Ciclo 1
Fundamentos de programación en Python

Sesión 1: Introducción lenguaje de programación Python

Programa Ciencias de la Computación e Inteligencia Artificial

Escuela de Ciencias Exactas e Ingeniería

Universidad Sergio Arboleda

Bogotá

Agenda

1. Introducción a Python
2. Introducción a Colab como entorno de desarrollo
3. Aspecto generales de Python
4. Ejercicios

1. ¿Qué es Python?

Python es un popular lenguaje de programación. Fue creado por Guido van Rossum y publicado en 1991.

Se utiliza para:

- El desarrollo web (del lado del servidor),
- Desarrollo de software,
- Matemáticas,
- Scripting de sistemas.

1.1 ¿Qué puede hacer Python?

- Python puede utilizarse en un servidor para crear aplicaciones web.
- Python se puede utilizar junto con el software para crear flujos de trabajo.
- Python puede conectarse a sistemas de bases de datos. También puede leer y modificar archivos.
- Python puede utilizarse para manejar big data y realizar matemáticas complejas.
- Python puede utilizarse para la creación rápida de prototipos o para el desarrollo de software listo para la producción.

1.2 ¿Por qué Python?

- Python funciona en diferentes plataformas (Windows, Mac, Linux, Raspberry Pi, etc).
- Python tiene una sintaxis sencilla similar a la del idioma inglés.
- Python tiene una sintaxis que permite a los desarrolladores escribir programas con menos líneas que otros lenguajes de programación.
- Python se ejecuta en un sistema de interpretación, lo que significa que el código puede ejecutarse tan pronto como se escribe. Esto significa que la creación de prototipos puede ser muy rápida.
- Python puede tratarse de forma procedimental, orientada a objetos o funcional.

1.3 Es bueno saber

- La versión principal más reciente de Python es Python 3. Sin embargo, Python 2, a pesar de que no se actualiza más que las actualizaciones de seguridad, sigue siendo bastante popular.
- En este curso se escribirá Python en una plataforma on line (Colab de Google).
- Es posible escribir Python en un Entorno de Desarrollo Integrado, como Thonny, Pycharm, Netbeans o Eclipse, que son particularmente útiles cuando se manejan grandes colecciones de archivos Python.

1. 4 Sintaxis de Python comparada con otros lenguajes de programación

- Python fue diseñado para ser legible, y tiene algunas similitudes con el idioma inglés con influencia de las matemáticas.
- Python utiliza nuevas líneas para completar un comando, a diferencia de otros lenguajes de programación que suelen utilizar punto y coma o paréntesis.
- Python se basa en la indentación, utilizando espacios en blanco, para definir el ámbito de aplicación, como el de los bucles, las funciones y las clases. Otros lenguajes de programación suelen utilizar corchetes para este fin.

```
print("Hello, World!")
```

2. Colab como entorno de desarrollo

¿Que es Colab?

Google Colab, también conocido como Collaboratory, es un entorno gratuito basado en Jupyter notebook que se ejecuta en los servidores de la nube de Google. No requiere instalación ni configuración de Python. Se tiene acceso gratuito a hardware CPU, GPU. Se puede compartir el código de manera similar que con Google Drive.

