

Taller de FutureMakers

Día 1

Inteligencia artificial (IA) y Machine Learning (aprendizaje automático) (ML)

Versión 1.0.1

Esta guía le ayudará a completar el taller de FutureMakers.
Siga a su propio ritmo.

Tu nombre:

*No lo olvides: la programación puede ser difícil, pero es divertido, ¡puedes hacerlo! No
tengas miedo de cometer **errores** - los errores son la forma en
que aprendemos. No tengas miedo de hacer **preguntas** - los
formadores/as están para ayudarte.*

Recuerda: ¡aprendemos cometiendo errores, y es bueno hacer preguntas!

Introducción

Bienvenido al taller de FutureMakers. Vamos a repasar una serie de actividades diseñadas para introducirte en la programación, el Machine Learning (aprendizaje automático) y la inteligencia artificial. No te preocupes si no tienes mucha experiencia en codificación, o incluso si nunca has programado antes. Este taller te mostrará cómo codificar en Python (Parte A), cómo usar herramientas emocionantes como detectores de rostros o traducción automática de idiomas (Parte B), y cómo construir tu propio sitio web interactivo (Parte C).

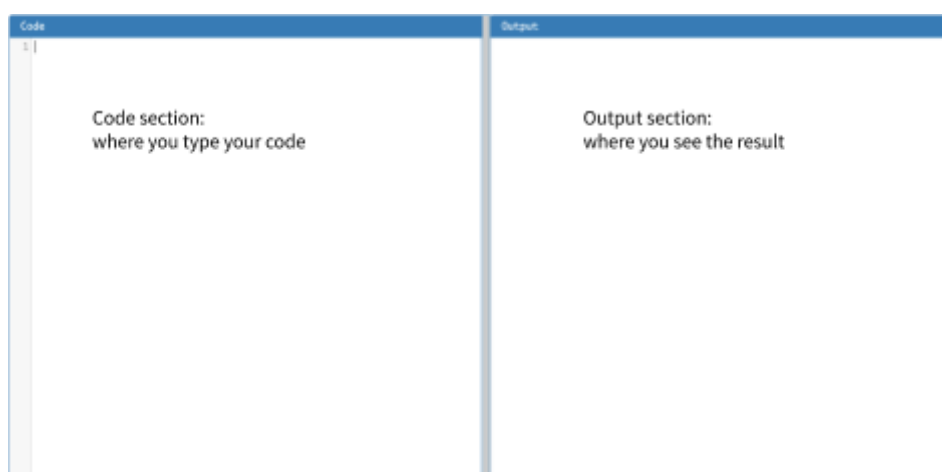
A lo largo de este taller, recuerda que la programación puede ser difícil, pero también es muy divertida. No tenga miedo de cometer errores y hacer preguntas - cometer errores es lo que aprendemos, e incluso los programadores más profesionales buscan frecuentemente en la Web cosas que no entienden. La programación no consiste en conocer todas las respuestas, sino en saber cómo buscarlas, a menudo haciendo una búsqueda en la Web o preguntando a un amigo o colega.

Parte A

Comenzando a programar (en Python)

CodeSkulptor es un sitio web diseñado para aprender a codificar en Python sin tener que configurar nada en tu ordenador. Abrir CodeSkulptor: <https://py3.codeskulptor.org/>. O simplemente busque el CodeSkulptor Python 3. **Asegúrese de que está utilizando la versión Python 3, no la versión Python 2.**

CodeSkulptor tiene dos caras:



Empieza por borrar todo lo del lado del código (a la izquierda), para que esté vacío.

¿Qué significa esto? El programador y el usuario

Recuerda: ¡aprendemos cometiendo errores, y es bueno hacer preguntas

El programador es la persona que escribe el código.

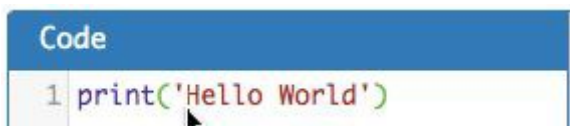
El usuario es la persona que ejecuta el código.

Por ejemplo, cuando estás jugando un juego de computadora, usted es el usuario. Cuando usted está probando su propio programa, usted es el programador y el

usuario. La codificación consta de tres pasos:

1. Escribe tu código

Aquí se introducen las instrucciones en el cuadro Código.

A screenshot of a code editor window with a blue header bar labeled "Code". Below the header, the text "1 print('Hello World')" is displayed in a monospaced font with syntax highlighting: "1" is in blue, "print" is in green, and the string is in red. A mouse cursor is visible over the code.

2. Ejecute su código

Haga clic en el botón Ejecutar:



...y CodeSkulptor ejecutará tu código Python e intentará seguir tus instrucciones.

3. Depurar

¿Escribiste las instrucciones correctas para que el ordenador las ejecutara? A menudo, lo que realmente sucede es diferente de lo que esperamos. Podríamos encontrar problemas - llamados bugs - en nuestro código.

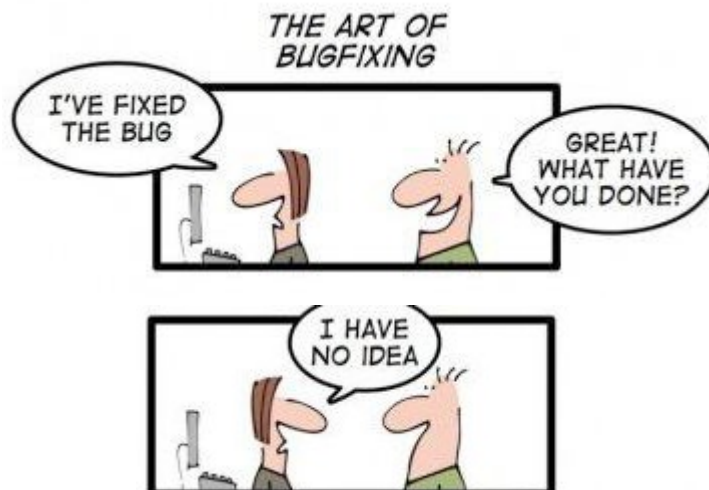
¿Qué significa esto? Bichos

Los errores son problemas con tu código, lo que significa que no hace lo que quieres. Cuando encuentre un error, aparecerá un mensaje de error (normalmente en rojo) que le dará alguna información sobre lo que está mal. En este mensaje de error, hemos intentado utilizar una variable llamada "saludo", pero no se ha podido encontrar.

