

# TPo

## Introducción



# Introducción a Google Colab



# ¿Qué es Google Colab?

Colab, o **Colaboratory**, permite escribir y ejecutar Python en nuestro navegador.

Nos da acceso a una máquina de manera remota.

Las ventajas son:

- No es necesaria ninguna configuración
- Es fácil de compartir
- Podemos utilizar fácilmente procesadores GPU (**G**raphics **P**rocessing **U**nit) que se diferencia del procesador “común”, el CPU (**C**entral **P**rocessing **U**nit) porque tiene más núcleos, más especializados y por lo tanto permite realizar cálculos más complejos de manera más rápida.

Una característica no muy amigable es que no se pueden editar al mismo tiempo los documentos de Google Colab (que a partir de ahora llamaremos Notebooks)

# ¿Cómo se ve Google Colab?

Vayan a <https://colab.research.google.com/>

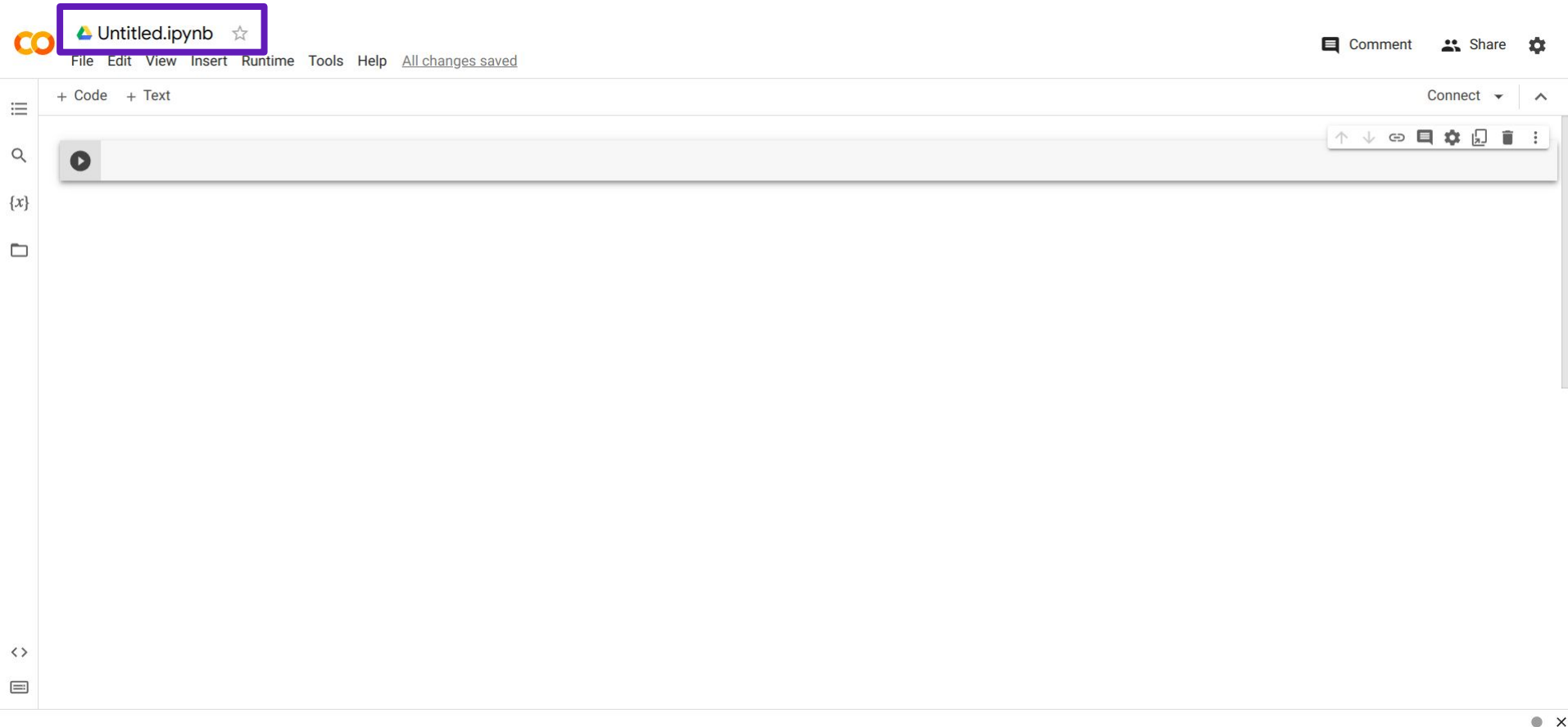
Y se les abre una ventana. En la parte inferior elijan: **New Notebook**

# Nueva Notebook

The image shows a Jupyter Notebook interface. At the top, the title "Nueva Notebook" is displayed in a large, bold, black font. Below the title, the Jupyter logo (two interlocking orange circles) is followed by the text "Untitled.ipynb" and a star icon. A menu bar contains the following items: File, Edit, View, Insert, Runtime, Tools, Help, and a link that says "All changes saved". On the right side of the top bar, there are three icons: a comment icon labeled "Comment", a share icon labeled "Share", and a settings gear icon.

