Sesión 3

Sentencias de Control : If - else
Sentencia de Control: While
Sentencia de Control: For (cadenas y listas)

Condicional: if - else

Programación Básica con Python

Graciela Meza Lovon, Yessenia Yari Ramos, Alvaro Mamani Aliaga

Expresiones Booleanas

- True = 1
- False = 0

- Tener en consideración que true y false no son expresiones booleanas.

Operadores de Comparación

- x == y # x es igual a y
- x != y # x no es igual a y
- x > y # x es mayor que y
- x < y # x es menor que y
- x >= y # x es mayor o igual a y
- x <= y # x es menor o igual a y

- Recordar que “=” en Python es un signo de asignación y que no existe =< ó =>

Operadores Lógicos

```
>>> a = 10
>>> b = 20
>>> c = 30
```

- **and**

a<b and a>c

False

- **or**

a<b or a>c

True

- **not**

not(a>b)

True

Operador Modulo

- Usa el símbolo %
- Calcula el residuo de la división entera de los números
- Ejemplo
  - a%b da como resultado el residuo de la división entera de a / b

```
print ("5%3 es: ", 5%3)
print ("6%3 es: ", 6%3)
print ("8%3 es: ", 8%3)
print ("7%5 es: ", 7%5)
```

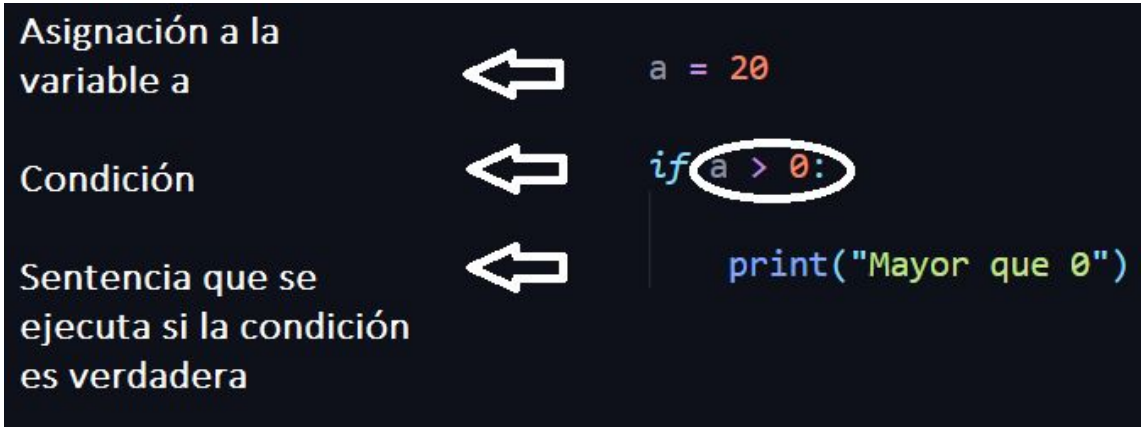
```
5%3 es:  2
6%3 es:  0
8%3 es:  2
7%5 es:  2
```

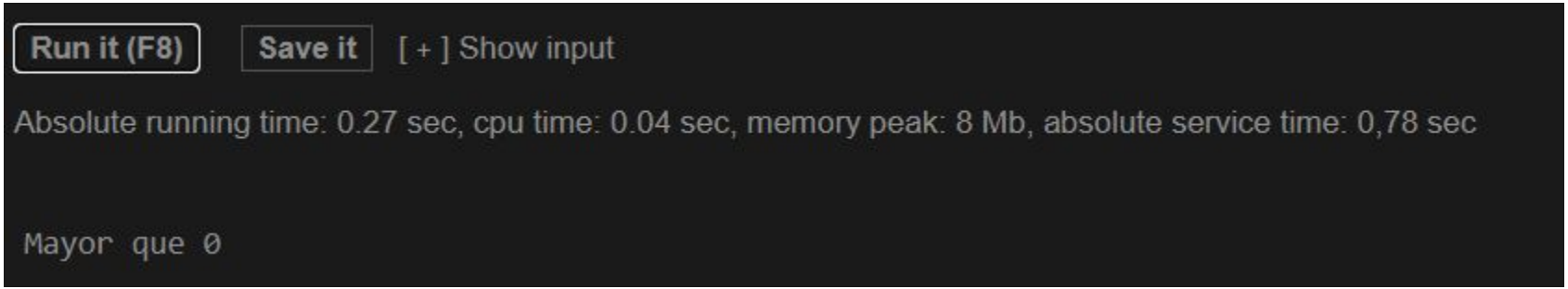
Ejecución Condicional

- Revisa condiciones y cambia el comportamiento del programa dependiendo de la evaluación de dichas condiciones
- La condición más simple es if

- En el bloque de sentencias puede tener una o varias sentencias.