A screenshot of an output window with a blue header bar labeled "Output". Below the header, the text "Line 1: NameError: name 'greeting' is not defined" is displayed in red.

Recuerda: ¡ aprendemos cometiendo errores, y es bueno hacer preguntas

La corrección de errores es una parte importante de la programación:



Ejercicio: Código de ejecución

Asegúrese de que la ventana Código esté vacía y, a continuación, introduzca el siguiente programa:

```
print('Hola Mundo')
```

A continuación, haga clic en Ejecutar.

Tu turno

Modificar el programa anterior para decir algo diferente.

Ejercicio: Hacer una pregunta al usuario

Introduzca el siguiente programa:

```
name = input('¿Cuál es su  
nombre?') greeting = "Hello " +  
name print(greeting)
```

¿Qué significa esto?

Una **variable** es una parte de su programa que puede recordar cosas.

Aquí tenemos dos variables: nombre (que almacena el nombre del usuario cuando lo introduce, como 'Alex') y saludo (que almacena el saludo, como 'Hello Alex'). Usamos el signo más (+) para pegar las dos partes del saludo ('Hola ' y 'Alex'). El signo más (+) se puede utilizar para unir palabras así como para sumar números.

Tu turno

Recuerda: ¡ aprendemos cometiendo errores, y es bueno hacer preguntas!

Cambiar el último programa para decir adiós al usuario.

Pregunta: ¿Por qué hay un espacio después de "Hola"? ¿Qué pasa si quitamos este espacio? Sácalo y vuelve a ejecutar el programa.

Ejercicio: hacer cosas muchas veces

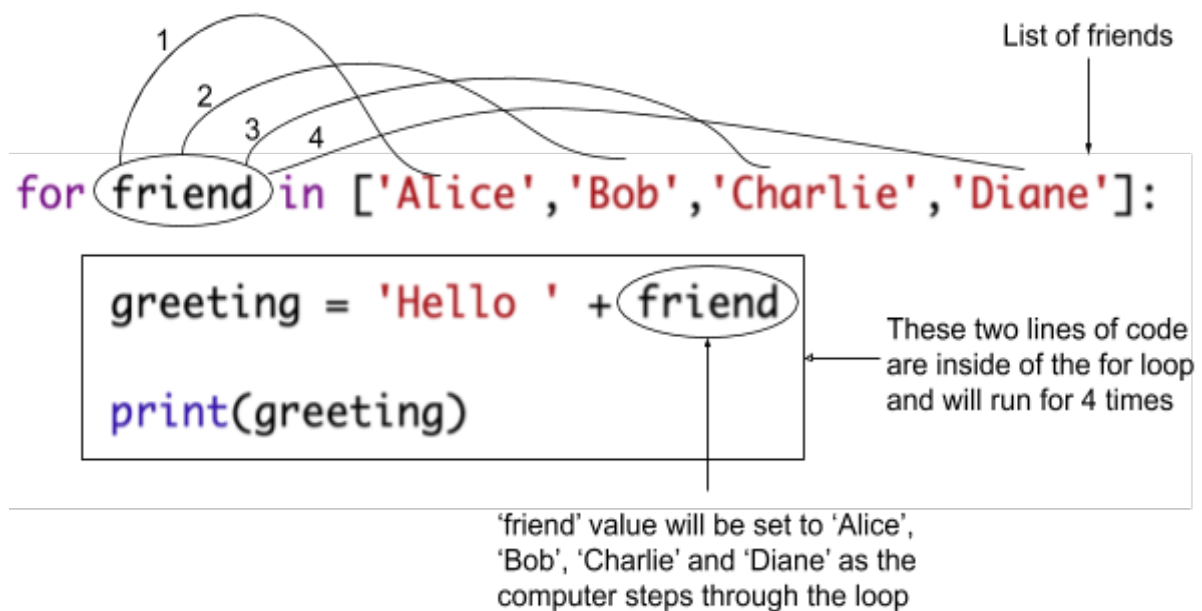
Antes de empezar, asegúrese de que la ventana de código esté vacía. Si desea guardar su código del ejercicio anterior, puede abrir una nueva pestaña para obtener una nueva ventana de CodeSkulptor (asegúrese de usar la versión Python 3).

Escriba el siguiente programa:

```
para un amigo en ['Alice', 'Bob', 'Charlie',  
    'Diane']: saludo = 'Hola ' + amigo  
    imprimir(saludar)
```

A continuación, haga clic en Ejecutar.

Esto se llama un **bucle**. Sabemos que es un bucle porque comienza con la palabra para. Dondequiera **que** veas, sabes que algo va a pasar varias veces.



Pregunta: ¿Por qué hay un espacio en blanco antes de algunas de las líneas?

Este espacio en blanco se denomina sangrado. Cada línea con una sangría está dentro de un bucle.

A veces, CodeSkulptor le ayudará a insertar automáticamente una sangría. Si desea crear uno usted mismo, pulse el tabulador.

¿Qué significa esto?

Recuerda: ¡ aprendemos cometiendo errores, y es bueno hacer preguntas!

Un **error** es un problema con su programa. A veces los errores ocurren porque cometiste un error. A veces suceden porque otro programador cometió un error.

Bug: ¿qué pasa si olvidamos la sangría?

Borre los sangrados para que su programa se vea así:

```
para su amigo en ['Alice', 'Bob', 'Charlie', 'Diane']:  
saludo ='Hola ' + impresión  
de amigo(saludo)
```

Ahora ejecute el programa. En la salida, verá

```
Línea 2: SyntaxError: mala entrada ('saludo')
```

Esto significa que su programa tiene un error. Aquí, el error es que no hay ninguna sangría. Debido a que hay un bucle, Python está esperando una línea sangrada; no encuentra una, así que no sabe qué poner en el bucle y nos muestra un error.

Tu turno

Vamos a añadir algunos amigos a la lista.

Esta parte del programa es una lista de amigos:

```
Alice, Bob, Charlie, Diane.
```

Puedes ver que el nombre de cada amigo está entre comillas, y están separados por comas. Añade más nombres a la lista. No olvides las comillas y las comas. Por ejemplo:

```
"Alice", "Bob", "Charlie", "Diane", "Jeremy".
```

A continuación, ejecute el programa de nuevo.

