



109- [PF] - Lab - Tipos de datos numéricos

Trabajo con tipos de datos numéricos

Información general sobre el laboratorio

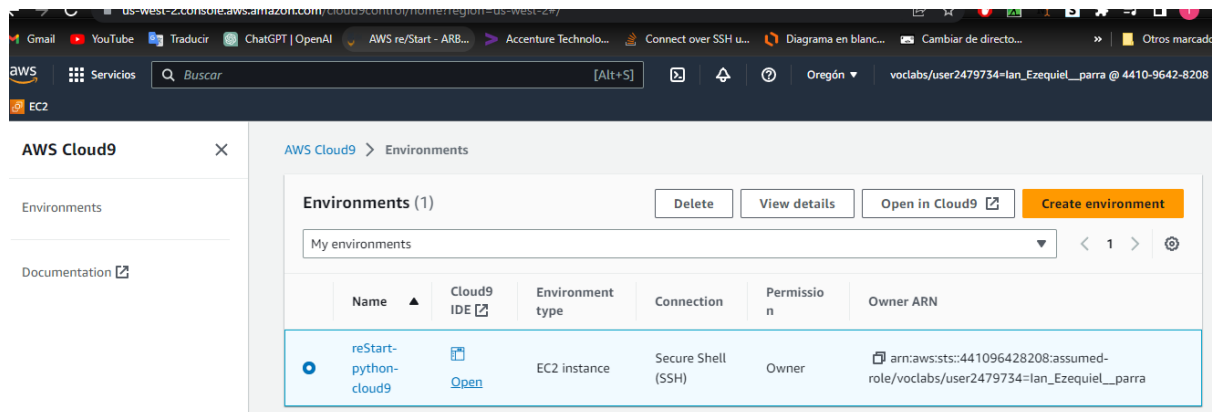
Python simplifica las matemáticas. De hecho, Python es un lenguaje popular entre los científicos de datos, quienes deben analizar grandes cantidades de datos. En este laboratorio, explorará los tipos de datos básicos que se utilizan para almacenar valores numéricos.

En este laboratorio, deberá realizar lo siguiente:

- utilizar el shell de Python
 - utilizar el tipo de dato int (entero)
 - utilizar el tipo de dato float (flotante)
 - utilizar el tipo de dato complex (complejo)
 - utilizar el tipo de dato bool (booleano)
-

Tiempo de finalización estimado

Acceso al IDE de AWS Cloud9



1. Para activar el entorno de su laboratorio, desplácese hasta la parte superior de estas instrucciones y seleccione **Start Lab** (Iniciar laboratorio).

Se abrirá el panel **Start Lab** (Iniciar laboratorio), donde se muestra el estado del laboratorio.

2. Espere hasta que aparezca el mensaje *Lab status: ready* (Estado del laboratorio: listo) y, a continuación, para cerrar el panel **Start Lab** (Iniciar laboratorio), haga clic en la **X**.

3. En la parte superior de estas instrucciones, elija **AWS**.

AWS Management Console se abrirá en una pestaña nueva del navegador. El sistema iniciará la sesión de forma automática.

Nota: Si no se abre una pestaña nueva del navegador, generalmente aparece un anuncio o un icono en la parte superior de este, el cual indica que el navegador no permite que se abran ventanas emergentes en el sitio. Haga clic en el anuncio o en el icono, y elija Allow pop ups (Permitir ventanas emergentes).

4. En AWS Management Console, elija **Services** (Servicios) > **Cloud9**. En el panel **Your environments** (Sus entornos), busque la tarjeta **reStart-python-cloud9** y elija **Open IDE** (Abrir IDE).

Se abre el entorno de AWS Cloud9.

Nota: Si se abre una ventana emergente con el mensaje `.c9/project.settings have been changed on disk` (Se ha modificado la configuración de `.c9/project` en el disco), elija **Discard** (Descartar) para ignorarlo. Del mismo modo, si una ventana de diálogo le pide que **Show third-party content** (Muestre contenido de terceros), elija **No** para rechazar la indicación.