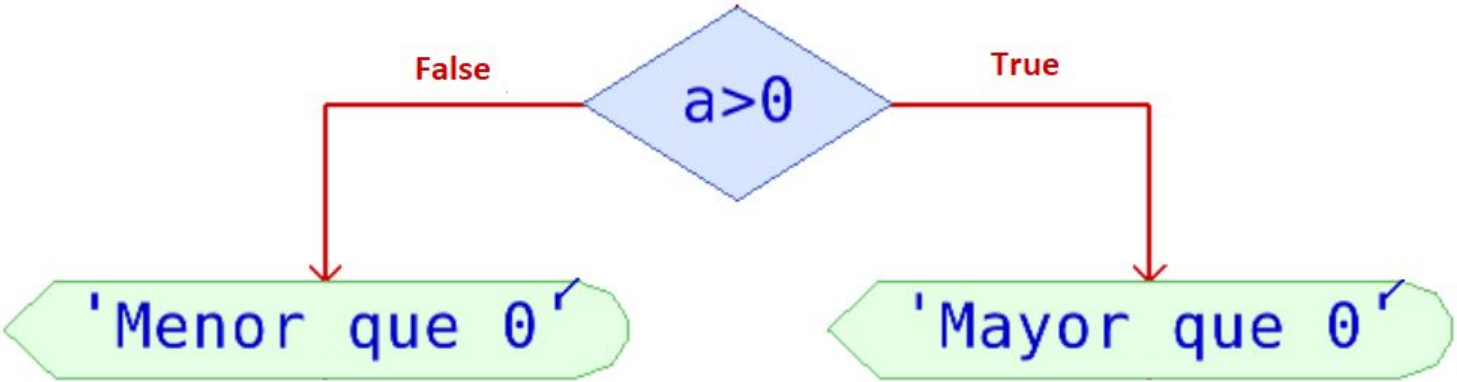
Ejecución Condicional





Ejecución Alternativa

- En el ejemplo anterior ¿qué pasaría si “ a = -10” ?
  - No debería imprimir nada en consola …..
- Pero si deseamos que algo imprima usamos la ejecución alternativa (ramas).



Ejecución Alternativa

Asignación a la variable a

Condición

Sentencia que se ejecuta si la condición es verdadera

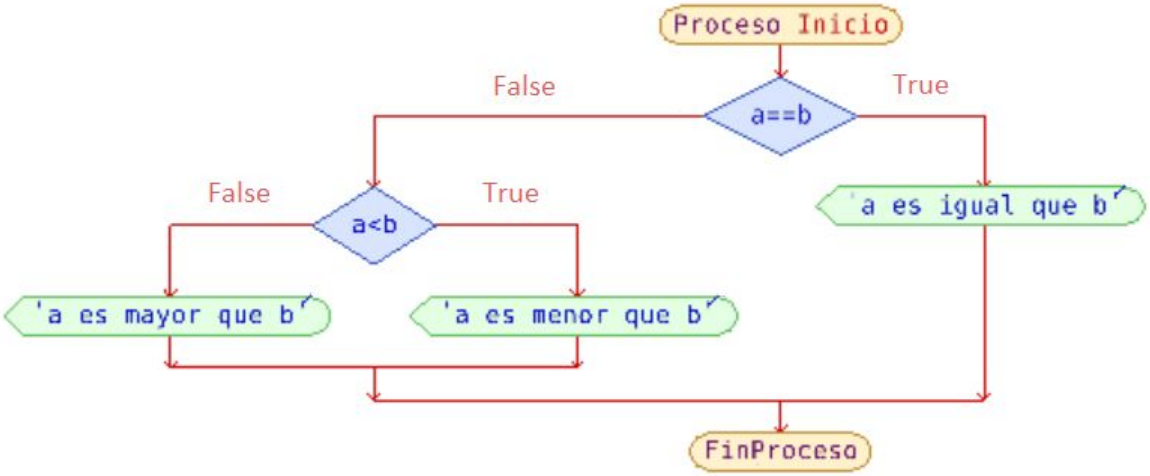
Sentencia que se ejecuta si la condición es falsa

```
a = 20

if a > 0:
    print("Mayor que 0")
else:
    print("Menor o igual que 0")

print("Salimos del if")
```

