Desconocido

### Diferencia entre cadenas y números

La entrada del*usuario* es lo que una persona introduce en el teclado, ya sea un carácter, una flecha pulsada o la tecla ENTER, o cualquier otra cosa. En Python, la entrada del usuario es una cadena, lo que significa que cuando escribes el número 10 en el teclado, Python guarda el número 10 como una cadena, no como un número.

Compara el número 10 y la cadena '10' . Aunque para nosotros la única diferencia entre ambos es que uno está rodeado de comillas, para un ordenador son completamente distintos.

Por ejemplo, comparemos el valor de la variable age con un número en una sentencia if , así

>>> if age == 10:  
 print("What's the best way to speak to a monster?")  
 print("From as far away as possible!")

Si primero establecemos la variable age con el número 10 :

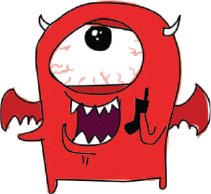
>>> age = 10  
>>> if age == 10:  
 print("What's the best way to speak to a monster?")  
 print("From as far away as possible!")  
   
What's the best way to speak to a monster?  
From as far away as possible!

Como puedes ver, se ejecuta la sentencia print .

A continuación, establecemos age a la cadena ‘10’ (con comillas), así:

>>> age = '10'  
>>> if age == 10:  
 print("What's the best way to speak to a monster?")  
 print("From as far away as possible!")  
  
>>>

Aquí, la sentencia print no se ejecuta porque Python no ve la cadena como un número.



Afortunadamente, Python tiene funciones que pueden convertir cadenas en números y números en cadenas. Por ejemplo, puedes convertir la cadena ‘10’ en un número con la función int :

>>> age = '10'  
>>> converted\_age = int(age)

La variable converted\_age contiene ahora el número 10 (en lugar de una cadena).

Para convertir un número en una cadena, utiliza str del siguiente modo:

>>> age = 10  
>>> converted\_age = str(age)

Ahora converted\_age contiene la cadena ‘10’ en lugar del número 10.

¿Recuerdas la sentencia if age == 10 que no imprimía nada cuando la variable se convertía en una cadena ( age = ’10’ )? Si convertimos primero la variable, obtendremos un resultado totalmente distinto:

>>> age = '10'  
>>> converted\_age = int(age)  
>>> if converted\_age == 10:  
 print("What's the best way to speak to a monster?")  
 print("From as far away as possible!")  
  
What's the best way to speak to a monster?  
From as far away as possible!

Pero debes saber esto: si intentas convertir un número con punto decimal (también llamados *números de coma flotante* , porque el punto puede "moverse" en un número), obtendrás un error porque la función int espera un *número entero* (un número sin punto decimal):

>>> age = '10.5'  
>>> converted\_age = int(age)  
Traceback (most recent call last):  
 File "<pyshell#35>", line 1, in <module>  
 converted\_age = int(age)  
ValueError: invalid literal for int() with base 10: '10.5'

Python envía un ValueError para indicarte que el valor que has intentado utilizar no es apropiado. Para solucionarlo, utiliza float en lugar de int , ya que la función float puede manejar números que no sean enteros:

>>> age = '10.5'  
>>> converted\_age = float(age)  
>>> print(converted\_age)  
10.5

También obtendrás un ValueError si intentas convertir una cadena que no contiene un número en dígitos:

>>> age = 'ten'  
>>> converted\_age = int(age)  
Traceback (most recent call last):  
 File "<pyshell#1>", line 1, in <module>  
 converted\_age = int(age)}  
ValueError: invalid literal for int() with base 10: 'ten'

Como hemos utilizado la palabra ten en lugar del número 10 , Python se echa las manos a la cabeza.

[anterior](ch05_8.html)[Subtema 9 de 11: (Ver todo)](ch05.html)[siguiente](ch05_10.html)