Consejo

Las palabras que quieras imprimir en la pantalla (como los nombres) siempre tienen que estar entre comillas, como 'Londres'.

```
print('Londres')
```

Sin embargo, los nombres de las variables (como el saludo) no necesitan estar entre comillas.

Ejercicio: bucles con números

Imagina que quieres decir algo 10 veces. Por ejemplo:

```
¡Tengo una  
banana! ¡Tengo  
dos plátanos!
```

Recuerda: ¡ aprendemos cometiendo errores, y es bueno hacer preguntas!

¡Tengo tres
plátanos!

Recuerda: ¡ aprendemos cometiendo errores, y es bueno hacer preguntas!

```
¡Tengo 4 plátanos!  
¡Tengo 5 plátanos!  
¡Tengo 6 plátanos!  
¡Tengo 7 plátanos!  
¡Tengo 8 plátanos!  
¡Tengo 9 plátanos!  
¡Tengo          10  
plátanos!
```

Podríamos hacer esto con huellas:

```
print('¡Tengo 1 banana!')  
print('¡Tengo 2 bananas!')  
... y así sucesivamente...
```

Pero esto sería aburrido y repetitivo. Pruebe el siguiente código:

```
para i in range(10):  
    print('I have ' + str(i) + ' bananas!')
```

Eso es mucho más limpio.

Sin embargo, todavía hay dos problemas, ¿puedes verlos? El siguiente código los corrige:

```
para i en  
    rango(10): j =  
        i + 1  
    si j == 1:  
        print ("Tengo ' + str(j) + ' banana!")  
    si no:  
        print('I have ' + str(j) + ' bananas!')
```

¿Puedes ver cómo?

Consejo: `range(x)` produce los números comenzando en 0 y continuando hasta que `x-1` es producido. Por ejemplo, el `rango(3)` es `[0, 1, 2]`.

Ejercicio: hacer algo una vez, luego algunas cosas repetidamente, luego algo una vez.

Introduzca el siguiente código:

```
name = input('¿Cuál es su nombre?')  
  
para thing_to_say en ['Hello', 'Have a nice day',  
    'Goodbye']: print(thing_to_say + name)  
  
print ('¡Diviértete!')
```


Recuerda: ¡ aprendemos cometiendo errores, y es bueno hacer preguntas!

Consejo: si ha escrito correctamente la línea de bucle (la línea que comienza por **for**) terminándola con dos puntos, CodeSkulptor sangrará automáticamente la siguiente línea. Si no lo hace, puede colocar la sangría usted mismo pulsando la tecla de tabulación.

¿Qué está pasando aquí? La primera línea, que recibe el nombre del usuario, se ejecuta una vez. Entonces la línea dentro del bucle, que dice algo al usuario, se está ejecutando varias veces.

Por último, la última línea, que dice `Diviértete!

Consejo: una línea que está sangrada puede estar en un bucle, por lo que puede ejecutarse varias veces. Una línea que *no* está sangrada no está en un bucle, por lo que sólo se ejecutará una vez.

Tu turno

Crea tu propio bucle y ejecútalo. ¡Puede hacer lo que quieras! Usted puede obtener información del usuario primero si lo desea, pero no tiene que hacerlo. Usted puede incluso obtener información del usuario *dentro* del bucle - esto significa que usted recibirá información varias veces.

Ejercicio: hacer algunas matemáticas

Antes de empezar, asegúrese de que la ventana de código esté vacía. Si desea guardar su código del ejercicio anterior, puede abrir una nueva pestaña para obtener una nueva ventana de CodeSkulptor (asegúrese de usar la versión Python 3).

Hay muchas cosas que podemos hacer con Python. Una de las más sencillas es usarla como calculadora.

- Cuando estamos programando, usamos el símbolo más (+) para sumar números y el símbolo menos (-) para restar números.
- Sin embargo, no usamos el símbolo de los tiempos (×) para multiplicar los números, sino que usamos la estrella (*).
- Y no usamos el símbolo de división (÷) para dividir números, usamos la barra (/) en su lugar.

Introduzca y ejecute el siguiente código:

```
imprimir(14 * 21)
```

Puede ver que la respuesta está impresa.

Tu turno: introduce dos números muy grandes (con 6 ó 7 dígitos cada uno) y multiplícalos. Python es mucho, mucho más potente que tu calculadora!

Ejercicio: matemáticas con nombres de variables

Podemos usar variables para almacenar números con un nombre en particular.

Recuerda: ¡ aprendemos cometiendo errores, y es bueno hacer preguntas!

Introduce y ejecuta el siguiente código:

```
número = 5
```

Recuerda: ¡ aprendemos cometiendo errores, y es bueno hacer preguntas!

```
otro_número = 10

total = número + otro_número
imprimir(total)
```

Bug: ¿qué pasa si intentamos pegar una palabra a un número? Introduzca el siguiente código:

```
número = 5
otro_número = 10

total = número + otro_número
imprimir('El total es ' +
total)
```

Ahora trata de ejecutarlo. Verá un error:

Línea 5: TypeError: no puede concatenar objetos 'str' e 'int'.

Concatenado" **sólo significa** "pegamento" o "pegamento". Python está confundido porque le pedimos que pegue una cadena (palabra - **str** es la abreviatura de cadena) a un entero (un número entero).

Python es muy inteligente, pero a veces no sabe cómo hacer cosas obvias. Para corregir este error, sólo necesitamos convertir nuestra variable entera en una cadena primero.

Ejercicio: convertir un entero en una cadena de caracteres

Antes de empezar, asegúrese de que la ventana de código esté vacía. Si desea guardar su código del ejercicio anterior, puede abrir una nueva pestaña para obtener una nueva ventana de CodeSkulptor (asegúrese de usar la versión Python 3).

Introduzca y ejecute el siguiente código:

```
mi_número entero = 7
mi_cadena = '7

print('Este es el número
entero:') print(my_integer)

print('Esta es la cadena:')
print(my_string)

print('Este es el entero convertido en una
cadena:') print(str(my_integer))

print ('¡Todos parecen iguales!')
```

Recuerda: ¡ aprendemos cometiendo errores, y es bueno hacer preguntas!

