Esta clase va a ser

- grabada

Clase 04. PYTHON

Controladores de Flujo I

Temario

03

Operador y expresiones

- ✓ Operadores
- ✓ Expresiones anidadas

04

Controladores de Flujo I

- ✓ Condicional
- ✓ Else
- ✓ Elif

05

Controladores de Flujo II

- ✓ Sentencias iterativas
- ✓ While
- ✓ For

CODERHOUSE

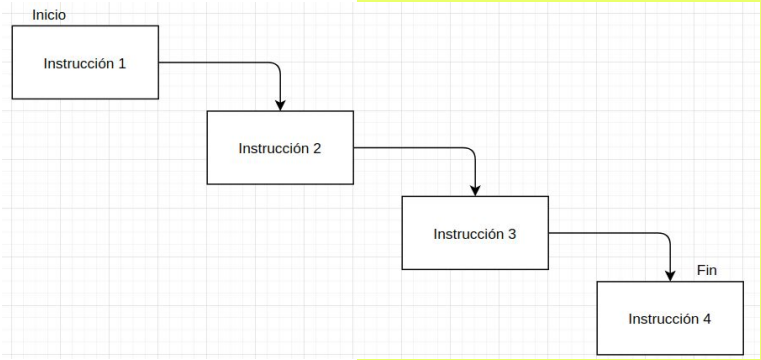
Objetivos de la clase

- **Conceptualizar** el flujo y diagrama de flujo
- **Reconocer** sus funcionalidades
- **Conceptualizar** sentencias de control
- **Utilizar** sentencia de control if

Flujo

¿Qué es el flujo?

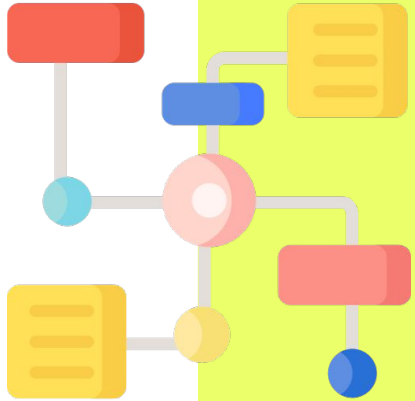
Veremos cómo controlar el flujo con Python, pero antes de eso, ¿Qué es el flujo?. El flujo es una forma de entender la sucesión de las instrucciones de un programa, estas instrucciones se ejecutan una después de otras de forma ordenada y suelen tener el objetivo final de manipular información.



CODERHOUSE

¿Qué es el flujo?

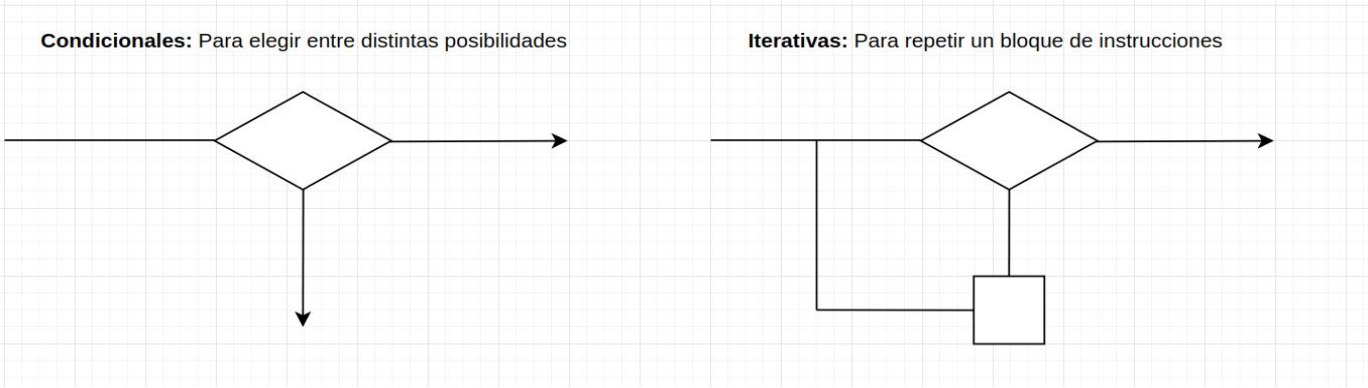
Sin embargo, para manipular datos no es suficiente con realizar cálculos o resolver expresiones, **necesitamos que de alguna forma nuestro programa pueda elegir, que sepa cómo actuar en función de determinadas situaciones, o incluso repetir una tarea si es necesario**.
Para estas situaciones, existen las sentencias de control de flujo.