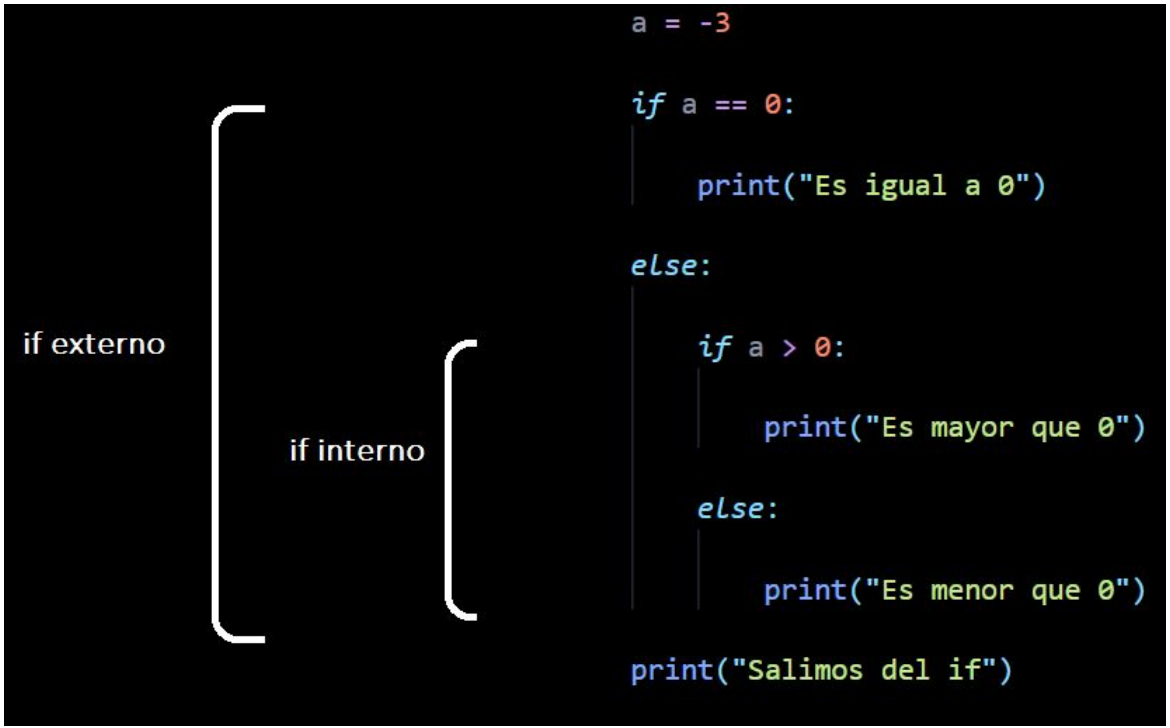
```
Mayor que 0
Salimos del if
```

Ejercicio 1

- Se tienen dos números a y b, queremos saber si a es mayor o menor o igual, ¿cómo resolvemos el problema?
- Se necesitan más de dos “if”



Condicionales Anidados




```python
a = -3

if a == 0:
    print("Es igual a 0")
else:
    if a > 0:
        print("Es mayor que 0")
    else:
        print("Es menor que 0")
print("Salimos del if")
```

```
Es menor que 0
Salimos del if
```

Ejercicio 2

- Indicar si un número está en el intervalo de 0 a 10, incluyendo 0 y 10

SOLUCIÓN 1

D: > EjerciciosPython > condicional1.py > ...

```
1   #intervalo 0 a 10
2   a = 5
3   if a >= 0:
4       if a <= 10:
5           print("a está en el intervalo de 0 a 10")
```

Ejercicio 2

- Indicar si un número está en el intervalo de 0 a 10

SOLUCIÓN 2

D: > EjerciciosPython > condicional1.py > ...

```
1   #intervalo 0 a 10
2   a = 5
3   if a >= 0 and  a <= 10:
4           print("a está en el intervalo de 0 a 10")
5
```

Ejercicio 3

- Escribir un programa que encuentre el máximo de tres números.