Ejercicio: declaraciones de if

Hemos visto cómo usar un bucle para hacer algo varias veces: esta es una de las técnicas más importantes en la programación. Otra técnica clave es la **ifstatement**, que es una manera de tomar una decisión: hacer una cosa bajo ciertas circunstancias, o hacer otra cosa.

Introduzca el siguiente programa en el panel de código y haga clic en Ejecutar.

```
temperatura = flotador(entrada ('¿Cuál es la
temperatura? '))) si la temperatura es > 30:
    imprimir ("No está muy
caliente"):
    print ("¡Está demasiado caliente!")
```

Una vez que se introduce una temperatura, el programa hace una elección y decide qué hacer.

Ejercicio: ejecutar algunos ejemplos de CodeSkulptor

Hay un botón a la izquierda de CodeSkulptor que muestra muchos ejemplos escritos por otras personas. Eche un vistazo, abra uno de los ejemplos, lea (pregunte a un instructor si tiene alguna pregunta) y haga clic en Ejecutar.

Luego, eche un vistazo a algunos ejemplos más, léalos y ejecútelos.

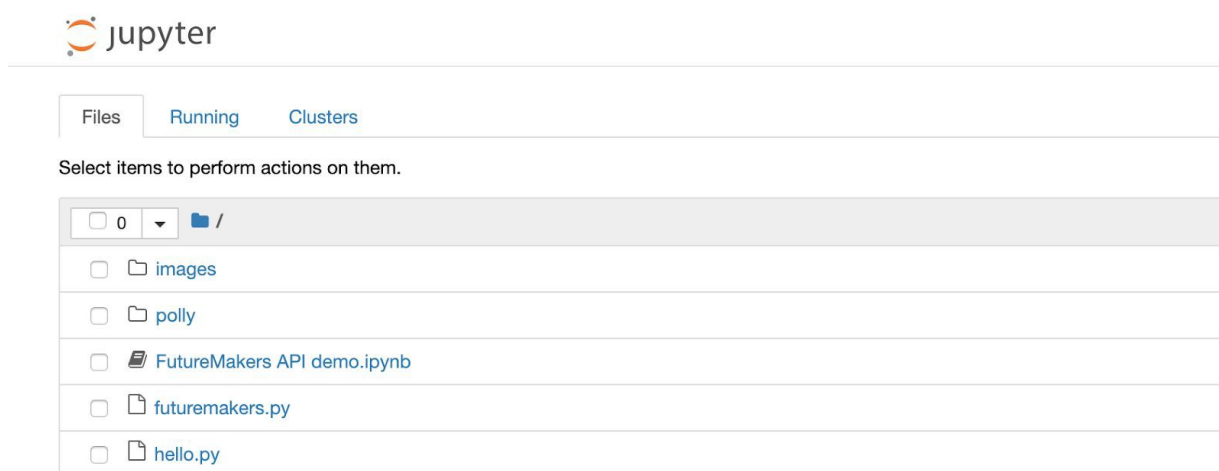
Recuerda: ¡ aprendemos cometiendo errores, y es bueno hacer preguntas!

Parte B

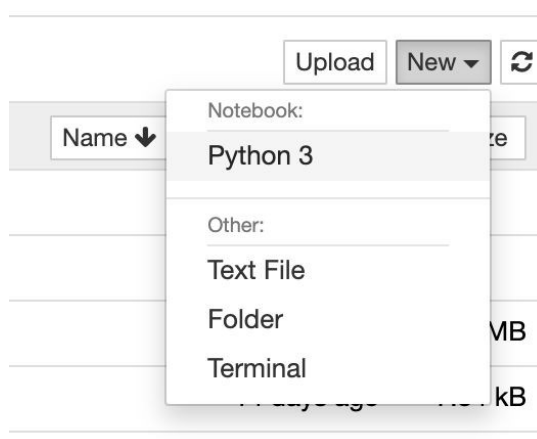
Chatbots y AI con el cuaderno Jupyter

Abra el Navegador Anaconda desde el menú Inicio y haga clic en Notebook. Esto abrirá el Jupyter Notebook, un entorno fantásticamente útil para desarrollar y ejecutar código Python.

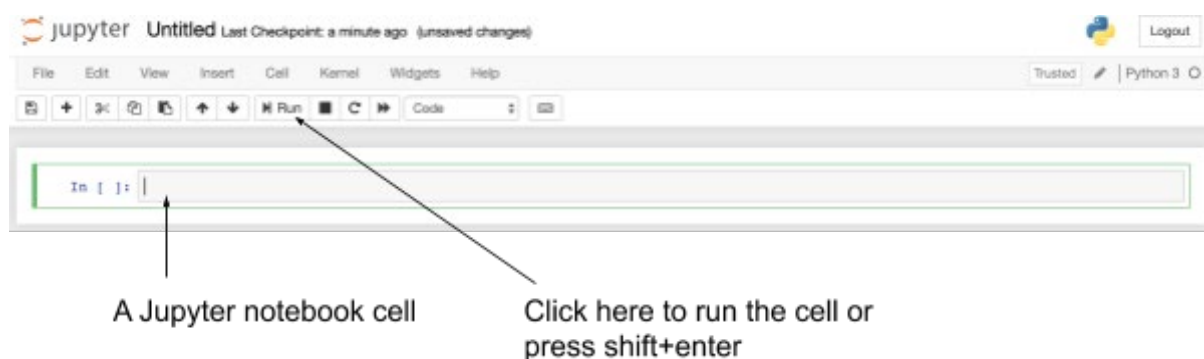
Asegúrese de navegar hasta la carpeta donde está almacenado el archivo `futuremakers.py`:



Haga clic en el botón 'Nuevo' en la parte superior derecha y seleccione Python 3 notebook



Recuerda: ¡ aprendemos cometiendo errores, y es bueno hacer preguntas!



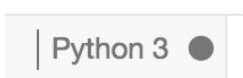
El código vive en pequeñas áreas llamadas **células**. Presione escape (esc), luego b para crear una nueva celda debajo de la actual. Presione esc, luego a para crear una nueva celda sobre la actual.

Para ejecutar el código en una celda, presione shift e ingrese al mismo tiempo.

