



Tecnológico
de Monterrey

| Educación
Continua

Fundamentos de Python

Programas LIVE

SS1 | Aprender

Bienvenida y presentación

Bienvenida



**“El viaje de mil millas
comienza con un paso”**

-Lao Tzu

Objetivo principal del módulo



- Utilizarás **estatutos y estructuras de datos** en **Python** que involucren **ciclos y estatutos condicionales**

Objetivos particulares de la sesión



- Discriminarás entre los diferentes **tipos de datos** que intervienen en un problema para **representarlo en Python**
- Definirás las **estructuras de datos** más convenientes para **representar un problema en Python**
- Integrarás los conocimientos adquiridos para **resolver un problema** a través de **Python**

Agenda

1

Sesión Síncrona 1

Aprender

2

Trabajo asíncrono 1

Profundizar | Ruta de Aprendizaje

3

Sesión Síncrona 2

Preparar para Aplicar

4

Trabajo asíncrono 2

Aplicar en el trabajo | Reto

1

2

3

4



Dinámica de participación y convivencia

Sesión grupal:

- Las dudas durante la sesión se manejarán a través del chat. Éstas serán respondidas en los momentos definidos para ello
- Participar en las dinámicas propuestas durante la sesión de acuerdo con las instrucciones dadas para cada una

Rooms:

- Los equipos se harán de manera aleatoria y serán sólo para las actividades de práctica
- Estar atentos a la notificación de zoom para unirse al room
- Seguir al pie de la letra las instrucciones para la dinámica en rooms
- Informar al moderador si te encuentras sólo en el room para reasignarte
- Te pedimos no salir del room que tienes asignado. Cualquier reasignación será realizada por parte del tutor o staff



Introducción

- **Python** es uno de los lenguajes más utilizados en **Data Science**. Conocerlo nos permite estar a la vanguardia y no quedarnos rezagados en el uso de tecnologías de punta



