Repartido de ejercicios: objetos, clases y mensajes

A continuación, encontrarás distintos ejercicios que buscan reforzar los conceptos trabajados en el curso hasta el momento de Programación Orientada a Objetos. Diseña cada uno de los programas utilizando las guías de diseño SRP y Expert que venimos trabajando. Además, recuerda agregar el código necesario en la clase Program para validar que tu programa se comporta como se espera.

Al final, encontrarás un desafío bonus. Este se trata de una consigna más compleja en la cual comenzarás con un programa sencillo y posteriormente lo modificarás, agregando nuevas funcionalidades para adaptar el funcionamiento de tu sistema y proporcionar información a distintos interesados en la empresa, reflejando lo que sucede en el mundo real donde los programas están en constante evolución.

Mucha suerte!!

1) Crea una clase “Circulo” con los atributos:
   - “radio”

   y los métodos:
   - GetPerimeter(), que retorna el perímetro del círculo
   - GetArea(), que retorna el área del círculo

   Referencia: https://edabit.com/challenge/kpReAapuDjgX2b4em

2) **Calculadora de precios de smoothies**

   Te han encargado crear un programa para la empresa “Smoothies R Us” que permita calcular el precio de los Smoothies que venden a sus clientes.

   Crea una clase “Ingrediente" con los atributos:
   - Nombre
   - Costo (representa el costo de agregar el ingrediente a un Smoothie)

   Luego, crea una clase “Smoothie” que contenga un nombre, un precio base y una lista de Ingredientes extra. Smoothie tiene los métodos:

   - AddIngredient(Ingrediente ing), que agrega un ingrediente a la lista de ingredientes. Un ingrediente no puede agregarse más de una vez.
   - GetTotalPrice(), que devuelva la suma del precio de todos los ingredientes de la lista más el precio base del smoothie

- GetFullName(), que devuelva el nombre inicial del Smoothie mostrando también todos los ingredientes agregados. Ejemplo para un “Batido de frutilla” al cual se le agregan los ingredientes extra “Vainilla”, “Caramelo” y “Copo de chantilly”:
  `“Batido de frutilla con vainilla, caramelo y copo de chantilly”`

Referencia: https://edabit.com/challenge/rYKtzcuCQ9FQ9t9pH

3) **Árbol genealógico**

Crea una clase “Person” con los atributos:

- Name, que representa el nombre de la persona
- Role, que representa el rol que tiene la persona
- FirstProgenitor, que representa uno de los progenitores de la persona
- SecondProgenitor, que representa el otro progenitor

Y los métodos:

- GetName(), que devuelve el nombre de la persona
- ShowFamilyTree(), que muestra el árbol genealógico de la persona de la siguiente forma:
  “Juan es hijo. Pedro es padre. Clara es madre. Carlos es abuelo. María es abuela.” - **Máximo 2 generaciones**

4) **Sistema de transferencias bancarias**

Próximamente abrirá el nuevo banco Programabank Uruguay. Para ello, necesitan que crees un programa que permita registrar las transferencias entre cuentas.

Para ello, crea una clase “Account” con los atributos:

- Number, que representa al número de cuenta (¡no puede haber dos cuentas con el mismo número!)
- Balance, que representa el balance de la cuenta y siempre debe ser positivo
- Titular, que representa al dueño de la cuenta y es una instancia de la clase Person del ejercicio 3.

Con los métodos:

- AddToBalance(double amount), que acredita un monto en la cuenta

- RemoveFromBalance(double amount), que debita un monto de la cuenta
- TransferTo(Account account, double amount), que transfiere monto de una cuenta a otra.

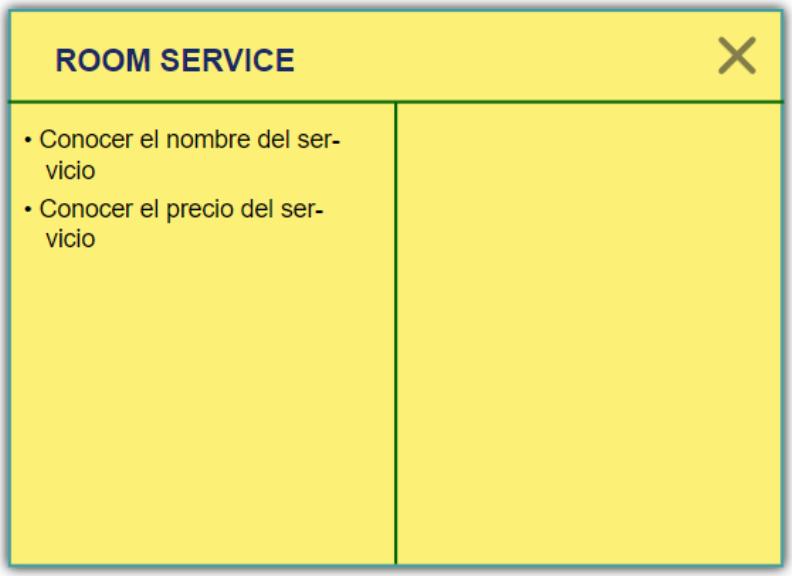
Además, debe crear una clase Bank que almacene todos los clientes del banco y sus cuentas.

