Codo a Codo 4.0

FULL STACK PYTHON

html - css3 - bootstrap - javascript
vue.js - sql - python - django

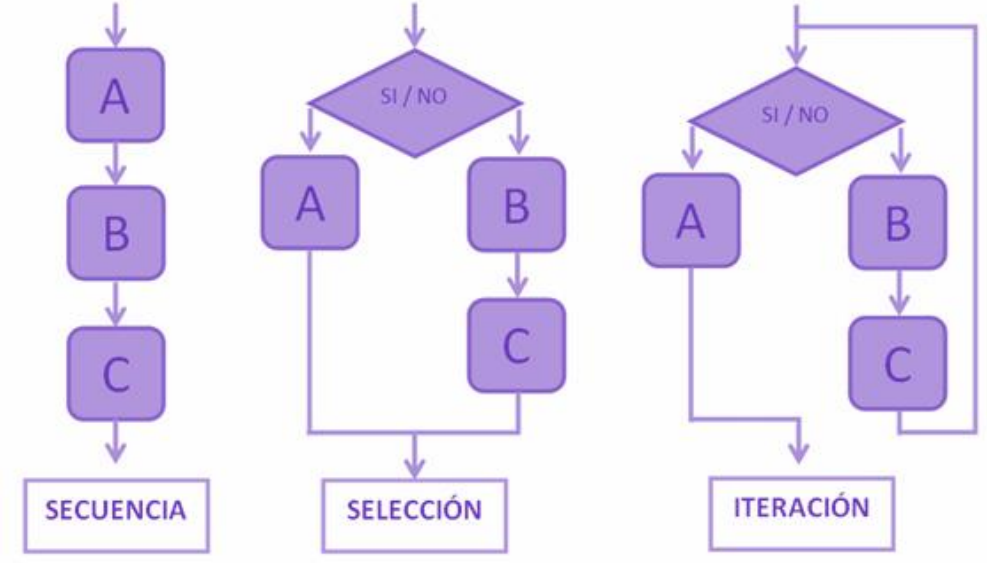
Clase 26

Python – Parte 2

Estructuras de control

En programación, las **estructuras de control** permiten modificar el flujo de ejecución de las instrucciones de un programa. Con ellas se puede:

- De acuerdo a si se cumple con una condición, ejecutar un grupo u otro de sentencias (***if***)
- Ejecutar un grupo de sentencias mientras se cumpla una condición (***while***)
- Repetir un grupo de sentencias un número determinado de veces (***for***)



Secuenciales: las instrucciones se ejecutan una después de la otra, en el orden en que están escritas, es decir, **en secuencia**.

Condicional (Selección o de decisión): ejecutan un bloque de instrucciones u otro, o saltan a un subprograma o subrutina según se cumpla o no una condición.

Iterativa (Repetitiva): inician o repiten un bloque de instrucciones si se cumple una condición o **mientras** se cumple una **condición**.

Estructuras secuenciales

Las acciones se ejecutan una seguida de la otra, es decir que se ejecuta una acción o instrucción y continúa el control a la siguiente. La ejecución es lineal y de arriba hacia abajo.

Todas las instrucciones se ejecutan una sola vez y finaliza el programa.

PY

```python
#Programa Suma: suma dos números enteros ingresados por teclado
nro1= int(input("Ingrese el primer número: "))
nro2= int(input("Ingrese el segundo número: "))
suma= nro1 + nro2
print("La suma es:", suma)
```

secuenciales_1.py

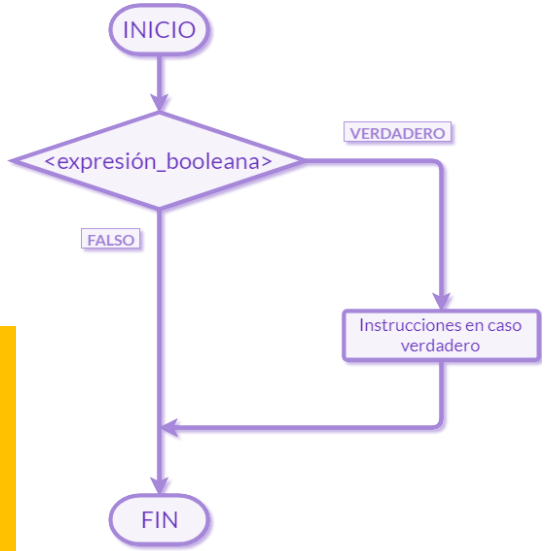
*Este programa pide dos números guardados en **nro1** y **nro2**, luego realiza la suma de ambos valores guardando el resultado en **suma** y finalmente muestra un mensaje por pantalla.*