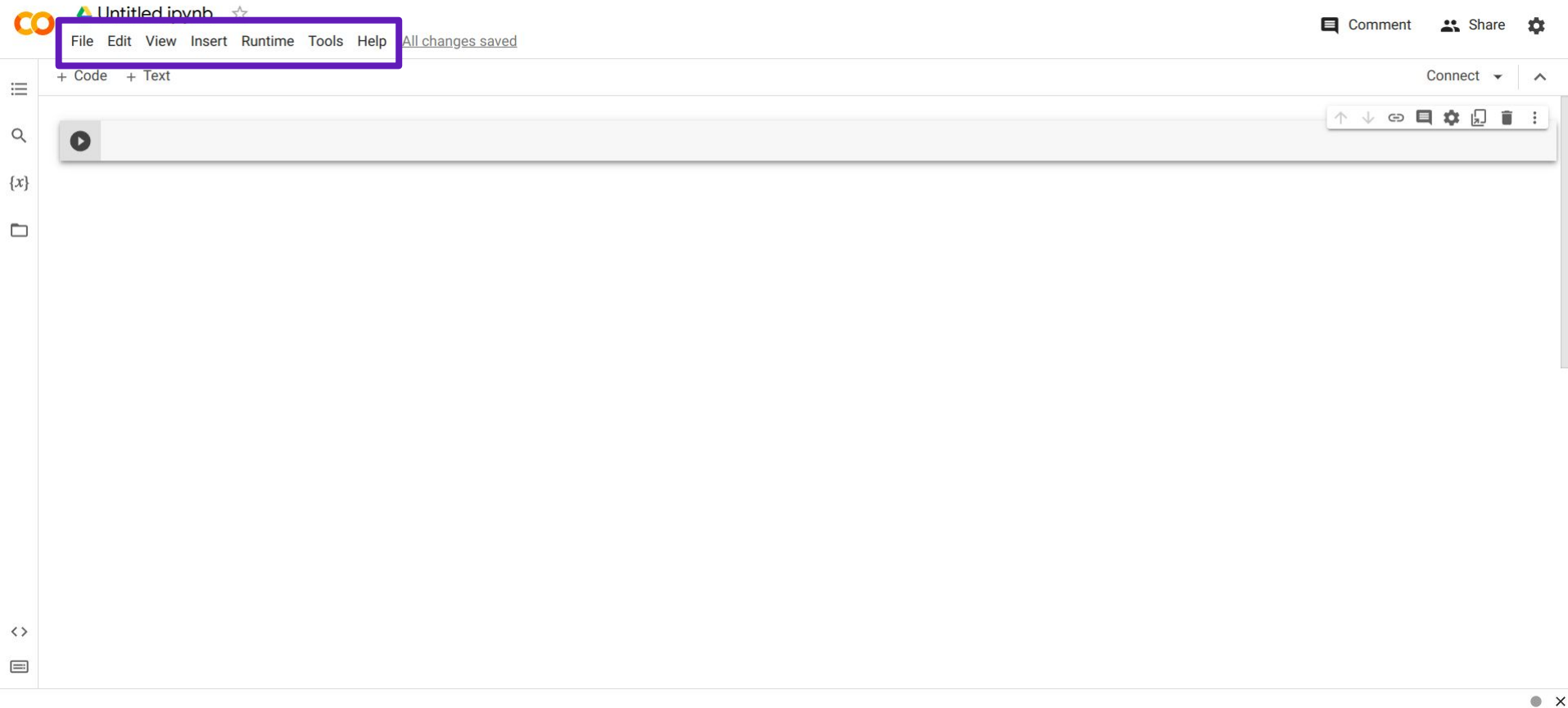
Below the menu bar, there is a toolbar with "+ Code" and "+ Text" buttons. To the right of these buttons is a "Connect" dropdown menu and an upward-pointing arrow icon. The main area of the notebook is a large, empty white space. On the left side of this area, there is a vertical sidebar with several icons: a list icon (three horizontal lines), a search icon (magnifying glass), a code icon (a bracketed 'x'), a folder icon, and a code editor icon (a bracketed '<>'). At the bottom right of the notebook area, there is a small gray bar with a play button icon (a circle with a right-pointing triangle) and a toolbar with icons for undo, redo, comment, settings, copy, paste, and a vertical ellipsis menu.

# Nombre de la “notebook”



The screenshot displays the JupyterLab web interface. At the top left, the 'co' logo is visible. The main title bar shows 'Untitled.ipynb' with a star icon, which is highlighted by a purple rectangular box. Below the title bar, a menu bar contains the following items: File, Edit, View, Insert, Runtime, Tools, Help, and a link for 'All changes saved'. On the right side of the title bar, there are three icons: a comment icon labeled 'Comment', a share icon labeled 'Share', and a settings gear icon. Below the menu bar, a toolbar shows '+ Code' and '+ Text' buttons. The main workspace area is currently empty, featuring a large play button icon on the left and a toolbar on the right with icons for undo, redo, link, comment, settings, copy, paste, and a trash can. The left sidebar contains icons for a menu, search, a variable '{x}', a file explorer, and a code editor icon. The bottom right corner of the interface has a close button 'x'.