Creación del archivo de ejercicio de Python

1. En la barra de menú, elija **File > New From Template > Python File** (Archivo > Nuevo a partir de plantilla > Archivo en Python).
Esta acción crea un archivo sin título.
2. Elimine el código de muestra del archivo de plantilla.
3. Elija **File > Save As...** (Archivo > Guardar como...), proporcione un nombre adecuado para el archivo de ejercicio (por ejemplo, *numeric-data.py*) y guárdelo en el directorio **/home/ec2-user/environment**.

Acceso a la sesión del terminal

1. En su IDE de AWS Cloud9, elija el icono **+** y seleccione **New Terminal** (Nuevo terminal).
Se abre una sesión de terminal.
2. Para ver el directorio en el que está trabajando actualmente, escriba `pwd`. Este comando lleva a **/home/ec2-user/environment**.
3. En este directorio, también debería ser capaz de localizar el archivo que creó en la sección anterior.

Ejercicio 1: Utilizar el shell de Python

En la pestaña del terminal, se puede iniciar un shell de Python escribiendo el siguiente comando:

```
python3voclabs:~/environment $ python3
Python 3.7.16 (default, Mar 10 2023, 03:25:26)
[GCC 7.3.1 20180712 (Red Hat 7.3.1-15)] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> █
```

```
python3
```

El shell de Python debería verse similar al siguiente ejemplo.

```
Python 3.6.12 (default, Aug 31 2020, 18:56:18)
[GCC 4.8.5 20150623 (Red Hat 4.8.5-28)] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

Los tres símbolos para “mayor que” (`>>>`) representan el símbolo del sistema donde el usuario puede escribir comandos de Python. En las siguientes actividades, practicará el uso del shell de Python mediante algunos comandos numéricos.

Adición

1. Escriba la siguiente entrada:

```
2 + 2
```

1. Presione ENTER (Intro).
2. Confirme que obtiene `4` como salida.

Sustracción

1. Escriba la siguiente entrada

```
4 - 2
```

1. Presione ENTER (Intro).
2. Confirme que obtiene `2` como salida.

Multiplicación

Para multiplicar, utilice el símbolo `*`:

1. Escriba la siguiente entrada:

```
2 * 2
```

1. Presione ENTER (Intro).
2. Confirme que obtiene *4* como salida.

División

Para dividir, utilice el símbolo `/`:

1. Escriba la siguiente entrada:

```
4 / 2
```

1. Presione ENTER (Intro).
2. Confirme que obtiene *2.0* como salida.

Salida del shell de Python

1. Para salir del shell de Python, escriba el siguiente comando:

```
quit()
```

```
python3voclabs:~/environment $ python3
python3voclabs:~/environment $ python3
Python 3.7.16 (default, Mar 10 2023, 03:25:26)
[GCC 7.3.1 20180712 (Red Hat 7.3.1-15)] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> 2 + 2
4
>>> 4 - 2
2
>>> 2 + 2
4
>>> 2 + 2
4
>>> 2 / 2
1.0
>>> quit()
voclabs:~/environment $
```

Ejercicio 2: Presentar el tipo de dato entero

Para obtener más información sobre los tipos de datos, utilizará algunas funciones integradas. Una *función* es un fragmento de código reutilizable con un nombre. Utiliza una función mediante las siguientes acciones:

- llamarla por su nombre
- incluir una lista de una o más entradas llamadas *argumentos*, que se encuentran entre paréntesis

Python tiene varias funciones integradas que puede utilizar para escribir programas más útiles.

Un conjunto de funciones se denomina *biblioteca*. El conjunto de funciones integradas de Python se denomina *Biblioteca estándar de Python*.

Edición de un archivo en Python

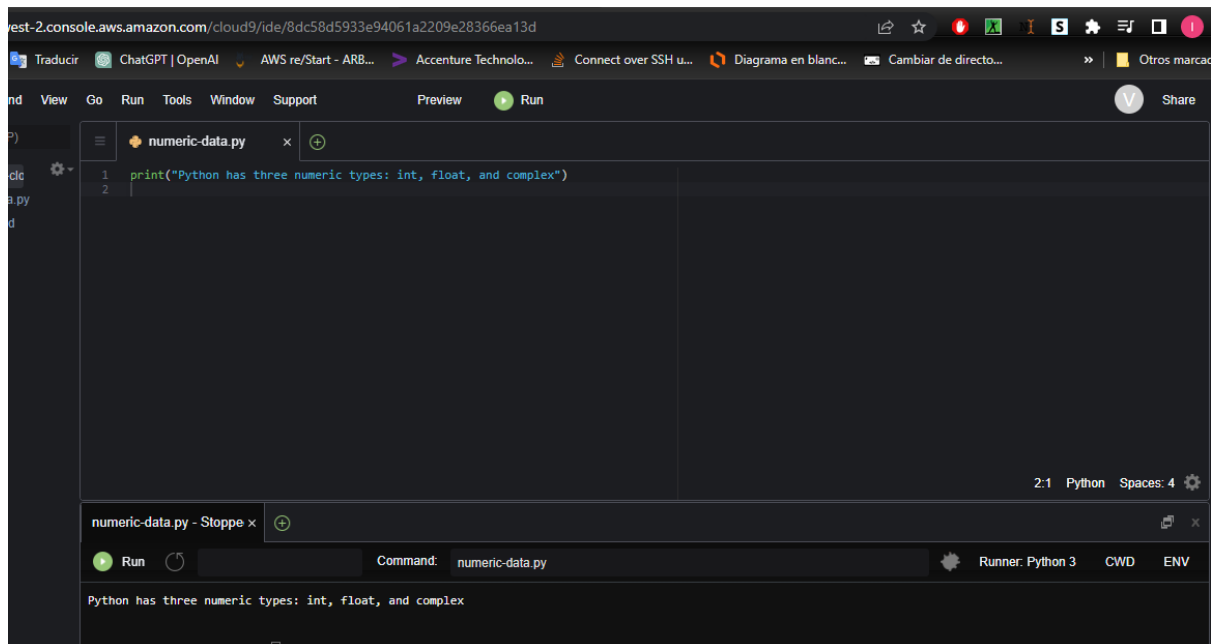
En vez de escribir los comandos uno por uno en el shell de Python, editará un archivo de texto que contiene una secuencia de comandos:

1. En el panel de navegación del IDE, elija el archivo que creó en la sección *Creación del archivo de ejercicios de Python* anterior.
2. En el archivo, escriba el siguiente código:

```
print("Python has three numeric types: int, float, and complex")
```

1. Para guardar el archivo, elija **File > Save** (Archivo > Guardar).

2. En la parte superior de la ventana del IDE, elija **Run** (Ejecutar) (el botón **Play** [Reproducir]).
3. En el panel inferior (consola) del IDE, confirme que el programa imprime el siguiente mensaje: *Python has three numeric types: int, float, and complex* (Python tiene tres tipos numéricos: entero, flotante y complejo).



Nota: Es posible que deba desplazarse hacia arriba para ver la salida de la consola.

1. En la pestaña del terminal, también puede ejecutar el programa escribiendo el siguiente comando, donde *<lab-python-file-name>* es el nombre del archivo que creó para este laboratorio:

```
python3 <lab-python-file-name>.py
```

1. Confirme que el texto que introdujo esté escrito como una salida estándar.

```
~ $ python3 <lab-python-file-name>.py
Python has three numeric types: int, float, and complex
```

Creación de una variable

Una variable es como una caja etiquetada que almacena información. Puede cambiar el contenido de la caja, pero la etiqueta seguirá siendo la misma. En esta

actividad, utilizará el nombre de variable *myValue*, pero almacenará diferentes tipos de datos en esa caja etiquetada.

1. Vuelva al archivo en Python y, en una nueva línea, escriba el siguiente código:

```
myValue=1
```

1. Utilice la función `print()` para escribir el valor de la variable en el shell: En el contexto de la programación, *escribir* significa agregar información en el shell.