Estructuras condicionales

Las estructuras condicionales / alternativas / selectivas o de decisión tienen como objetivo seleccionar caminos alternos, en base a una condición, que es una pregunta. En base a la respuesta, que va a ser ***verdadero o falso / si o no*** voy a tomar un camino u otro. La palabra clave asociada a esta estructura es ***if***.

Estructura condicional simple

Voy a tener un único camino (verdadero).




```
if condición:
    sentencia 1
    sentencia 2
    …
instrucción fuera del if
```

Si la condición resulta VERDADERA se ejecutarán las instrucciones que continúan dentro del if (identadas), en caso contrario no se ejecutará nada dentro del if.

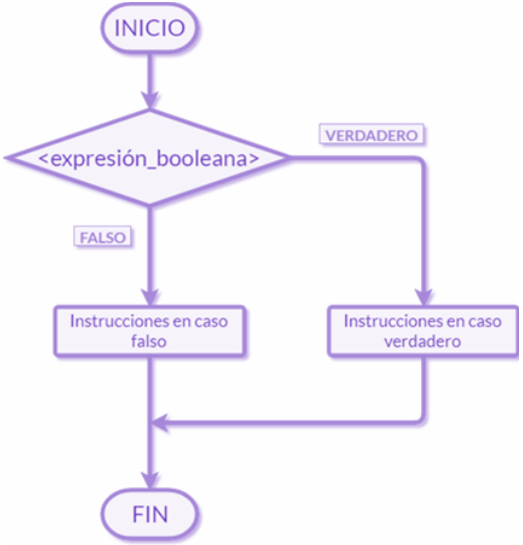
EJERCICIO: Ingresar un número y determinar si es menor que 10.

condicionales_1.py

Estructuras condicionales

Estructura condicional doble (if-else)

Voy a tener dos únicos caminos (verdadero/falso). La condición determina cuál de las dos se ejecuta.




```
if condición:
    sentencia 1
    sentencia 2
else:
    sentencia 3
instrucción fuera del if
```

*Si se debe ejecutar alguna acción cuando la condición es falsa, esta debe estar precedida por la palabra **else:**, a la altura del if al que se asocia. Éste es opcional, o sea que no tiene que colocarse si en el caso de que la condición sea falsa no hay que ejecutar ninguna sentencia.*

IMPORTANTE: En Python la indentación (sangría en español) del código es importante ya que **delimita** la ejecución de las instrucciones por **bloque de verdad** y por **bloque de falsedad**. Estas sangrías deben respetarse.

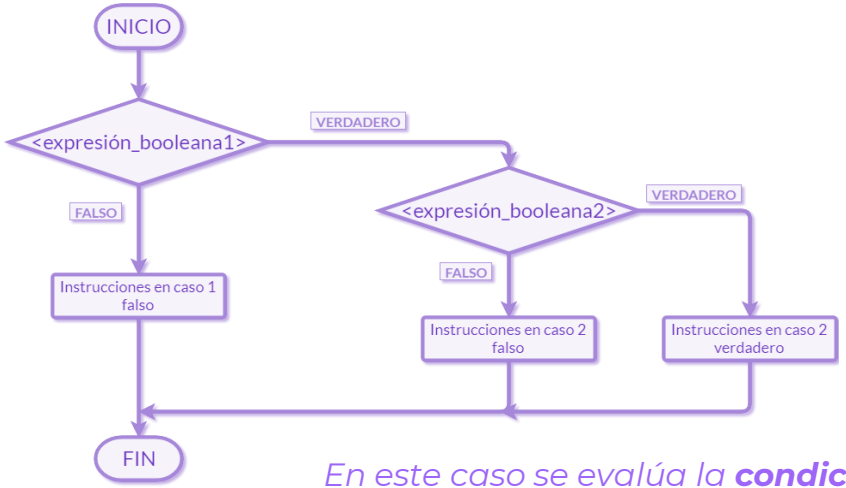
EJERCICIO: Ingresar un número distinto de cero y decir si es par o impar.

