

Todas y Todos podemos programar

Academias de invierno UOH

Contenidos

Tercera clase programación

A mi computador le gusta tomar decisiones.

- a. Presentación de instrucciones de decisión (if, elif, else) en Python
- b. Presentación de comparaciones con múltiples decisiones.
- c. Controlando el tiempo en el computador.

Mi computador es el mejor para repetir instrucciones.

- a. Presentación de instrucción de iteración “While” en Python
- b. Presentación de instrucción de iteración “For” en Python
- c. ¿Cómo manejamos más de una variable?, listas en Python.

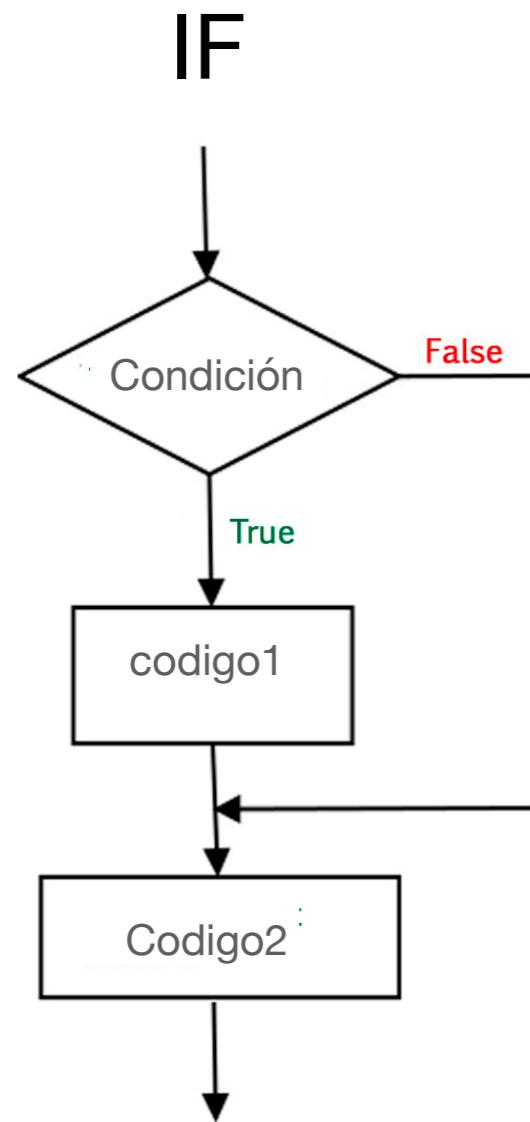
A mi computador le gusta tomar decisiones

Presentación de instrucciones de decisión (if, elif, else) en Python



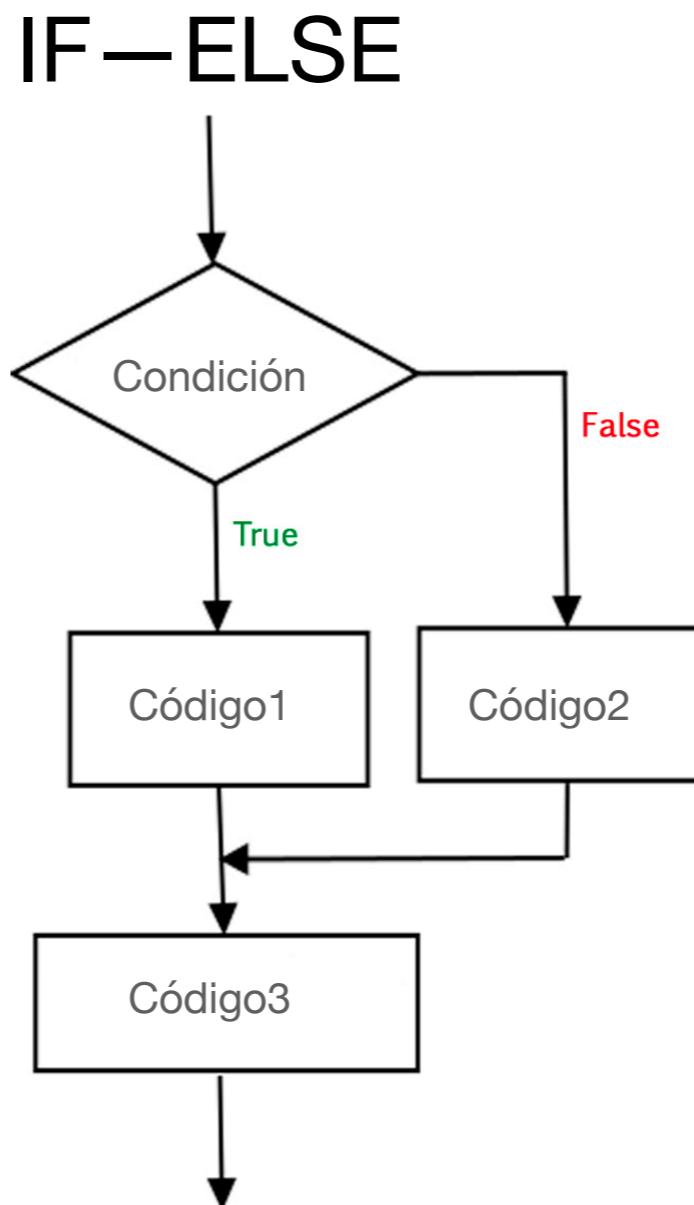
A mi computador le gusta tomar decisiones

