### Una lista de diccionarios

El diccionario alien\_0 contiene información variada sobre un alienígena, pero no tiene espacio para almacenar información sobre un segundo alienígena, y mucho menos sobre una pantalla llena de alienígenas. ¿Cómo puedes gestionar una flota de alienígenas? Una forma es hacer una lista de alienígenas en la que cada alienígena sea un diccionario de información sobre ese alienígena. Por ejemplo, el siguiente código construye una lista de tres alienígenas:

**aliens.py**

alien\_0 = {'color': 'green', 'points': 5}  
alien\_1 = {'color': 'yellow', 'points': 10}  
alien\_2 = {'color': 'red', 'points': 15}  
  
❶ aliens = [alien\_0, alien\_1, alien\_2]  
  
for alien in aliens:  
 print(alien)

Primero creamos tres diccionarios, cada uno de los cuales representa a un alienígena diferente. Almacenamos cada uno de estos diccionarios en una lista llamada aliens ❶. Por último, recorremos la lista e imprimimos cada alienígena:

{'color': 'green', 'points': 5}  
{'color': 'yellow', 'points': 10}  
{'color': 'red', 'points': 15}

Un ejemplo más realista incluiría más de tres alienígenas con un código que generara automáticamente cada alienígena. En el siguiente ejemplo, utilizamos range() para crear una flota de 30 alienígenas:

# Make an empty list for storing aliens.  
aliens = []  
  
# Make 30 green aliens.  
❶ for alien\_number in range(30):  
❷ new\_alien = {'color': 'green', 'points': 5, 'speed': 'slow'}  
❸ aliens.append(new\_alien)  
  
# Show the first 5 aliens.  
❹ for alien in aliens[:5]:  
 print(alien)  
print("...")  
  
# Show how many aliens have been created.  
print(f"Total number of aliens: {len(aliens)}")

Este ejemplo comienza con una lista vacía para contener todos los alienígenas que se crearán. La función range() ❶ devuelve una serie de números, que sólo indican a Python cuántas veces queremos que se repita el bucle. Cada vez que se ejecuta el bucle, creamos un nuevo alien ❷ y luego añadimos cada nuevo alien a la lista aliens ❸. Utilizamos una rebanada para imprimir los cinco primeros alienígenas ❹, y finalmente, imprimimos la longitud de la lista para demostrar que realmente hemos generado la flota completa de 30 alienígenas:

{'color': 'green', 'points': 5, 'speed': 'slow'}  
{'color': 'green', 'points': 5, 'speed': 'slow'}  
{'color': 'green', 'points': 5, 'speed': 'slow'}  
{'color': 'green', 'points': 5, 'speed': 'slow'}  
{'color': 'green', 'points': 5, 'speed': 'slow'}  
...  
  
Total number of aliens: 30

Todos estos alienígenas tienen las mismas características, pero Python considera a cada uno un objeto independiente, lo que nos permite modificar cada alienígena individualmente.

¿Cómo podrías trabajar con un grupo de alienígenas como éste? Imagina que en un aspecto del juego algunos alienígenas cambian de color y se mueven más rápido a medida que avanza la partida. Cuando llegue el momento de cambiar los colores, podemos utilizar un bucle for y una sentencia if para cambiar el color de los alienígenas. Por ejemplo, para cambiar los tres primeros aliens a amarillo, aliens de velocidad media que valen 10 puntos cada uno, podríamos hacer esto:

# Make an empty list for storing aliens.  
aliens = []  
  
# Make 30 green aliens.  
for alien\_number in range (30):  
 new\_alien = {'color': 'green', 'points': 5, 'speed': 'slow'}  
 aliens.append(new\_alien)  
  
for alien in aliens[:3]:  
 if alien['color'] == 'green':  
 alien['color'] = 'yellow'  
 alien['speed'] = 'medium'  
 alien['points'] = 10  
  
# Show the first 5 aliens.  
for alien in aliens[:5]:  
 print(alien)  
print("...")

Como queremos modificar los tres primeros alienígenas, hacemos un bucle a través de una porción que incluya sólo a los tres primeros alienígenas. Ahora todos los alienígenas son verdes, pero no siempre será así, así que escribimos una sentencia if para asegurarnos de que sólo modificamos los alienígenas verdes. Si el alienígena es verde, cambiamos el color a 'yellow', la velocidad a 'medium', y el valor en puntos a 10, como se muestra en la siguiente salida:

{'color': 'yellow', 'points': 10, 'speed': 'medium'}  
{'color': 'yellow', 'points': 10, 'speed': 'medium'}  
{'color': 'yellow', 'points': 10, 'speed': 'medium'}  
{'color': 'green', 'points': 5, 'speed': 'slow'}  
{'color': 'green', 'points': 5, 'speed': 'slow'}  
...

Podrías ampliar este bucle añadiendo un bloque elif que convierta a los alienígenas amarillos en rojos y rápidos, con un valor de 15 puntos cada uno. Sin mostrar de nuevo todo el programa, ese bucle tendría el siguiente aspecto:

for alien in aliens[0:3]:  
 if alien['color'] == 'green':  
 alien['color'] = 'yellow'  
 alien['speed'] = 'medium'  
 alien['points'] = 10  
 elif alien['color'] == 'yellow':  
 alien['color'] = 'red'  
 alien['speed'] = 'fast'  
 alien['points'] = 15

Es habitual almacenar varios diccionarios en una lista cuando cada diccionario contiene muchos tipos de información sobre un objeto. Por ejemplo, podrías crear un diccionario para cada usuario de un sitio web, como hicimos en *user.py*  en la página 99, y almacenar los diccionarios individuales en una lista llamada users. Todos los diccionarios de la lista deben tener una estructura idéntica, para que puedas recorrer la lista y trabajar con cada objeto diccionario de la misma manera.

[anterior](c06_22.html)[Subtema 23 de 28: (Ver todo)](c06.html)[siguiente](c06_24.html)