Desconocido

## 2 Cálculos y Variables



Ahora que has instalado Python y sabes cómo iniciar la Shell de Python, estás preparado para hacer algo con él. Empezaremos con algunos cálculos sencillos y luego aprenderemos a utilizar variables. Las variables son una forma de almacenar cosas en un programa informático, y pueden ayudarte a escribir programas útiles.

### Calcular con Python

Normalmente, cuando te piden que halles el producto de dos números como 8 × 3,57, utilizarías una calculadora o un lápiz y papel. ¿Qué te parece utilizar la Shell de Python para realizar tus cálculos? Vamos a probarlo.

Inicia el intérprete de comandos Python haciendo doble clic en el icono IDLE de tu escritorio, o si utilizas Ubuntu, haz clic en el icono IDLE del menú Aplicaciones. En la pantalla, introduce este cálculo:

>>> 8 \* 3.57  
28.56

Al introducir un cálculo de multiplicación en Python, se utiliza el símbolo del asterisco ( \* ) en lugar del signo de multiplicación (×).

¿Qué tal si probamos con una ecuación un poco más útil?

Supón que estás cavando en el patio de tu casa y descubres una bolsa con 20 monedas de oro. Al día siguiente, bajas a hurtadillas al sótano y metes las monedas dentro del invento replicante a vapor de tu abuelo (por suerte, *sólo* caben las 20 monedas dentro). Oyes un silbido y un estallido y, unas horas después, salen disparadas otras 10 monedas relucientes.

¿Cuántas monedas tendrías en tu cofre del tesoro si hicieras esto todos los días durante un año? Sobre el papel, las ecuaciones podrían tener este aspecto

10 × 365 = 3650  
3650 + 20 = 3670

Claro que es bastante fácil hacer estos cálculos con una calculadora o en papel, pero también podemos hacerlos con la Shell de Python. Primero, multiplicamos 10 monedas por 365 días en un año para obtener 3650. Después, sumamos las 20 monedas originales para obtener 3670.

>>> 10 \* 365  
3650  
>>> 3650 + 20  
3670

Ahora, ¿qué pasaría si un cuervo viera el oro brillante que hay en tu habitación, y cada semana entrara volando y consiguiera robar tres monedas? ¿Cuántas monedas te quedarían al final del año? Así es como se ve este cálculo en la Shell de Python:

>>> 3 \* 52  
156  
>>> 3670 - 156  
3514

Primero, multiplicamos 3 monedas por 52 semanas del año. El resultado es 156. Restamos ese número de nuestro número total de monedas (3670), lo que nos dice que nos quedarían 3514 monedas al final del año.

Aunque podrías hacer fácilmente este cálculo con una calculadora, trabajarlo en la Shell de Python es beneficioso para aprender a escribir programas informáticos sencillos. En este libro, aprenderás a ampliar estas ideas para escribir programas aún más útiles.

### Operadores de Python

Puedes hacer sumas, restas, multiplicaciones y divisiones en el Shell de Python, entre otras operaciones matemáticas que exploraremos más adelante. Los símbolos básicos que utiliza Python para realizar operaciones matemáticas, llamados *operadores* , se enumeran en [la Tabla 2-1 .](ch02.xhtml#ch02tab01)

Tabla**2-1:** Operadores básicos de Python

| **Símbolo** | **Operación** |
| --- | --- |
| + | Suma |
| - | Resta |
| \* | Multiplicación |
| / | División |

La *barra inclinada* ( / ) se utiliza para dividir porque es similar a la línea de división que utilizarías al escribir una fracción. Por ejemplo, si tuvieras 100 piratas y 20 barriles grandes y quisieras calcular cuántos piratas podrías esconder en cada barril, podrías dividir 100 piratas entre 20 barriles (100 ÷ 20) introduciendo 100 / 20 en la Shell de Python. Sólo tienes que recordar que la barra diagonal es aquella cuya parte superior cae hacia la derecha.

### El orden de las operaciones

En los lenguajes de programación utilizamos paréntesis para controlar el orden de las operaciones. Una *operación* es cualquier cosa que utilice un operador. La multiplicación y la división tienen un orden superior a la suma y la resta, por lo que se realizan primero. En otras palabras, si introduces una ecuación en Python, la multiplicación o la división se realizan antes que la suma o la resta.