Sentencias de control

Se dividen en dos tipos, las de control condicional y las de control iterativo. (A las siguientes imágenes se le denomina diagrama de flujo)

Nos centraremos en las sentencias de control condicional.



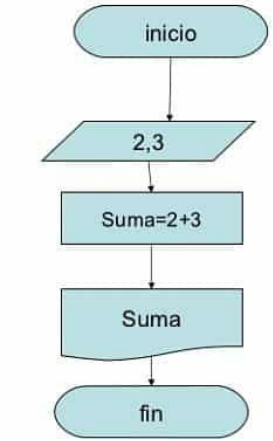
CODERHOUSE

Diagramas de flujo

Expresan nuestros algoritmos en forma de diagrama mediante una representación gráfica basada en figuras geométricas que varían según la estructura de código.

Una app recomendada que se suele utilizar es

[Diagrams]



Punto de inicio del programa

Entrada de datos 2,3

Proceso

Salida

Fin

CODERHOUSE

Condicional

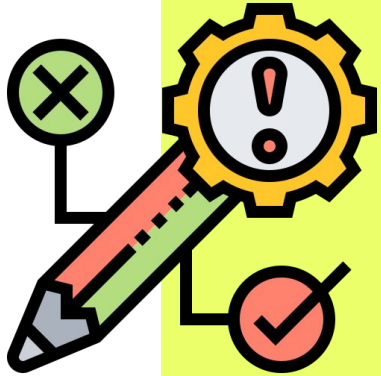
Condicional

En la vida diaria, actuamos de acuerdo a la evaluación de condiciones, de manera mucho más frecuente de lo que en realidad creemos: si el semáforo está en verde, cruzar la calle. Si no, esperar a que el semáforo se ponga en verde.

A veces, también evaluamos más de una condición para ejecutar una determinada acción: si llega la factura de la luz y tengo dinero, pagar la factura.

Condicional

Las sentencias de control condicionales, son aquellas que nos permiten evaluar si una o más condiciones se cumplen, para decir qué acción vamos a ejecutar. La evaluación de condiciones, solo puede arrojar 1 de 2 resultados: **True** o **False** (verdadero o falso).



Condicional

Para describir la evaluación a realizar sobre una condición, se utilizan los operadores relacionales (==, !=, >, <, etc.). Y, para evaluar más de una condición simultáneamente se utilizan los operadores lógicos (not, and, or).

Las sentencias de control de flujo condicionales se definen mediante el uso de tres palabras claves reservadas:

- ✔ if (si)
- ✔ elif (si no, si)
- ✔ else (si no)

Sentencia If

Dentro de las sentencias condicionales el if (si) posiblemente sea la más famosa y utilizada en la programación, esto debido a que nos permite controlar el flujo del programa y dividir la ejecución en diferentes caminos.

Al utilizar esta palabra reservada **if** le estamos indicando a Python que queremos ejecutar una porción de código, o bloque de código, solo si se se cumple una determinada condición, es decir, si el resultado es **True**.

Sentencia If

Primero definimos una variable **edad** y le asignamos un valor entero **30**. Después, a través del condicional, le decimos que queremos imprimir "**Es un adulto**" en pantalla, solo si se cumple la condición de que **edad** sea mayor o igual a 18.

Veamos el siguiente ejemplo:

```
edad =  30
if edad >= 18:
    print('Es un adulto')
if True:
    print('Se cumple la condición')
```

Indentación

Python se basa en la sangría (espacio en blanco al comienzo de una línea) para definir el alcance en el código. Otros lenguajes de programación a menudo usan corchetes para este propósito.

El siguiente código nos arrojará un error:

```
a = 25
b = 50
if b > a:
print("b es más grande que a")
```

Puedes probarlo para verificarlo.

If

Recordemos que Python admite las condiciones lógicas habituales de las matemáticas:

- ✔ Es igual a : a == b
- ✔ No es igual a: a != b
- ✔ Menos que: a < b
- ✔ Menor o igual que: a <= b
- ✔ Mayor que: a > b
- ✔ Mayor o igual que: a >= b

También podemos apoyarnos del uso de operadores lógicos como ser: AND, OR, NOT.

