



115- [PF] - Lab - Bucles

Ejercicio 1: Trabajo con un bucle while

Un bucle `while` hace que una sección del código se repita hasta que se cumpla una determinada condición. En este ejercicio, creará un script en Python que pedirá al usuario adivinar un número.

Impresión de las reglas del juego

1. En el panel de navegación del IDE, elija el archivo `.py` que creó en la sección *Creación del archivo de ejercicios de Python* anterior.
2. Utilice la función `print()` para informar al usuario acerca del juego:

```
print("Welcome to Guess the Number!")  
print("The rules are simple. I will think of a number, and you will try to guess it.")
```

1. Guarde y ejecute el archivo.
2. Confirme que el script se ejecuta de forma correcta y que la salida se muestra según lo previsto.

Importación aleatoria y escritura de un bucle while

Utilizará el comando `import` para incluir el código que escribió otra persona. Hasta ahora, ha utilizado funciones integradas. Recuerde que una función es un fragmento

de código reutilizable.

1. Al inicio del archivo, incluya el módulo de Python (que es un tipo de biblioteca) llamado `random`.

Nota: Las instrucciones `import` se colocan al inicio del script por convención.

```
import random
```

1. Coloque el cursor en la línea siguiente a la segunda instrucción `print()`. Luego, escriba una instrucción que generará un número aleatorio entre 1 y 10 mediante el uso de la función `randint()` del módulo `random`.

```
number = random.randint(1,10)
```

1. Monitoree si el usuario adivinó su número con la creación de una variable llamada *isGuessRight*:

```
isGuessRight = False
```

1. Para gestionar la lógica del juego, cree un bucle `while`:

```
while isGuessRight != True:
    guess = input("Guess a number between 1 and 10: ")
    if int(guess) == number:
        print("You guessed {}. That is correct! You win!".format(guess))
        isGuessRight = True
    else:
        print("You guessed {}. Sorry, that isn't it. Try again.".format(guess))
```

Nota: El bucle `while` repetirá el código dentro del bucle hasta que se adivine el número correcto, lo que está representado por la condición `isGuessRight != True` en el código. Además, Python utiliza la sangría con espacios para determinar los bloques lógicos, es decir, qué instrucciones se consideran parte del

bucle `while`. Puede poner sangría en una línea si coloca el cursor junto a una instrucción y presiona TAB.

1. Guarde el archivo.

Escritura de un pseudocódigo

Antes de ejecutar el script en Python, describa la lógica del bucle `while` en oraciones escritas (sin código). Esta técnica se denomina *pseudocodificación*.

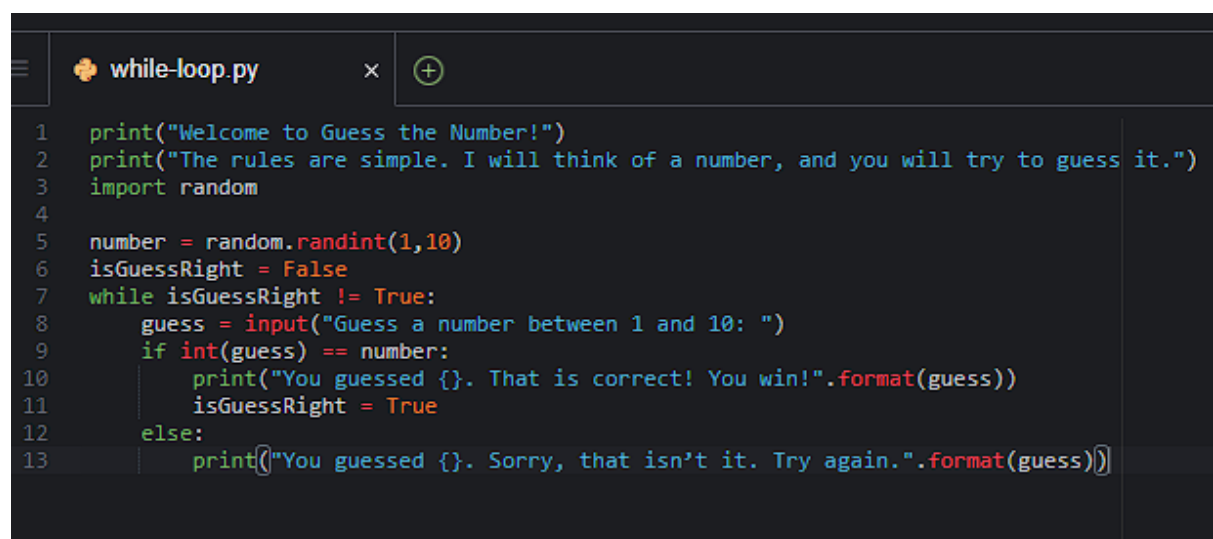
Por ejemplo:

- Si el usuario no ha adivinado la respuesta correcta, ingrese el bucle.
- Pida al usuario que adivine el número.
- ¿Es el número correcto?
- Si la respuesta es correcta, dígaselo al usuario y salga del bucle.
- Si no ha adivinado el número, indique al usuario que fue una suposición incorrecta y continúe con el bucle.

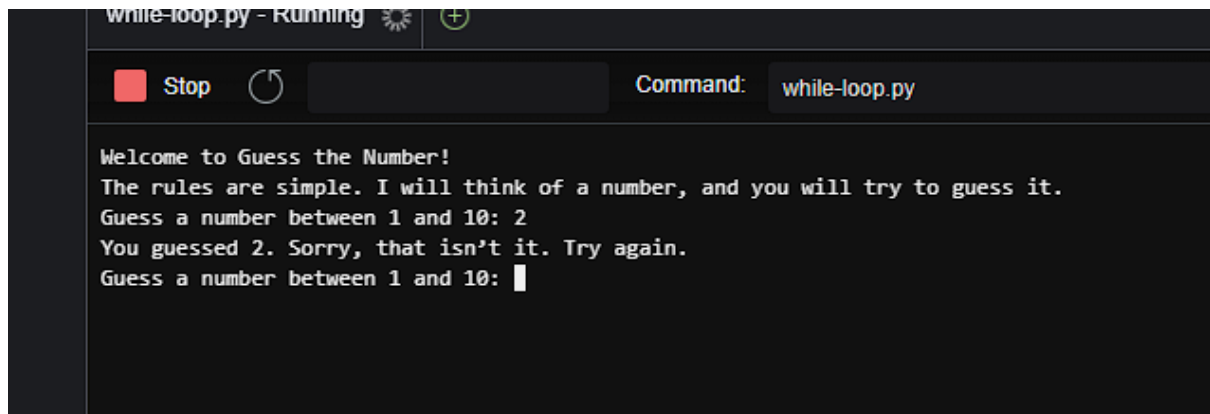
Ejecución del script

Ahora ejecute el script en Python y compruebe si funciona.

1. Ejecute el archivo.
2. Confirme que el script se ejecuta de forma correcta y que la salida se muestra según lo previsto.



```
1 print("Welcome to Guess the Number!")
2 print("The rules are simple. I will think of a number, and you will try to guess it.")
3 import random
4
5 number = random.randint(1,10)
6 isGuessRight = False
7 while isGuessRight != True:
8     guess = input("Guess a number between 1 and 10: ")
9     if int(guess) == number:
10         print("You guessed {}. That is correct! You win!".format(guess))
11         isGuessRight = True
12     else:
13         print("You guessed {}. Sorry, that isn't it. Try again.".format(guess))
```



Adición de comentarios

Es útil escribir comentarios en el código. Python ignora las líneas de comentarios y comienza con un signo numeral (#). En la mayoría de los teclados, puede crear este símbolo si presiona MAYÚS+3. Agregue sus propios comentarios para que lo ayuden a recordar qué hace el código.

Informe al usuario sobre el script

En esta actividad, iniciará un nuevo script en Python, empezando por informar al usuario qué hace el script.

1. En la barra de menú, elija **File > New From Template > Python File** (Archivo > Nuevo a partir de plantilla > Archivo en Python).
2. Elimine el código de muestra que se proporciona del archivo de plantilla.
3. Elija **File > Save As...** (Archivo > Guardar como...) y guárdelo como *for-loop.py*.
4. Utilice la función `print()` para informar al usuario sobre lo que hace el script:

```
print("Count to 10!")
```

1. Guarde y ejecute el archivo.
2. Confirme que el script se ejecuta de forma correcta y que la salida se muestra según lo previsto.

