Ejercicio 20.2

Usando la lista pedidos_sandwiches del ejercicio 20.1, asegurate de que el sándwich de pastrón aparezca al menos tres veces en la lista. Agregá código al principio del programa para mostrar un mensaje diciendo que la sandwichería se quedó sin pastrón. Luego, usá un bucle while para eliminar todas las apariciones de pastrón de pedidos_sandwiches. Asegurate de que ningún sándwich de pastrón termine en sandwiches_terminados.

Ejercicio 20.3

Escribí un programa que le pregunte a los usuarios sobre su destino de vacaciones soñado. Mostrá un mensaje como: Si pudieras visitar un lugar en el mundo, ¿a dónde irías? Luego, incluí un bloque de código que imprima los resultados de la encuesta.

Ejercicio 21.1

Escribí una función llamada mostrar_mensaje() que imprima una oración diciendo qué estás aprendiendo en este capítulo. Llamá a la función y asegurate de que el mensaje se muestre correctamente.

Ejercicio 21.2

Escribí una función llamada libro_favorito() que reciba un parámetro, titulo. La función debería imprimir un mensaje como: “Uno de mis libros favoritos es Alicia en el país de las maravillas”. Llamá a la función incluyendo un título de libro como argumento.

Ejercicio 22.1

Escribí una función llamada hacer_camiseta() que reciba un tamaño y el texto de un mensaje que se debe imprimir en la camiseta. La función debería imprimir una oración resumiendo el tamaño de la camiseta y el mensaje impreso en ella.

Llamá a la función una vez usando argumentos posicionales para hacer una camiseta. Llamá a la función una segunda vez usando argumentos con palabra clave.

Ejercicio 22.2

Modificá la función hacer_camiseta() para que las camisetas sean grandes por defecto, con un mensaje que diga Me encanta Python. Hacé una camiseta grande y una mediana con el mensaje predeterminado, y una camiseta de cualquier tamaño con un mensaje diferente.

Ejercicio 22.3

Escribí una función llamada describir_ciudad() que reciba el nombre de una ciudad y su país. La función debería imprimir una oración simple, como Reikiavik está en Islandia. Dále un valor

predeterminado al parámetro para el país. Llamá a la función para tres ciudades diferentes, siendo al menos una de ellas de un país distinto al predeterminado.

Ejercicio 23.1

Escribí una función llamada ciudad_pais() que reciba el nombre de una ciudad y su país. La función debería devolver una cadena de texto formateada de esta manera: Santiago, Chile.

Llamá a la función con al menos tres pares ciudad-país y muestra los valores que devuelve.

Ejercicio 23.2

Escribí una función llamada hacer_album() que construya un diccionario describiendo un álbum de música. La función debería recibir el nombre de un artista y el título de un álbum, y debería devolver un diccionario que contenga esta información. Usá la función para crear tres diccionarios representando diferentes álbumes. Imprimí cada valor de retorno para mostrar que los diccionarios están guardando correctamente la información del álbum.

Usá None para agregar un parámetro opcional a hacer_album() que permita almacenar la cantidad de canciones de un álbum. Si la línea de llamada incluye un valor para la cantidad de canciones, agregalo al diccionario del álbum. Hacé al menos una llamada nueva a la función que incluya la cantidad de canciones de un álbum.

Ejercicio 23.3

Comenzá con tu programa del ejercicio 23.2. Escribí un bucle while que permita a los usuarios ingresar el artista y el título de un álbum. Una vez que tengas esa información, llamá a hacer_album() con los datos proporcionados por el usuario e imprimí el diccionario que se crea. Asegurate de incluir un valor para salir del bucle while.

Ejercicio 24.1

Creá una lista que contenga una serie de mensajes de texto cortos. Pasá la lista a una función llamada mostrar_mensajes(), que imprima cada mensaje de texto.

Ejercicio 24.2

Comenzá con una copia de tu programa del ejercicio 24.1. Escribí una función llamada enviar_mensajes() que imprima cada mensaje de texto y mueva cada mensaje a una nueva lista llamada mensajes_enviados mientras se imprime. Después de llamar a la función, imprimí ambas listas para asegurarte de que los mensajes se movieron correctamente.

Ejercicio 24.3

Comenzá con tu trabajo del ejercicio 24.2. Llamá a la función enviar_mensajes() con una copia de la lista de mensajes. Después de llamar a la función, imprimí ambas listas para mostrar que la lista original ha retenido sus mensajes.

Ejercicio 25.1

Escribí una función que acepte una lista de ingredientes que una persona quiere en un sándwich. La función debería tener un parámetro que pueda recopilar tantos ingredientes como se pasen en la llamada a la función y debería imprimir un resumen del sándwich que se está pidiendo. Llamá a la función tres veces, usando una cantidad diferente de argumentos en cada llamada.