```
print(myValue)
```

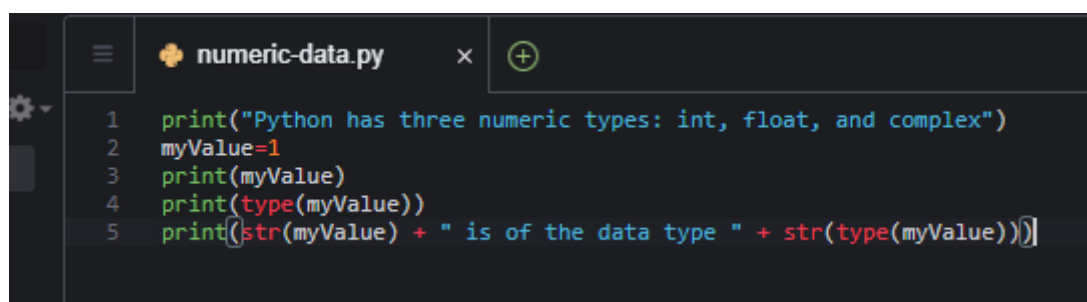
1. Para obtener el tipo de dato en la variable, utilice la función integrada `type()` :

```
print(type(myValue))
```

1. Para combinar números y texto, utilice la función integrada `str()` , la cual convierte un argumento en un conjunto de letras denominado *cadena*. En este caso, está convirtiendo el tipo de dato `int` (entero) al tipo de dato de *cadena*:

```
print(str(myValue) + " is of the data type " + str(type(myValue)))
```

1. Guarde el archivo.



```
1 print("Python has three numeric types: int, float, and complex")
2 myValue=1
3 print(myValue)
4 print(type(myValue))
5 print(str(myValue) + " is of the data type " + str(type(myValue)))
```

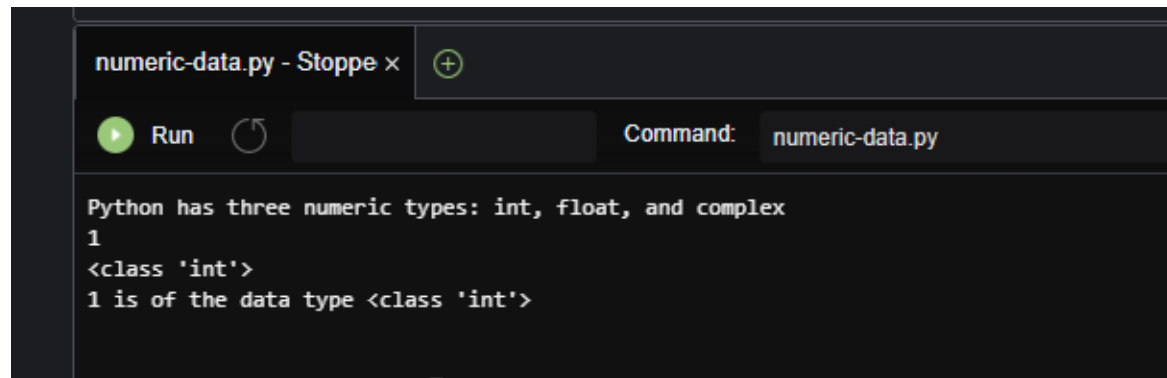
1. Para ejecutar el archivo, elija **Run** (Ejecutar).
2. En el panel inferior del IDE, confirme que tiene la siguiente salida:

```
Python has three numeric types: int, float, and complex
1
<class 'int'>
```



```
1 is of the data type <class 'int'>
~ $
```

Nota: Es posible que deba desplazarse hacia arriba para ver la salida.

A screenshot of a terminal window titled 'numeric-data.py - Stoppe x'. The window has a 'Run' button and a 'Command:' field containing 'numeric-data.py'. The output text is: 'Python has three numeric types: int, float, and complex', '1', '<class 'int'>', and '1 is of the data type <class 'int'>'.