# Barra de Menú similar a cualquiera de Google docs





Untitled.ipynb ☆

File Edit View Insert Runtime Tools Help [All changes saved](#)

Comment

Share



+ Code + Text

Connect



**Barra Lateral**







Untitled.ipynb ☆

File Edit View Insert Runtime Tools Help [All changes saved](#)

Comment Share ⚙

+ Code + Text

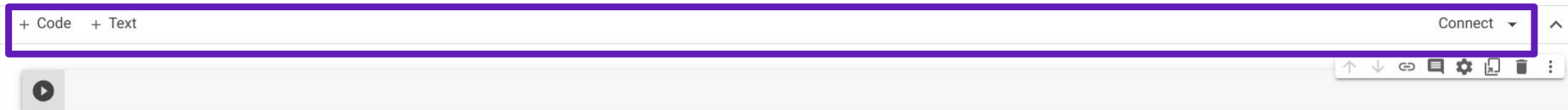
Connect ^



## Celda

Aquí es donde ingresaremos nuestro código!

## Barra de Acceso Rápido



Permite crear celdas de código y texto rápidamente (ya lo vamos a ver).

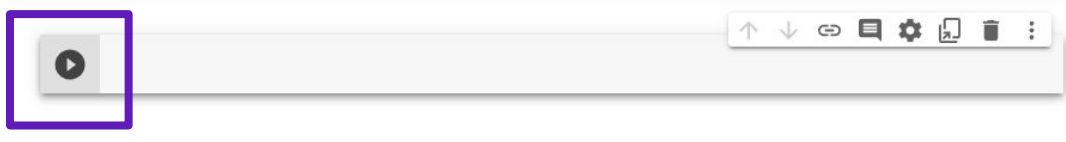
“Connect”

Permite conectar a un “Runtime” de manera rápida.

# Celdas

Hay distintos tipos de Celdas. Para ver cuales existen investiguemos el Menú **Insert**

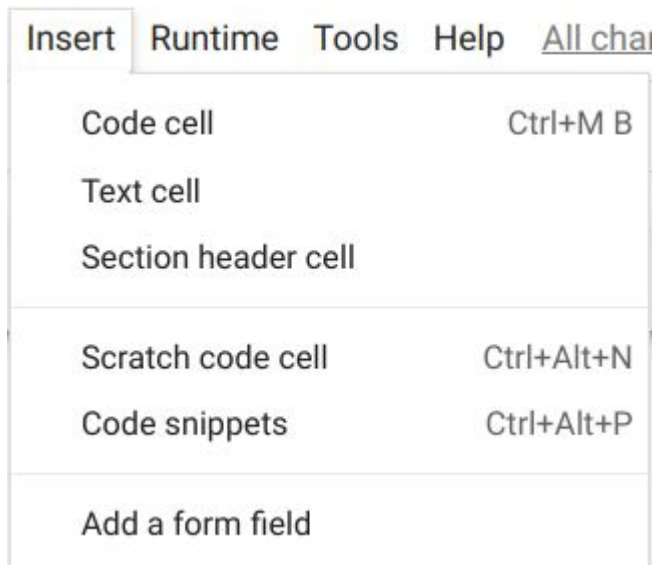
**Code Cell (Celda de Código):** En estas celdas se ingresa el código que luego podemos ejecutar pulsando el símbolo del play



**Text Cell (Celda de Texto):** Google Colab combina texto en el código para, por ejemplo, poder explicar, hacer reportes, etc.

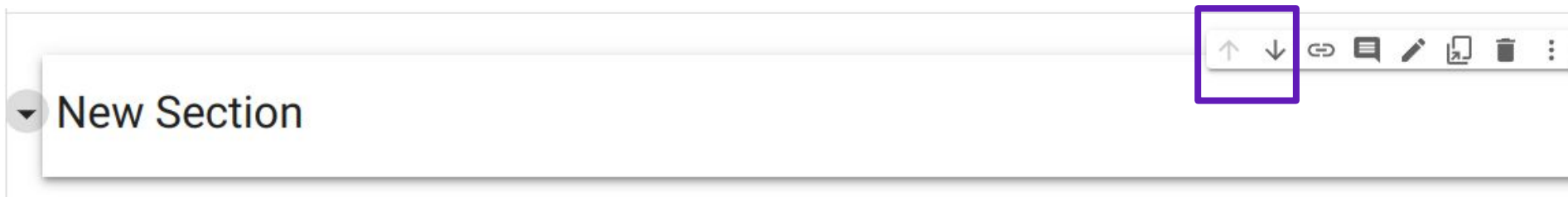
**Section Header Cell:** Es similar a una Text Cell, pero define el comienzo de una sección.

Ahora vamos a crear nuestra notebook para el día de hoy!