Tecnológico
de Monterrey

Educación
Continua

Panorámica de la sesión

1

Sesión Síncrona 1
Aprender

Tema 1: Ambientes de programación

Tema 2: Estructuras de datos

Tema 3: Estatutos condicionales

Tema 4: Estatutos de repetición

Cierre

Panorámica de la sesión

1

Sesión Síncrona 1
Aprender

Tema 1: Ambientes de programación

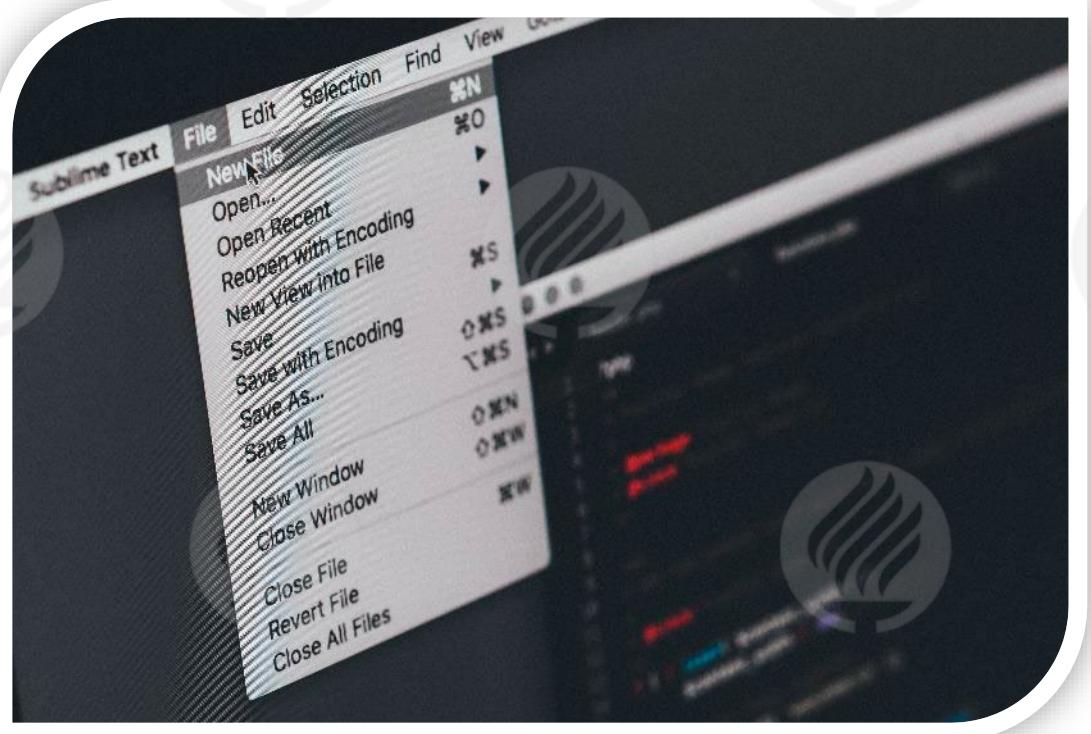
Tema 2: Estructuras de datos

Tema 3: Estatutos condicionales

Tema 4: Estatutos de repetición

Cierre

Propósito | Tema 1



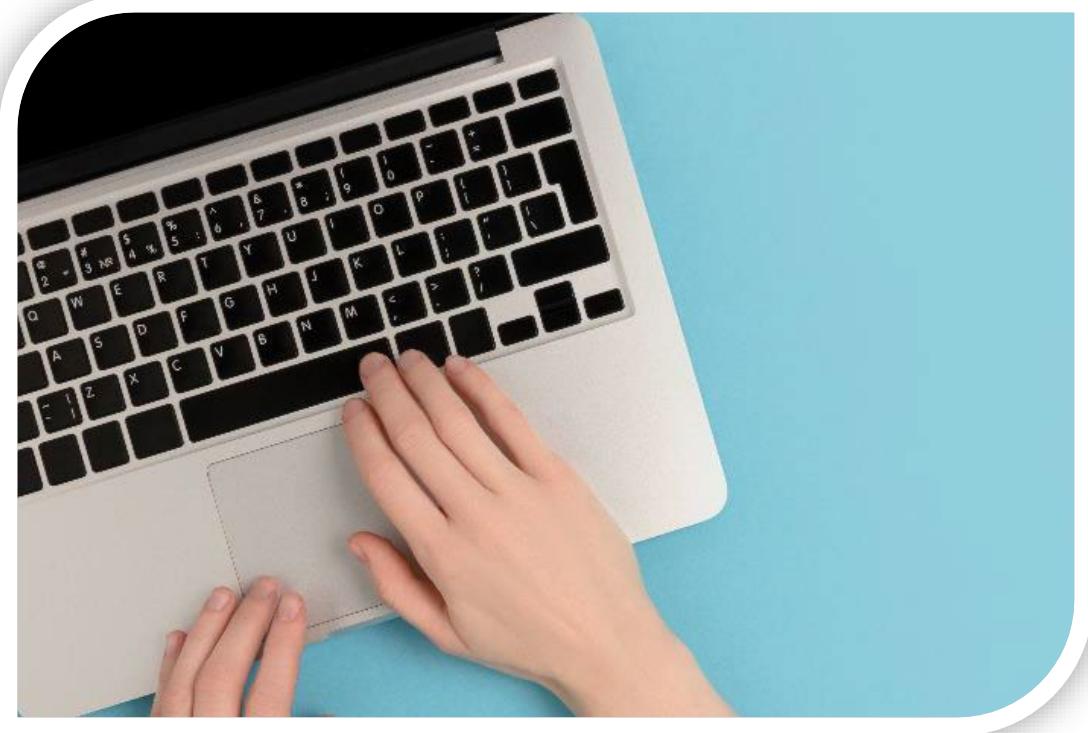
- Muchas veces usamos solo un **editor de texto** para programar, pero podemos perdonos de algunas ventajas que nos ofrecen los **ambientes de programación integrales**
- Aprender a usar un **ambiente de programación** nos permitirá avanzar más rápido en este curso y no tener que preocuparnos por los detalles de **compilación y ejecución de nuestros scripts**



Google Colab (Colaboratory)

- Permite **escribir y ejecutar** código en Python directamente en el navegador
- El código se **ejecuta en un servidor** de Google
- Los documentos se escriben en un ambiente interactivo llamado **Colab Notebook**

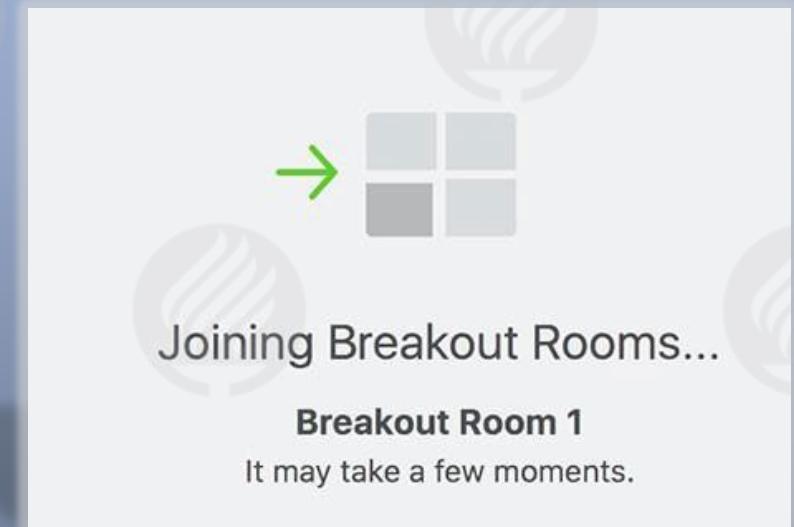
Actividad | Aprendizaje activo



- Entra a **Google Colab** en la liga:
<https://colab.research.google.com>
- Crea un nuevo **Notebook**
- Agrega y ejecuta los siguientes **fragmentos de código**:
 - `print('Hello World')`
 - `print('My name is', 'NOMBRE')`
- Comparte con tus compañeros

Actividad | BreakOut Rooms (BOR)

- 1. Al unirte,
aparecerá un
mensaje**
- 2. Después estarás
con un grupo
pequeño de
compañeros**





- Como puedes ver, ambos **ambientes de programación** te permiten trabajar con scripts de Python de forma **interactiva y colaborativa**
- Actualmente, ¿utilizas alguna **herramienta de programación** en tu trabajo?
- ¿Consideras que alguno de estos **ambientes** facilitaría la **colaboración en tu trabajo**?

Cierre: Concepto clave | Tema 1

Los ambientes de programación en línea no solo proveen herramientas de compilación y ejecución, además habilitan la colaboración y uso de recursos distribuidos

Panorámica de la sesión

1

Sesión Síncrona 1
Aprender

Tema 1: Ambientes de programación

Tema 2: Estructuras de datos

Tema 3: Estatutos condicionales

Tema 4: Estatutos de repetición

Cierre

Propósito | Tema 2



- **Python** es capaz de reconocer el **tipo de dato** que queremos guardar en una variable, sin embargo, es necesario **conocerlos y entenderlos para poder usarlos adecuadamente**
- Si **no** podemos identificarlos correctamente, no vamos a ser capaces de **detectar posibles errores en el código**

