Universidad Andrés Bello®

Unab Online

Repetición

León, R. (2021). *Repetición* [apunte].

Chile. UNAB

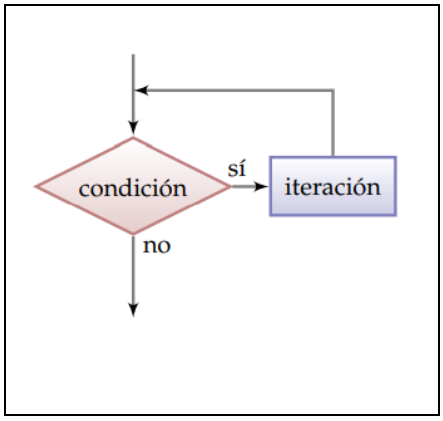
REPETICIÓN

1. Ciclos

Como se mencionó anteriormente un *ciclo* es un conjunto de sentencias que son ejecutadas varias veces, hasta que una condición de término es satisfecha.

La sentencia *while (mientras)* ejecuta un bloque de instrucciones mientras una condición sea verdadera. Cada vez que el bloque de instrucciones es ejecutado se denomina *iteración*. La condición es evaluada antes de cada iteración. Si la condición al inicio es falsa, entonces el ciclo no se ejecuta ninguna vez.

Figura 1
Funcionamiento de la sentencia **while**.



La sintaxis de la sentencia *while* es la siguiente:

```
while condicion:
    sentencias
```

Por ejemplo, en la Figura 2 se muestra un programa en donde se multiplican dos números enteros sin utilizar el operador de la multiplicación (*).

Figura 2
Ejemplo sentencia **while.**

Programa

`a = int(input("ingrese a: "))`
`b = int(input("ingrese b: "))`
`p = 0`
`n = b`
`while n > 0:`
`    n = n - 1`
`    p = p + a`
 	
`print(f"El producto de {a} y {b} es {p}")`

Salida 1 del programa
ingrese a: 5 ingrese b: 3 El producto de 5 y 3 es 15
Salida 2 del programa
ingrese a: 12 ingrese b: 7 El producto de 12 y 7 es 84

La sentencia *for (para)* permite ejecutar ciclos con rangos. Es decir, permite ejecutar un bloque de instrucciones una cantidad fija de veces. Para contabilizar la cantidad de veces, utiliza una *variable de control*, que va tomando valores distintos en cada iteración.
Una de las sintaxis para utilizar un *for* con rango es la siguiente:

for variable **in rango**(fin) :
    sentencias para cada valor de la variable de control

En la primera iteración la variable de control tiene valor 0. Al final de cada iteración el valor de la variable de control aumenta automáticamente. El ciclo termina justo antes que la variable obtenga el valor fin.

En la Figura 3 se muestra un programa que imprime los cubos de los números del 0 al 10.

Figura 3
Ejemplo sentencia **for** con rango.

Programa

```
for i in range(11):
    cubo = i ** 3
    print(f"El cubo de {i} es {cubo}")
```

Salida del programa

```
El cubo de 0 es 0
El cubo de 1 es 1
El cubo de 2 es 8
El cubo de 3 es 27
El cubo de 4 es 64
El cubo de 5 es 125
El cubo de 6 es 216
El cubo de 7 es 343
El cubo de 8 es 512
El cubo de 9 es 729
El cubo de 10 es 1000
```


Un *rango* es una sucesión de números enteros equiespaciados. Incluyendo la mostrada anteriormente, hay tres formas de definir un rango:
range(final)
range(inicial,final)
range(inicial,final,incremento)

El valor inicial siempre es parte del rango. El valor final nunca es incluido en el rango. El incremento indica la diferencia entre dos valores consecutivos del rango. Si el valor inicial es omitido, se asume que es 0. Si el incremento es omitido, se asume que es 1. En la Tabla 1 se muestra algunos ejemplos de rangos.

Tabla 1
Ejemplo de rangos.

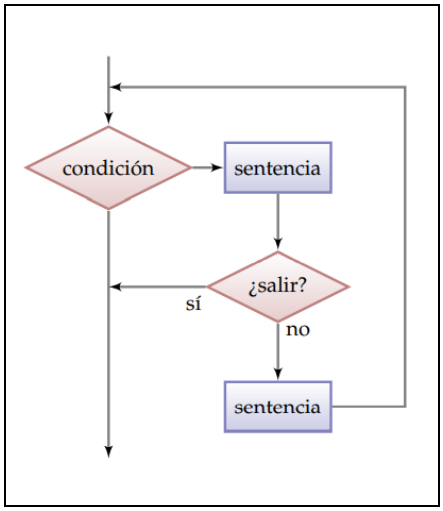
Rango	Valores
range(9)	0,1,2,3,4,5,6,7,8
range(3,13)	3,4,5,6,7,8,9,10,11,12
range(3,13,2)	3,5,7,9,11
range(11,4)	Ningún valor
range(11,4,-1)	11,10,9,8,7,6,5

En general, el ciclo *for* con rango se usa cuando el número de iteraciones es conocido antes de entrar al ciclo.