condicionales_1.py

Estructuras condicionales

Estructura condicional anidada (if-else)

Voy a tener varios caminos que se van abriendo de a dos (verdadero/falso).




```
if condición1:
    if condición2:
        sentencia 1
    else:
        sentencia 2
else:
    sentencia 3
instrucción fuera del if
```

Cada ELSE se corresponde con el IF más próximo que no haya sido emparejado, y deben tener la misma indentación.

*En este caso se evalúa la **condición1** y en caso de ser verdadera se evalúa la **condición2** (anidada). Si la segunda condición resulta **verdadera** se ejecuta la sentencia 1, si resulta **falsa** se ejecuta la sentencia 2. En caso de que la primer condición **no se cumple** se ejecuta la sentencia 3.*

EJERCICIO: Ingresar un número entero e imprimir un mensaje indicando si es cero, par o impar.

condicionales_1.py

Estructuras condicionales

Estructura condicional anidada (if-elif-else)

Simplifica la escritura de la estructura anidada, reemplaza al else...if

```
if condición1:
    sentencia 1
elif condición2:
    sentencia 2
    sentencia 3
elif condición3:
    sentencia 4
else:
    sentencia 5
instrucción fuera del if
```

*En este caso se evalúa la **condición1** y en caso de ser verdadera se ejecuta la **sentencia 1**, luego se van evaluando las siguientes condiciones y en caso de ser cierta alguna de ella se ejecutan las sentencias correspondientes. Con else se ejecutan las sentencias en caso de no cumplirse las condiciones precedentes.*

EJERCICIO: Leer un número entero e imprimir un mensaje indicando si es cero, uno, dos o cualquier otro número.

condicionales_1.py

Estructuras condicionales

Utilización de operadores lógicos (AND / OR)

Podemos reducir la cantidad de if anidados usando los operadores lógicos and (y) y or (o) dentro de la misma condición.

```py
if (condición1) and (condición2):
if (condición1) or (condición2):
```

- Una condición con **AND** es verdadera si ***todas*** las condiciones que conecta son verdaderas. La condición con OR es verdadera si ***por lo menos una*** de las dos condiciones es verdadera.
- Una condición **AND** es falsa si ***alguna de las condiciones*** que conecta es falsa y una condición OR es falsa si ***todas las condiciones*** son falsas.

Tablas de verdad: Muestran el valor de verdad de una proposición compuesta, para cada combinación de verdad que se pueda asignar.

Cond 1	Cond 2	Y (and)
V	V	**V**
V	F	**F**
F	V	**F**
F	F	**F**

Cond 1	Cond 2	O (or)
V	V	**V**
V	F	**V**
F	V	**V**
F	F	**F**

Cond	NO (not)
V	**F**
F	**V**

Orden de precedencia: 1° not; 2° and y 3° or

Estructuras condicionales

AND: Ejemplo

En un aviso del diario piden **ingenieros en sistemas con 5 años de experiencia como mínimo**, para ocupar un puesto laboral. A la convocatoria se presenta:

- Un **licenciado en sistemas** con **6 años de experiencia**: NO LO TOMAN, pues la primera condición es falsa.
- Un **ingeniero en sistemas** con **4 años de experiencia**: NO LO TOMAN, pues la segunda condición es falsa.
- Un **analista programador** con **4 años de experiencia**: NO LO TOMAN, pues las 2 condiciones son falsas.
- Un **ingeniero en sistemas** con **7 años de experiencia**: LO TOMAN, pues las 2 condiciones son verdaderas.