Tipos de datos | Tema 2.1



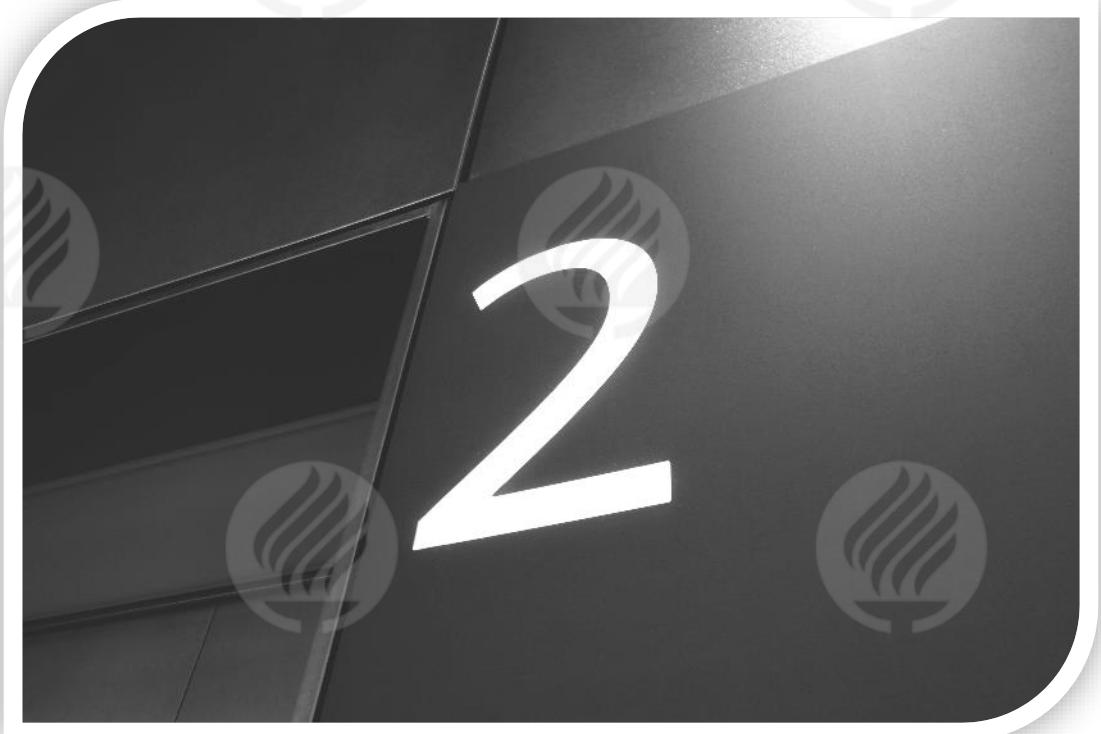
1. Booleanos
2. Numéricos
 - Enteros
 - Flotantes
 - Complejos
3. Cadenas de caracteres

Booleanos | Tema 2.1



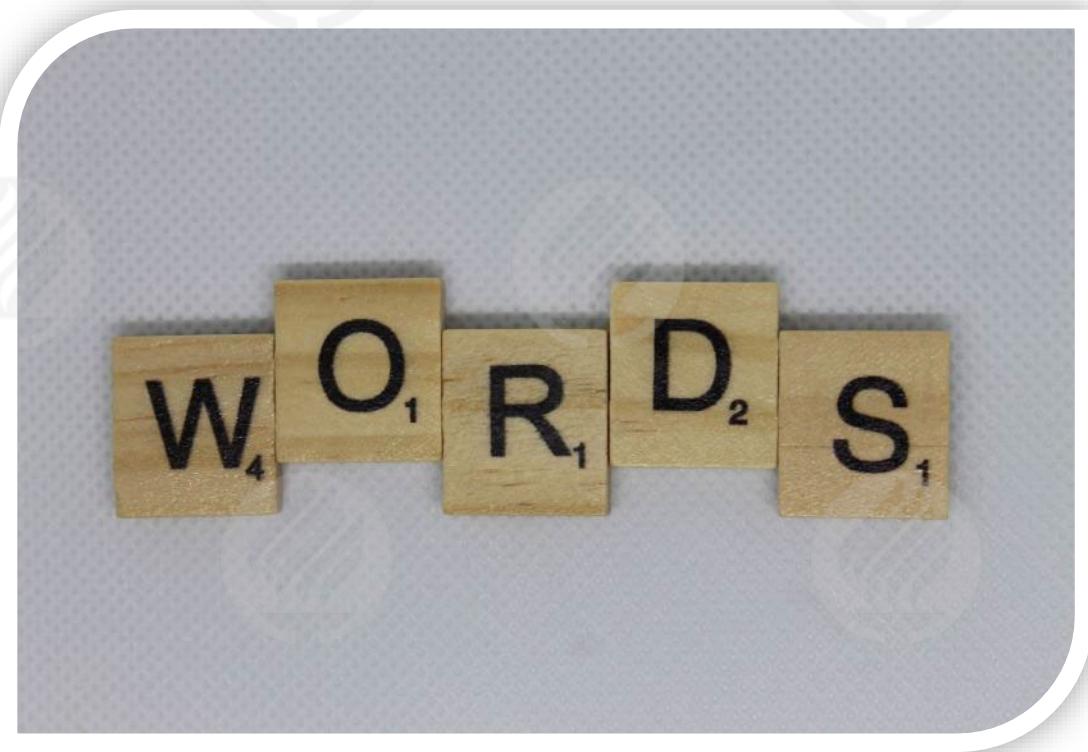
- Pueden tener **dos valores**:
 - a. True
 - b. False
- Se pueden utilizar dentro de **estructuras de control** como **if, while y for**