Cuando el kernel de Python está listo, el indicador en la parte superior derecha se ve así:



Cuando el kernel de Python está ocupado, por otro lado, se ve así:



Si el kernel se atasca, reinícialo usando el menú o presionando ctrl-c. Una vez que reinicie el kernel (o abra de nuevo el portátil), todo el código y la salida de la última vez estará allí, pero no habrá variables o funciones presentes en la memoria; tendrá que ejecutar el código de nuevo para hacer algo.

Es una buena idea organizar su cuaderno de manera que las células puedan funcionar de forma natural y en orden.

Ejercicio: código de ejecución en el cuaderno

- Abrir un nuevo cuaderno.
- Clic en la primera celda.
- Presione esc, luego b para crear una nueva celda después de la actual. Haga esto unas cuantas veces más para tener muchas celdas disponibles.
- Escriba algún código en la primera celda; imprimir algo es una buena idea.
- Presionar shift+enter para ejecutar la primera celda.

Ejercicio: iniciar y detener el kernel

¿Qué significa esto? El kernel de Python es simplemente Python mismo: el programa que lee tu código Python y lo ejecuta.

Recuerda: ¡ aprendemos cometiendo errores, y es bueno hacer preguntas!

Escriba el siguiente código en la celda de un cuaderno:

```
mientras que True:  
    print ("¡Sigo corriendo!")
```

```
In [ ]: while True:  
        print("I'm still running!")
```

Este código se ejecutará para siempre y no se detendrá por sí solo. Ejecutar la celda (shift+enter).

Observe que Python está ocupada:

Python 3 ●

Ahora detenga el kernel, ya sea usando el botón Detener o usando los menús: Kernel > Interrumpir. Ahora sabes cómo detener el kernel si lo necesitas más tarde.

Ejercicio: definir algunas variables

Asegúrese de tener cuatro celdas vacías antes de comenzar este ejercicio. Comience reiniciando el kernel (menú del kernel > Reiniciar). Esto asegura que Python está listo y que no quedan variables del código anterior.

Ahora llene las celdas así:

Escriba esto en la primera celda	<code>alices_dog = 'Rover'.</code>	Ejecute esta celda ahora
Escriba esto en la segunda celda	<code>bobs_dog = 'Fido</code>	<i>No dirijas esta celda</i>
Escriba esto en la tercera celda	<code>print(alices_dog)</code>	Ejecute esta celda ahora
Escriba esto en la cuarta celda	<code>print(bobs_dog)</code>	Ejecute esta celda ahora

Notará que `print(alices_dog)` funciona e imprime 'Rover'.

Sin embargo `print(bobs_dog)` no funciona: hay un error. Esta línea provoca un error:

NameError: nombre 'lol' no está definido

¿Qué ha pasado aquí? Python sabe de la variable `alices_dog`, *porque corrimos esa celda*. Pero Python no sabe de la variable `bobs_dog`, *porque no ejecutamos esa celda*.

Tienes que ejecutar una celda para que el código en ella funcione. De lo contrario, el código no tendrá ningún efecto. Puede ejecutar una celda más de una vez, y puedes

Recuerda: ¡ aprendemos cometiendo errores, y es bueno hacer preguntas!

ejecutarla de nuevo cuando lo desee.

¿Qué significa esto? Funciones

Recuerda: ¡aprendemos cometiendo errores, y es bueno hacer preguntas!

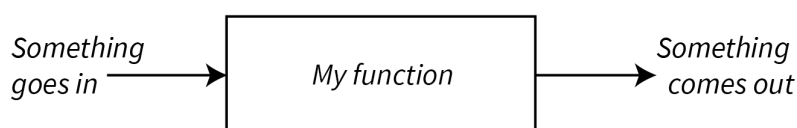
En la programación, a menudo queremos hacer lo mismo una y otra vez. Para ello, utilizamos funciones. Una función es como una pequeña máquina que toma una entrada, le hace algo y devuelve una salida.

Algunos ejemplos de funciones podrían ser:

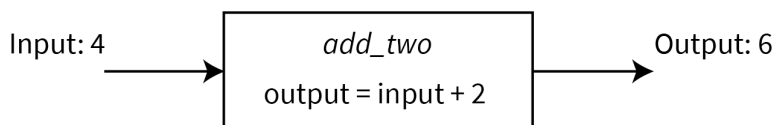
- Una función para devolver TRUE si un número es menor de 10, y FALSE en caso contrario;
- Una función para añadir dos a un número;
- Una función para poner en mayúsculas una palabra;
- Una función para verificar si una fecha es un miércoles.

Puede imaginar funciones como cajas reutilizables que pueden procesar entradas y convertirlas en salidas, como ésta:

1. What a function looks like generally



2. Example function which adds two to a number



Ejercicio: hacer una función

Inicie este ejercicio reiniciando el núcleo (menú del núcleo > Reiniciar).

Hagamos una función - un trozo de código que podemos reutilizar varias veces dándole un nombre.

Escriba lo siguiente en una celda, pero no la ejecute todavía.

```
def add_100(number):  
    número de devolución + 100
```

```
In [ ]: def add_100(number):  
        return number + 100
```

Consejo: no olvide sangrar la línea que está dentro de la función.

Una función es como una máquina que hace algo útil para nosotros. Algo entra y algo sale. Este código define una función que toma un número, le añade 7 y nos devuelve el resultado.

Recuerda: ¡ aprendemos cometiendo errores, y es bueno hacer preguntas!

Una función en python se verá como esta plantilla:

```
def my_python_function(input_variable_1, input_variable_2):  
    output = input_variable_1, input_variable_2  
    salida de retorno
```

Esta plantilla sólo añade dos variables de entrada juntas, pero puede hacer lo que quiera en una función. Las funciones pueden tener cualquier número de variables de entrada, o ninguna. La palabra clave **def** es la abreviatura de *define* - muestra que estamos definiendo una nueva función. La palabra clave **return** es lo que usamos para enviar la salida de vuelta de la función.

Ahora, aún sin ejecutar la primera celda, intentemos usar nuestra función en otra celda. Busque o haga una celda vacía y escriba el siguiente código:

```
print(add_100(7))
```

Esta línea le pide a Python que tome 7, le agregue 100 e imprima el

resultado. Ahora trata de ejecutar la célula. Hay un error - tenemos

otro error:

```
NameError: nombre 'add_100' no está definido
```

¿Por qué está pasando esto? Como antes, es porque no hemos ejecutado la célula con la función en ella. Así que vuelve y ejecuta la celda en la que escribiste la función. Luego ejecute la celda que contiene `print(add_100(7))`. Esta vez, funcionará, e imprimirá 107.

