Aprende a programar con Minecraft

### **Controlar bucles con una variable de recuento**

Las variables de recuento son una forma habitual de almacenar el número de veces que se ha repetido un programa. Ya has visto estas variables en acción unas cuantas veces. Veamos otro ejemplo:

count = 0  
while count < 5:  
print(count)  
count += 1

La condición del bucle while comprueba que el valor de la variable de recuento es menor que 5. En el cuerpo del bucle, he cambiado el valor de la variable cuenta para registrar el número de veces que se ha repetido la cuenta. Añadir al valor de una variable de recuento se llama *incrementar*.

La última línea de este código incrementa en 1 el valor de la variable de recuento. Cada vez que se repita el código, comprobará el nuevo valor de la variable de recuento para ver si es menor que 5. Cuando sea igual o mayor que 5, el bucle se detendrá.

Si te olvidas de incrementar la variable count, acabarás con un *bucle infinito*, que repetirá el bucle eternamente, como se muestra en el siguiente ejemplo:

count = 0  
while count < 5:  
print(count)

El valor de count es siempre 0 porque nunca se incrementa. Por lo tanto, la condición del bucle siempre será Verdadero, y el bucle se repetirá *eternamente*. Si no me crees, ¡prueba a ejecutar el código!

0  
0  
0  
0  
0  
--snip--

Para interrumpir la ejecución de este programa infinito, pulsa CTRL-C. Para corregir el código, sólo tienes que añadir la cuenta de líneas += 1 al cuerpo del bucle. Ahora no quedarás atrapado en un bucle infinito. ¡Uf!

Los recuentos no siempre tienen que incrementarse en 1. En algunas situaciones puedes querer incrementar el recuento en un valor diferente. En el siguiente ejemplo, la cuenta se incrementa en 2 cada vez; el resultado es que el código imprime todos los números pares entre 0 y 100:

count = 0  
while count < 100:  
print(count)  
count += 2

También puedes contar hacia atrás utilizando un número negativo para *disminuir* el valor de la cuenta. El siguiente código cuenta *hacia* atrás desde 100 hasta 1:

count = 100  
while count > 0:  
print(count)  
count -= 1

La única diferencia entre este ejemplo y los anteriores es la condición. Aquí he utilizado un comparador mayor que(>). Mientras la cuenta sea mayor que 0, el bucle continúa; cuando la cuenta llega a 0, el bucle se detiene.

**NOTA**

*La variable utilizada para controlar un bucle no siempre se llama* recuento*.* *Podrías llamarla* repeticiones *o cualquier otra cosa que desees. Si miras el código de otras personas, verás una gran variedad de nombres diferentes.*

#### **Misión 34: La maldición acuosa**

Intentemos algo un poco desagradable y escribamos una maldición para el jugador que dure poco tiempo. Las maldiciones en los videojuegos pueden *debilitar* al personaje de alguna manera, como ralentizándolo o debilitándolo, a menudo sólo durante un rato.

Crearemos un programa de maldición que coloque un bloque de agua que fluya en la posición del jugador una vez por segundo durante 30 segundos. Esto hará que al jugador le resulte difícil moverse sin ser empujado por el agua que fluye.

El siguiente código coloca un bloque de agua que fluye en la posición del jugador:

*waterCurse.py*

from mcpi.minecraft import Minecraft  
mc = Minecraft.create()  
  
pos = mc.player.getPos()  
mc.setBlock(pos.x, pos.y, pos.z, 8)

Este código colocará un bloque de agua en la posición actual del jugador sólo una vez. Tu tarea es hacer que se repita. El código final debe repetirse 30 veces, y cada iteración del bucle debe durar 1 segundo.

Guarda este código como *waterCurse.py* en la carpeta *whileLoops* y ejecútalo una vez para asegurarte de que funciona. Deberías ver aparecer un único bloque de agua en la posición del jugador antes de que el programa se detenga.

Vamos a ver qué hay que añadir a continuación para que esta maldición dure. Utiliza lo que has aprendido sobre los bucles while y las variables de recuento para hacer lo siguiente:

1. Añade una variable de recuento al programa.
2. Añade un bucle al programa para repetir las dos últimas líneas de código. El bucle debe repetirse 30 veces.
3. Incrementa la variable cuenta al final del bucle.
4. Importa el módulo de tiempo (en la primera línea de tu programa) y añade una suspensión de 1 segundo en la última línea del bucle while.