Numéricos | Tema 2.1



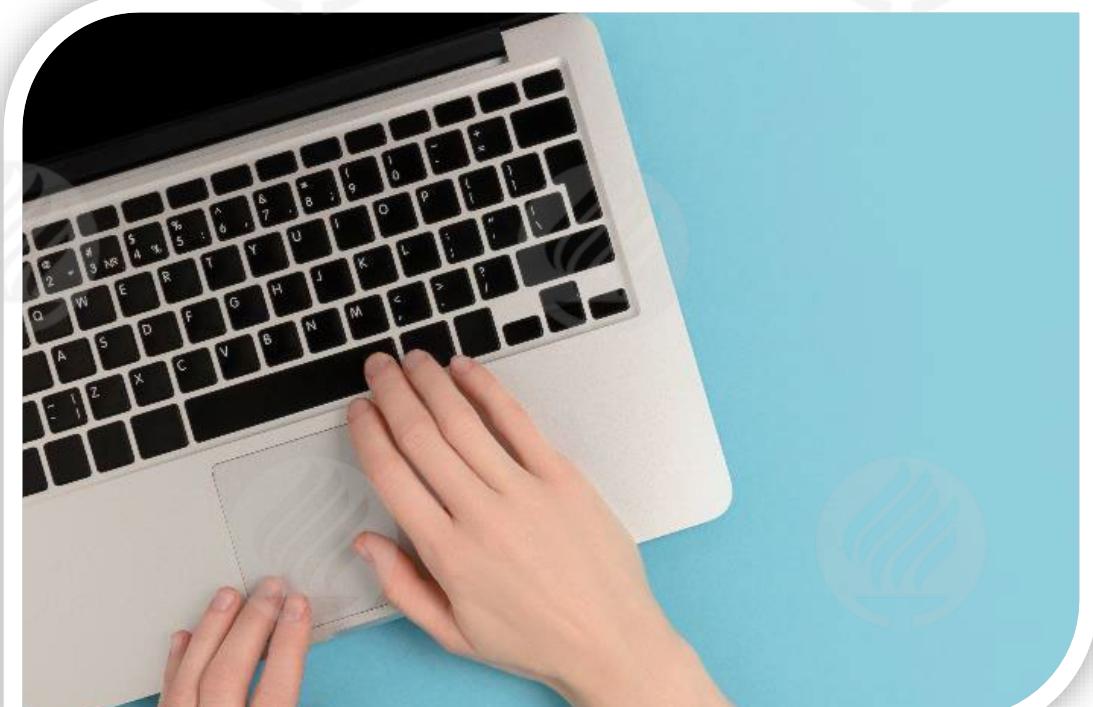
- Los **enteros** pueden expresarse en diferentes bases:
 - ✓ decimal
 - ✓ binaria (0b)
 - ✓ octal (0o)
 - ✓ hexadecimal (0x)
- Los **flotantes** incluyen una parte entera y una decimal
- Los **complejos** incluyen una parte real y una parte imaginaria

Strings | Tema 2.1



- Las cadenas de caracteres o **strings** contienen texto
- No existen las variables **char** como en otros lenguajes, pero se simulan con una **cadena de caracteres de un solo elemento**
- Algunos **métodos de utilidad** son: *capitalize, count, find, lower, replace, rsplit, startswith, strip, swapcase, title, upper*

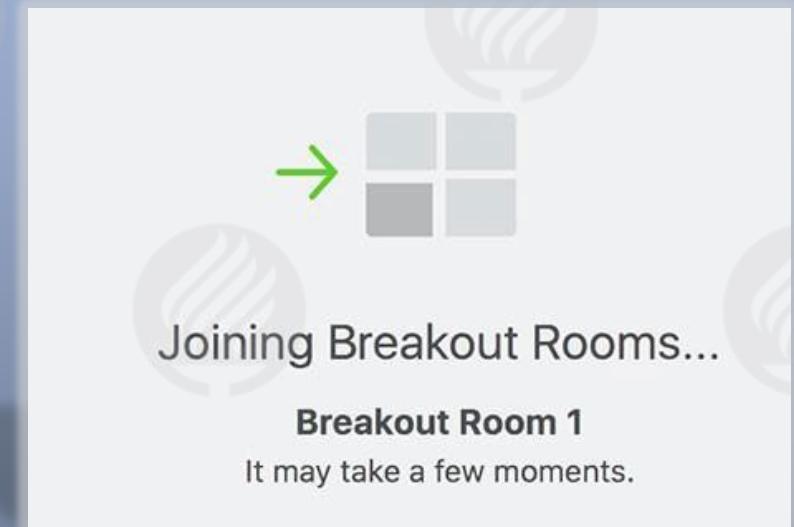
Actividad | Aprendizaje activo



- Crea una nueva libreta en **Colab**:
 - Longitud de una *string*
 - Número de veces que contiene un carácter
 - Capitaliza como título
 - Separa una *string* por espacios
 - Reemplaza las 's' por '\$'

Actividad | BreakOut Rooms (BOR)

- 1. Al unirte,
aparecerá un
mensaje**
- 2. Después estarás
con un grupo
pequeño de
compañeros**





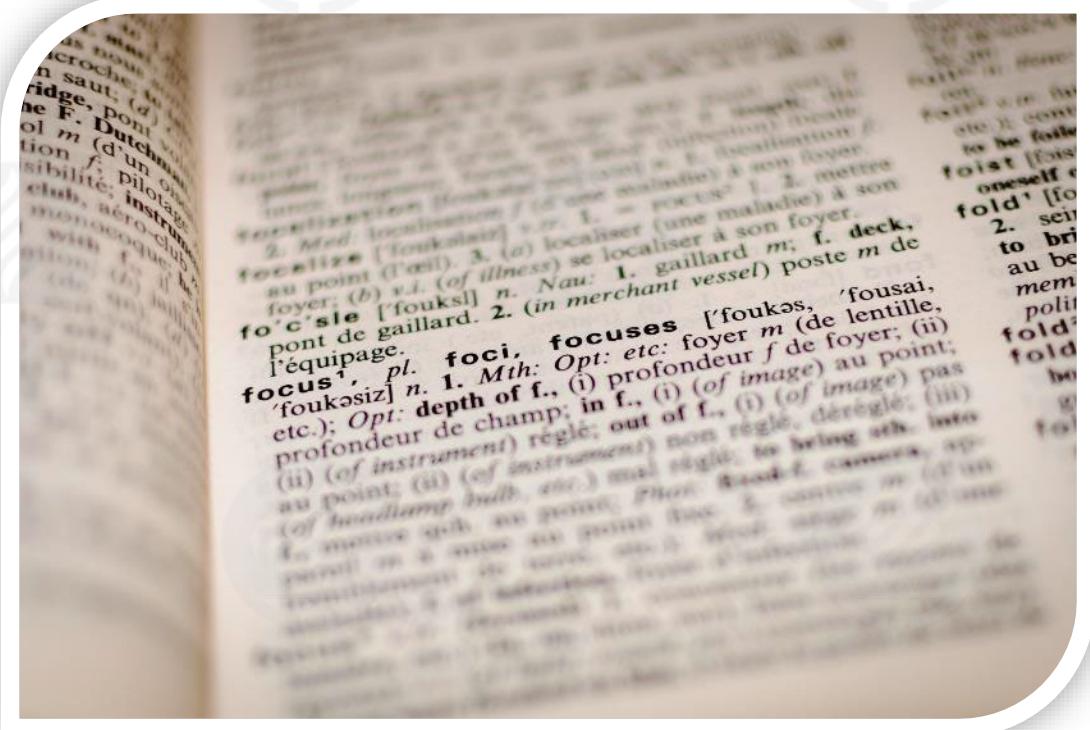
- Podemos utilizar los **tipos de datos** disponibles en Python para **representar y documentar la información** que nos rodea:
 1. ¿Cuántos kilómetros por litro rinde tu automóvil?
 2. ¿De quién recibes más correos en una semana?
 3. ¿Cuántos hijos tienes?
 4. ¿Tienes un gato en casa?