Tu turno

Definir otra función que hace algo interesante.

Consejo: una función no tiene que tomar un valor de entrada, y no tiene que devolver nada. Por ejemplo:

Código de la función	Notas	Cómo se utiliza la función	Lo que se ve en la pantalla cuando se utiliza la función
			función función

Recuerda: ¡ aprendemos cometiendo errores, y es bueno hacer preguntas!

<pre>def say_hello(): print('Hola!')</pre>	No toma ninguna entrada No devuelve nada (sólo impresiones) Hola)	di_hola()	Hola!
--	---	-----------	-------

Recuerda: ¡ aprendemos cometiendo errores, y es bueno hacer preguntas!

<pre>def say_hello_to(name): print('Hola, ' + nombre + '!')</pre>	Toma entrada No devuelve nada (sólo imprime un archivo saludo)	<pre>say_hello_to('Alice')</pre>	Hola Alice!
<pre>def my_favourite_singer(): volver 'Justin Bieber</pre>	No toma ninguna entrada Devuelve a denominar	<pre>print(mi_cantante_favorito)</pre>	Justin Bieber
<pre>def doble(número): devolver 2 * número</pre>	Toma entrada También devuelve algo (dos veces el original número)	<pre>print(doble(7))</pre>	14

Uso de los servicios de inteligencia artificial (IA) de Amazon con el cuaderno Jupyter

Hagamos algo emocionante, utilizando servicios de inteligencia artificial de última generación para traducir y hablar texto y reconocer personas y objetos. La mayoría de las tareas que estamos a punto de realizar eran imposibles hace cinco años!

Ejercicio: importar la biblioteca FutureMakers

Comience un nuevo cuaderno - es bueno tener un espacio de trabajo claro y fresco.

Una de las mejores cosas de Python es que es fácil usar el código de otras personas.

¿Qué significa esto?

Una **biblioteca** es un programa que está destinado a ser utilizado por otros. Por ejemplo, existen librerías Python para mostrar imágenes en la pantalla, para conocer el tiempo en cualquier parte del mundo, o para hacer una serie de imágenes en una animación como un GIF. Python hace que sea muy fácil usar las miles de librerías escritas por otros usuarios de Python. Aquí usaremos la biblioteca de **Futuremakers**, que fue creada por los instructores de FutureMakers para ayudarle a utilizar la IA y el aprendizaje automático.

Empecemos por importar la biblioteca de **futuros fabricantes**. Este es un archivo llamado **futuremakers.py** que debería estar en el directorio **python** dentro de la carpeta FutureMakers.

Recuerda: ¡ aprendemos cometiendo errores, y es bueno hacer preguntas!

Una vez que haya abierto un nuevo cuaderno, escriba la siguiente línea en una celda y ejecute la celda.

de la importación de los futuros fabricantes *

La célula debería tener este aspecto:

Recuerda: ¡ aprendemos cometiendo errores, y es bueno hacer preguntas!

```
In [1]: from futuremakers import *
```

¿Qué significa esto? *

El símbolo de estrella * se utiliza a menudo en la programación para significar "todo". Así que esta línea significa

Desde la biblioteca de Futuremakers, importe todo.

Ahora probemos que nuestra importación funcionó. Escriba esta línea en una nueva celda y ejecútela:

```
test_futuremakers()
```

La célula debería tener este aspecto:

```
In [2]: test_futuremakers()
```

```
The FutureMakers library is imported and ready to use.
```

Esto le pide a Python que busque la función test_futuremakers (que está dentro del módulo futuremakers) y la ejecute. Deberías ver el mensaje

La biblioteca FutureMakers se importa y se imprime un mensaje de bienvenida

debajo de la celda. Si ve un error, pida ayuda a un instructor.

Si está interesado en ver el código Python para la biblioteca FutureMakers, simplemente abra futuremakers.py (puede hacerlo desde la página de carpetas de Jupyter Notebook) y eche un vistazo. Hay mucho código -unas 100 líneas- y la mayor parte se trata de usar Internet para hacer peticiones a los servidores de Amazon, que son los que hacen la mayor parte del trabajo.

Ejercicio: traducción de texto

Ahora estamos listos para usar algunas de las otras funciones de la biblioteca de FutureMakers. Traduzcamos algo de texto.

Escriba esta línea en una nueva celda y ejecútela:

```
translate_english_to_spanish('Tengo un perro muy bueno.')
```

Deberías ver la salida "Tengo un perro muy bueno".

¿Qué pasó aquí?

Recuerde: ¡aprendemos cometiendo errores, y es una buena idea hacer

La biblioteca de FutureMakers utilizó Internet para hacer una solicitud a los servidores de Amazon (ya sea en Londres o en algún lugar de los Estados Unidos). Al igual que cuando usted accede a una página web, la biblioteca envió su sentencia a los servidores de Amazon y les pidió que la tradujeran al español. El

Recuerde: ¡aprendemos cometiendo errores, y es una buena idea hacer

Los servidores de Amazon devolvieron la respuesta y la biblioteca se la devolvió a usted. Finalmente, el cuaderno Jupyter lo mostró como texto.

Tu turno

Intenta traducir un texto diferente del inglés al español.

¿Qué pasa si introduces algún texto que ya está en español? ¿Qué sucede si ingresas algún texto que no esté en inglés o español?

Ejercicio: traducción de diferentes idiomas

Para traducir entre diferentes idiomas, necesitamos decirle a Amazon Translate en qué idioma se encuentra actualmente nuestro texto y a qué idioma queremos traducir. Hacemos esto usando códigos de lenguaje, como **en** para el inglés, **fr** para el francés, y **es** para el español (*español*).

Por ejemplo, podemos traducir del inglés al francés:

```
translate('en', 'fr', 'I have a good dog')
```

"J'ai un bon chien."

...luego del francés al español:

```
"J'ai un bon chien.") ``Tengo un buen  
perro.''
```

...luego del español al inglés:

```
"Tengo un buen perro".
```

Pruébalo por ti mismo. Si usas una frase más compleja, Amazon podría equivocarse.