Iteración: while

Programación Básica con Python

Graciela Meza Lovon, Yessenia Yari Ramos, Alvaro Mamani Aliaga

Contenido

- La sentencia while
- Tablas de una dimensión
- Tablas de dos dimensiones

La sentencia while

- Se usa para realizar tareas repetitivas.
- Sintaxis:

```
while <expresión_booleana>:
    <1era sentencia>
    …
    <Última sentencia>
```

La sentencia while

```
while <expresión_booleana>:
    <1era sentencia>
    …
    <Última sentencia>
```

- **Pasos:**

1. Evalúa la condición, resultando en False (falso) o True (cierto).
2. Si la condición es falsa (False), se sale de la sentencia while y continúa la ejecución con la siguiente sentencia (afuera del while).
3. Si la condición es cierta (True), se ejecutan cada una de las sentencias del cuerpo y regresa al paso 1.

La sentencia while

- Ejemplo: Escriba un programa que muestre en la pantalla “Hola Mundo” 10 veces.

La sentencia while

```
x=1                                  #variable contador
while x<=10:
    print ("Hola Mundo")
    x = x + 1
```

La sentencia while

```
x=1
while x<=10:
    print ("Hola Mundo")
    x = x + 1
```

```
Salida:
Hola Mundo
Hola Mundo
Hola Mundo
Hola Mundo
Hola Mundo
Hola Mundo
Hola Mundo
Hola Mundo
Hola Mundo
Hola Mundo
```

La sentencia while

- Ejemplo: Escriba un programa que muestre en pantalla los números enteros en el rango de [0,10] en orden ascendente.

La sentencia while

```
x=0
while x<=10:  # x<11
    print(x)
    x = x + 1
```

La sentencia while

```
x=0
while x<=10:
    print(x)
    x = x + 1
```

Salida:

```
0
1
2
3
4
5
6
7
8
9
10
```

La sentencia while

- Ejemplo: Escriba un algoritmo que muestre en pantalla la cuenta regresiva del 10 al 1, y que luego muestre en pantalla la palabra **Despegue**

La sentencia while

```
n=10
while n > 0:                # n>=1
    print (n)
    n = n-1
print ("Despegue")
```

La sentencia while

```
n=10
while n > 0:
    print(n)
    n = n-1
print ("Despegue")
```

Salida:

```
10
9
8
7
6
5
4
3
2
1
¡Despegue!
```

La sentencia while

- En algunos casos esto no es tan fácil de asegurar el número de veces que se ejecutará el while.

```
n=10
while n != 1:
    print(n)
    if n%2 == 0:
        n = n/2
    else:
        n = n*3+1
```

Tablas

- Ejemplo: Escriba un programa que imprima la Tabla del 2.

```
n = 1
while n <= 10:
    print('2 x',n,'\t=\t',2*n)
    n = n + 1
```

Tablas

- Ejemplo: Escriba un programa que imprima la Tabla del 2.

```
n = 1
while n <= 10:
    print('2 x',n,'\t=\t',2*n)
    n = n + 1
```

```
Salida:

2 x 1 = 2
2 x 2 = 4
2 x 3 = 6
2 x 4 = 8
2 x 5 = 10
2 x 6 = 12
2 x 7 = 14
2 x 8 = 16
2 x 9 = 18
2 x 10= 20
```

Tablas

- Escriba un programa que imprima los cuadrados de los números enteros del 1 al 10.

Tablas

- Escriba un programa que imprima los cuadrados de los números enteros del 1 al 10 que sean pares

Iteración: for

Programación Básica con Python

Graciela Meza Lovon, Yessenia Yari Ramos, Alvaro Mamani Aliaga

Sentencia for y los iterables

- El bucle for se utiliza para recorrer los elementos de un objeto iterable y ejecutar un bloque de código.
- Un iterable es un objeto que se puede iterar sobre él, es decir, que permite recorrer sus elementos uno a uno.