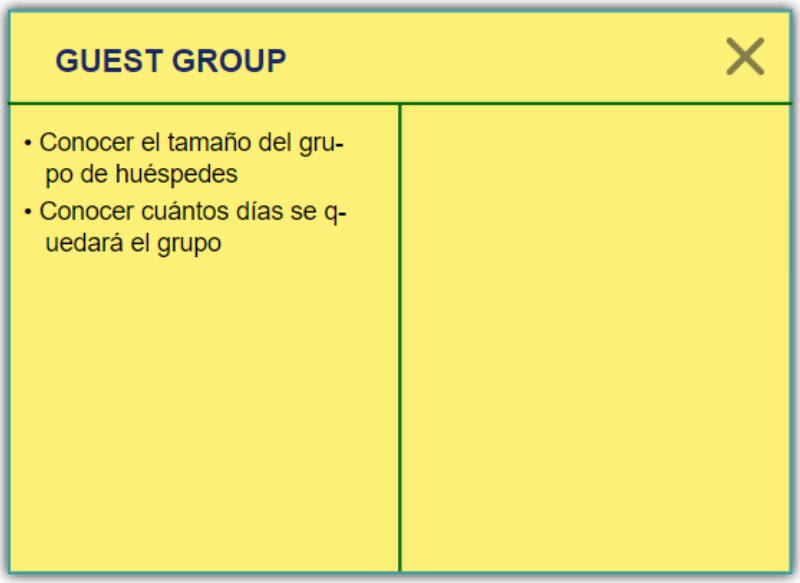
5) **Gestión de habitaciones del hotel**

La cadena de hoteles de lujo “BugFree Resorts” abrirá un nuevo hotel en Punta del Este, y han decidido contratarte para escribir el programa que gestionará las habitaciones, así como el check-in y check-out de huéspedes. Debes programarlo siguiendo buenas prácticas de POO, y el encargado te ha proporcionado las siguientes tarjetas CRC como punto de partida:

ROOM	
• Conocer el número de la habitación • Conocer la cantidad máxima de huéspedes que puede recibir • Conocer el grupo de huéspedes alojado actualmente • Conocer la cuenta de la habitación a pagar • Poder ser ocupada por un grupo de huéspedes • Poder ser liberada • Solicitar un servicio a la habitación	• GUEST GROUP • ROOM SERVICE • ROOM BILL

ROOM BILL	
• Conocer el precio de la habitación por noche • Conocer cuántos días será ocupada la habitación • Conocer los servicios extra ordenados • Calcular el total de la cuenta • Agregar un servicio a la habitación a la cuenta • Borrar el estado de la cuenta e iniciar una nueva	• ROOM SERVICE


ROOM SERVICE
• Conocer el nombre del servicio
• Conocer el precio del servicio


GUEST GROUP
• Conocer el tamaño del grupo de huéspedes
• Conocer cuántos días se quedará el grupo

6) Sistema de Biblioteca:

Crea las siguientes clases con sus respectivos métodos y atributos. Como paso previo a escribir el código, generen las tarjetas CRC correspondientes a cada clase.

Clases:

Libro: Atributos como Titulo, Autor, Disponible (booleano).

Miembro: Atributos como Nombre, ID, LibrosPrestados (lista de libros).

Biblioteca: Atributos como Nombre, ListaLibros, ListaMiembros.

Métodos:

Libro.Prestar(): Cambia el estado del **Libro** a no disponible.

Libro.Devolver(): Cambia el estado del **Libro** a disponible.

Miembro.PrestarLibro(Libro libro): Permite prestar a un **Miembro** un **Libro** si el mismo está disponible, actualiza la lista de libros prestados una vez finalizado el préstamo.

Miembro.DevolverLibro(Libro libro): Permite a un **Miembro** devolver un **Libro**.

Biblioteca.AgregarLibro(Libro libro): Añade un **Libro** a la lista de la **Biblioteca**.

Biblioteca.AgregarMiembro(Miembro miembro): Añade un **Miembro** a la lista de la **Biblioteca.**

Biblioteca.MostrarLibrosDisponibles(): Muestra todos los libros disponibles.