Tu turno

Traduce una frase de un idioma que conoces a un idioma que no conoces tan bien, y luego tradúcela de vuelta. Utilice la siguiente tabla para buscar códigos de idioma. Si un idioma no aparece en esta tabla, significa que Amazon Translate no puede usarlo todavía.

Árabe y chino
(simplificado) zh
Chino (Tradicional) zh-TW
Checo cs
Danish da
Dutch nl
English en
Finnish fi

Francés fr
Alemán de
Hebreo
hebreo hindi
hi Indonesio
id Italiano it
Japonés ja
Coreano ko

Recuerde: ¡aprendemos cometiendo errores, y es una buena idea hacer

Malayo ms
Noruego no
Persa fa
Polaco pl
Portugués pt

Ruso ru
Español es
Sueco sv
Turco tr

Ejercicio: hablar texto en voz alta (sintetizar el habla)

Ahora utilizaremos otro servicio de inteligencia artificial relacionado con el lenguaje: la síntesis de voz. Le pediremos a Amazon que genere un archivo de audio a partir del texto que queramos; luego lo reproduciremos, como si el servicio de Amazon pudiera hablar. Esto utiliza un servicio de Amazon llamado **Polly**.

Compruebe que el volumen ha subido, luego escriba esto en una nueva celda y ejecútelo:

```
hablar ('Hola, FutureMakers!')
```

Después de un breve retardo, un reproductor de audio se abrirá y reproducirá su discurso sintetizado. Este audio fue generado por poderosos sistemas de aprendizaje automático dentro de uno de los edificios de Amazon.

Si desea volver a reproducir el archivo, puede volver a llamar a la función de **voz**. Si desea encontrar el archivo de audio (por ejemplo, para enviárselo a un amigo), se encuentra en la carpeta **polly**; cada vez que ejecuta la función de **voz**, se guarda un nuevo archivo de audio en esta carpeta.

Tu turno: trata de pedirle a Amazon que diga algo más largo. Experimente con puntos y comas para generar pausas en el discurso.

Ejercicio: buscar, guardar y mostrar imágenes

Para los próximos ejercicios, trabajaremos con imágenes. La mejor forma de hacerlo es buscar imágenes en Google Imágenes y guardarlas en la carpeta de **imágenes** dentro de la carpeta FutureMakers para que podamos trabajar con ellas fácilmente.

Intentemos guardar una imagen ahora.

- Abra Chrome o Firefox y vaya a www.google.com (o use la barra de búsqueda).
- Buscar algo (un animal o una persona famosa son buenos lugares para empezar).
- Una vez que esté viendo los resultados de la búsqueda, haga clic en Imágenes.
- Haga clic en una de las imágenes para que sea más grande que todas las demás.
- **En Chrome**, haga clic con el botón derecho en la imagen y haga clic en **Abrir imagen en la nueva pestaña**. El objetivo es que la imagen se abra en una ventana por sí sola; si esto no sucede inmediatamente, es posible que tenga que hacer clic con el botón derecho del ratón en la imagen de nuevo y seleccionar **Abrir imagen en una nueva pestaña de nuevo**. A continuación, haga clic con el botón derecho del ratón en la imagen de nuevo y haga clic en **Guardar imagen como**.

Recuerda: ¡ aprendemos cometiendo errores, y es bueno hacer preguntas!

- **En Firefox**, haga clic con el botón derecho en la imagen y haga clic en **Ver imagen**. El objetivo es que la imagen se abra en una ventana por sí sola; si esto no sucede de inmediato, puede ser que

Recuerda: ¡ aprendemos cometiendo errores, y es bueno hacer preguntas!

necesita hacer clic con el botón derecho del ratón en la imagen de nuevo y seleccionar **Ver imagen de nuevo**. A continuación, haga clic con el botón derecho del ratón en la imagen de nuevo y haga clic en **Guardar imagen como**.

- Cuando haga **Guardar imagen como**, el navegador le preguntará dónde desea guardar la imagen. Busque la carpeta **FutureMakers** (que se encuentra en su escritorio), luego busque la carpeta de **imágenes** dentro de ella y guarde la imagen dentro de la carpeta de **imágenes**. Cuando guarde la imagen, dele un buen nombre como cat.png o cat.jpg (conservar la extensión), no un nombre difícil de usar como 8978_cat_small_96.jpg.
- En Windows, haga doble clic en su imagen en la carpeta de **imágenes** para comprobar que está allí.

Para los siguientes ejercicios, es mejor usar imágenes de tamaño mediano - no tan pequeñas que no tengan muchos detalles, pero no tan grandes que puedan llenar fácilmente toda la pantalla.

Ejercicio: reconocer a las celebridades

Ahora usaremos los servicios de Amazon para tratar de reconocer a las celebridades. Probemos primero con una imagen que ya está en la carpeta Futuremakers/images: friends.jpg.

Veamos primero la imagen. En una nueva celda, escriba esta línea y ejecútela:

```
show_image('imágenes/amigos.jpg')
```

Esto mostrará la imagen justo debajo de la celda.

Ahora pidamos a Amazon que reconozca a la gente en esta imagen. En una nueva celda, escriba esta línea y ejecútela:

```
find_celebrities('images/friends.jpg')
```

Después de un breve retraso, debería ver la siguiente línea:

```
Amazon Rekognition encontró 4 celebridades en images/friends.jpg: David Schwimmer, Matt LeBlanc, Lisa Kudrow y Courteney Cox.
```

Amazon no ha hecho el trabajo a la perfección: sólo ha reconocido a cuatro personas, sin contar a Jennifer Aniston y Matthew Perry.

Tu turno: busca una imagen de una celebridad más moderna (o una que contenga más de una persona), guárdala en la carpeta de **imágenes**, luego usa `find_celebrities` para intentar detectar a la persona o personas famosas. También puede usar `show_image` para ver la imagen en el cuaderno de Jupyter.

¿Por qué no probar las imágenes (suministradas) `cage.jpg` y `patrick_stewart.jpg`?
¿Por qué tarda tanto tiempo en procesarse `cage.jpg`?

Recuerda: ¡ aprendemos cometiendo errores, y es bueno hacer preguntas!

Ejercicio: detección de rostros

Recuerda: ¡ aprendemos cometiendo errores, y es bueno hacer preguntas!