Ejemplo con AND:

```
a = 195
b = 30
c = 400
if a > b and c > a:
  print("Ambas condiciones son verdaderas")
```

If

Ejemplo con OR:

```
a = 195
b = 50
c = 500
if a > b or a > c:
  print(“Al menos una de las condiciones es verdadera")
```

Ejemplo con NOT:

```
x = 10
if not x > 15:
    print("False")
```

If

If en una sola linea – Ejemplo 1:

```
a = 150
b = 35
if a > b: print(“a es mayor que b")
```

If en una sola linea – Ejemplo 2:

```
a = 5
b = 150
print("A") if a > b else print("B")
```

If en una sola linea – Ejemplo 3:

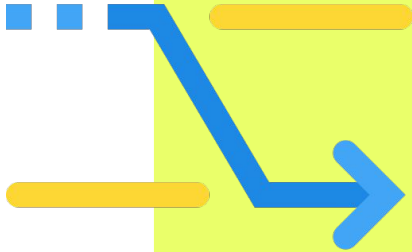
```
a = 150
b = 330
print("A") if a > b else print("=") if a == b else print("B")
```

Else

Sentencia Else

Dentro de las sentencias condicionales el **else** (sino) es una especie de “hermano” de **if** el cual se puede encadenar al final de un bloque de código **if** para comprobar los casos contrarios, es decir los **False.**

Al utilizar esta palabra reservada **else** le estamos indicando a Python que queremos ejecutar una porción de código, o bloque de código, sólo si no se cumple ninguna de las condiciones antes dichas, es decir, si el resultado es **False** siempre.



SENTENCIA ELSE

Veamos el siguiente ejemplo:

```
numero =  24
if numero > 36:
   print("El número es grande")
else:
   print("El número es chico")
```

1

Primero definimos una variable número y le asignamos un valor entero 24.

2

Después, a través del condicional, le preguntamos si el número es mayor a 36, si es así, queremos imprimir “El número es más grande” en pantalla, de lo contrario queremos que imprima “El número es más chico”.

Múltiples If

Veamos un ejemplo de cómo podemos trabajar con múltiples ifs anidados:

Ejemplo 1:

```
x = 25
if x > 10:
  print("por encima de diez,")
  if x > 20:
    print("y también por encima de 20!")
  else:
    print("pero no por encima de 20")
```

Ejemplo 2:

```
x = 15
if x > 10:
  print("por encima de diez,")
  if x > 20:
    print("y también por encima de 20!")
  else:
    print("pero no por encima de 20")
```

Break

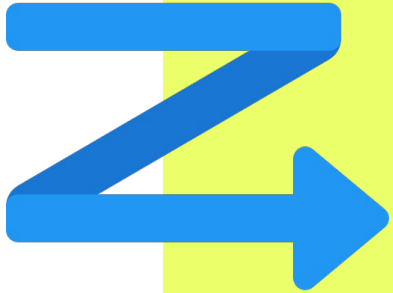
¡10 minutos y volvemos!

Elif

Sentencia Elif

La última sentencia condicional que podemos encontrar es el **elif** (si no, si), también podríamos decir que es un hermano de **if,** ya que se utiliza en continuación al **if** para poder encadenar muchísimas más comprobaciones.

Al utilizar esta palabra reservada elif le estamos indicando a Python que queremos ejecutar una porción de código, o bloque de código, solo si la condición anterior no se cumple, es decir, si el resultado del **if** o algún **elif** fue **False**.



SENTENCIA ELIF

```
a = 2 + 3
if a == 4:
    print ("A es igual a cuatro")
elif a == 5:
    print ("A es igual a cinco")
elif a == 6:
    print ("A es igual a seis")
else:
    print ("No se cumple la condición")
```

Como podemos observar, la primera condición valida si A es igual a 4, como esto no es verdadero, se evalúa la siguiente condición, si A es igual a 5, si no A es igual a 6? .