Presentación de instrucciones de decisión (if, elif, else) en Python



```
a = 11
if(a > 10):
    print("el valor es mayor a 10")

print("el valor es:",a)
```

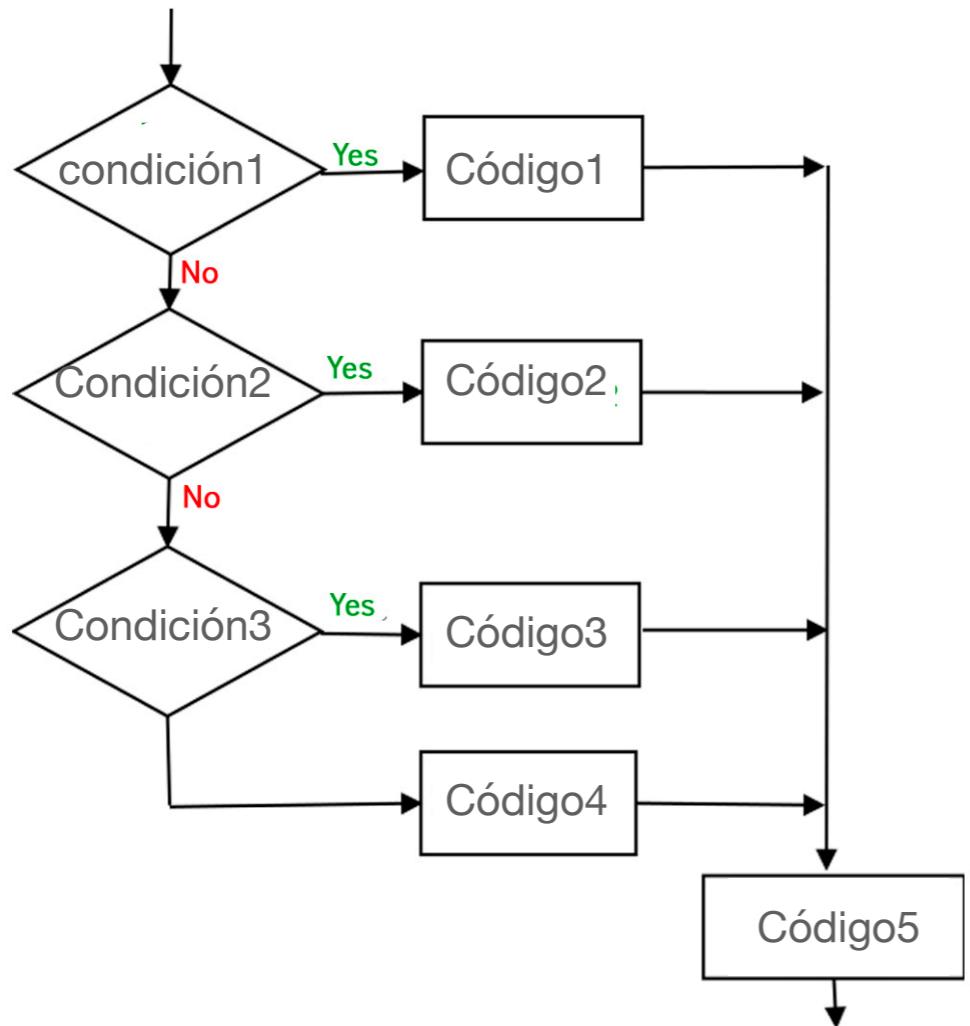


```
texto = "Alex"
if(texto == "Alex"):
    print("Hola Alex")
else:
    print("No eres alex")
```

A mi computador le gusta tomar decisiones

Presentación de instrucciones de decisión (if, elif, else) en Python

IF-elif-else



```
#if-elif-else
nombre= "Alex"
if(nombre == "Alex"):
    print("Hola Alex")
    print("Se que te gusta jugar al futbol")
elif(nombre == "Roberto"):
    print("Hola Roberto")
    print("Me contaron que te gusta leer libros")
elif(nombre == "Camila"):
    print("Hola Camila")
    print("Se que te gusta programar")
else:
    print("Hola ",nombre)
    print("No se nada de ti")

print("Bienvenido")
```

A mi computador le gusta tomar decisiones

Presentación de comparaciones con múltiples decisiones.

```
edad = 20
nombre = "Carlos"
ciudad = "Talca"

if(nombre == "Carlos"):
    if(ciudad == "Rancagua"):
        if(edad < 20):
            print("Hola Carlos de Rancagua, tienes menos de 20 años")
        else:
            print("Hola Carlos de Rancagua, tienes más de 20 años")
    else:
        print("La persona no es de rancagua")
else:
    print("El nombre de la persona no es Carlos")
```

Que hace este código?

```
edad = 20
nombre = "Alex"
ciudad = "Rancagua"

if(nombre == "Carlos" and ciudad == "Rancagua"):
    if(edad < 20):
        print("Hola Carlos de Rancagua, tienes menos de 20 años")
    else:
        print("Hola Carlos de Rancagua, tienes más de 20 años")
else:
    if(nombre == "Carlos"): print("es Carlos pero no es de Rancagua")
    if(ciudad == "Rancagua"): print("No es Carlos, pero es de Rancagua")
```

A mi computador le gusta tomar decisiones

Controlando el tiempo en el computador.