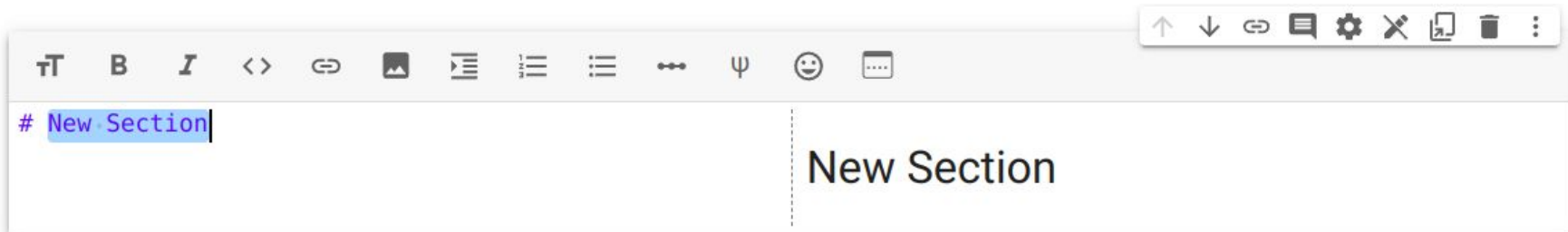
# Creando nuestra Notebook

Inserta una “Section Header Cell”



Con las Flechitas que figuran en el menú de la celda, se puede subir y bajar la celda a la altura del documento que uno desee

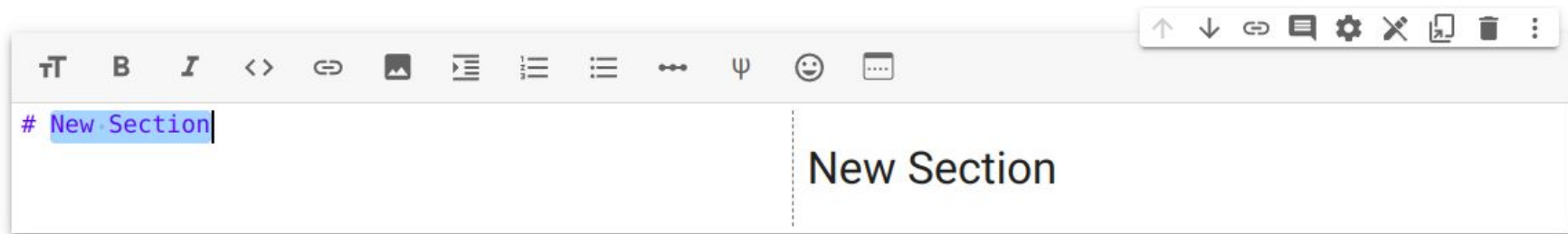
Para editar la celda haz doble click en ella



# Creando nuestra Notebook

En la parte de la **Izquierda** aparece el texto sin formatear que ingresamos.

En la parte de la **Derecha** aparece el texto que ingresamos con el formato aplicado.



Nombra a esta sección:

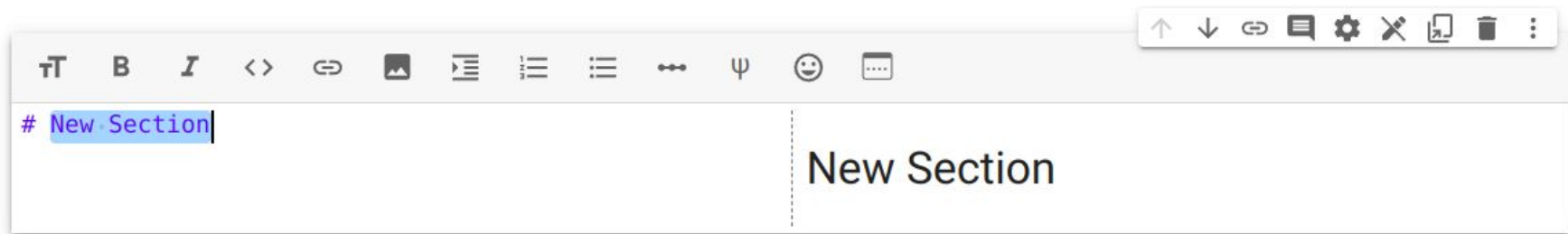
**# Nociones básicas de programación en Python**

El símbolo “#” permite definir distintos niveles de secciones

# Creando nuestra Notebook

En la parte de la **Izquierda** aparece el texto sin formatear que ingresamos.

En la parte de la **Derecha** aparece el texto que ingresamos con el formato aplicado.



Nombra a esta sección:

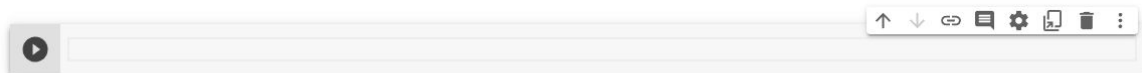
**# Nociones básicas de programación en Python**