Por ejemplo, en la siguiente ecuación, los números 30 y 20 se multiplican primero, y el número 5 se añade a su producto:

>>> 5 + 30 \* 20  
605

Esta ecuación es otra forma de decir: "Multiplica 30 por 20, y luego añade 5 al resultado". El resultado es 605. Podemos cambiar el orden de las operaciones añadiendo paréntesis alrededor de los dos primeros números, así

>>> (5 + 30) \* 20  
700

El resultado de esta ecuación es 700 (no 605) porque los paréntesis le dicen a Python que haga primero la operación dentro de los paréntesis, y luego haga la operación fuera de los paréntesis. Este ejemplo está diciendo: "Suma 5 a 30, y luego multiplica el resultado por 20".

Los paréntesis se pueden *anidar* , lo que significa que puede haber paréntesis dentro de paréntesis, como éste:

>>> ((5 + 30) \* 20) / 10  
70.0

En este caso, Python evalúa primero los paréntesis interiores, luego los exteriores y, por último, el operador de división. En otras palabras, esta ecuación está diciendo: "Suma 5 a 30, luego multiplica el resultado por 20 y divide ese resultado por 10". Esto es lo que ocurre

1. Sumando 5 a 30 se obtiene 35.
2. Multiplicando 35 por 20 se obtiene 700.
3. Dividiendo 700 por 10 se obtiene la respuesta final de 70.

Si no hubiéramos utilizado paréntesis, el resultado sería ligeramente distinto:

>>> 5 + 30 \* 20 / 10  
65.0

En este caso, 30 se multiplica primero por 20 (dando 600), y luego 600 se divide por 10 (dando 60). Por último, se añade 5 para obtener el resultado de 65.

**NOTA**

*Recuerda que la multiplicación y la división siempre van antes que la suma y la resta, a menos que se utilicen paréntesis para controlar el orden de las operaciones.*

### Las variables son como etiquetas

La palabra *variable* en programación describe un lugar para almacenar información, como números, texto, listas de números y texto, etc. Una variable es esencialmente una etiqueta para algo.

Por ejemplo, para crear una variable llamada fred , utilizamos un signo igual ( = ) y luego le decimos a Python de qué información debe ser etiqueta la variable. Aquí, creamos la variable fred y le decimos a Python que es una etiqueta para el número 100 (ten en cuenta que esto no significa que otra variable no pueda tener el mismo valor):

>>> fred = 100

Para encontrar el valor de una variable, introduce print en la Shell de Python, seguido del nombre de la variable entre paréntesis, así:

>>> print(fred)  
100

También podemos decirle a Python que cambie la variable fred para que etiquete otra cosa. Por ejemplo, así es como se cambia fred por el número 200:

>>> fred = 200  
>>> print(fred)  
200

En la primera línea, decimos que fred etiqueta a un número 200 . En la segunda línea, imprimimos el valor de fred , sólo para confirmar el cambio. Python imprime el resultado en la última línea.

También podemos utilizar más de una etiqueta (o variable) para el mismo elemento:

>>> fred = 200  
>>> john = fred  
>>> print(john)  
200

En este ejemplo, le estamos diciendo a Python que queremos que el nombre (o variable) john etiquete lo mismo que fred utilizando el signo igual entre john y fred .

Por supuesto, fred probablemente no sea un nombre muy útil para una variable, porque lo más probable es que no nos diga nada sobre para qué se utiliza la variable. Llamemos a nuestra variable number\_of\_coins en lugar de fred , así:

>>> number\_of\_coins = 200  
>>> print(number\_of\_coins)  
200

Esto deja claro que estamos hablando de 200 monedas. Los nombres de las variables pueden estar formados por letras, números y el carácter de subrayado ( \_ ), pero no pueden empezar por un número. Puedes utilizar desde letras sueltas (como a ) hasta frases largas para los nombres de las variables. (Una variable no puede contener un espacio, así que utiliza un guión bajo para separar las palabras.) A veces, si estás haciendo algo rápido, lo mejor es un nombre de variable corto. El nombre que elijas debe depender de lo significativo que necesites que sea el nombre de la variable.

Ahora que ya sabes cómo crear variables, veamos cómo utilizarlas.