Sintaxis

```
for <elem> in <iterable>:
    <Primera sentencia>
    ...
    <Última sentencia>
```

- **elem** es la variable que toma el valor del elemento dentro del iterable en cada paso del bucle.
- La sentencia **for** finaliza su ejecución cuando se recorren todos los elementos del iterable.

Sentencia for y la clase range

- El tipo de dato range(max), devuelve un iterable cuyos valores van desde 0 hasta max - 1
  - Sirve para implementar el bucle for basado en una secuencia numérica
  - Ejemplo: Escriba un programa que imprima los enteros consecutivos del 0 a 5

0 1 2 3 4 5

```
for x in range(6):
    print(x)
```

```
Salida:
0
1
2
3
4
5
```

Sentencia for y la clase range

- El tipo de datos range se puede invocar con uno, dos e incluso tres parámetros
- Ejemplo: Escriba un programa que imprima los enteros consecutivos del 0 a 5

1 2 3 4 5

```
for y in range(1,6):
    print(y)
```

```
Salida:
1
2
3
4
5
```

Sentencia for y la clase range

- El tipo de datos range se puede invocar con uno, dos e incluso tres parámetros
- Ejemplo: Imprimir los enteros consecutivos pares en el rango de 4 a 10

4 6 8 10

```
for ele in range(4,11,2):
    print(ele)
```

```
Salida:
4
6
8
10
```

Iteraciones Anidadas

Programación Básica con Python

Graciela Meza Lovon, Yessenia Yari Ramos, Alvaro Mamani Aliaga

Tablas

- Escriba un programa que imprima los primeros 6 múltiplos de 2.

Tablas

```
i = 1
while i <= 6:
    print (i*2)
    i = i + 1
```

Tablas

```
Salida:
2
4
6
8
10
12
```

Tablas

- Escriba que imprima los primeros 6 múltiplos de 2 en una sola línea. Los múltiplos deben ser mayores o iguales que 2.

Tablas

```
i = 1
while i <= 6:
    print (i*2, end='\t')
    i = i + 1
```

Tablas

```
Salida:
2  4  6  8  10 12
```

Tablas de dos dimensiones

- Pasemos de tablas simples, es decir de una entrada a tablas de 2 entradas.
- Ejemplo: Escriba un programa que imprima la tabla de los primeros 6 múltiplos de los primeros 10 números enteros positivos.

Tablas de dos dimensiones

- **Ejemplo:** Escriba un programa que imprima la tabla de los primeros 6 múltiplos de los primeros 10 números enteros positivos.
  - Una forma de resolver
  - Note que todo el programa es una repetición de la impresión de los múltiplos.

```
a = 1
i = 1
while i <= 6:
    print (i*a, end='\t')
    i = i + 1

a = 2
i = 1
while i <= 6:
    print (i*a, end='\t')
    i = i + 1

a = 3
i = 1
while i <= 6:
    print (i*a, end='\t')
    i = i + 1
…
a = 10
```

Tablas de dos dimensiones

- Ejemplo: Escriba un programa que imprima la tabla de los primeros 6 múltiplos de los primeros 10 números enteros positivos.

  - Otra forma

```
a = 1
while a <= 10:
    i = 1
    while i <= 6:
        print (i*a, end='\t')
        i = i + 1
    a = a + 1
    print()
```

Tablas de dos dimensiones

```
Salida:
1  2  3  4  5  6
2  4  6  8  10 12
3  6  9  12 15 18
4  8  12 16 20 24
5  10 15 20 25 30
6  12 18 24 30 36
7  14 21 28 35 42
8  16 24 32 40 48
9  18 27 36 45 54
10 20 30 40 50 60
```

Las sentencias break y continue

- La sentencia **continue** se usa para omitir el resto del código dentro de un bucle solo para la iteración actual. El bucle no termina pero continúa con la siguiente iteración.

```
i = 0
while i <= 5:
    i +=1
    print(i)
    if i == 3:
        continue
    print(i)
```

Las sentencias break y continue

- La sentencia break termina el ciclo que lo contiene. El control del programa fluye a la declaración inmediatamente después del cuerpo del bucle.

```
i = 0
while i <= 5:
    i +=1
    print(i)
    if i == 3:
        break
    print(i)
print("estoy fuera del while")
```

¡Gracias!