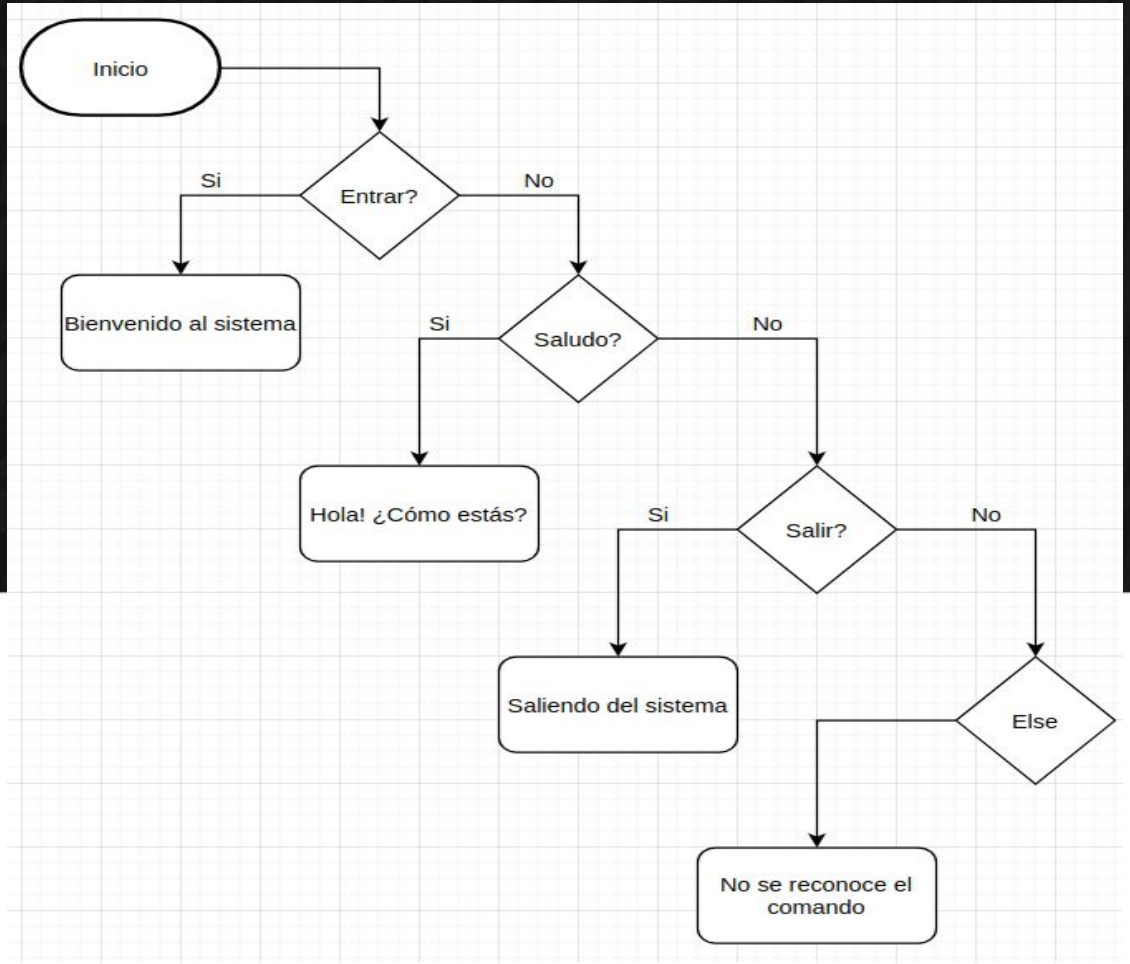
Finalmente, se define un bloque else por default que se ejecutará cuando ninguna de las condiciones anteriores sea verdadera.

Pregunta! ¿Cuál sería el resultado de este ejemplo?

¿Para qué sirve la sentencia Elif?

```
comando =  "SALIR"
if comando == "ENTRAR":
    print("Bienvenido al sistema.")
elif comando == "SALUDO":
    print("Hola! ¿Cómo estás?")
elif comando == "SALIR":
    print("Saliendo del sistema.")
else:
    print("No se reconoce el comando.")
```

Básicamente nos sirve para poder darle múltiples opciones al programa.