Escritura del bucle for

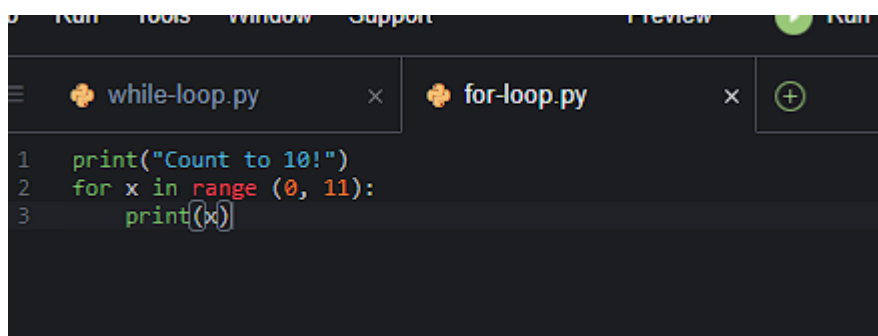
En Python, puede incluir una gran cantidad de funcionalidad en pocas palabras. Esta característica hace que Python sea relativamente fácil de escribir en comparación con otros lenguajes de programación, pero también puede hacer que el código Python sea más difícil de leer. En esta actividad, utilizará la

instrucción `for`, pero también dedicará un poco de tiempo analizándola después de verla en acción.

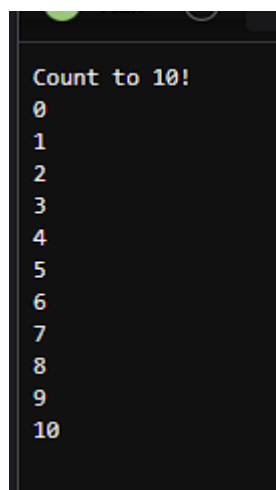
1. Regrese al script en Python. Para contar hasta 10, escriba el siguiente código.

Nota: Python utiliza la sangría para determinar que la instrucción `print` está dentro de la instrucción del bucle `for`:

```
for x in range (0, 11):  
    print(x)
```



1. Guarde y ejecute el archivo.
2. Confirme que el script se ejecuta de forma correcta y que la salida se muestra según lo previsto.



Aquí hay una explicación de lo que sucedió en esas dos líneas. La instrucción `for` utiliza las palabras clave `for ... in` para indicar a la computadora que recorra la lista. La función `range()` genera una lista. La función `range()` toma un

número inicial y un número final, pero el número final no está incluido. Por lo tanto, pasa `11` para que la función deje de contar en 10. La letra `x` actúa como una variable. Cada vez que se ejecuta el bucle, la variable `x` se asigna a la siguiente variable en el bucle y se muestra en la pantalla.

¡Felicitaciones! Ha trabajado con los bucles `while` y `for` en Python.

Finalizar laboratorio

¡Felicitaciones! Ha llegado al final del laboratorio.

1. Elige **End Lab** (Finalizar laboratorio) en la parte superior de esta página y, a continuación, selecciona Yes (Sí) para confirmar que deseas finalizar el laboratorio.

Un panel muestra el mensaje *DELETE has been initiated... You may close this message box now* (Se ha iniciado la ELIMINACIÓN... Ya puedes cerrar este mensaje).

1. Aparece brevemente el mensaje *Ended AWS Lab Successfully* (El laboratorio de AWS finalizó correctamente), que indica que el laboratorio ha finalizado.

Recursos adicionales

Para obtener más información sobre AWS Training and Certification, consulte <https://aws.amazon.com/training/>.

Sus comentarios son bienvenidos y valorados. Si desea compartir alguna sugerencia o corrección, proporcione los detalles en nuestro [Formulario de contacto de AWS Training and Certification](#).

© 2022 Amazon Web Services, Inc. y sus empresas afiliadas. Todos los derechos reservados. Este contenido no puede reproducirse ni redistribuirse, total ni parcialmente, sin el permiso previo por escrito de Amazon Web Services, Inc. Queda prohibida la copia, el préstamo o la venta de carácter comercial.