El símbolo “#” permite definir distintos niveles de secciones

# Creando nuestra Notebook

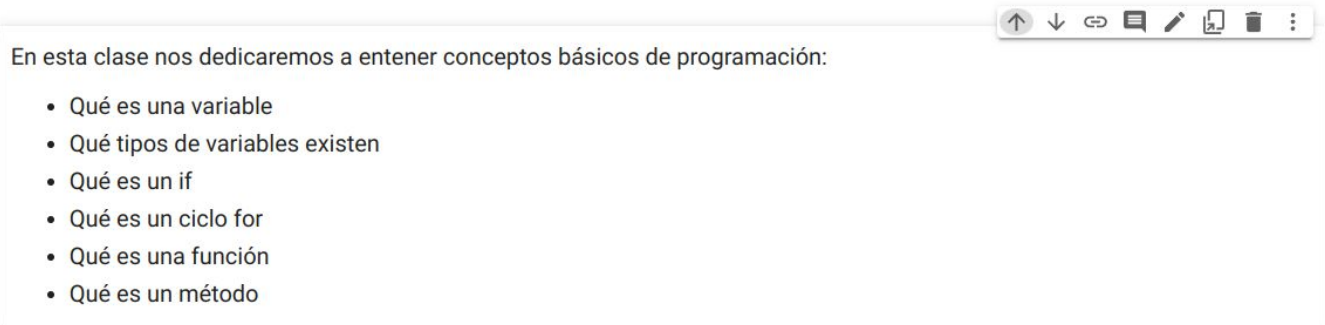
Tu notebook ahora debería verse así:

▼ Nociones básicas de programación en Python



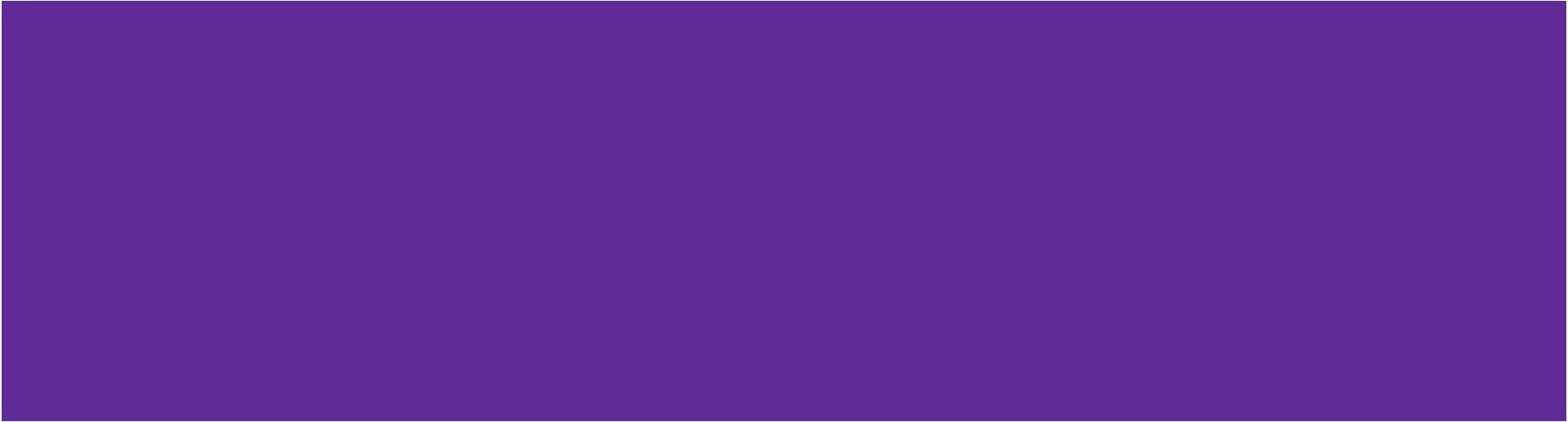
Prueba Ingresar ahora una celda de texto utilizando ya sea la barra de acceso rápido (**+Code**) o bien desde el menú **Insert → Text Cell**

▼ Nociones básicas de programación en Python



[ ]

# ¿Qué es programar?





# ¿Qué es un programa o un script?

- Un *programa* o *script* es básicamente una serie de instrucciones (o *algoritmo*) que fueron escritas usando una gramática específica que puede ser interpretada por una computadora.
- Hay varias gramáticas posibles, llamadas *lenguajes de programación*. En esta materia vamos a utilizar principalmente **Python**.
- Casi toda nuestra vida diaria puede pensarse como un **algoritmo**!
- Algunos ejemplos de **algoritmos** en nuestra vida diaria son:
  - **Una receta de cocina** (hay que seguir las instrucciones de arriba a abajo en orden)
  - La **fórmula resolvente de la cuadrática** en matemática (menos b mas menos la raíz cuadrada...).

# La base de la programación

- Diferentes lenguajes de programación tienen diferentes herramientas al momento de programar y es un mundo muy amplio, pero hay **tres** que tienen un papel central en la mayoría de los lenguajes de programación:
  - las **Variables**,
  - los **Condicionales**
  - los **Ciclos**.
- A continuación vamos a ver una pequeña introducción a cada una de ellas.
- Lo importante **no** es aprender exactamente cómo se usan (los paréntesis, los espacios, los iguales, etc), todo esto se Googlea.
- Lo que **sí es importante** es aprender la lógica detrás de cada herramienta y cuando conviene aplicarla

# Nociones básicas de programación en Python



# Variables

Las variables son contenedores que nos permiten almacenar datos, por ejemplo:



```
x = 38
y = "Hola Mundo"
print(x)
print(y)
```

Creamos una variable x

Con el símbolo del igual “=” **le asignamos** el valor **38**

Creamos una variable y

Con el símbolo del igual “=” **le asignamos** el valor “**Hola Mundo**”

**Usamos la función print para imprimir los datos almacenados en la variable x y en la variable y**

**Ejecutamos el código haciendo click en el play**

Nos aparece el resultado:

✓  
0s

```
x = 38
y = "Hola Mundo"
print(x)
print(y)
```

```
38
Hola Mundo
```



## Tip!

El símbolo “#” permite agregar **comentarios** al código de manera que sea más legible en el futuro.