Ejercicio 25.2

Con:
La definición de construir_perfil() espera un nombre y un apellido, y luego permite que el usuario pase tantos pares clave-valor como quiera. Los dos asteriscos antes del parámetro **info_usuario hacen que Python cree un diccionario llamado info_usuario, que contiene todos los pares clave-valor adicionales que recibe la función. Dentro de la función, podés acceder a los pares clave-valor en info_usuario de la misma manera en que lo harías con cualquier diccionario.

Creá un perfil tuyo llamando a construir_perfil(), usando tu nombre y apellido junto con tres pares clave-valor adicionales que te describan.

Ejercicio 25.3

Escribí una función que almacene información sobre un auto en un diccionario. La función siempre debería recibir el fabricante y el modelo del auto. Luego, debería aceptar una cantidad arbitraria de keyword arguments. Llamá a la función con la información requerida y dos pares clave-valor adicionales, como el color o una característica opcional. Tu función debería funcionar para una llamada como esta:

```
auto = hacer_auto('toyota', 'corolla', color='gris',
airbags=True)
```

Imprimí el diccionario que devuelve la función para asegurarte de que toda la información se almacenó correctamente.

Ejercicio 26.1

Colocá estas funciones en un archivo separado llamado printing_functions.py:

```
def imprimir_modelos(diseños_no_imprimidos,
modelos_completados):
    """
```

```
    Simula imprimir cada diseño, hasta que no queden más.
    Mueve cada diseño a modelos_completados después de
imprimirlo.
    """
    while diseños_no_imprimidos:
        diseño_actual = diseños_no_imprimidos.pop()
        print(f"Imprimiendo modelo: {diseño_actual}")
        modelos_completados.append(diseño_actual)


def mostrar_modelos_completados(modelos_completados):
    """Mostrar todos los modelos que fueron impresos."""
    print("\nLos siguientes modelos han sido impresos:")
    for modelo_completado in modelos_completados:
        print(modelo_completado)
```

Escribí una declaración de importación al comienzo del programa principal y modificá el archivo para usar las funciones importadas.

Ejercicio 26.2

Usando un programa que hayas escrito y que tenga una función, guardá esa función en un archivo separado. Importá la función en tu archivo principal y llamala usando cada uno de estos enfoques:

import nombre_módulo

from nombre_módulo import nombre_función

from nombre_módulo import nombre_función as nf

import nombre_módulo as nm

from nombre_módulo import *

Ejercicio 27.1

Hacé una clase llamada Restaurante. El método __init__() de Restaurante tiene que guardar dos atributos: nombre_restaurante y tipo_cocina.

Creá un método llamado describir_restaurante() que imprima esa información y otro método llamado abrir_restaurante() que imprima un mensaje indicando que el restaurante está abierto.

Creá una instancia llamada restaurante a partir de tu clase. Imprimí los dos atributos por separado y después llamá a ambos métodos.

Ejercicio 27.2

Usando la clase del ejercicio 27.1, creá tres instancias distintas y llamá a describir_restaurante() para cada una.

Ejercicio 27.3

Hacé una clase llamada Usuario. Creá dos atributos llamados nombre y apellido, y agregá otros atributos que normalmente se guardarían en un perfil de usuario.

Creá un método llamado describir_usuario() que imprima un resumen con la información del usuario.

Hacé otro método llamado saludar_usuario() que imprima un saludo personalizado.

Creá varias instancias que representen diferentes usuarios y llamá a ambos métodos para cada uno.

Ejercicio 28.1

Usá tu programa del ejercicio 27.1. Agregá un atributo llamado clientes_atendidos con un valor por defecto de 0.

Creá una instancia llamada restaurante a partir de esta clase.

Imprimí la cantidad de clientes que el restaurante ha atendido.

Modificá este valor y volvé a imprimirlo.

Agregá un método llamado establecer_clientes_atendidos() que te permita definir la cantidad de clientes atendidos. Llamá a este método con un nuevo número y volvé a imprimir el valor.

Agregá otro método llamado incrementar_clientes_atendidos() que permita sumar una cantidad específica al número de clientes atendidos. Llamá a este método con cualquier número que represente, por ejemplo, la cantidad de clientes atendidos en un día.

Ejercicio 28.2

Agregá un atributo llamado intentos_login a la clase Usuario del ejercicio 27.3.

Escribí un método llamado incrementar_intentos_login() que aumente el valor de intentos_login en 1.

Escribí otro método llamado reiniciar_intentos_login() que restablezca el valor de intentos_login a 0.

Creá una instancia de la clase Usuario y llamá a incrementar_intentos_login() varias veces.

Imprimí el valor de intentos_login para asegurarte de que se incrementó correctamente. Luego, llamá a reiniciar_intentos_login() y volvé a imprimir intentos_login para verificar que volvió a 0.