Pruébelo usted mismo

9-1. Restaurante: haz una clase llamada Restaurante. El método \_\_init \_\_ () para Restaurant debe almacenar dos atributos: un restaurant\_name y un cuisine\_type. Cree un método llamado describe\_restaurant () que imprima estas dos piezas de información, y un método llamado open\_restaurant () que imprima un mensaje que indique que el restaurante está abierto. Cree una instancia llamada restaurant de su clase. Imprima los dos atributos individualmente y luego llame a ambos métodos .

9-2. Tres restaurantes: comience con su clase desde el ejercicio 9-1. Cree tres instancias diferentes de la clase y llame a describe\_restaurant () para cada instancia .

9-3. Usuarios: crea una clase llamada Usuario. Cree dos atributos llamados first\_name y last\_name, y luego cree varios otros atributos que normalmente se almacenan en un perfil de usuario. Cree un método llamado describe\_user () que imprima un resumen de la información del usuario. Cree otro método llamado greet\_user () que imprima un saludo personalizado para el usuario. Cree varias instancias que representen a diferentes usuarios y llame a ambos métodos para cada usuario.

9-4. Número servido: comience con su programa desde el Ejercicio 9-1 (página 166). Agregue un atributo llamado number\_served con un valor predeterminado de 0. Cree una instancia llamada restaurante de esta clase. Imprima el número de clientes que el restaurante ha atendido y luego cambie este valor e imprímalo de nuevo. Agregue un método llamado set\_number\_served () que le permite establecer el número de clientes que han sido atendidos. Llame a este método con un nuevo número e imprima el valor nuevamente. Agregue un método llamado increment\_number\_served () que le permite incrementar la cantidad de clientes que han sido atendidos. Llame a este método con cualquier número que desee que pueda representar a cuántos clientes se atendieron en, digamos, un día hábil .

9-5. Intentos de inicio de sesión: agregue un atributo llamado login\_attempts a su clase de usuario del ejercicio 9-3 (página 166). Escriba un método llamado increment\_login\_attempts () que incremente el valor de login\_attempts en 1. Escriba otro método llamado reset\_login\_attempts () que restablezca el valor de login\_attempts a 0. Cree una instancia de la clase User y llame a increment\_login\_attempts () varias veces. Imprima el valor de login\_attempts para asegurarse de que se incrementó correctamente y luego llame a reset\_login\_attempts (). Imprima login\_attempts nuevamente para asegurarse de que se restableció a 0.

9-6. Puesto de helados: Un puesto de helados es un tipo específico de restaurante. Escriba una clase llamada IceCreamStand que herede de la clase Restaurante que escribió en el ejercicio 9-1 (página 166) o el ejercicio 9-4 (página 171). Cualquiera de las versiones de la clase funcionará; solo elige el que más te guste. Agregue un atributo llamado sabores que almacena una lista de sabores de helado. Escribe un método que muestre estos sabores. Cree una instancia de IceCreamStand y llame a este método .

9 -7. Administrador: un administrador es un tipo especial de usuario. Escriba una clase llamada Admin que herede de la clase de usuario que escribió en el ejercicio 9-3 (página 166) o el ejercicio 9-5 (página 171). Agregue un atributo, privilegios, que almacene una lista de cadenas como "puede agregar publicación", "puede eliminar publicación", "puede prohibir usuario", etc. Escriba un método llamado show\_privileges () que enumere el conjunto de privilegios del administrador. Cree una instancia de Admin y llame a su método .

9-8. Privilegios: escriba una clase de Privilegios separada. La clase debe tener un atributo, privilegios, que almacene una lista de cadenas como se describe en el ejercicio 9-7. Mueva el método show\_privileges () a esta clase. Cree una instancia de Privileges como atributo en la clase Admin. Cree una nueva instancia de Admin y use su método para mostrar sus privilegios .

9-9. Actualización de la batería: use la versión final de electric\_car.py de esta sección. Agregue un método a la clase Battery llamado upgrade\_battery (). Este método debe verificar el tamaño de la batería y establecer la capacidad en 85 si aún no lo está. Haga un automóvil eléctrico con un tamaño de batería predeterminado, llame a get\_range () una vez y luego llame a get\_range () por segunda vez después de actualizar la batería. Debería ver un aumento en la autonomía del automóvil.

9-10. Restaurante importado: con su última clase de restaurante, guárdelo en un módulo. Haga un archivo separado que importe Restaurante. Cree una instancia de Restaurant y llame a uno de los métodos de Restaurant para mostrar que la declaración de importación funciona correctamente.0

9-11. Administrador importado: comience con su trabajo desde el ejercicio 9-8 (página 178). Almacene las clases Usuario, Privilegios y Administrador en un módulo. Cree un archivo separado, cree una instancia de administrador y llame a show\_privileges () para mostrar que todo funciona correctamente.

9-12. Múltiples módulos: almacene la clase de usuario en un módulo y almacene las clases de privilegios y de administrador en un módulo separado. En un archivo separado, cree una instancia de administrador y llame a show\_privileges () para mostrar que todo sigue funcionando correctamente.0

9-13. OrderedDict Rewrite: comience con el ejercicio 6-4 (página 108), donde utilizó un diccionario estándar para representar un glosario. Vuelva a escribir el programa usando la clase OrderedDict y asegúrese de que el orden de la salida coincida con el orden en el que se agregaron los pares clave-valor al diccionario.

9-14. Dados: el módulo aleatorio contiene funciones que generan números aleatorios de diversas formas. La función randint () devuelve un número entero en el rango que proporcionas. El siguiente código devuelve un número entre 1 y 6: from random import randintx = randint (1, 6) Hacer una clase Die con un atributo llamado lados, que tiene un valor predeterminado de 6. Escribe un método llamado roll\_die () que imprima un número aleatorio entre 1 y el número de lados que tiene el dado. Haz un dado de 6 caras y tíralo 10 veces. Haz un dado de 10 caras y un dado de 20 caras. Tira cada dado 10 veces .

9-15. Módulo de la semana de Python: Un recurso excelente para explorar la biblioteca estándar de Python es un sitio llamado Módulo de la semana de Python. Vaya a http://pymotw.com/ y mire la tabla de contenido. Encuentre un módulo que le parezca interesante y lea sobre él, o explore la documentación de las colecciones y módulos aleatorios

6-4. Glosario 2: ahora que sabe cómo recorrer un diccionario, limpie

subir el código del ejercicio 6-3 (página 102) reemplazando su serie de impresión

declaraciones con un bucle que recorre las claves y los valores del diccionario.

Cuando esté seguro de que su ciclo funciona, agregue cinco términos Python más a su

glosario. Cuando vuelva a ejecutar su programa, estas nuevas palabras y significados

debe incluirse automáticamente en la salida

6-3. Glosario: se puede utilizar un diccionario de Python para modelar un diccionario real.

Sin embargo, para evitar confusiones, llamémoslo glosario.

• Piense en cinco palabras de programación que haya aprendido en el

capítulos. Utilice estas palabras como claves en su glosario y almacene sus

significados como valores.

• Imprima cada palabra y su significado como una salida con un formato prolijo. Podrías

imprima la palabra seguida de dos puntos y luego su significado, o imprima la palabra

en una línea y luego imprima su significado con sangría en una segunda línea. Utilizar el

carácter de nueva línea (\ n) para insertar una línea en blanco entre el significado de cada palabra

emparejar en su salida