> # comentario sobre la operación.
>>> c *= b
>>> # comentario sobre la operación.
>>> c & d
>>> # comentario sobre la operación.
>>> c | d
>>> # comentario sobre la operación.
>>> c & c
>>> # comentario sobre la operación.
>>> d | d
>>> # comentario sobre la operación.
```

(20pt) Como comentario dentro del código (usando “#” para iniciar una línea comentada) explique qué realiza cada operación.

Ejercicio 2

(20 puntos)

En un script llamado “ejercicio02.py” cree las variables

```
>>> a = 3.1416
>>> b = “Una aproximación del valor de Pi es “
>>> c = True
>>> d = False
>>> e = 1
>>> f = 0
```

(7pt) Usando “type()” indique qué tipo de variables son.

(7pt) Comente (usando “#”) cuál es el resultado de estas operaciones, cuándo funcionan y cuándo fallan.

```
b + a
int(b) + int(a)
float(b) + float(a)
str(b) + str(a)
c + e
d | e
bool(a)
bool(b)
bool(f)
```

(6pt) Escriba en el mismo código una variable `h` tipo string tal que

```
>>> bool(h)
False
```

Ejercicio 3

(20 puntos)

Cree un archivo llamado "ejercicio03.py" que:

- (10pt) use un ciclo “for” para imprimir los impares positivos menores que 20
- (10pt) use un ciclo “while” para imprimir los pares positivos menores que 20

Ejercicio 4

(20 puntos)

Cree un archivo llamado "ejercicio04.py" que:

- (10pt) Indique si un número entero guardado como la variable “number” es un número primo.
- (10pt) Al ejecutarse desde la terminal, pueda tener argumentos de entrada, esto lo veremos en clase.

```
:~$ python ejercicio04.py 2017
2017 es primo
:~$
```

Tal vez sea útil usar `np.range(x)`, con `x` un número entero.

Ejercicio 5

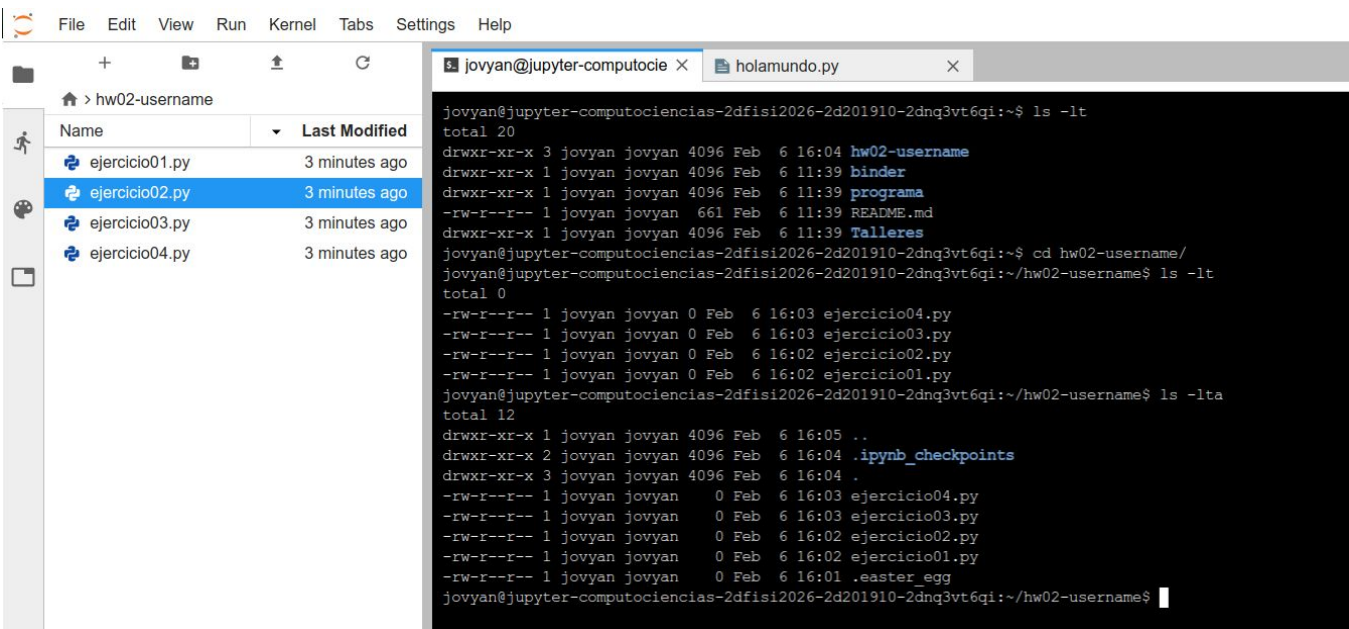
(20 puntos)

- (10pt) **No hay ejercicio05.py.**

Para saber si leyeron por completo esta guía antes de empezar a responder, el primer archivo que deben crear dentro de la carpeta “hw02-username” en clase se llamará “.easter_egg”.

```
mkdir hw02-username
cd hw02-username
ls -lt
touch .easter_egg
ls -al
```

Con esto podemos revisar en qué momento crearon los archivos.



- (10pt) Comprima la carpeta en formato tar.

Al terminar de trabajar, debe tener cuatro archivos

```
tar -czvf hw02-username.tar hw02-username
```

Entregue este archivo comprimido.