EJERCICIO: Realizar un programa que permita ingresar un número y decir si tiene de dos cifras o no.

condicionales_1.py

Estructuras condicionales

OR: Ejemplo

Tengo invitados en casa y voy a comprar 1 kilo de helado. Sé que los únicos gustos que comen son chocolate o vainilla. Después de ir a varias heladerías encontré:

- **Hay chocolate** pero **no hay vainilla**. LO COMPRO, pues la primer condición es verdadera.
- Sólo **hay vainilla**, **chocolate no**. LO COMPRO, pues la segunda condición es verdadera.
- **Hay chocolate y vainilla**. LO COMPRO, pues las dos condiciones son verdaderas.
- Hay **americana y dulce de leche**. NO LO COMPRO, pues ninguna de las condiciones es verdadera.

EJERCICIO: Ingresar dos números enteros y decir si alguno es divisible por el otro.

condicionales_1.py

Jerarquía en las comparaciones (en orden de cálculo):

1° Aritméticos (+, -, *, /, en el orden de siempre)

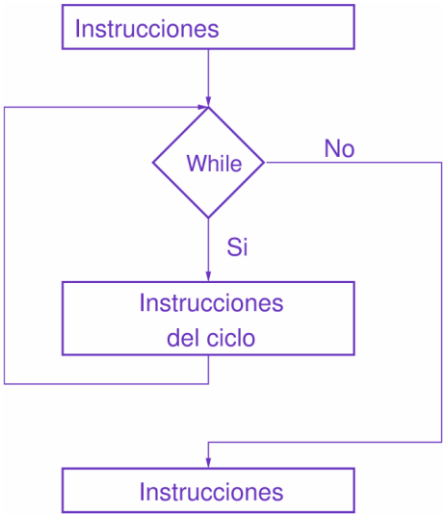
2° Relacionales

3° Y luego los lógicos (or / and)

Pero los paréntesis pueden romper esa jerarquía.

Estructuras repetitivas

Permiten repetir un código dependiendo de una condición o de un contador. Si se cumple la condición ejecuto una serie de operaciones y vuelvo a hacer la misma pregunta. Cuando la respuesta a esa pregunta ya no sea más verdadera, es decir, sea falsa, salimos del ciclo repetitivo y seguimos con el flujo normal del programa.



Dentro de las estructuras ***iterativas o repetitivas*** nos encontramos con:

- **Ciclos Exactos**: Donde conocemos la cantidad exacta de repeticiones (while/for) porque ese valor es aportado al iniciar el programa o por el usuario antes de que se inicie el ciclo. En este caso encontramos el **While/For**.
- **Ciclos Condicionales**: Donde no se conoce de antemano la cantidad de repeticiones, sino que va a depender de una condición que puede variar. Va a cortar cuando el usuario ingrese un determinado valor, se puede repetir 1 vez, n veces o ninguna vez, es variable. En este caso encontramos el ***While***.

A diferencia de otros lenguajes, en Python no existe el Do...While

IMPORTANTE: Debemos evitar los **bucles infinitos**, es decir aquellos cuya condición no se cumple nunca y no permiten salir del bucle

Estructuras repetitivas | While

Permite ejecutar un bloque de sentencias mientras la condición a la que precede es **verdadera**. El ciclo finaliza cuando la condición del while es **falsa**. Puede corresponder a los ***ciclos exactos*** o a los ***ciclos condicionales***.

```
while condición:
    sentencia 1
    sentencia 2
siguiente sentencia fuera del while
```

*Si la condición es **verdadera** "entra" al while y se ejecutan las sentencias que están debajo de esta instrucción indentadas.*

*Si la condición es **falsa** "no entra" al while y ejecuta la próxima instrucción fuera del while (la primera que no tiene inentación respecto de ese while).*

Esta instrucción no permite saber de antemano cuantas veces se va a repetir el bloque de sentencias que abarca, ya que el corte se produce por el valor de verdad de la condición.

EJERCICIO: Leer números y mostrarlos hasta que se ingrese un número negativo (ciclo condicional).