Listas | Tema 2.2



- Las **listas** en Python nos permiten guardar **múltiples valores** (incluso de diferentes tipos) en una sola variable:
 - a. Ordenadas
 - b. Dinámicas
 - c. Permiten duplicidad
- Se representan entre **corchetes cuadrados** y sus elementos se separan por comas

Diccionarios | Tema 2.2



- Los diccionarios en **Python** almacenan pares de **claves y valores**
 - a. No ordenados
 - b. Dinámicos
 - c. No permiten duplicados
- Se representan entre **llaves** y los pares **clave:valor** se separan por **dos puntos**

Actividad | Aprendizaje activo



Crea una nueva libreta en Colab:

- Longitud de una lista
- Longitud de un diccionario
- Lista de palabras
- Frecuencia de palabras



- Algunas veces es más fácil utilizar las **estructuras de datos** que ya están disponibles en Python para **representar información compleja**:
 1. ¿Cómo representarías la lista de empleados?
 2. ¿Cómo representarías la nómina?
 3. ¿Cómo almacenarías un inventario?
 4. ¿Cómo almacenarías las mediciones de temperatura de todo un año?

Cierre: Concepto clave | Tema 2

Los tipos y estructuras de datos en Python nos permiten representar la información que nos rodea de forma que podemos almacenarla, manipularla y visualizarla fácilmente

Panorámica de la sesión

1

Sesión Síncrona 1
Aprender

Tema 1: Ambientes de programación

Tema 2: Estructuras de datos

Tema 3: Estatutos condicionales

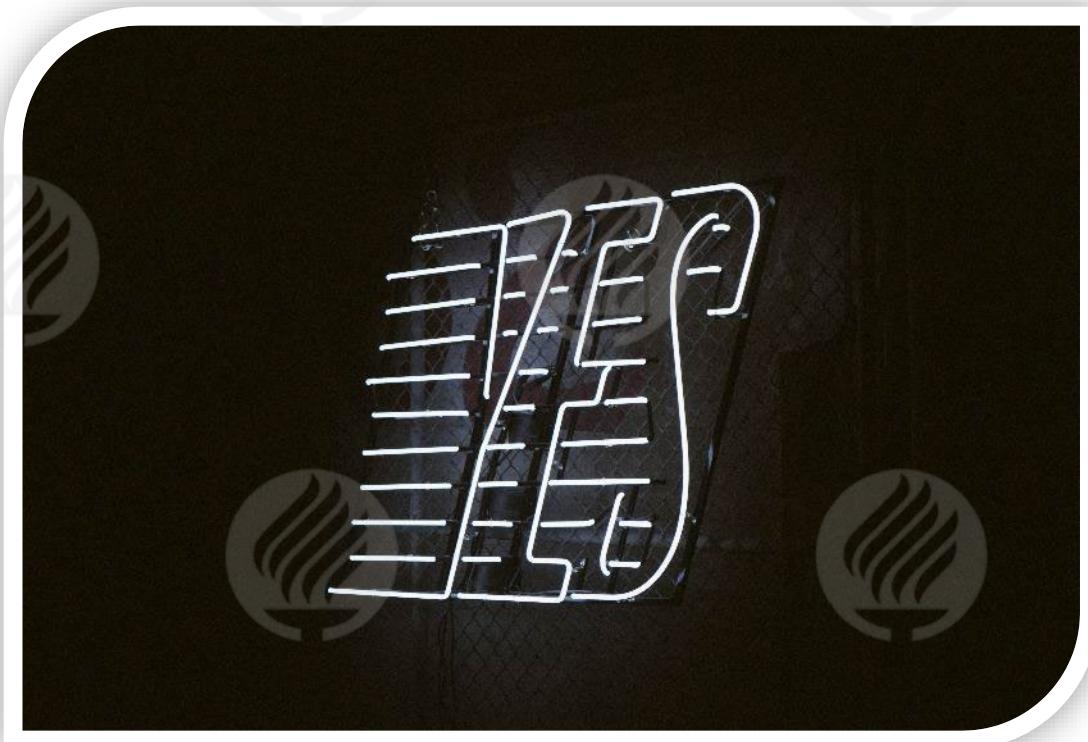
Tema 4: Estatutos de repetición

Cierre

Propósito | Tema 3



- Los **estatutos condicionales** nos permiten realizar programas que deciden cómo deben **proceder de acuerdo con las condiciones dadas**
- **Sin condicionales**, los programas serían incapaces de ejecutar instrucciones diferentes y **siempre realizarían el mismo proceso**



- La **estructura de control if** nos permite ejecutar una sección de código **solo si se cumple una condición**
- La palabra **if** va seguida de la **expresión condicional** sin paréntesis y termina con **dos puntos**
- El **código** que se va a ejecutar va **indentado hacia la derecha**



- **elif** se utiliza para **evaluar una segunda condición**, es decir, si la primera condición no se cumplió, trata con esta otra condición
- **else** se utiliza para indicar el código que se va a ejecutar en caso de que **ninguna condición precedente se haya cumplido**
- Ambos estatutos se terminan con **dos puntos** y el código a ejecutar va **indentado a la derecha**