2. Colab como entorno de desarrollo

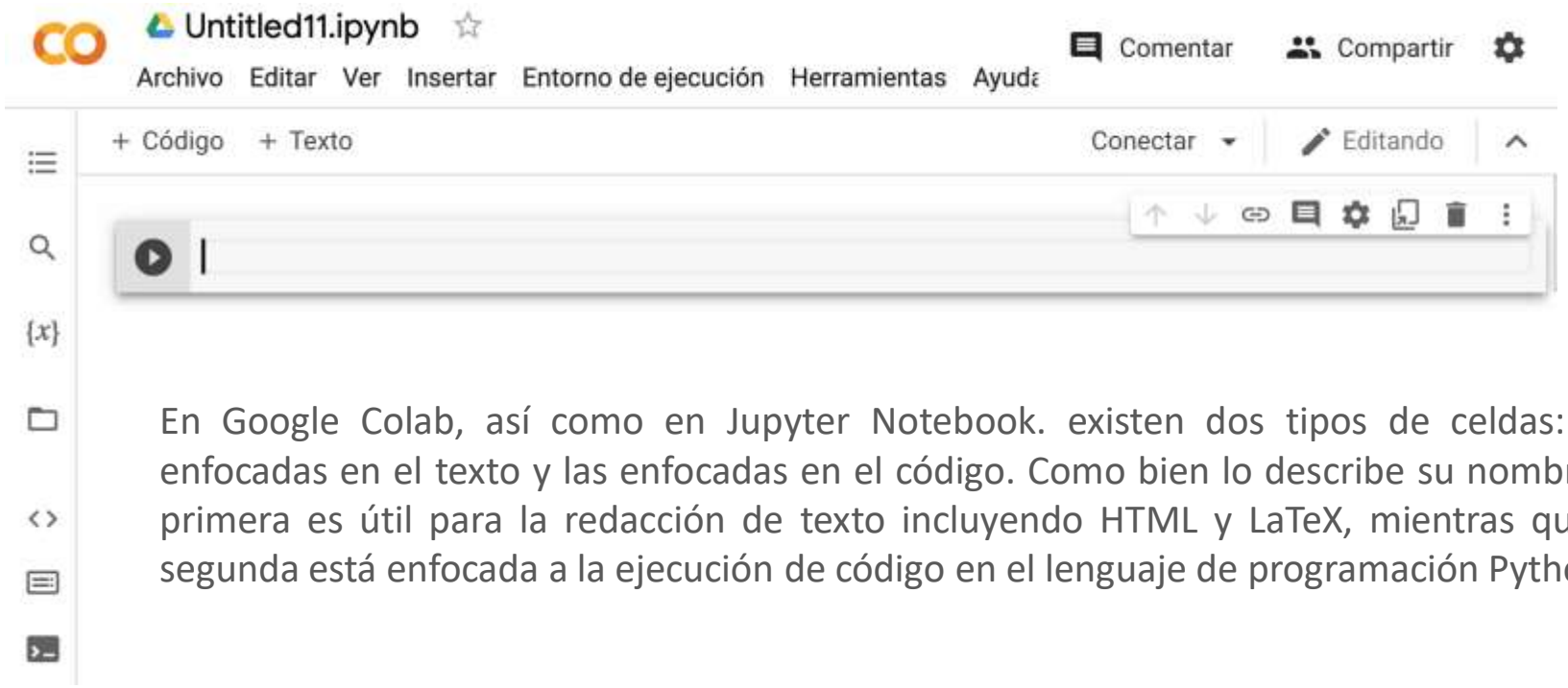
¿Cómo acceder a Colab?

Para acceder a este servicio lo primero que se requiere es una cuenta en Gmail. El correo del proyecto MINTIC esta en el dominio de colab.

Una vez que tienes la cuenta, hay que ingresar a ella y luego, en otra ventana del navegador, ir a la siguiente dirección https://colab.research.google.com/. Lo anterior te llevará a la página de inicio de Collaboratory y en ella encontrarás un excelente tutorial para iniciar

2.1 Aspectos básicos de una notebook en Colab

Cuando se inicia una notebook obtendremos una vista como la siguiente:



En Google Colab, así como en Jupyter Notebook. existen dos tipos de celdas: Las enfocadas en el texto y las enfocadas en el código. Como bien lo describe su nombre la primera es útil para la redacción de texto incluyendo HTML y LaTeX, mientras que la segunda está enfocada a la ejecución de código en el lenguaje de programación Python.

2.1 Aspectos básicos de una notebook en Colab

Para agregar una celda de texto será necesario dar clic en el siguiente botó:

Para agregar una celda de código será necesario dar clic en el botón siguiente:

2.1 Aspectos básicos de una notebook en Colab

La acción anterior insertará una celda en la cual podrás escribir y ejecutar código, de la siguiente manera:

3.1 Indentación en Python

- La sangría se refiere a los espacios al principio de una línea de código.
- Mientras que en otros lenguajes de programación la sangría en el código es sólo para la legibilidad, la sangría en Python es muy importante.
- Python utiliza la sangría para indicar un bloque de código.

```
if 5 > 2:
  print("Cinco es mayor a 2!")
```

- Python te dará un error si te saltas la sangría:

```
if 5 > 2:
print("Cinco es mayor a 2!")

  File "<ipython-input-5-1cf66c84ec30>", line 2
    print("Cinco es mayor a 2!")
        ^
IndentationError: expected an indented block
```

3.2 Variable en Python

Una variable puede tener un nombre corto (como x e y) o un nombre más descriptivo (edad, nombre del coche, volumen_total).

