Desconocido

## C Solución de problemas



En este apéndice encontrarás información sobre cómo solucionar algunos problemas menos comunes con Python. Si ejecutas versiones antiguas de algunos sistemas operativos, es posible que experimentes estos problemas.

### Errores "TK" al importar Turtle en Ubuntu

Si utilizas una versión antigua de Ubuntu Linux y obtienes errores al importar tortuga, puede que necesites instalar un programa llamado  *tkinter .* Para ello, abre el Centro de Software de Ubuntu e introduce **python-tk** en el cuadro de búsqueda. En la ventana debería aparecer "Tkinter-Escribir aplicaciones Tk con Python". Haz clic en **Instalar** para instalar este paquete. Esto no debería ser necesario si estás ejecutando una versión más reciente de Ubuntu; si es posible, deberías pedir al propietario de tu ordenador que lo actualice por ti.

### Error de atributo con Turtle

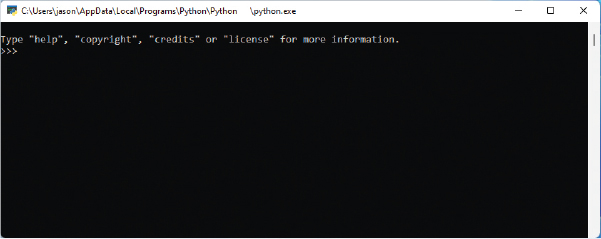
Algunos programadores noveles experimentan extraños errores de *atributo* al intentar utilizar tortuga:

>>> import turtle  
>>> t = turtle.Turtle()  
Traceback (most recent call last):  
 File "<stdin>", line 1, in <module>  
AttributeError: module 'turtle' has no attribute 'Turtle'

La causa más típica de este error es que hayas creado un archivo llamado tortuga *.py* en tu carpeta de inicio. En este caso, cuando introduces import turtle , obtienes el archivo que has creado y no el módulo turtle de Python. Si eliminas o cambias el nombre de ese archivo, el módulo correcto debería importarse correctamente.

### Problemas al ejecutar Tortuga

Si tienes problemas al utilizar el módulo turtle , y la propia ventana de la tortuga no parece funcionar, prueba a utilizar la consola de Python en lugar del intérprete de comandos de Python, como se indica a continuación:

* En Windows, introduce **Python** en el cuadro de búsqueda y haz clic en **Python 3.1 *x en*** la lista de Aplicaciones. En su lugar, puedes utilizar el símbolo del sistema de Windows (haz clic en el icono de Windows y escribe **cmd** en el cuadro de búsqueda). Cuando se abra, tendrás que introducir la ruta a un programa llamado *python.exe .* Si has instalado Python 3.10, la ruta podría ser algo así: *AppData\Local\Programas\Python \Python310\python.exe .* Sin embargo, depende mucho de la versión de Python que hayas instalado, por lo que este método debería ser probablemente el último recurso (puedes ver el resultado de ejecutarlo en [la Figura C-1](app03.xhtml#app03fig01) ).
* 
* Figura*C-1: Ejecutar la consola de Python desde el símbolo del sistema de Windows*
* En macOS, haz clic en el icono de búsqueda de Spotlight en la esquina superior derecha de la pantalla (debería parecer una lupa) e introduce **Terminal** en el cuadro de entrada. A continuación, introduce python3 cuando se abra el terminal.
* En Ubuntu Linux, abre el terminal desde el menú **Mostrar aplicaciones** e introduce python3 .10 (ten en cuenta que tu número de versión puede ser diferente).
* En Raspberry Pi, haz clic en el icono Terminal de la barra de menús de la parte superior, o haz clic en **Terminal** en el menú **Accesorios** e introduce /usr/local/opt/python-3.10.0 (esto sólo funcionará si has seguido las instrucciones de instalación de Raspberry Pi del [Capítulo 1](ch01.xhtml#ch01) ; ten en cuenta que tu número de versión puede ser diferente).

La consola de Python es similar a la Shell de Python (IDLE), pero no tiene resaltado de sintaxis (texto coloreado), opciones de guardado fácil y otras características beneficiosas. Sin embargo, si tienes problemas para ejecutar tortuga en el Shell de Python, utilizar la consola de Python puede ayudarte.

### La clase no tiene argumentos

Un error común con el que tropiezan algunos lectores es un TypeError , normalmente visto por primera vez en el [Capítulo 11 .](ch11.xhtml#ch11) Es posible que veas un error similar al siguiente:

b = Ball(canvas, 'red')  
Traceback (most recent call last):  
 File "/usr/lib/python3.10/idlelib/run.py", line 573, in runcode  
 exec(code, self.locals)  
 File "<pyshell#4>", line 1, in <module>  
TypeError: Ball() takes no arguments

El motivo suele ser la falta de guiones bajos. La clase Ball se define primero así:

class Ball:  
 def \_\_init\_\_(self, canvas, color):  
 self.canvas = canvas  
 self.id = canvas.create\_oval(10, 10, 25, 25, fill=color)  
 self.canvas.move(self.id, 245, 100)

Sin embargo, si escribes mal la función \_\_init\_\_ con un solo guión bajo a cada lado ( \_init\_ ), Python ya no la reconocerá como función de inicialización. Esta es la razón por la que llamar a Ball(...) con cualquier argumento produce un error: Python piensa que no hay ninguna función de inicialización a la que llamar (de hecho, crea una función de inicialización por defecto para ti que no tiene parámetros).