Inicio
Entrar?
Si
No
Bienvenido al sistema
Saludo?
Si
No
Hola! ¿Cómo estás?
Salir?
Si
No
Saliendo del sistema
Else
No se reconoce el comando

PARA RECORDAR

Cuando se tiene varios **if's** se ven las múltiples condiciones y si todo está bien, nos mostrará el resultado de cada **if**.
Sin embargo, en el caso de múltiples **elif**, comprueba las condiciones de arriba a abajo hasta que se cumpla una de ellas, y de ser así, las demás no se comprueban.

Generaciones digitales

Escribir un programa que indique la generación correspondiente para un año de nacimiento indicado

Duración: 20 **minutos.**

ACTIVIDAD EN CLASE

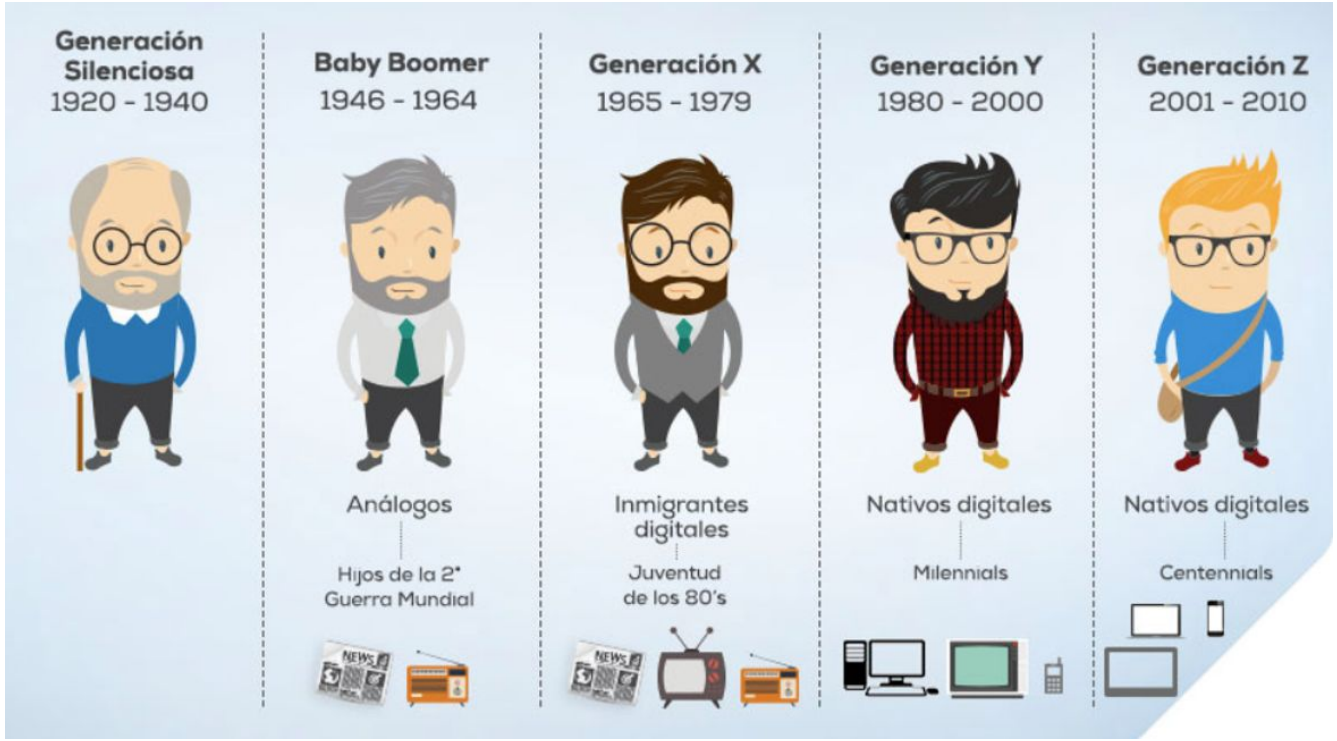
Generaciones digitales

Descripción de la actividad.

Escribir un programa que indique la generación correspondiente para un año de nacimiento indicado. Trabajarán con el notebook de clase Clase_4.ipynb

Importante: Para los años que no pertenezcan a ninguna generación, se deberá colocar: “No existe generación asociada”

Generaciones digitales



Crédito Bancario

Aprobación de Crédito bancario

Duración: **20 minutos.**

ACTIVIDAD EN CLASE

Aprobación de Crédito bancario

Descripción de la actividad.