Amazon Rekognition también puede detectar rostros en una imagen - no sólo los rostros de celebridades, sino cualquier rostro - y estimar sus edades, sus expresiones y (curiosamente) si tienen barba.

Probemos primero con una imagen de muestra. Veámoslo primero; Escriba esta línea en una nueva celda y ejecútela:

```
show_image('images/patrick_stewart.jpg')
```

Para detectar caras, escriba esta línea en una nueva celda y ejecútela:

```
detect_faces('images/patrick_stewart.jpg')
```

Debería ver la siguiente salida:

Amazon Rekognition encontró 1 cara en esta imagen.
Creo que esta persona tiene entre 60 y 80 años. Estoy 100 por ciento seguro de que no tienen barba. Estoy 68% seguro de que se sienten felices.

Tu turno: elige una nueva imagen de Google Imágenes, guárdala en la carpeta FutureMakers/images con un nombre fácil de recordar, visualízala en el portátil e intenta detectar caras. Vea si está de acuerdo con Amazon Rekognition! También puede probar imágenes que contengan más de una persona.

Ejercicio: detección de objetos

Amazon Rekognition también puede detectar objetos en una imagen. Probemos esto con una imagen difícil que contiene muchos objetos.

Para mostrar la imagen, escriba esta línea en una nueva celda y ejecútela:

```
show_image('images/mike_wilks.jpg')
```

Esta es una imagen de un libro llamado The Ultimate Alphabet de Mike Wilks. Contiene muchos objetos que comienzan con la letra S.

Ahora, para detectar objetos, escriba esta línea en una nueva celda y ejecútela:

```
detect_objects('images/mike_wilks.jpg')
```

Amazon Rekognition debe encontrar entre 20 y 30 objetos. Sin embargo, no todos comienzan con S! Amazon Rekognition está haciendo todo lo posible para describir los objetos en la imagen.

Tu turno: elige una nueva imagen de Google Imágenes, guárdala en la carpeta FutureMakers/images con un nombre fácil de recordar, visualízala en el portátil e intenta detectar objetos. Vea si está de acuerdo con Amazon Rekognition! ¿Qué tan bueno es para

Recuerda: ¡ aprendemos cometiendo errores, y es bueno hacer preguntas!

diferenciar entre cosas similares, como diferentes tipos de perros?

Recuerda: ¡ aprendemos cometiendo errores, y es bueno hacer preguntas!

Ejercicio: conexión de los servicios de inteligencia artificial entre sí

Ahora hemos probado cada uno de los servicios de inteligencia artificial y aprendizaje automático que utilizaremos:

- Traducción
- Síntesis de voz
- Detección de celebridades
- Detección de rostros y estimación de emociones
- Reconocimiento de objetos

Ahora tratemos de combinarlos. Por ejemplo, haremos una función que le pedirá a Amazon que primero traduzca una frase al inglés y luego la hable.

Escriba esta función en una celda y ejecútela (nada sucederá cuando la ejecute - tendrá que llamar a la función desde otra celda):

```
def translate_and_speak(french_sentence):  
    english_sentence = translate('fr', 'en', french_sentence)  
    speak(english_sentence)
```

Asegúrate de hacer bien la sangría. Utilice el tabulador para hacer una sangría; ambas líneas dentro de la función deben estar sangradas.

Esta función traduce primero una frase del francés al inglés y luego la habla. Hagamos una frase en francés para probarlo:

```
translate('en', 'fr', "I'm cold")
```

La salida por debajo de la celda será:

"J' ai froid".

Consejo: hay un apóstrofe en la frase "Tengo frío". Esto significa que tenemos que usar comillas dobles alrededor de esta oración, porque si usamos comillas simples, Python confundirá el apóstrofe con una sola comilla.

Ahora podemos copiar y pegar la frase en francés en una llamada a nuestra nueva función. Esto traducirá la oración al inglés y luego la hablará:

```
translate_and_speak("J' ai froid")
```

Tu turno: modifica el ejemplo anterior para que se traduzca de un idioma diferente al inglés. Entonces Pruébalo.

Ejercicio: conectar dos servicios más

Recuerda: ¡ aprendemos cometiendo errores, y es bueno hacer preguntas!

En este ejemplo, le pediremos a Amazon que diga los objetos que encuentra en una imagen.

Hagamos una nueva función primero. Escriba este código en una nueva celda y ejecútelo. Esta función detecta primero los objetos y luego dice los resultados.

```
def
    detectar_objetos_y_hablar(imagen):
    objetos = detectar_objetos(imagen)
    hablar(objetos)
```

Ahora vamos a probarlo. En primer lugar, busque una imagen en línea con sólo uno o dos objetos - no queremos que la descripción tome demasiado tiempo para hablar. Guárdelo en la carpeta de imágenes con un nombre fácil. Entonces corre, por ejemplo:

```
detectar_objetos_y_hablar('images/my_new_image.jpg')
```

Tu turno

Piense en otra manera de combinar dos servicios, y póngalos juntos. Si no está seguro, no dude en preguntar a uno de los instructores.

Ejercicio: ideación

- Trabajar en equipo.
- Echa un vistazo a la lista de Objetivos de Desarrollo Sostenible: <https://www.globalgoals.org>
- Piense en un problema del mundo real que a usted y a sus compañeros de equipo les gustaría resolver juntos.
- Investigue un poco para entender mejor el problema.
- Trata de entender y sentir empatía con las personas que más lo sufren.
- Haga una lluvia de ideas sobre cómo la IA puede ayudarle a resolver este problema.
- Aplica tus superpoderes: creatividad, empatía, inteligencia emocional, juicio, razonamiento, comunicación, trabajo en equipo y las habilidades de IA que aprendiste hoy.
- Piensa en cómo, dónde y en qué forma se puede aplicar la IA para mejorar la vida de estas personas o para salvar nuestro planeta.

Ejercicio final: presentación del trabajo

- Preséntese
- Problema que está tratando de resolver con AI (SDG)
- ¿Qué tipo de inteligencia artificial crearía usted para abordar este problema?
- Cómo su IA aborda el problema
- Cualquier alternativa a tu idea
- Cualquier consideración sobre el diseño del producto para hacer su idea más accesible
- Cualquier preocupación ética que tenga
- Qué desea hacer a continuación