Un código bien comentado permite que cualquier otra persona pueda entenderlo, incluyendo nuestro yo del futuro



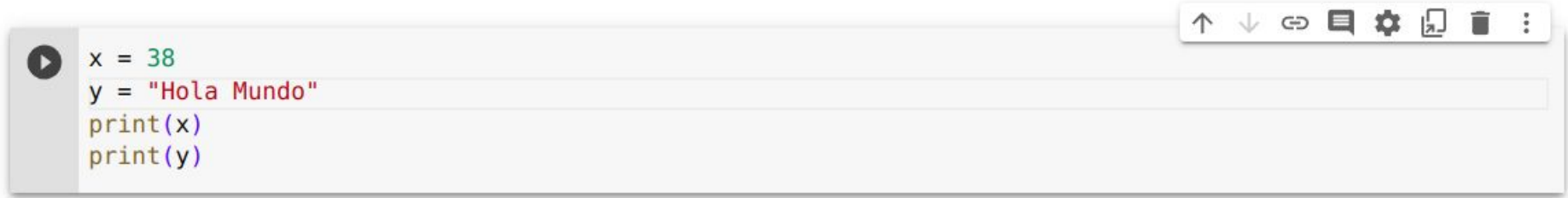
```
# Guardo 38 en la variable x
x = 38
# Guardo "Hola Mundo" en la variable y
y = "Hola Mundo"
print(x) # Imprimo el contenido de x
print(y) # Imprimo el contenido de y
```

```
38
Hola Mundo
```



# Nombre de Variables

Las variables tienen un nombre: en nuestro caso **x** e **y**



```
x = 38
y = "Hola Mundo"
print(x)
print(y)
```

Los nombres de las variables:

- Pueden ser cortos o largos (y descriptivos).
- Son **case sensitive**, es decir **no es lo mismo** si uso minúsculas o mayúsculas

```
mivariable = "Hola Mundo"
MIVARIABLE = "Hola Mundo"
mi_variable = "Hola Mundo"
MI_VARIABLE = "Hola Mundo"
miVariable = "Hola Mundo"
```

Son todas variables con **distinto nombre y por lo tanto distintas variables**

# Nombres Ilegales Variables

Los nombres de las variables **NO PUEDEN**:

- Empezar con un número

**2miVariable** ❌

**miVariable2** ✅

- Tener el símbolo -

**mi-Variable** ❌

**mi\_Variable** ✅

- Tener espacios

**mi variable** ❌

**mi\_variable** ✅

# Tipos de Datos que se pueden almacenar en una variable

Existen numerosos tipos de datos, los más comunes son:

Tipo texto o *string*: `str`

“Hello word” → Noten que va entre “comillas dobles” o también puede usarse ‘comillas simples’

Tipo numérico: `int`, `float` (1 2      0.2)

Tipo booleanos: `bool`

True False 1 0

Tipos secuenciales: `list`, `tuple`, `range`

Tipos mapeables: `dict`



# Algoritmos Cotidianos: Receta Chocotorta

- Elementos necesarios
  - Chocolinas
  - Dulce de leche
  - Queso crema
  - Leche o café para remojar las chocolinas
  - Opcionalmente, cacao en polvo o azúcar impalpable
- Algoritmo
  - Mezclar el dulce de leche con el queso crema
  - Preparar una fuente donde entre la chocotorta
  - Repetir 4 veces:
    - Remojar varias chocolinas en leche o café y colocarlas en la fuente, formando una capa.
    - Colocar una capa de la mezcla sobre la capa de galletitas.
  - Enfriar por 1 hora en refrigerador.
  - Si queremos, espolvorear con cacao en polvo o azúcar impalpable.



# Algoritmos cotidianos: traducir a programa

- Algoritmo
  - Mezclar el dulce de leche con el queso crema
  - Preparar una fuente donde entre la chocotorta
  - Repetir 4 veces:
    - Remojar varias chocolinas en leche o café y colocarlas en la fuente, formando una capa.
    - Colocar una capa de la mezcla sobre la capa de galletitas.
  - Enfriar por 1 hora en refrigerador.
  - Si queremos, espolvorear con cacao en polvo o azúcar impalpable.

```
# Queremos enseñarle a una máquina  
# a hacer esta chocotorta y para hacerla  
# fácil supongamos que le podemos dar  
# comandos con print, por ejemplo:
```

```
print("Mezclar el dulce de leche con el queso  
crema")
```

```
# Una posible versión del algoritmo sería:
```