`sleep(...)`

`sleep(segundos)`

Retrasar la ejecución durante un número determinado de segundos. El argumento puede ser un número de punto flotante para una precisión de subsegundos.

```
import time

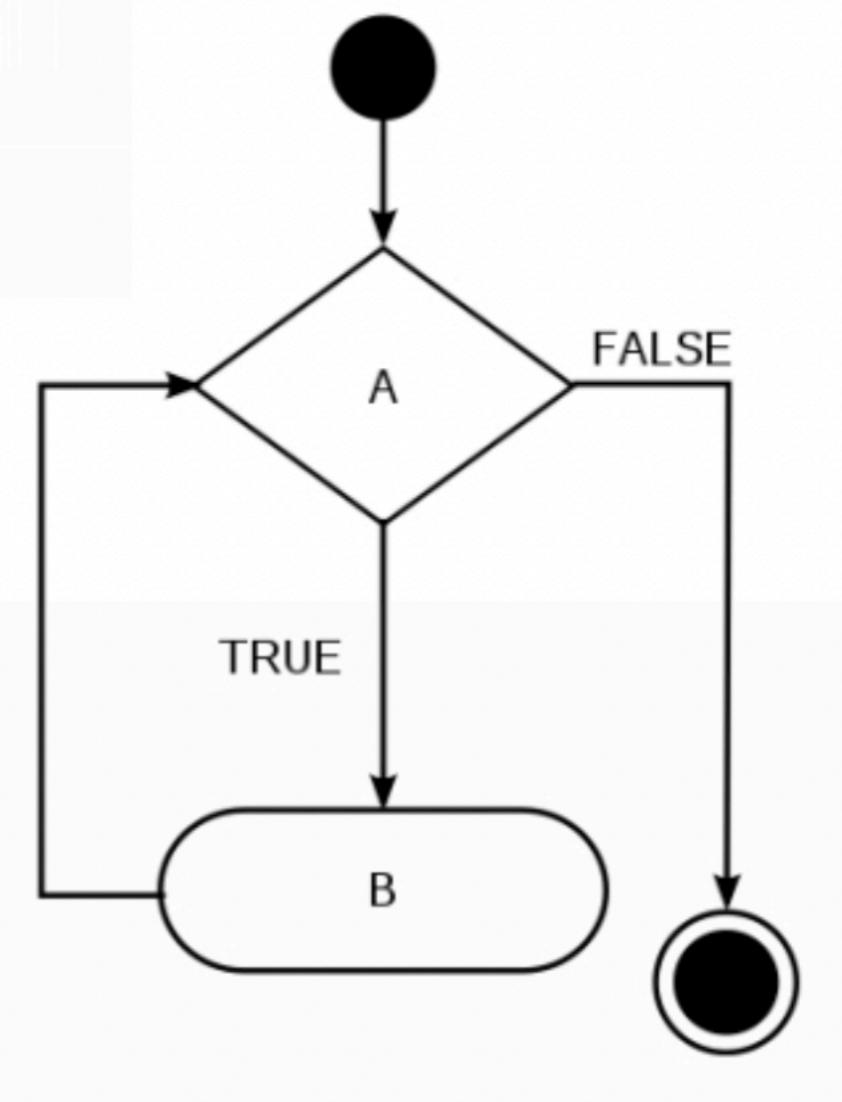
a = 10
nombre = input("Ingresa un nombre:")
if(nombre == "Roberto"):
    time.sleep(10)
    print("Roberto esta esperando")
else:
    time.sleep(1)
    print("Hola", nombre)
```

Que hace este código?

Mi computador es el mejor para repetir instrucciones.

