
Acceso a los ejercicios resueltos.

1. Diseñar una clase llamada “Vehículo” con los atributos marca, modelo, año y precio. Crear dos objetos pertenecientes a esa clase e imprimir en pantalla la marca, el modelo y el precio de cada vehículo (mediante __str__).
2. En este ejercicio utilizaremos la misma clase que en el ejercicio anterior y añadiremos un método llamado “nombre_completo” que retorne en una cadena los atributos marca y modelo concatenados y separados por un guión (Seat-Ibiza). Crear dos objetos y probar el método.
3. Diseñar una clase Python llamada “Producto” con los atributos nombre, categoría, precio y cantidad. Diseña en esta clase el método __str__ de forma que retorne todos los atributos en un dato de tipo cadena (str).
   Crear dos productos pertenecientes a esa clase y mostrar todos los datos de aquel producto que tenga mayor precio.
4. Modificar la clase “Producto” del ejercicio anterior añadiendo un método que actualice la cantidad de un producto sumándole un valor pasado como parámetro. Mostrar los datos de un producto antes y después de ser modificada su cantidad.

   En los siguientes ejercicios utilizaremos esta lista de productos:

```
productos = [Producto("tomate", "fruta", 2.3, 100),
             Producto("patata", "verdura", 1.5, 200),
             Producto("cebolla", "verdura", 1.8, 150),
             Producto("manzana", "fruta", 3.2, 50),
             Producto("pera", "fruta", 2.7, 75)]
```

5. Visualizar en pantalla aquellos productos que pertenezcan a la categoría ‘verdura’.
6. Visualizar en pantalla aquellos productos cuyo precio esté entre 1.5 y 2.5 (incluidos).
7. Obtener la media de los precios de los productos de la categoría ‘verdura’.
8. Contar cuántos productos tienen un precio entre 2 y 3 (excluidos).
9. Calcular la media de todos los precios de los productos.
10. Diseñar una clase llamada “Mates” con los **métodos estáticos** que se describen:
    a. mayor. Recibe dos números como argumento y retorna el mayor.
    b. producto. Recibe tres números como argumento y retorna su producto.
    c. potencia. Recibe una base y un exponente como argumentos y retorna la base elevada al exponente.

    Probar los métodos programados.
11. Diseñar la clase “Empleado” con los atributos identificador, nombre, departamento, salario. Tener en cuenta que el salario será privado. Define un método para obtenerlo y otro para modificarlo.
    Programa el método __eq__(), de forma que indique si dos empleados son iguales o no en función de su salario.
    Crear varios empleados, mostrar sus datos y comparar si son iguales o no.
12. Diseñar una aplicación Python que trabaje con objetos de la clase “Partido”. Cada partido tendrá como atributos equipo local, equipo visitante, goles local, goles visitante, campeonato y fecha.

La aplicación consta también de una clase llamada “GestionPartidos” que tendrá como atributo de clase una lista de partidos y los métodos siguientes:

- Filtrar por equipo local. Recibe un equipo local como argumento e imprime todos los partidos de ese equipo actuando como local.
- Ganados del equipo. Recibe un equipo como argumento y retorna cuántos partidos ganó ese equipo, independientemente de si actuó como local o como visitante.
- Mostrar los partidos del año pasado como parámetro.
- Mostrar los partidos de una fecha pasada como parámetro.
- Añadir un nuevo partido a la lista de partidos.
- Cuenta partidos. Retorna el número de partidos de la lista.

Prueba las clases y los métodos creados.

13. Para este ejercicio utilizaremos la lista de productos vista antes. Accederemos al primer objeto de la lista y le añadiremos el atributo ***caducado*** con el valor ***True***. Recorremos la lista y visualizamos el atributo __dict__ de cada objeto.

14. Programar una agenda de contactos.
    Definir la clase Contacto con los atributos nombre, teléfono y correo del contacto.
    a. Añade el método __str__ para que retorne todos los atributos con el formato: **Nombre - teléfono - correo.**
    b. Programa también __repr__ para que retorne los datos del contacto con el formato: **Contacto(nombre, teléfono, correo).**
    c. Programa el método __eq__ para determinar si dos contactos son iguales. En este caso serán iguales si coinciden todos los valores de sus atributos.

    Programa la clase Agenda. Esta clase tendrá una lista de contactos y los métodos.
    - Buscar contacto por nombre y retornar el contacto.
    - Obtener el teléfono de un contacto. Retornar el teléfono.
    - Obtener el correo de un contacto. Retornar el correo.
    - Cambiar el teléfono de un contacto. Retornar True si se pudo hacer el cambio, False en caso contrario.
    - Cambiar el correo de un contacto. Retornar True si se pudo hacer el cambio, False en caso contrario.
    - Listar todos los contactos.
    - Obtener el número de contactos.

15. Diseñar una aplicación Python que simule el funcionamiento de un carrito de la compra. Para ello debes tener en cuenta que al carrito se añadirán una o varias unidades de una serie de productos.
    Necesitas crear un listado de productos disponibles. De cada producto se guarda un id, nombre, categoría, precio y stock.

El carrito debe permitir:

- mostrar lista de productos.
- añadir nuevos productos.
- eliminar un producto.
- actualizar las unidades de un producto.
- calcular el importe total.