2. Salir de un ciclo

Además de las condiciones propias de las sentencias anteriores (*while* y *for*) para terminar las repeticiones, siempre es posible salir de un ciclo en medio de una iteración. La sentencia *break* se utiliza para tales casos.

Figura 4
Funcionamiento de la sentencia **break**.



Por ejemplo, un programa para saber si un número es primo o compuesto, se puede construir considerando terminar de forma prematura la búsqueda de divisores en caso de que se encuentre al primero de ellos. En la Figura 5 se puede observar el funcionamiento de la sentencia *break* para saber si un número es primo o no.

Figura 5
Ejemplo sentencia **break**.

Programa

```
n = int(input("Ingrese n: "))
es_primo= True
for d in range(2,n):
    #se encuentra un numero divisible.
    if n % d == 0:
        es_primo = False
        break
#la variable es_primo es de tipo booleana.
if es_primo:
    print(f"{n} es primo")
else:
    print(f"{n} es compuesto")
```

Salida 1 del programa

Ingrese n: 11
11 es primo

Salida 2 del programa

Ingrese n: 1616
1616 es compuesto

Salida 3 del programa

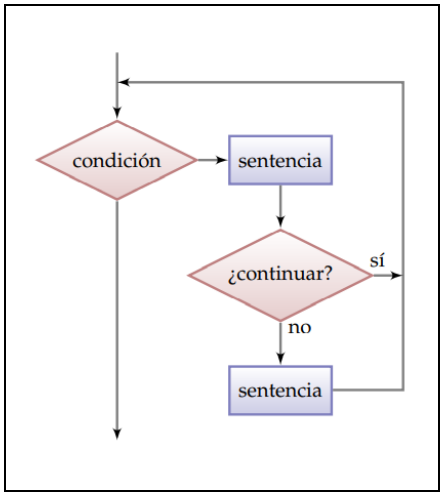
Ingrese n: 1543
1543 es primo

Notar que la sentencia *break* siempre aparecerá dentro de una sentencia *if* ya que, de otro modo, el ciclo terminaría siempre en la primera iteración.

3. Saltar a la siguiente iteración

La sentencia *continue* se utiliza para saltar a la siguiente iteración sin que termine de ejecutarse la que está en curso.

Figura 6
Funcionamiento de la sentencia **continue**.



Por ejemplo, en la Figura 7 se muestra un programa en donde se calcula el cuadrado y el cubo de los números del 1 al 20, excepto los que son múltiplos de 4.

Figura 7
Ejemplo sentencia **continue**.

Programa

```
for num in range(1,21):
    if num % 4 == 0:
        continue
    a2 = num ** 2
    a3 = num ** 3
    print(f"{num}^2 = {a2} -- {num}^3 = {a3}")
```

Salida del programa

```
1^2 = 1 -- 1^3 = 1
2^2 = 4 -- 2^3 = 8
3^2 = 9 -- 3^3 = 27
5^2 = 25 -- 5^3 = 125
6^2 = 36 -- 6^3 = 216
7^2 = 49 -- 7^3 = 343
9^2 = 81 -- 9^3 = 729
10^2 = 100 -- 10^3 = 1000
11^2 = 121 -- 11^3 = 1331
```

```
13^2 = 169 -- 13^3 = 2197
14^2 = 196 -- 14^3 = 2744
15^2 = 225 -- 15^3 = 3375
17^2 = 289 -- 17^3 = 4913
18^2 = 324 -- 18^3 = 5832
19^2 = 361 -- 19^3 = 6859
```

"Para complementar la información y contenidos presentados, puedes dirigirte a":

1. **Downey, A. (2013). Think Python. Green Tea Press.** Accesado el 2 de diciembre de 2019 en https://greenteapress.com/wp/think-python-2e/

2. **Python software foundation, Python v3 Documentation.** Accesado el 2 de diciembre de 2019 en https://docs.python.org/3/

3. Marzal Varó, A., Gracia Luengo, I., & García Sevilla, P. (2014). Introducción a la programación con Python 3. Universitat Jaume I. https://doi.org/10.6035/sapientia93 (Capítulo 1, secciones 1.1 y 1.2, páginas: 11 - 16)

Bibliografía

1. Hinojosa, A. (2016). Python paso a paso. RA-MA Editorial y Publicaciones.

2. Marzal, A., Gracia, I., García, P. (2014). Introducción a la Programación con Python 3. Universitat Jaumé Publicaciones.

3. Bonvallet, R. (2013). Apuntes de Programación. Editorial USM.