EJERCICIO: Desarrollar un programa que cuente hasta 4, utilizando una sola variable para contar (ciclo exacto).

EJERCICIO: Ingresar números enteros y sumar solamente los números positivos, el programa finalizará ingresando 0 (cero) (ciclo condicional).

repetitivas_1.py

Estructuras repetitivas | While

Uso de contadores y acumuladores

El **contador** se incrementará en 1 por cada “vuelta” del ciclo iterativo, mientras que el acumulador irá incrementando (+) o decrementando (-) en cada iteración.

Ejemplo: Ingresar 5 valores por teclado, obtener su suma y su promedio.

PY

```
cont= 1
suma= 0
while cont <= 5:
    num= int(input("Ingrese un número: "))
    suma = suma + num   # Acumulamos, es equivalente suma += num
    cont = cont + 1     # Incrementamos, es equivalente cont += 1

print("La suma es:", suma)
print("El promedio es:", suma/cont)
```

*Este ejemplo corresponde a un **ciclo exacto**.*

Estructuras repetitivas | For

Esta sentencia de repetición se utiliza cuando sabemos exactamente la cantidad de repeticiones que se deben hacer (corresponde entonces a los ***ciclos exactos***). El bloque de instrucciones se repetirá un número determinado de veces (iteraciones o repeticiones).

```
for i in range(inicio, fin, paso):
    sentencia1
    sentencia2
primer sentencia fuera del for
```

i: *Es la variable que controla el número de veces que se ejecuta el bloque de instrucciones que queda dentro del for.*

inicio: *Es el valor inicial de i*

fin: *Es el valor hasta el que se va a ir incrementando i de a tantas unidades como este indicado en* ***paso****. El ciclo se repite mientras i sea menor que fin.*

Algunos parámetros de los que están entre paréntesis pueden no escribirse; por ejemplo, si solo se escribe **un número**, este corresponde al valor de ***fin*** y asume que el valor inicial es cero y el paso es uno. Y si se escriben solo ***dos valores*** dentro del paréntesis, se asume que son el de ***inicio*** y ***fin*** y que el paso es uno.

```
for i in range(fin):
```
→ Se asume que inicio = 0 y paso = 1

```
for i in range(inicio, fin):
```
→ Se asume que paso = 1

Estructuras repetitivas | For

EJERCICIO: Mostrar los números del 1 al 10.

EJERCICIO: Mostrar los múltiplos de 5 comprendidos entre 12 y 80.

EJERCICIO: Imprimir los números del 9 al 1.

EJERCICIO: Mostrar los números del 10 al 20

EJERCICIO: Ingresar cinco números y mostrar su suma

repetitivas_1.py

*Estos ejemplos corresponden a los llamados **ciclos exactos**.*

Componentes de una Estructura Iterativa

Los componentes de un ciclo son:

- Expresiones de inicialización
- Condición/es lógica/s
- Acciones a realizar dentro del ciclo
- Condiciones de terminación
- Expresiones de finalización

Material complementario

- **Ejemplo ejercicio estructura secuencial:** https://www.youtube.com/watch?v=8OJT494xm74
- **Estructura Condicional:** https://www.youtube.com/watch?v=kIkAhld32O8
- **Tablas de Verdad:** https://www.youtube.com/watch?v=Smzj5xOTi4I
- **Ejemplos ejercicios if-else y elif:** https://www.youtube.com/watch?v=PKFKoAN2zEo
- **While para ciclos condicionales:** https://youtu.be/Ll8Q48_yPIM
- **Ejemplos While:**
  - https://www.youtube.com/watch?v=I6T_qjYiDDM&list=PLb_E6BNMg5j7-MJ0ctjvKQlv2PU7qbMDb&index=21
  - https://www.youtube.com/watch?v=vhW_cIidSQI&list=PLb_E6BNMg5j7-MJ0ctjvKQlv2PU7qbMDb&index=22
- **For para ciclos exactos:** https://www.youtube.com/watch?v=TPXPoUkUNqg&list=PLb_E6BNMg5j7-MJ0ctjvKQlv2PU7qbMDb&index=17