

FUNDAMENTOS DE ALGORITMOS

MÓDULO 1

RECURSO DE APRENDIZAJE 5

ESTRUCTURAS DE CONTROL EN PYTHON

ITERACIÓN




PONTIFICIA
UNIVERSIDAD
CATÓLICA DE
VALPARAÍSO



Escuela de Ingeniería

Informática



ESTRUCTURAS ITERATIVAS EN PYTHON

Repaso

El lenguaje Python incluye 2 estructuras de control iterativas :

while condición :

instrucción 1
instrucción 2
:
instrucción N

for variable **in** objeto :

instrucción 1
instrucción 2
:
instrucción N



Sentencia
break

Sentencia break

- La sentencia **break** permite “quebrar” o **detener** la ejecución de una **estructura iterativa** (**while** o **for**) en cualquier punto de la misma **si se cumple** una determinada **condición lógica**.
- Una vez que se ejecuta la sentencia **break**, el **ciclo** que la contiene se **detiene** y el programa **continuará** ejecutando la siguiente instrucción fuera del **ciclo**.

Ejemplo uso del **break**

- Escriba un **programa** que lea un conjunto de números enteros e indique para cada uno de ellos si es **par o impar** hasta que se ingrese un **valor igual a cero**; en cuyo caso se debe detener el **ciclo**.

Entrada	Resultado
12 5 0	12 ES PAR 5 ES IMPAR FIN PROGRAMA
17 128 46 0	7 ES IMPAR 128 ES PAR 46 ES PAR FIN PROGRAMA

Solución en Python

```
1 # mientras sea verdadero ==> Loop Infinito controlado
2 while True :
3
4     # se lee un número entero
5     numero = int(input())
6
7     # en este punto si numero es 0 para y termina el ciclo
8     if numero == 0 :
9         break
10
11     elif numero % 2 == 0 :
12         print(numero,"ES PAR")
13     else :
14         print(numero,"ES IMPAR")
15
16 # instrucción fuera del ciclo
17 print("FIN PROGRAMA")
```



Sentencia
continue

Sentencia continue

- La sentencia **continue** permite **finalizar** la iteración actual de una **estructura iterativa** (**while** o **for**) en cualquier punto de la misma **si se cumple** una determinada **condición lógica**.
- Una vez que se ejecuta la sentencia **continue**, el flujo de control del programa vuelve al **comienzo del ciclo** para iniciar la **siguiente iteración**, ignorando (no ejecutando) todas las instrucciones que quedan por ejecutar dentro de la **iteración actual**.

Ejemplo uso del continue

- Escriba un **programa** que imprima los **números enteros impares** que se encuentran en el rango de **a hasta b**. Los valores de a y b serán ingresados por la/el usuaria(o).

Entrada	Resultado
4 15	Números impares entre 4 y 15 : 5 7 9 11 13 15
5 20	Números impares entre 5 y 20 : 5 7 9 11 13 15 17 19

Solución en Python

```
1  # inicialmente se leen los valores de a y b
2  a = int(input())
3  b = int(input())
4
5  # se escribe título
6  print("Números impares entre",a,"y",b,":")
7
8  # se itera con la variable numero
9  # tomando valores desde a hasta b
10 for numero in range( a , b+1 ) :
11
12     # se verifica si numero es par
13     # si lo fuera continua con el siguiente valor
14     if numero % 2 == 0 :
15         continue
16
17     # si no fue par, es decir si es impar imprime el valor de numero
18     print(numero , end = " ")
```



ESTRUCTURAS
ITERATIVAS
ANIDADAS

Anidamiento

- Una **estructura iterativa** puede incluir dentro de sus instrucciones **cuantas estructuras iterativas** sean necesarias para resolver un determinado problema. Estas a su vez puede contener a otras y así sucesivamente.
- Cuando una **estructura iterativa** está dentro de otra se le denomina **ciclo anidado**.

Ejemplo Estructuras Iterativas Anidadas

- Escriba un **programa** que imprima la **tabla de multiplicar del 1 al 10** de los primeros **n números naturales**. El valor de **n** será ingresado por la/el usuaria(o). **Se asegura que $n \geq 1$** .

Entrada	Resultado
2	<pre>Tabla de Multiplicar del 1 al 10 de 1 1 x 1 = 1 1 x 2 = 2 1 x 3 = 3 1 x 4 = 4 1 x 5 = 5 1 x 6 = 6 1 x 7 = 7 1 x 8 = 8 1 x 9 = 9 1 x 10 = 10 Tabla de Multiplicar del 1 al 10 de 2 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</pre>

Solución Python

```
1 # inicialmente se lee valor de n
2 n= int(input())
3
4 # ciclo con i tomando valores desde 1 a n
5 ▼ for i in range ( 1 , n+1 ) :
6
7     # escribe título ==> tabla del i
8     print("Tabla de Multiplicar del 1 al 10 de", i )
9
10    # ciclo interno con j - siempre desde 1 a 10
11 ▼ for j in range (1,11) :
12
13        # escribe valor de i, j y la multiplicación
14        print( i , "x" , j , "=" , i*j )
15
```