Reglas para las variables de Python:

- Un nombre de variable debe comenzar con una letra o el carácter de subrayado
- Un nombre de variable no puede empezar por un número
- Un nombre de variable sólo puede contener caracteres alfanuméricos y guiones bajos (A-z, 0-9 y _)
- Los nombres de las variables distinguen entre mayúsculas y minúsculas (edad, Age y AGE son tres variables diferentes)

3.2 Variable en Python

Muchos valores a múltiples variables

Python permite asignar valores a múltiples variables en una sola línea:

```
x, y, z = "Azul", "Rojo", "Verde"
print(x)
print(y)
print(z)
```

```
Azul
Rojo
Verde
```

3.3 Variable en Python

Salida de variables

La función ***print()*** de Python se utiliza a menudo para dar salida a las variables.

```
x = "Python es un lenguaje"
print(x)
```

```
Python es un lenguaje
```

```
x = "Python"
y = "es un"
z = "lenguaje"
print(x, y, z)
```

```
Python es un lenguaje
```

```
x = "Python "
y = "es un"
z = " lenguaje"
print(x+y+z)
```

```
Python es un lenguaje
```

3.4 Tipos de datos en Python

- En programación, el tipo de datos es un concepto importante.
- Las variables pueden almacenar datos de diferentes tipos, y diferentes tipos pueden hacer diferentes cosas.
- Python tiene los siguientes tipos de datos incorporados por defecto, en estas categorías:

Tipos de texto	`str`
Tipos numéricos	`int`, `float`, `complex`
Tipos de secuencia	`list`, `tuple`, `range`
Tipo de mapeo	`dict`
Tipos de conjuntos	`set`, `frozenset`
Tipo booleano	`bool`
Tipos binarios	`bytes`, `bytearray`, `memoryview`

3.4 Tipos de datos en Python

A continuación se describen alguos tipos de datos que se manejan en Python:

```
#Tipos de datos
a=1
b=1.2
c='Hola'
d=True
e=[1,2,3,4,5]
```

```
print(a, type(a))
print(b, type(b))
print(c, type(c))
print(d, type(d))
print(e, type(e))
```

```
1 <class 'int'>
1.2 <class 'float'>
Hola <class 'str'>
True <class 'bool'>
[1, 2, 3, 4, 5] <class 'list'>
```

3.5 Numeros en Python

A continuación se describen alguos tipos de datos que se manejan en Python:

```
x = 1     # int
y = 2.8   # float
z = 1j    # complex
print(type(x))
print(type(y))
print(type(z))
```

```
<class 'int'>
<class 'float'>
<class 'complex'>
```

```
x = 35e3
y = 12E4
z = -87.7e100

print(type(x))
print(type(y))
print(type(z))
```

```
<class 'float'>
<class 'float'>
<class 'float'>
```

```
x = 3+5j
y = 5j
z = -5j
print(type(x))
print(type(y))
print(type(z))
```

```
<class 'complex'>
<class 'complex'>
<class 'complex'>
```

3.5 Numeros en Python

Conversión de tipo

```
x = 1    # int
y = 2.8  # float
z = 1j   # complex
#convertir a float:
a = float(x)
#convertir de float a int:
b = int(y)
#convertir de int a complejo:
c = complex(x)
```

```
print(a)
print(b)
print(c)
print(type(a))
print(type(b))
print(type(c))
```

```
1.0
2
(1+0j)
<class 'float'>
<class 'int'>
<class 'complex'>
```

3.6 Booleanos en Python

- En programación a menudo necesitas saber si una expresión es Verdadera o Falsa.
- Puedes evaluar cualquier expresión en Python, y obtener una de las dos respuestas, Verdadero o Falso.
- Cuando comparas dos valores, la expresión se evalúa y Python devuelve la respuesta booleana:

```
print(10 > 9)
print(10 == 9)
print(10 < 9)
```

```
True
False
False
```

```
print(bool("Hola"))
print(bool(15))
```

```
True
True
```

```
x = "Hello"
y = 15

print(bool(x))
print(bool(y))
```

```
True
True
```

4. Ejercicios

- Crear en colab y evaluar los tipos de datos que tiene Python
- Crear en colab un programa que imprima por pantalla mensajes que utilicen todos los símbolos de Python
- Crear en colab un programa que evalué las compuertas AND y OR y NOT
- Crear en colab un programa que sume dos numero complejos

Preguntas