Actividad | Aprendizaje activo

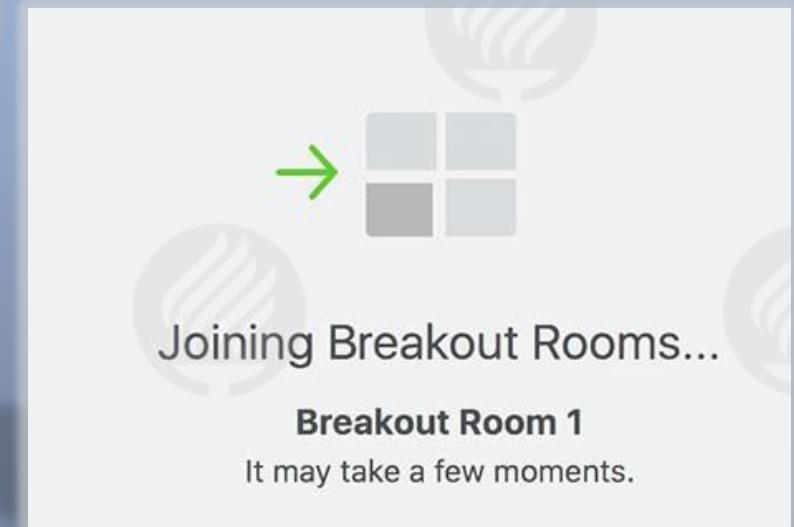


Crea una nueva libreta en Colab:

- Número mayor
- Temperatura
- Triángulo
- Ganancia o pérdida

Actividad | BreakOut Rooms (BOR)

- 1. Al unirte,
aparecerá un
mensaje**
- 2. Después estarás
con un grupo
pequeño de
compañeros**





- Los **estatutos condicionales** te permiten crear programas que **pueden decidir**:
 1. ¿Debes comprar o vender?
 2. ¿Tienes acceso a la información?
 3. ¿En qué categoría debes colocar un producto?

Cierre: Concepto clave | Tema 3

Los **estatutos condicionales** son una herramienta poderosa que le da a los programas la **capacidad de decisión**