Para aprobar un crédito, el cliente debe ser mayor de edad. Además, debe tener una antigüedad en el sistema financiero de mínimo 3 años y un ingreso mayor a 2500 dólares. En caso no tenga la antigüedad suficiente, su ingreso mensual debe ser como mínimo 4000 dólares. Si no cumple ninguna de las condiciones, no se aprueba el crédito

Datos iniciales

- ✔ edad = 15
- ✔ antigüedad = 10
- ✔ ingreso = 1500

Marvel vs. CapCom

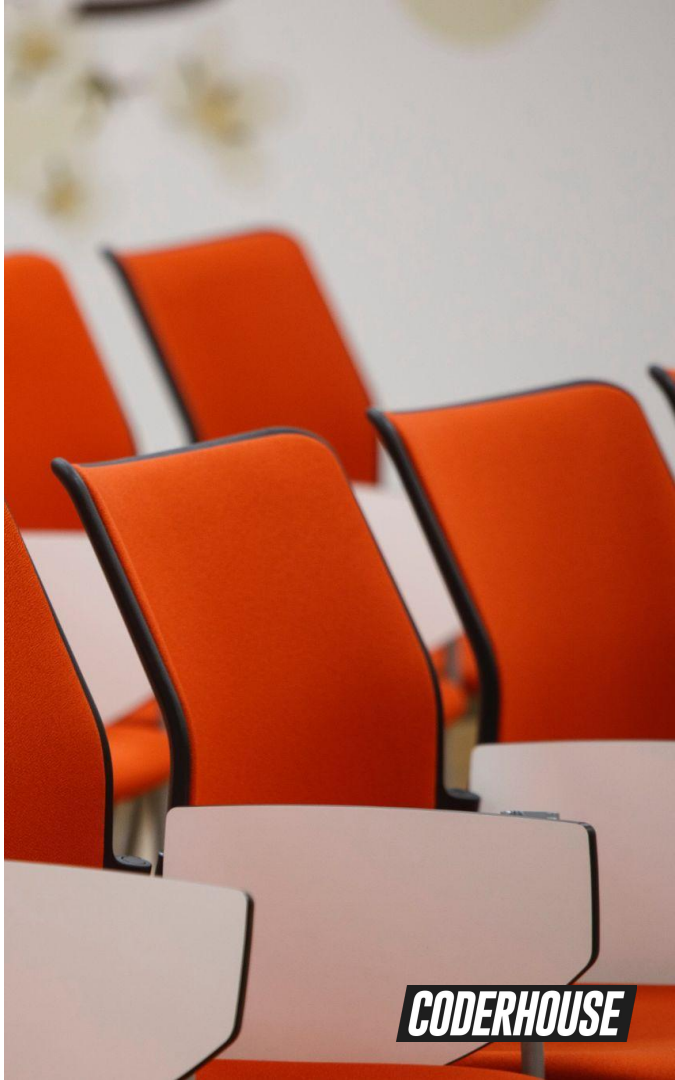
Duración: **20 minutos.**

ACTIVIDAD EN CLASE

Marvel vs. CapCom

Un curso se ha dividido en dos grupos diferentes: A y B de acuerdo al nombre y a una preferencia (Marvel o Capcom). El grupo **A** está formado por fans de **Marvel** con un **nombre anterior a la M** y los fans de **Capcom** con un **nombre posterior a la N** y el grupo **B por el resto**. Escribir un programa que pregunte al usuario su nombre y preferencia, y muestre por pantalla el grupo que le corresponde.
Ej.:
¿Cómo te llamas? Alan
¿Cuál es tu preferencia (M o C)? C
Tu grupo es B
Para preguntarle al usuario, recuerda usar input.

Ejemplo: Desafío



¿Quieres saber más?
Te dejamos material ampliado de la clase

CODERHOUSE

MATERIAL AMPLIADO

Recursos

- ✓ Pseudocódigo y Diagramas de flujo
- ✓ Funciones Listas
- ✓ Tipo Tuplas

Disponible en nuestro repositorio.

¿Preguntas?

Resumen
de la clase hoy

- ✓ Flujo
- ✓ Sentencia de control
- ✓ Diagrama de Flujo
- ✓ Sentencias

Opina y valora esta clase

CODERHOUSE

Muchas gracias.

#DemocratizandoLaEducación