Guarda el programa y pruébalo. Mientras caminas por el mundo del juego, el programa debería crear un bloque de agua cada segundo durante 30 segundos. Si te quedas atascado, vuelve a los pasos de la Misión nº 33[(página 129](ch07.xhtml#page_129)) para obtener ayuda.

La[Figura 7-2](ch07.xhtml#ch7fig2) muestra la maldición en acción.



*Figura 7-2: ¡Oh, no! Me persigue una pequeña inundación.*

**OBJETIVO EXTRA: UNA INUNDACIÓN MÁS RÁPIDA**

¿Cómo harías que el bucle se repitiera el doble de rápido (cada medio segundo) sin dejar de durar 30 segundos?

#### **Bucles while infinitos**

En la mayoría de los casos, es muy importante que la condición booleana de tu bucle while se convierta finalmente en Falso; de lo contrario, el bucle iterará eternamente y tu ordenador podría bloquearse.

Pero hay ocasiones en las que puedes querer programar un bucle infinito. Por ejemplo, los videojuegos suelen utilizar un bucle infinito para comprobar la entrada del usuario y gestionar el movimiento del jugador. Por supuesto, estos videojuegos incluyen un botón de Salir para que puedas pausar o detener los bucles infinitos cuando necesites tomarte un descanso.

Una forma sencilla de crear un bucle infinito es utilizar una condición Verdadero al definir un bucle while, como se muestra aquí:

while True:  
print("Hello")

Este código se repetirá eternamente, imprimiendo la cadena "Hola" una y otra vez. Tanto si pretendías crear un bucle infinito como si no, pulsar CTRL-C en el intérprete de comandos de Python es una forma habitual de detenerlo. En IDLE puedes seleccionar **Shell** image **Reiniciar Shell** para detener también el bucle.

Ten en cuenta que cualquier código que se coloque *después de* un bucle while infinito nunca se ejecutará. En el siguiente ejemplo, la última línea de código es inalcanzable debido al bucle while infinito que le precede:

while True:  
print("Hello")  
print("This line is never reached")

Aunque los bucles infinitos a veces pueden ser complicados, también puedes crearlos para hacer un montón de cosas chulas. ¡Intentémoslo a continuación!

#### **Misión 35: Camino de flores**

El programa que escribirás en esta misión es como el de la Misión 34, pero en lugar de colocar bloques de agua, crearás un rastro de flores detrás del jugador. ¡Las flores son mucho más bonitas que las inundaciones!

Abre el archivo *waterCurse.py* en la carpeta *whileLoops* y guárdalo como *flowerTrail.py*.

Para que aparezca un rastro infinito de flores mientras el jugador camina por el juego, haz los siguientes cambios en el programa:

1. Cambia la condición del bucle while a True.
2. Elimina la variable cuenta y el incremento.
3. Cambia el argumento del tipo de bloque en la función setBlock() de 8 a 38.
4. Reduce el valor del argumento en la función sleep () a 0,2 para que aparezcan cinco flores cada segundo.
5. Guarda el programa y ejecútalo. [La Figura 7-3](ch07.xhtml#ch7fig3) muestra lo que deberías ver.



*Figura 7-3: ¡Mira qué flores más bonitas!*

**OBJETIVO EXTRA: UN RASTRO DE DESTRUCCIÓN**

El programa *flowerTrail.py* es muy flexible. Prueba a cambiar el tipo de bloque que coloca el programa. Un tipo de bloque divertido es el TNT explosivo(setBlock(x, y, z, 46, 1)). Fíjate en el argumento extra 1 después de 46, que es el tipo de bloque TNT. El 1 establece el estado del TNT para que detone con sólo golpearlo, sin necesidad de pedernal y acero. ¡Sólo tienes que pulsar el botón izquierdo del ratón unas cuantas veces cuando apuntes a la TNT para hacerla explotar!

[anterior](ch07_2.html)[Subtema 3 de 6: (Ver todo)](ch07.html)[siguiente](ch07_4.html)