Panorámica de la sesión

1

Sesión Síncrona 1
Aprender

Tema 1: Ambientes de programación

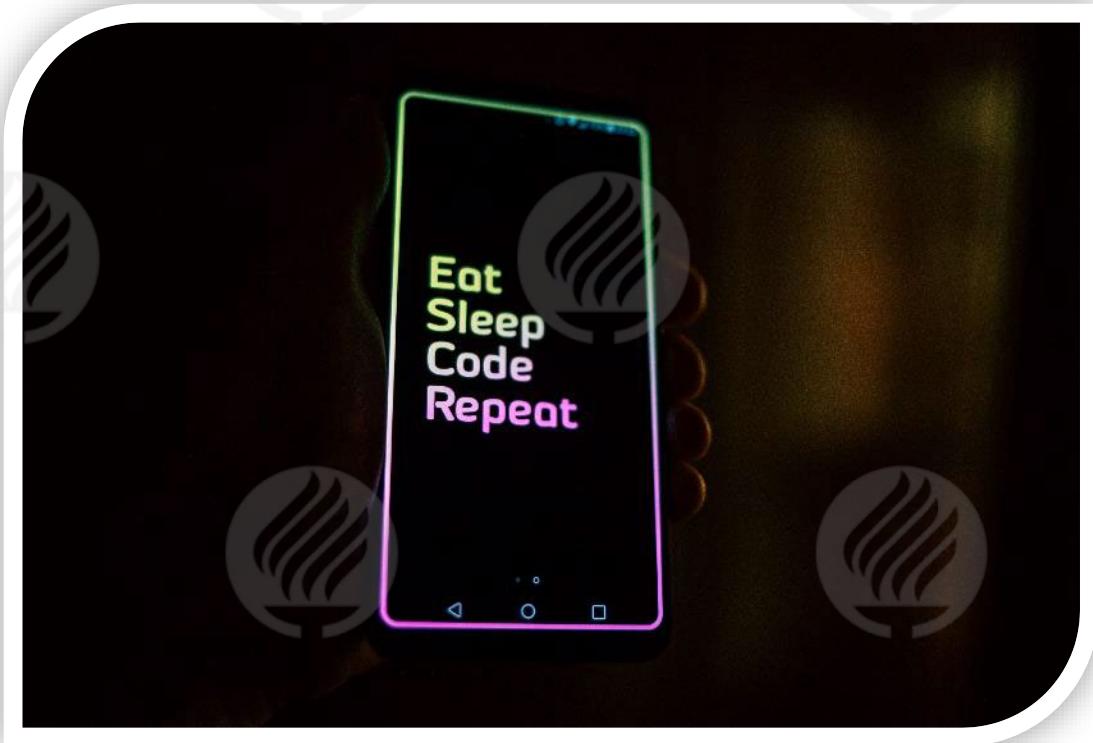
Tema 2: Estructuras de datos

Tema 3: Estatutos condicionales

Tema 4: Estatutos de repetición

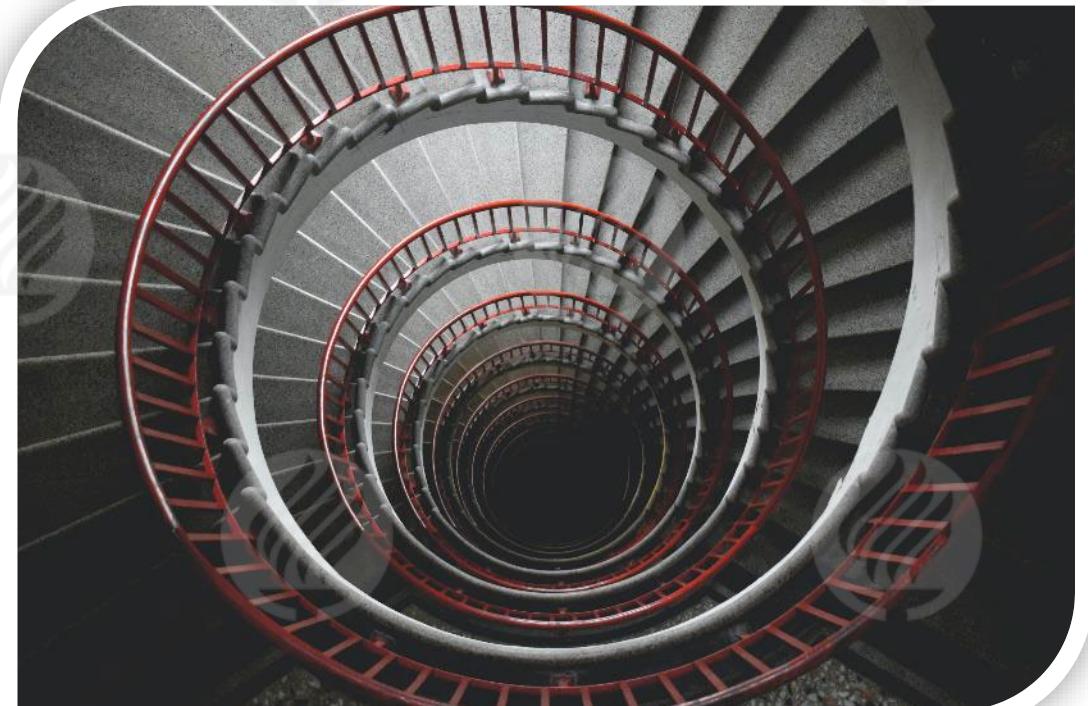
Cierre

Propósito | Tema 4



- Los **estatutos de repetición** nos permiten ejecutar un mismo fragmento de código **tantas veces como sea necesario**
- **No utilizarlos** implicaría más código y por tanto **mayor probabilidad de cometer errores**

While | Tema 4.1



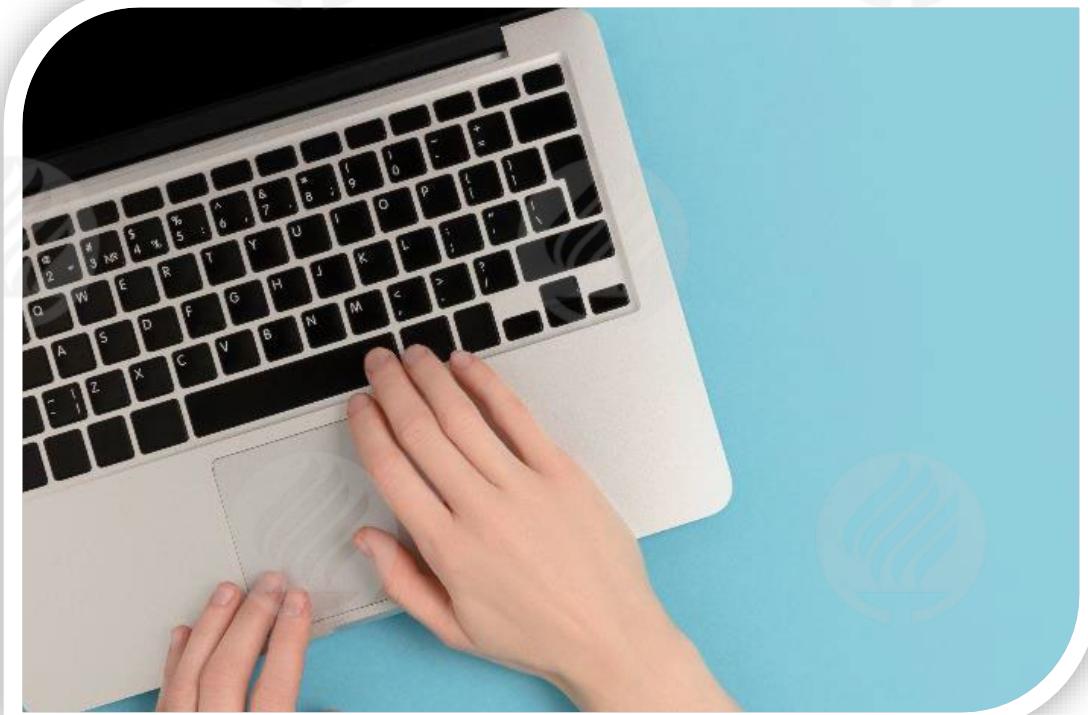
- El **estatuto while** nos permite crear **ciclos indeterminados**, es decir, repetir un mismo fragmento de código **mientras una condición se cumpla**
- El **estatuto break** nos permite **salir del ciclo** aún cuando la condición siga siendo verdadera
- El **estatuto continue** detiene la ejecución del ciclo actual y **continua con la siguiente**

While | Tema 4.1



- La **estructura** consiste en la palabra reservada **while** seguida de la **condición** y **dos puntos**
- El **código** que se va a ejecutar en caso de que la condición se cumpla **se indenta hacia la derecha**

Actividad | Aprendizaje activo



Crear una nueva libreta en Colab:

- Suma y promedio
- Tabla del 6
- Potencia
- División



- El **estatuto while** te permite crear programas que **pueden repetir**:
 1. Sumar todas las facturas
 2. Calcular el promedio de ventas
 3. Organizar todos los correos en tu bandeja de entrada