# Algoritmos cotidianos: traducir a programa

- Algoritmo
  - Mezclar el dulce de leche con el queso crema
  - Preparar una fuente donde entre la chocotorta
  - Repetir 4 veces:
    - Remojar varias chocolinas en leche o café y colocarlas en la fuente, formando una capa.
    - Colocar una capa de la mezcla sobre la capa de galletitas.
  - Enfriar por 1 hora en refrigerador.
  - Si nos gusta, espolvorear con cacao en polvo o azúcar impalpable.

```
print("Mezclar el dulce de leche con el queso  
crema")  
print("Preparar una fuente")
```

```
print("Remojar chocolinas en leche")  
print("Crear la capa 1 de chocolinas")  
print("Colocar capa de mezcla")  
print("Remojar chocolinas en leche")  
print("Crear la capa 2 de chocolinas")  
print("Colocar capa de mezcla")  
print("Remojar chocolinas en leche")  
print("Crear la capa 3 de chocolinas")  
print("Colocar capa de mezcla")  
print("Remojar chocolinas en leche")  
print("Crear la capa 4 de chocolinas")  
print("Colocar capa de mezcla")
```

```
print("Enfriar en refrigerador")  
print("Espolvorear con cacao en polvo")
```

# Algoritmos cotidianos: traducir a programa

- Algoritmo
  - Mezclar el dulce de leche con el queso crema
  - Preparar una fuente donde entre la chocotorta
  - Repetir 4 veces:
    - Remojar varias chocolinas en leche o café y colocarlas en la fuente, formando una capa.
    - Colocar una capa de la mezcla sobre la capa de galletitas.
  - Enfriar por 1 hora en refrigerador.
  - Si nos gusta, espolvorear con cacao en polvo o azúcar impalpable.

```
print("Mezclar el dulce de leche con el queso  
crema")  
print("Preparar una fuente")
```

```
print("Remojar chocolinas en leche")  
print("Crear la capa 1 de chocolinas")  
print("Colocar capa de mezcla")  
print("Remojar chocolinas en leche")  
print("Crear la capa 2 de chocolinas")  
print("Colocar capa de mezcla")  
print("Remojar chocolinas en leche")  
print("Crear la capa 3 de chocolinas")  
print("Colocar capa de mezcla")  
print("Remojar chocolinas en leche")  
print("Crear la capa 4 de chocolinas")  
print("Colocar capa de mezcla")
```

Estoy  
repetiendo  
4 veces  
lo mismo  
(casi)

```
print("Enfriar en refrigerador")  
print("Espolvorear con cacao en polvo")  
# ¿Qué pasa si un día quiero remojar en café?  
# ¿Qué pasa si no quiero espolvorear con nada?
```

# Mejorando el programa: Usemos Variables

- Algo que puede variar es
- En este caso la variable `remojar_en` contiene el valor "cafe" (entre comillas porque es un *string*).
- Fíjense que los nombres de las variables son descriptivos: `remojar_en` y `espolvorear_con`

```
remojar_en="cafe" # puede ser "leche" o "cafe"  
espolvorear_con="cacao en polvo" # puede ser "cacao en  
polvo" o "azúcar impalpable"
```

```
print("Mezclar el dulce de leche con el queso crema")  
print("Preparar una fuente")
```

```
print(f"Remojar chocolinas en {remojar_en}")  
print("Crear la capa 1 de chocolinas")  
print("Colocar capa de mezcla")  
print(f"Remojar chocolinas en {remojar_en}")  
print("Crear la capa 2 de chocolinas")  
print("Colocar capa de mezcla")  
print(f"Remojar chocolinas en {remojar_en}")  
print("Crear la capa 3 de chocolinas")  
print("Colocar capa de mezcla")  
print(f"Remojar chocolinas en {remojar_en}")  
print("Crear la capa 4 de chocolinas")  
print("Colocar capa de mezcla")
```

```
print("Enfriar en refrigerador")  
print(f"Espolvorear con {espolvorear_con}")
```

# Mejorando el programa: Usemos Variables

- Algo que puede variar es
- En este caso la variable `remojar_en` contiene el valor "cafe" (entre comillas porque es un *string*).
- Fíjense que los nombres de las variables son descriptivos: `remojar_en` y `espolvorear_con`

**Esto también se puede mejorar!**

```
remojar_en="cafe" # puede ser "leche" o "cafe"  
espolvorear_con="cacao en polvo" # puede ser "cacao en  
polvo" o "azúcar impalpable"
```

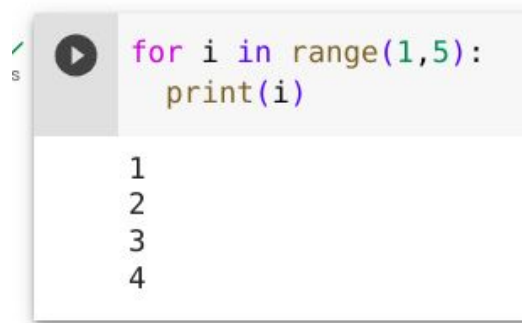
```
print("Mezclar el dulce de leche con el queso crema")  
print("Preparar una fuente")
```

```
print(f"Remojar chocolinas en {remojar_en}")  
print("Crear la capa 1 de chocolinas")  
print("Colocar capa de mezcla")  
print(f"Remojar chocolinas en {remojar_en}")  
print("Crear la capa 2 de chocolinas")  
print("Colocar capa de mezcla")  
print(f"Remojar chocolinas en {remojar_en}")  
print("Crear la capa 3 de chocolinas")  
print("Colocar capa de mezcla")  
print(f"Remojar chocolinas en {remojar_en}")  
print("Crear la capa 4 de chocolinas")  
print("Colocar capa de mezcla")
```

```
print("Enfriar en refrigerador")  
print(f"Espolvorear con {espolvorear_con}")
```