7) Ejercicio: Gestión de Producción Cinematográfica

Estás diseñando un sistema para gestionar la producción de películas en un gran estudio cinematográfico. En este sistema, debes modelar la creación y gestión de películas, los actores que participan en ellas, los directores que las dirigen, y el estudio que las produce.

Requisitos:

Las películas deben estar organizadas por género, tener una duración específica y un elenco de actores que las interpretan.

Cada actor puede participar en múltiples películas, y cada película puede tener varios actores.

Un director es responsable de dirigir las películas, y puede haber dirigido múltiples películas a lo largo de su carrera.

Las películas son producidas por un estudio cinematográfico, que puede haber producido una cantidad considerable de películas.

Tarea:

Diseña un sistema para gestionar este proceso de producción cinematográfica. Asegúrate de distribuir correctamente las responsabilidades entre las distintas partes del sistema, de manera que cada elemento tenga una función clara y específica, siguiendo el Principio de Responsabilidad Única (SRP) y el Patrón Expert.

Piensa cómo descomponer el sistema en partes lógicas y cómo estas partes interactuarán entre sí para cumplir con los requisitos.

Desafío bonus

Una tienda de ropa desea revolucionar su sistema de ventas, agregándole la posibilidad de manejar promociones, a la vez que proporciona información valiosa a los ejecutivos de la empresa acerca de las ventas realizadas para tomar decisiones.

1) En primer lugar, deberás programar unas clases que te permitan almacenar los datos de las ventas. Además, documenta tus clases (las que crees ahora y también cualquier modificación que hagas más adelante) utilizando tarjetas CRC.
    a. La clase Producto debería almacenar por lo menos el nombre y el precio de venta del producto.
    b. La clase LineaVenta representa cada una de las líneas del detalle de la venta, indicando producto vendido y cantidad del mismo.
    c. La clase Venta almacena qué productos (y cuántos de cada uno) se vendieron, y debería permitir calcular el monto final cobrado al cliente incluyendo los descuentos de las promociones que apliquen.
    d. La clase Tienda almacena las ventas realizadas, así como las promociones que la tienda tiene vigentes.

2) Ahora, los primeros en fila para agregar funcionalidades a tu programa es el departamento de Marketing. Necesitan agregar promociones a los productos para impulsar las ventas y la marca de la empresa. Debes agregar a tu programa todo lo necesario para poder aplicar promociones a los productos vendidos. Las promociones pueden implicar un descuento de x% en el precio de un producto, o algo más complejo como “2da unidad al x%” (es decir, si llevo dos productos de $100, por el segundo pagaré solo $100-x). Además, las promociones a aplicar serán siempre sobre un solo producto, no existen descuentos por, por ejemplo, combinar distintos productos.

    Piensa cómo afecta este requisito tu diseño inicial, y realiza las modificaciones necesarias para implementarlo siguiendo el principio SRP y el patrón Expert.

    Pista: si tu diseño inicial permitía tener varias LineaVenta con el mismo producto en la misma venta, puede ser un buen momento para revisar pros y contras de esa elección.

3) Ahora, comenzamos con la parte de obtener información valiosa para la empresa. En primer lugar, el área de Logística necesitará saber cuáles son los productos más vendidos en la tienda para asegurar un stock suficiente. Para ello, deberás analizar todos los registros de ventas de la tienda, y armar una lista con los 5 productos más vendidos (con el más vendido en la primera posición), indicando también la cantidad vendida de cada producto.
4) Por último, el área Comercial necesitará información de las ganancias para mantener al día los acuerdos comerciales con los proveedores y modificar las promociones que hagan falta. Para ello, necesitan que registres en cada Producto su costo (es decir, el precio al cual la empresa lo compra al proveedor). Luego, deberás revisar los registros de ventas, y calcular la ganancia promedio de cada producto, tomando en cuenta el precio de venta cuando se vende a un precio reducido debido a promociones. Por último, tu algoritmo deberá devolver una lista con los 5 productos que dan mayor ganancia y los 5 que dan la peor.

   Por ejemplo, supongamos un producto con precio de venta de $100, costo de $40 y una promoción de “2da unidad con 40% de descuento”.

   Ese producto se vende en dos ventas: en la primera, se vende una sola unidad a $100, y en la segunda se venden dos unidades a $160 ($100 + $100 x 60%).

   En la primera venta el margen de ganancia es de $60 ($100 - $40) por unidad, mientras que en la segunda el margen de ganancia es de $40 (($160 - $80) / 2) por unidad.

   Por lo tanto, el margen de ganancia general para ese producto es el promedio de la ganancia por cada unidad, es decir ($60 + $40 + $40) / 3 = $46,67.