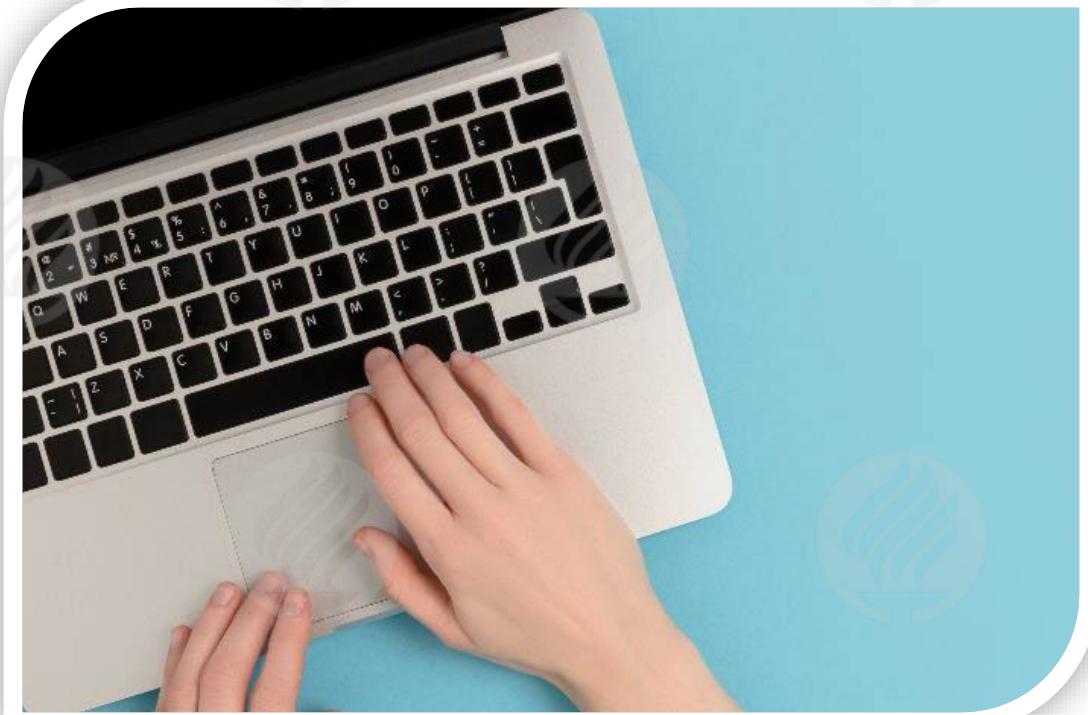


- El **estatuto for** nos permite crear **ciclos determinados**, es decir, repetir un mismo fragmento de código una **cantidad determinada de veces**
- El **estatuto break** nos permite **salir del ciclo** aún cuando la condición siga siendo verdadera
- El **estatuto continue** detiene la ejecución del ciclo actual y **continua con la siguiente**



- Los **ciclos for** nos permiten también **iterar** sobre:
 1. Una cadena de caracteres
 2. Una lista
 3. Un rango

Actividad | Aprendizaje activo

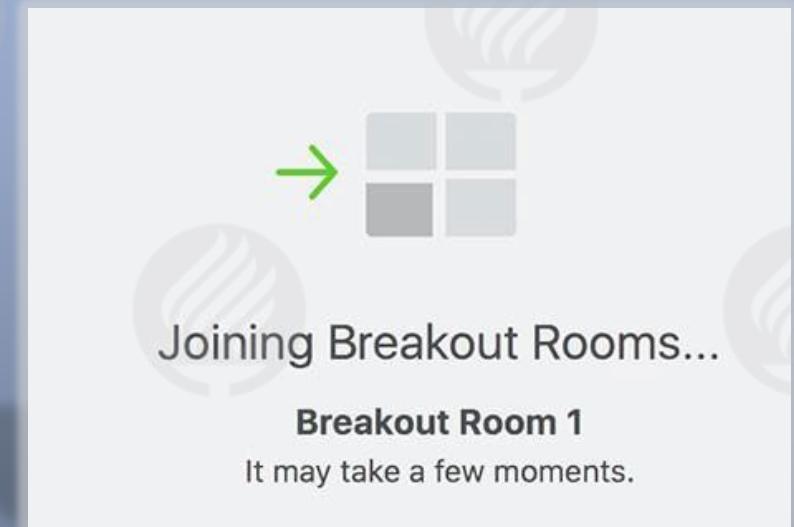


Crea una nueva libreta en Colab:

- Imprime elementos de una lista
- Imprime claves de un diccionario
- Número de vocales y consonantes
- Promedio de una lista

Actividad | BreakOut Rooms (BOR)

- 1. Al unirte,
aparecerá un
mensaje**
- 2. Después estarás
con un grupo
pequeño de
compañeros**





- El **estatuto for** te permite crear programas que pueden **repetir una cantidad conocida de veces**:
 1. Promediar la temperatura de todos los días de un año
 2. Calcular ganancia o pérdida
 3. Recordarte todas tus citas del día

Cierre: Concepto clave | Tema 4

Los estatutos de repetición son una herramienta poderosa que le da a los programas la capacidad de iterar

Panorámica de la sesión

1

Sesión Síncrona 1
Aprender

Tema 1: Ambientes de programación

Tema 2: Estructuras de datos

Tema 3: Estatutos condicionales

Tema 4: Estatutos de repetición

Cierre

Cierre de la sesión

Cierre | Sesión Sincrónica 1 [Aprender]



1. ¿Qué información puedes **representar** para analizarla, manipularla, almacenarla y visualizarla?
2. ¿En qué situaciones debes **tomar decisiones**?
3. ¿Cuándo es necesario **repetir un proceso varias veces**?
4. ¿Qué posibilidades te abren las habilidades recién adquiridas en **Python**?

Cierre: Concepto clave

Los tipos y estructuras de datos te permiten representar la información, mientras que los estatutos condicionales y de repetición te permiten decidir e iterar

Siguientes pasos | Semana de trabajo asíncrono [Profundizar]



1 Sesión Síncrona 1
Aprender

2 Trabajo asíncrono 1
Profundizar | Ruta de Aprendizaje

3 Sesión Síncrona 2
Preparar para Aplicar

4 Trabajo asíncrono 2
Aplicar en el trabajo | Reto

- Lo aprendido el día de hoy te va a permitir continuar con el **trabajo asíncrono** de la sección de Profundizar
- Ya cuentas con las **herramientas básicas** que son la base para **comprender y utilizar conceptos más avanzados** como el uso de estándares, la definición de funciones y el manejo de archivos de texto



Tecnológico de Monterrey | 2021

Prohibida la reproducción total o parcial de esta
obra sin expresa autorización del Tecnológico
de Monterrey

Gracias | Programas LIVE