# Mejorando el programa: Ciclos

- Los ciclos son estructuras que nos permiten repetir un mismo código varias veces. El ciclo más común es el `for`, donde el código dentro del `for` se ejecuta varias veces.
- `i` es una variable que indica en qué iteración se encuentra el ciclo.
- `i` va tomando variables en (in) el rango 1 a 5 (sin incluir el 5)



```
for i in range(1,5):  
    print(i)
```

1  
2  
3  
4

```
remojar_en="cafe" # puede ser "leche" o "cafe"  
espolvorear_con="cacao en polvo" # puede ser "cacao  
en polvo" o "azúcar impalpable"  
  
print("Mezclar el dulce de leche con el queso crema")  
print("Preparar una fuente")  
  
for i in range(1,5):  
    print(f"Remojar chocolinas en {remojar_en}")  
    print(f"Crear la capa {i} de chocolinas")  
    print("Colocar capa de mezcla")  
  
print("Enfriar en refrigerador")  
print(f"Espolvorear con {espolvorear_con}")
```

# El programa aún no cumple con lo especificado

- Algoritmo
  - Mezclar el dulce de leche con el queso crema
  - Preparar una fuente donde entre la chocotorta
  - Repetir 4 veces:
    - Remojar varias chocolinas en leche o café y colocarlas en la fuente, formando una capa.
    - Colocar una capa de la mezcla sobre la capa de galletitas.
  - Enfriar por 1 hora en refrigerador.
  - **Si nos gusta, espolvorear con cacao en polvo o azúcar impalpable.**

```
remojar_en="cafe" # puede ser "leche" o "cafe"
espolvorear_con="cacao en polvo" # puede ser "cacao
en polvo" o "azúcar impalpable"

print("Mezclar el dulce de leche con el queso crema")
print("Preparar una fuente")

for i in range(1,5):
    print(f"Remojar chocolinas en {remojar_en}")
    print(f"Crear la capa {i} de chocolinas")
    print("Colocar capa de mezcla")

print("Enfriar en refrigerador")
print(f"Espolvorear con {espolvorear_con}")
```



# Mejorando el programa: Condicionales

- Los condicionales permiten crear secciones de código que se van a ejecutar solo si se cumple (o no se cumple) una condición.
- El condicional más común es el if,  
  
Pregunta si una **condición** es ó VERDADERA ó FALSA.
- Si la condición es **VERDADERA**, entonces se ejecuta el código dentro del if.
- En este caso estoy preguntando si la variable `espolvorear` es igual (`==`) a la palabra "si", en cuyo caso escribo la frase usando `print`.

```
remojar_en="cafe" # puede ser "leche" o "cafe"
espolvorear_con="cacao en polvo" # puede ser "cacao
en polvo" o "azúcar impalpable"

espolvorear="si" # puede ser "si" o "no"

print("Mezclar el dulce de leche con el queso crema")
print("Preparar una fuente")

for i in range(1,5):
    print(f"Remojar chocolinas en {remojar_en}")
    print(f"Crear la capa {i} de chocolinas")
    print("Colocar capa de mezcla")

print("Enfriar en refrigerador")

if espolvorear == "si":
    print(f"Espolvorear con {espolvorear_con}")

# otra opción es:

# if espolvorear != "no":
#     print(f"Espolvorear con {espolvorear_con}")
```

# Mejorando el programa: Listas

- Las listas permiten almacenar numerosos elementos en una única variable.
- Se crean colocando los datos entre corchetes.

```
ingredientes=["chocolinas","dulce de leche","queso crema"]
```

- Los elementos de la lista están ordenados: cada elemento tiene un lugar en la lista. Si agregamos uno, se agrega al final.
- Los elementos de la lista pueden modificarse y pueden estar duplicados.
- Los elementos de la lista están indexados. El primer elemento tiene el índice 0, el segundo elemento tiene el índice 1, etc.

```
remojar_en="cafe" # puede ser "leche" o "cafe"  
espolvorear_con="cacao en polvo" # puede ser "cacao en polvo" o "azúcar impalpable"  
espolvorear="si" # puede ser "si" o "no"
```

```
ingredientes=["chocolinas","dulce de leche","queso crema"]
```

```
print(f"Mezclar el {ingredientes[1]} con el {ingredientes[2]}")  
print("Preparar una fuente")
```

```
for i in range(1,5):  
    print(f"Remojar {ingredientes[0]} en {remojar_en}")  
    print(f"Crear la capa {i} de {ingredientes[0]}")  
    print("Colocar capa de mezcla")
```

```
print("Enfriar en refrigerador")
```

```
if espolvorear == "si":  
    print(f"Espolvorear con {espolvorear_con}")
```