**Instrucción break**

En Python, la instrucción break le proporciona la oportunidad de cerrar un bucle cuando se activa una condición externa. Debe poner la instrucción break dentro del bloque de código bajo la instrucción de su bucle, generalmente después de una instrucción if condicional.

Veamos un ejemplo en el que se utiliza la instrucción break en un bucle for:

number = 0

for number in range(10):

if number == 5:

break # break here

print('Number is ' + str(number))

print('Out of loop')

Copy

En este pequeño programa, la variable number se inicia en 0. Luego, una instrucción for construye el bucle siempre que la variable number sea inferior a 10.

En el bucle for, existe una instrucción if que presenta la condición de que *si* la variable number es equivalente al entero 5, *entonces* el bucle se romperá.

En el bucle también existe una instrucción print() que se ejecutará con cada iteración del bucle for hasta que se rompa el bucle, ya que está después de la instrucción break.

Para saber cuándo estamos fuera del bucle, hemos incluido una instrucción print() final fuera del bucle for.

Cuando ejecutemos este código, el resultado será el siguiente:

Output

Number is 0

Number is 1

Number is 2

Number is 3

Number is 4

Out of loop

Esto muestra que una vez que se evalúa el entero number como equivalente a 5, el bucle se rompe porque se indica al programa que lo haga con la instrucción break.

La instrucción break hace que un programa interrumpa un bucle.

**Instrucción continue**

La instrucción continue da la opción de omitir la parte de un bucle en la que se activa una condición externa, pero continuar para completar el resto del bucle. Es decir, la iteración actual del bucle se interrumpirá, pero el programa volverá a la parte superior del bucle.

La instrucción continue se encuentra dentro del bloque de código abajo de la instrucción del bucle, generalmente después de una instrucción if condicional.

Usando el mismo programa de bucle for que en la sección anterior [Instrucción break](https://www.digitalocean.com/community/tutorials/how-to-use-break-continue-and-pass-statements-when-working-with-loops-in-python-3#break-statement), emplearemos la instrucción continue en vez de la instrucción break:

number = 0

for number in range(10):

if number == 5:

continue # continue here

print('Number is ' + str(number))

print('Out of loop')

Copy

La diferencia al usar la instrucción continue en vez de una instrucción break radica en que nuestro código continuará a pesar de la interrupción cuando la variable number se evalúe como equivalente a 5. Veamos el resultado:

Output

Number is 0

Number is 1

Number is 2

Number is 3

Number is 4

Number is 6

Number is 7

Number is 8

Number is 9

Out of loop

Aquí, Number is 5 nunca aparece en el resultado, pero el bucle continúa después de ese punto para imprimir líneas para los números 6 a 10 antes de cerrarse.

Puede usar la instrucción continue para evitar código condicional profundamente anidado o para optimizar un bucle eliminando los casos frecuentes que desee rechazar.

La instrucción continue hace que un programa omita determinados factores que surgen dentro de un bucle, pero luego continuará con resto de este.

**Instrucción Pass**

Cuando se activa una condición externa, la instrucción pass permite manejar la condición sin que el bucle se vea afectado de ninguna manera; todo el código continuará leyéndose a menos que se produzca la instrucción break u otra instrucción.

Igual que con las demás instrucciones, la instrucción pass se encuentra dentro del bloque de código abajo de la instrucción del bucle, normalmente después de una instrucción if condicional.

Usando el mismo bloque de código que antes, sustituiremos la instrucción break o continue con una instrucción pass:

number = 0

for number in range(10):

if number == 5:

pass # pass here

print('Number is ' + str(number))

print('Out of loop')

Copy

La instrucción pass que se produce después de la instrucción condicional if le indica al programa que continúe ejecutando el bucle e ignore el hecho de que la variable number se evalúa como equivalente a 5 durante una de sus iteraciones.

Ejecutaremos el programa y consideraremos el resultado:

Output

Number is 0

Number is 1

Number is 2

Number is 3

Number is 4

Number is 5

Number is 6

Number is 7

Number is 8

Number is 9

Out of loop

Al usar la instrucción pass en este programa, observamos que el programa se ejecuta exactamente como lo haría si no hubiera instrucción condicional en el programa. La instrucción pass le indica al programa que ignore esa condición y continúe ejecutando el programa como de costumbre.

La instrucción pass puede crear clases mínimas o actuar como marcador de posición al trabajar en un nuevo código y pensar en un nivel de algoritmo antes de preparar detalles.