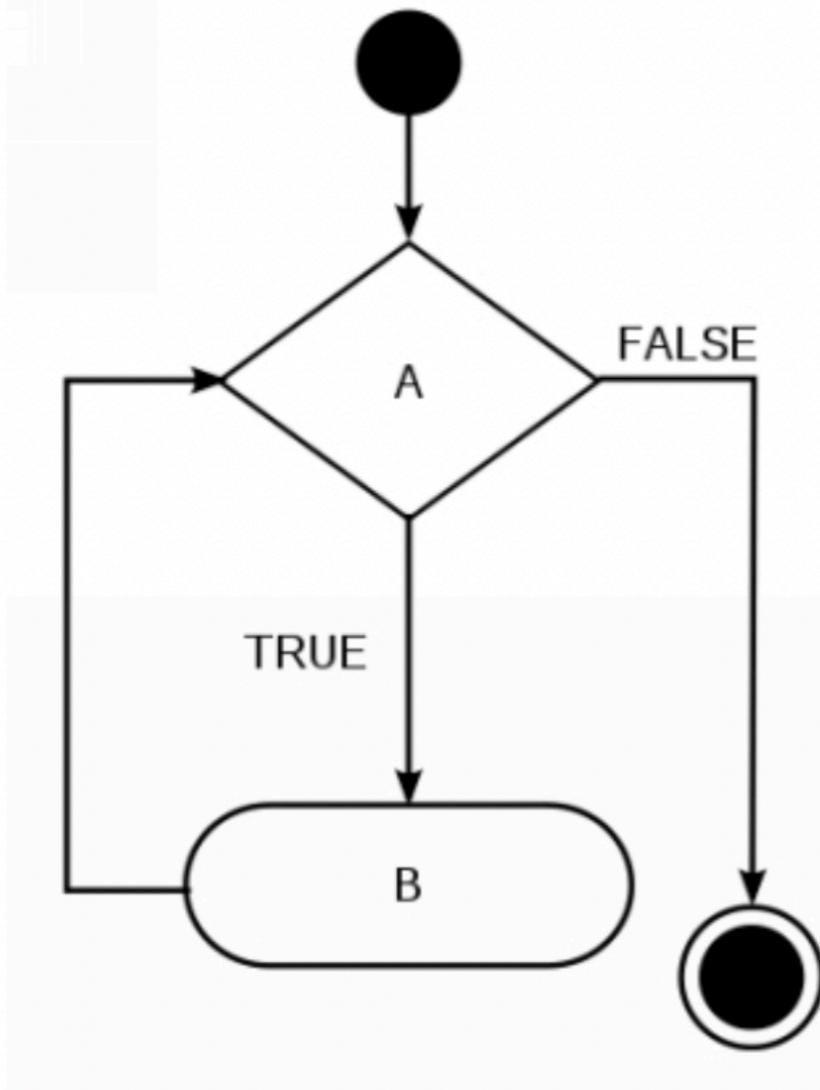
Presentación de instrucción de iteración “While” en Python

- Una computadora puede seguir haciendo lo mismo una y otra vez para siempre.
- Un bucle (loop) es un comando para repetir algo hasta que suceda algo más. Da vueltas y vueltas, una y otra vez, como un círculo.
- Uno de los tipos de bucles más populares es el bucle **while**.
 - Mientras _____ es Verdadero, haz _____



Mi computador es el mejor para repetir instrucciones.

Presentación de instrucción de iteración “While” en Python

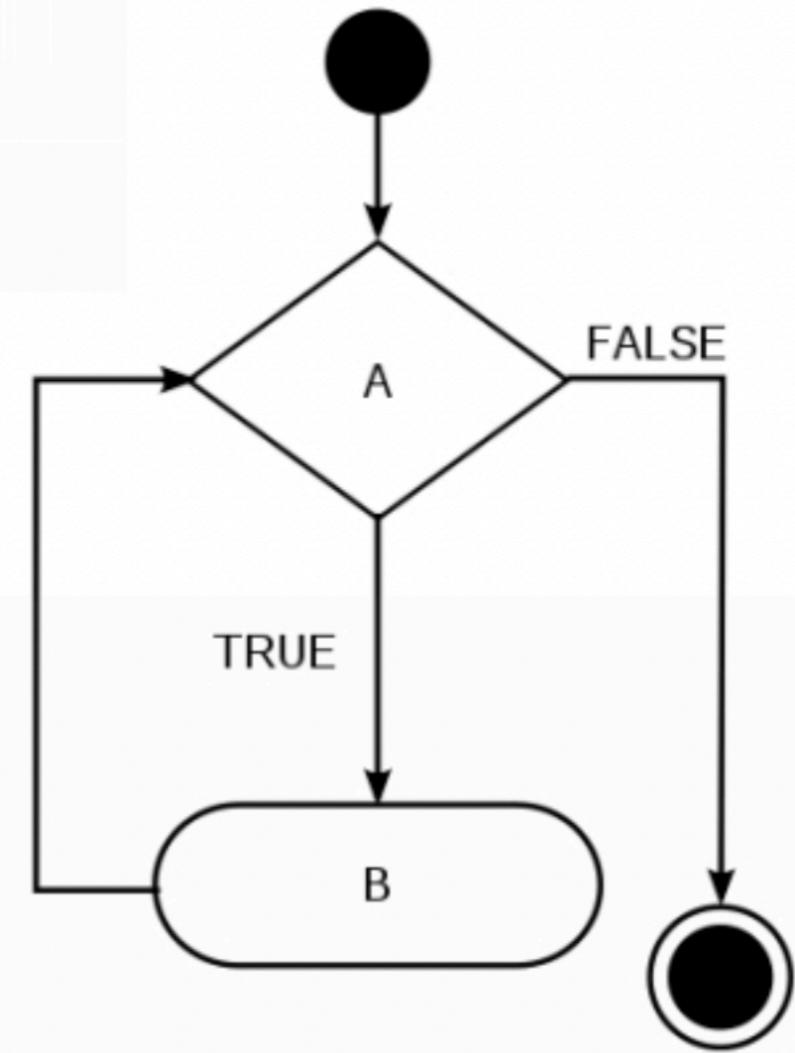


```
while 2 > 1:  
    print("Hola --- no puedo parar...")  
  
i = 1  
while i < 5:  
    print(i)  
  
i = 1  
while i < 5:  
    print(i)  
    i=i+1  
  
i = 1  
while i > 5:  
    print(i)  
    i=i+1
```