```
numeric-data.py - Stoppe x
Python has three numeric types: int, float, and complex
1
<class 'int'>
1 is of the data type <class 'int'>
```

Ejercicio 3: Presentar el tipo de dato flotante

El tipo de dato entero solo almacena números enteros. Si desea almacenar un número con decimales, como 3.14, necesita un nuevo tipo de dato denominado *flotante*.

1. Vuelva al archivo en Python y, en una nueva línea, escriba el siguiente código:

```
myValue=3.14
```

1. Para escribir el valor de la variable en el shell, utilice la función `print()`:

```
print(myValue)
```

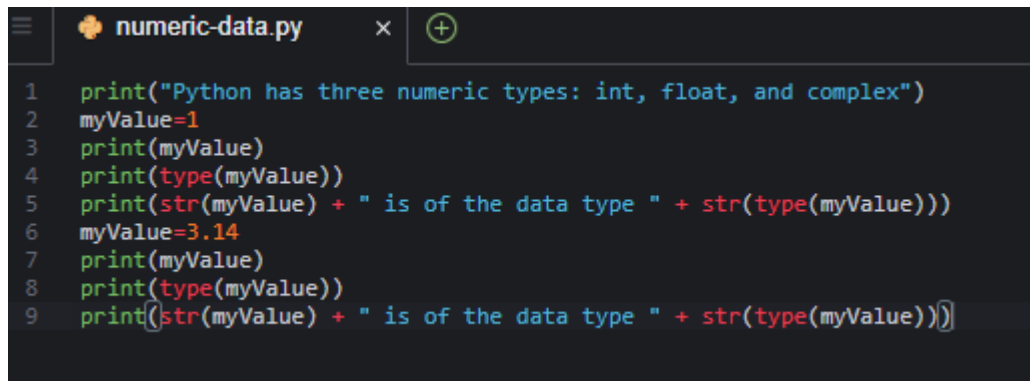
1. Obtenga el tipo de dato en la variable con la función integrada `type()`:

```
print(type(myValue))
```

1. Para combinar números y texto, utilice la función integrada `str()`:

```
print(str(myValue) + " is of the data type " + str(type(myValue)))
```

1. Guarde el archivo.
- 2.

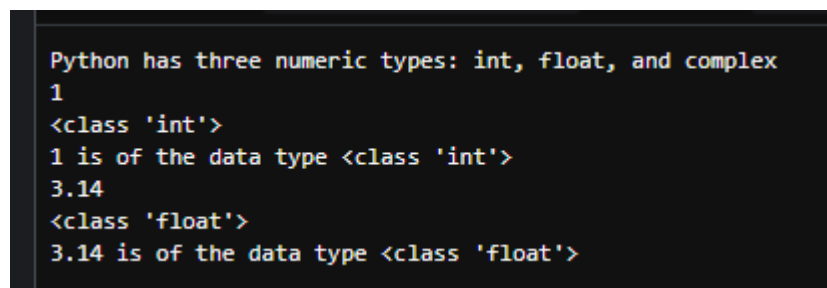


```
1 print("Python has three numeric types: int, float, and complex")
2 myValue=1
3 print(myValue)
4 print(type(myValue))
5 print(str(myValue) + " is of the data type " + str(type(myValue)))
6 myValue=3.14
7 print(myValue)
8 print(type(myValue))
9 print(str(myValue) + " is of the data type " + str(type(myValue)))
```

1. Para ejecutar el archivo, elija **Run** (Ejecutar).
2. En el panel inferior del IDE, confirme que ve la siguiente salida:

```
Python has three numeric types: int, float, and complex
1
<class 'int'>
1 is of the data type <class 'int'>
3.14
<class 'float'>
3.14 is of the data type <class 'float'>
~ $
```

Nota: Recuerde que es posible que deba desplazarse hacia arriba para ver la salida.



```
Python has three numeric types: int, float, and complex
1
<class 'int'>
1 is of the data type <class 'int'>
3.14
<class 'float'>
3.14 is of the data type <class 'float'>
```

Ejercicio 4: Presentar el tipo de dato complejo

En la matemática avanzada, un número imaginario es un número complejo, el cual se puede escribir como un número real que se multiplica por la unidad imaginaria i . El tipo de dato complejo es complicado porque tiene que representar una letra y un número, como $5j$.

1. Vuelva al archivo en Python e escriba el siguiente código:

```
myValue=5j
```

1. Escriba el valor de la variable con la función `print()`:

```
print(myValue)
```

1. Obtenga el tipo de dato en la variable con la función `type()`:

```
print(type(myValue))
```

1. Para combinar números y texto, utilice la función integrada `str()`:

```
print(str(myValue) + " is of the data type " + str(type(myValue)))
```

1. Guarde el archivo.