### Utilización de variables

¿Recuerdas nuestra ecuación para calcular cuántas monedas tendrías al final del año si pudieras crear mágicamente nuevas monedas con el misterioso invento de tu abuelo en el sótano? Teníamos esta ecuación:

>>> 20 + 10 \* 365  
3670  
>>> 3 \* 52  
156  
>>> 3670 - 156  
3514

Podemos convertirla en una sola línea de código:

>>> 20 + 10 \* 365 - 3 \* 52  
3514

No es muy fácil de leer, pero ¿y si convertimos los números en variables? Prueba a introducir lo siguiente:

>>> found\_coins = 20  
>>> magic\_coins = 10  
>>> stolen\_coins = 3

Estas entradas crean las variables found\_coins , magic\_coins , y stolen\_coins .

Ahora, podemos volver a introducir la ecuación así:

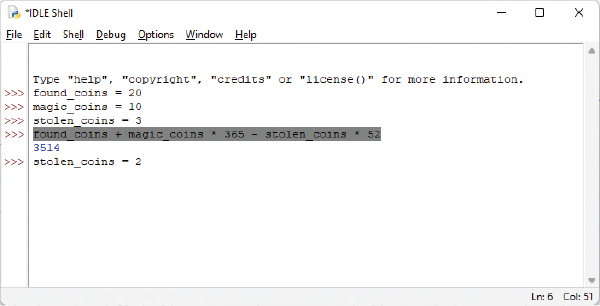
>>> found\_coins + magic\_coins \* 365 - stolen\_coins \* 52  
3514

Puedes ver que esto nos da la misma respuesta. Así que a quién le importa, ¿verdad? Ah, pero aquí está la magia de las variables. ¿Qué pasa si pones un espantapájaros en tu ventana y el cuervo sólo roba dos monedas en lugar de tres? Cuando utilizamos una variable, podemos simplemente cambiar la variable para que contenga ese nuevo número, y cambiará en todas las partes de la ecuación en las que se utilice. Podemos cambiar la variable stolen\_coins por 2 introduciendo esto:

>>> stolen\_coins = 2

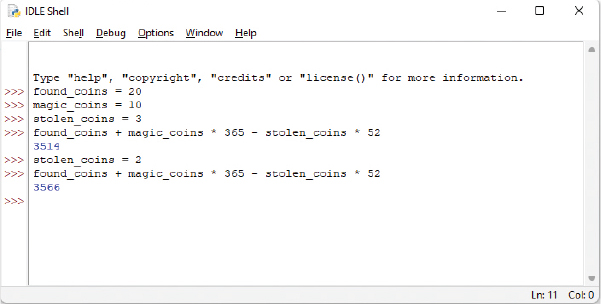
A continuación, podemos copiar y pegar la ecuación para calcular de nuevo la respuesta, de este modo

1. Selecciona el texto a copiar haciendo clic con el ratón y arrastrando desde el principio hasta el final de la línea, como se muestra en la [Figura 2-1 .](ch02.xhtml#ch02fig01)



*Figura 2-1: Seleccionar en el Shell de Python*

1. Mantén pulsada la tecla CTRL (o, si utilizas un Mac, la tecla COMANDO ⌘) y pulsa C para copiar el texto seleccionado. (A partir de ahora lo verás como CTRL-C).
2. Haz clic en la última línea de consulta (después de stolen\_coins = 2 ).
3. Mantén pulsada la tecla CTRL (o COMANDO) y pulsa V para pegar el texto seleccionado. (A partir de ahora lo verás como CTRL-V).
4. Pulsa ENTER para ver el nuevo resultado.



*Figura 2-2: Pegar en la Shell de Python*

Es mucho más fácil que volver a escribir toda la ecuación.

Puedes probar a cambiar las demás variables, y luego copiar (CTRL-C) y pegar (CTRL-V) el cálculo para ver el efecto de tus cambios. Por ejemplo, si golpeas los lados del invento de tu abuelo en el momento adecuado, y éste escupe  tres monedas de más cada vez, verás que acabas teniendo 4661 monedas al final del año:

>>> magic\_coins = 13  
>>> found\_coins + magic\_coins \* 365 stolen\_coins \* 52  
4661

Por supuesto, utilizar variables para una ecuación sencilla como ésta sigue siendo sólo *ligeramente* útil. Aún no hemos llegado a lo *realmente* útil. Por ahora, sólo recuerda que las variables son una forma de etiquetar las cosas para poder utilizarlas más adelante.

### Lo que has aprendido

En este capítulo has aprendido a hacer ecuaciones sencillas utilizando operadores de Python y a utilizar paréntesis para controlar el orden de las operaciones (el orden en que Python evalúa las partes de las ecuaciones). Luego creamos variables para etiquetar valores y utilizamos esas variables en nuestros cálculos.