Mi computador es el mejor para repetir instrucciones.

Presentación de instrucción de iteración “For” en Python

- Con el ciclo **while**, no especificamos (decimos claramente) cuántas veces repetir el ciclo. No dijimos, repite 5 veces..
- Con el ciclo **for**, especificamos cuántas veces repetir el ciclo.
- Una de las funciones que podemos usar con bucles for es **range()**. La función **range()** se puede usar para crear una lista de números dentro de un cierto rango.



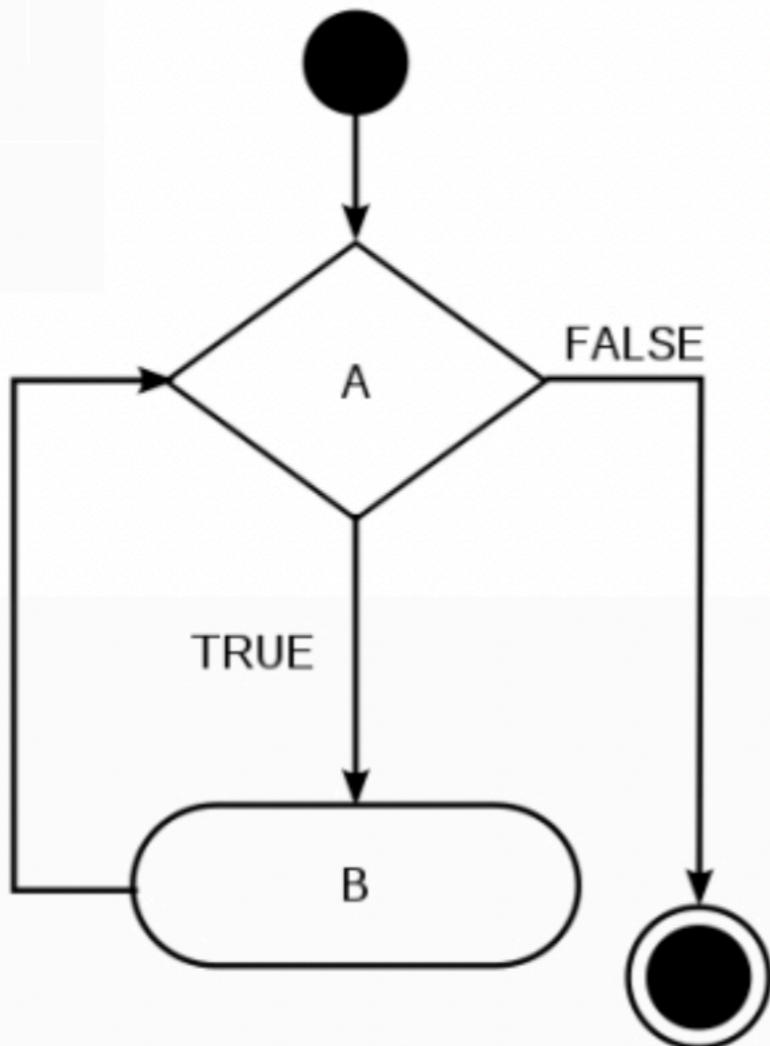
range(3)



[0,1,2]

Mi computador es el mejor para repetir instrucciones.

Presentación de instrucción de iteración “For” en Python



```
for i in range(1,10):  
    print(i)
```