```
myValue=5j
print(myValue)
print(type(myValue))
print(str(myValue) + " is of the data type " + str(type(myValue)))
```

2. Para ejecutar el archivo, elija **Run** (Ejecutar).

3. En el panel inferior del IDE, confirme que tiene la siguiente salida:

```
Python has three numeric types: int, float, and complex
1
<class 'int'>
1 is of the data type <class 'int'>
3.14
<class 'float'>
```

```
3.14 is of the data type <class 'float'>
5j
<class 'complex'>
5j is of the data type <class 'complex'>
~ $
```

Nota: Recuerde que es posible que deba desplazarse hacia arriba para ver la salida.

```
5j
<class 'complex'>
5j is of the data type <class 'complex'>
```

Ejercicio 5: Presentar el tipo de dato booleano

El tipo de datos bool (booleano) comprende los nombres permanentes *True* (Verdadero) y *False* (Falso), que se representan mediante los numerales *1* y *0*, donde *1* = *Verdadero* y *0* = *Falso*. El tipo de dato booleano se implementa como un subconjunto de los enteros y no se considera un tipo de dato real. Sin embargo, en algunos lenguajes de programación, se implementa como un tipo de dato diferente. Estos ejercicios denominan al tipo booleano de Python un *tipo de dato simulado*.

1. Vuelva al archivo de texto y escriba el siguiente código:

```
myValue=True
```

1. Escriba el valor de la variable en el shell con la función `print()`:

```
print(myValue)
```

1. Obtenga el tipo de dato en la variable con la función integrada `type()`:

```
print(type(myValue))
```

1. Para combinar números y texto, utilice la función integrada `str()`:

```
print(str(myValue) + " is of the data type " + str(type(myValue)))
```

1. Guarde el archivo.

```
myValue=True
print(myValue)
print(type(myValue))
print(str(myValue) + " is of the data type " + str(type(myValue)))
```

1. Elija **Run** (Ejecutar) (el botón **Play** [Reproducir]).
2. En el panel inferior del IDE, confirme que se muestra la salida correcta.
3. Vuelva al archivo **.py** y escriba el siguiente código:

```
myValue=False
```

1. Utilice la función `print()` para escribir el valor de la variable en el shell:

```
print(myValue)
```

1. Para obtener el tipo de dato en la variable, utilice la función integrada `type()` :

```
print(type(myValue))
```

1. Para combinar números y texto, utilice la función integrada `str()` :

```
print(str(myValue) + " is of the data type " + str(type(myValue)))
```

```
1 myValue=True
2 print(myValue)
3 print(type(myValue))
4 print(str(myValue) + " is of the data type " + str(type(myValue)))
5 myValue=False
6 print(myValue)
7 print(type(myValue))
8 print(str(myValue) + " is of the data type " + str(type(myValue)))
```

1. Guarde el archivo.
2. Elija **Run** (Ejecutar) (el botón **Play** [Reproducir]).
3. En el panel inferior del IDE, confirme que tiene la siguiente salida:

```
Python has three numeric types: int, float, and complex
1
<class 'int'>
1 is of the data type <class 'int'>
3.14
<class 'float'>
3.14 is of the data type <class 'float'>
5j
<class 'complex'>
5j is of the data type <class 'complex'>
True
<class 'bool'>
True is of the data type <class 'bool'>
False
<class 'bool'>
False is of the data type <class 'bool'>
~ $
```

```
True
<class 'bool'>
True is of the data type <class 'bool'>
False
<class 'bool'>
False is of the data type <class 'bool'>
```

¡Felicitaciones! Ha aprendido acerca de los tres tipos de datos numéricos de Python: entero, flotante y complejo. Además, recibió información introductoria sobre el tipo de dato simulado de Python denominado *booleano*. Tenga en cuenta que el tipo booleano corresponde, en realidad, a los números *0* y *1*, que representan los valores *True* (Verdadero) y *False* (Falso).____