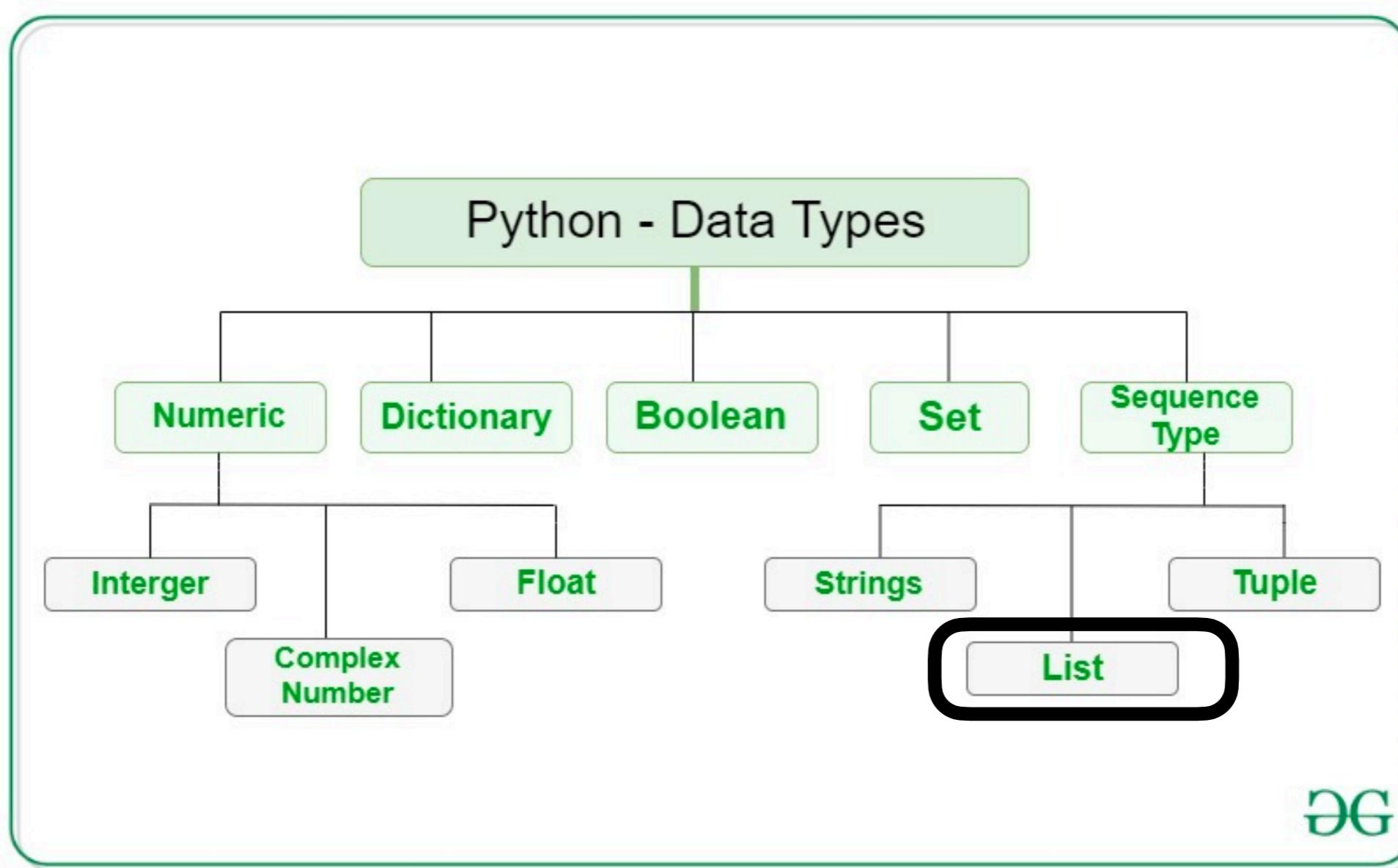
```
S="palabra"  
for i in S:  
    print(i)
```

```
for i in S:  
    for j in range(10):  
        print(i,j)
```

Mi computador es el mejor para repetir instrucciones.

¿Cómo manejamos más de una variable?, listas en Python.

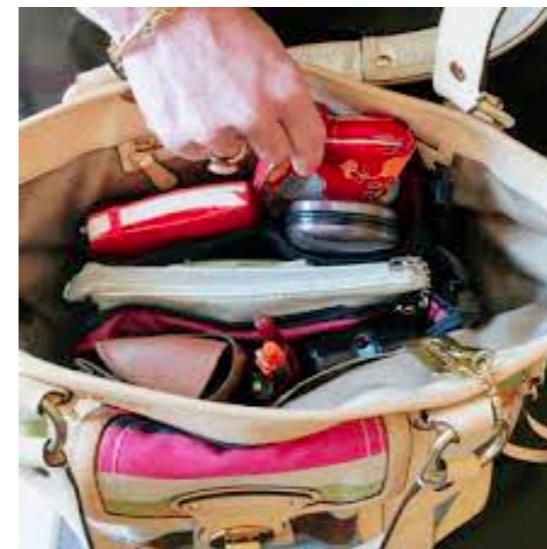
Tipos de variables en python:



Mi computador es el mejor para repetir instrucciones.

¿Cómo manejamos más de una variable?, listas en Python.

- Una lista, se define como una colección de elementos en un orden determinado.
- Ejemplos: letras del alfabeto, dígitos del 0 al 9, o los nombres de todas las personas de tu familia.
- Sin embargo, los elementos de una lista no tienen que estar necesariamente relacionados de ninguna manera en particular.



Mi computador es el mejor para repetir instrucciones.

¿Cómo manejamos más de una variable?, listas en Python.

Creando una lista



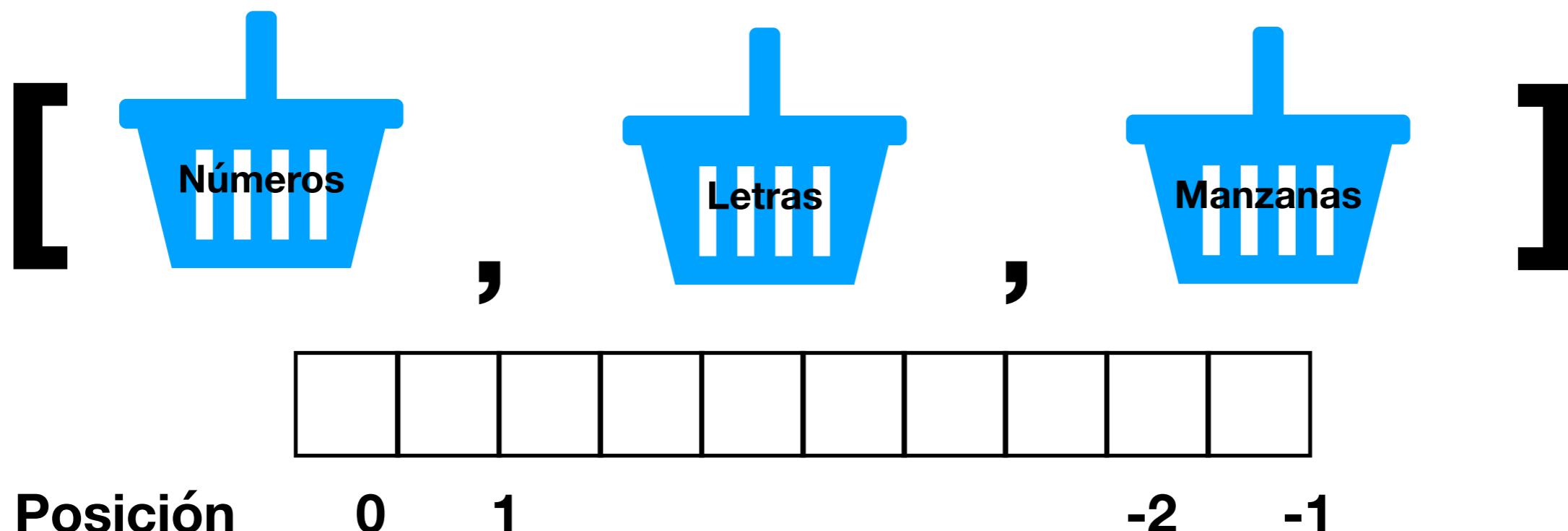
```
frutas = ['manzanas', 'peras', 'platanos', 'frutillas']
print(frutas)
```

```
['manzanas', 'peras', 'platanos', 'frutillas']
```

Mi computador es el mejor para repetir instrucciones.

¿Cómo manejamos más de una variable?, listas en Python.

Como accedemos a los elementos en una lista?



```
# acceso a los elementos en una lista
frutas = ['manzanas', 'peras', 'platanos', 'frutillas']
mensaje = "Mi fruta favorita son las/los " + frutas[0].title() + "."
print(mensaje)
```

↳ Mi fruta favorita son las/los Manzanas.

Mi computador es el mejor para repetir instrucciones.

¿Cómo manejamos más de una variable?, listas en Python.

Como modificamos los elementos en una lista?



```
▶ # modificar elementos en una lista  
frutas = ['manzanas', 'peras', 'platanos', 'frutillas']  
print(frutas)  
frutas[3] = "naranjas"  
print(frutas)
```

```
⇨ ['manzanas', 'peras', 'platanos', 'frutillas']  
['manzanas', 'peras', 'platanos', 'naranjas']
```

Las listas son mutables

Mi computador es el mejor para repetir instrucciones.

¿Cómo manejamos más de una variable?, listas en Python.

Como agregamos los elementos en una lista?



```
# añadir elementos en una lista
frutas = ['manzanas', 'peras', 'platanos', 'frutillas']
print(frutas)
frutas.append('naranjas')
print(frutas)

['manzanas', 'peras', 'platanos', 'frutillas']
['manzanas', 'peras', 'platanos', 'frutillas', 'naranjas']
```

```
# añadir elementos en una posición específica a una lista
frutas = ['manzanas', 'peras', 'platanos', 'frutillas']
print(frutas)
frutas.insert(2, 'naranjas')
print(frutas)

['manzanas', 'peras', 'platanos', 'frutillas']
['manzanas', 'peras', 'naranjas', 'platanos', 'frutillas']
```

Mi computador es el mejor para repetir instrucciones.

¿Cómo manejamos más de una variable?, listas en Python.

Como eliminamos los elementos en una lista?



```
# eliminar elementos en una posición específica a una lista  
frutas = ['manzanas', 'naranjas', 'peras', 'platanos', 'frutillas']  
print(frutas)  
del frutas[1]  
print(frutas)
```

```
[ 'manzanas', 'naranjas', 'per  
[ 'manzanas', 'peras', 'platan
```

```
# eliminar elementos con función pop  
frutas = ['manzanas', 'naranjas', 'peras', 'platanos', 'frutillas']  
print(frutas)  
fruta_eliminada= frutas.pop()  
print(frutas)  
frutas.remove('peras')  
print(frutas)
```

```
[ 'manzanas', 'naranjas', 'peras', 'platanos', 'frutillas']  
[ 'manzanas', 'naranjas', 'peras', 'platanos']  
[ 'manzanas', 'naranjas', 'platanos']
```

Mi computador es el mejor para repetir instrucciones.

¿Cómo manejamos más de una variable?, listas en Python.

Como ordenamos los elementos en una lista?



- Generalmente, las listas se crean de manera **dinámica**, por lo que no siempre puedes controlar el orden en el que tus usuarios proporcionan sus datos.

Mi computador es el mejor para repetir instrucciones.

¿Cómo manejamos más de una variable?, listas en Python.

Como ordenamos los elementos en una lista?



```
▶ frutas = ['manzanas', 'peras', 'platanos', 'frutillas']
print(frutas)
frutas.sort()
print(frutas)
```

```
['manzanas', 'peras', 'platanos', 'frutillas']
['frutillas', 'manzanas', 'peras', 'platanos']
```

Mi computador es el mejor para repetir instrucciones.

¿Cómo manejamos más de una variable?, listas en Python.

Recorriendo los elementos de una lista?



```
▶ 1 frutas = ['manzanas', 'sandía', 'peras', 'platanos', 'frutillas']
  2 print(frutas)
  3 for i in frutas:
  4     print(i)
```

```
[ 'manzanas', 'sandía', 'peras', 'platanos', 'frutillas' ]
manzanas
sandía
peras
platanos
frutillas
```

Mi computador es el mejor para repetir instrucciones.

¿Cómo manejamos más de una variable?, listas en Python.

Errores comunes con listas.



- Un error de índice (IndexError) significa que Python no puede averiguar el índice solicitado.
- Un error de tipo ValueError puede producirse al tratar de remover un valor que no se encuentra en la lista.

Mi computador es el mejor para repetir instrucciones.

¿Cómo manejamos más de una variable?, listas en Python.

Errores comunes con listas.

```
[20] #errores comunes de indice
     frutas = ['manzanas', 'peras', 'platanos', 'frutillas']
     print(frutas)
     print(frutas[-5])

['manzanas', 'peras', 'platanos', 'frutillas']

-----
IndexError                                     Traceback (most recent call last)
<ipython-input-20-b86b898bb2dc> in <module>()
      1 frutas = ['manzanas', 'peras', 'platanos', 'frutillas']
      2 print(frutas)
----> 3 print(frutas[-5])

IndexError: list index out of range
```

SEARCH STACK OVERFLOW

```
▶ #errores comunes de valor
    frutas = ['manzanas', 'peras', 'platanos', 'frutillas']
    print(frutas)
    del frutas[5]
    print(frutas)

[] ['manzanas', 'peras', 'platanos', 'frutillas']

-----
IndexError                                     Traceback (most recent call last)
<ipython-input-21-f0feb5a39f37> in <module>()
      2 frutas = ['manzanas', 'peras', 'platanos', 'frutillas']
```

Una serpiente que programa?

Google Colab

The screenshot shows a Google Colab notebook titled "TPP_C01.ipynb". The notebook contains the following sections and code snippets:

- Todas y todos podemos programar**

Este taller busca entregar competencias básicas de programación a estudiantes de tercero y Cuarto Medio.
- Hello world en python**

```
[2]: 1 print("hello world")
      hello world
```
- Obteniendo información**

```
[3]: 1 !python --version
      Python 3.7.13
```
- Creando mi primera variable**

```
[4]: 1 mensaje = "hello world"
```

The Colab interface includes a toolbar at the top with various icons for file operations, a sidebar with navigation and search tools, and a bottom navigation bar with other open files.

Una serpiente que programa?

GitHub

The screenshot shows a GitHub repository page for 'adigenova/tpp'. The repository is public and contains 1 branch and 0 tags. The most recent commit was made by 'adigenova' adding class 1, 4 hours ago. The repository description is 'Curso Introductorio de Python para alumnos de Tercero y cuarto Medio'. The README.md file contains the following content:

```
Todas y Todos Podemos Programar (TTP)

Curso Introductorio de Python para alumnos de Tercero y cuarto Medio.

Programa del curso

Programa que detalla el contenido de cinco clases teóricas y cinco talleres prácticos.

Bitacora de clases

Archivo guía con el contenido y código publicados en el repositorio GitHub del taller.
```

The repository has 0 stars, 1 watching, and 0 forks. It also lists releases, packages, and languages used.

<https://github.com/adigenova/tpp>

Información

- Laboratorio Práctico
 - 11:30 a 13:00
 - Sala A 505
 - 2 ayudantes para el taller.
 - Google Colab y mi primer programa en python

**Preguntas?
Muchas gracias.**