Desafío - Ticketera "Teckit"

- Para realizar este desafío debes haber estudiado previamente todo el material disponibilizado correspondiente a la unidad.
- Una vez terminado el desafío, comprime la carpeta y sube el `.zip`

Enunciado

Un cliente te ha elegido para desarrollar un software que le permita registrar las entradas vendidas y utilizadas de los eventos con los que trabaja una ticketera.

El cliente desea que se pueda hacer seguimiento de:

1. La información que necesita saber de los eventos es: nombre, edad mínima de ingreso y una lista de entradas. En el caso de las entradas, existen de dos tipos:

- Entradas normales, que se caracterizan por tener precio, un numero de asiento, un cliente y un vendedor;
- Entradas VIP, que tienen lo mismo que las entradas normales , pero además tienen derecho a retirar un regalo en el evento.

2. De sus clientes, la información que le interesa saber es : rut, el nombre y la fecha de nacimiento.
3. De sus vendedores necesita los mismos datos que del cliente más la cantidad de entradas vendidas

Además el cliente menciona que:

- Los eventos se reservan el derecho de admisión según la edad del cliente.
- Las entradas solo pueden ser usadas una vez y solo por el cliente asociado al rut.
- Se debe llevar la cuenta de las entradas usadas y de las vendidas para cada evento.

Para desarrollar esto, debes instanciar las clases necesarias para generar un escenario que demuestre que es posible realizar lo solicitado.

Ejemplo:

Crear un evento, crear varios clientes y un vendedor e imprimir por consola cada cosa que pase, por ejemplo, se vendió una entrada para un cliente posteriormente se utilizó una entrada para ingresar a un evento por ese cliente. Donde el output podría ser algo asi:

Caso: Crear evento:

```
Creando evento <NombreEvento> (Edad minima: <edadMinima>)
```

Caso: Venta de entrada:

```
Vendiendo entrada a cliente <NombreCliente> (<RutCliente>) para evento
<NombreEvento>
```

Caso: Ingreso al evento con una entrada, consultada por el rut del cliente Cuando el evento no está en curso:

```
Usando entrada con cliente <NombreCliente> (<RutCliente>) para evento
<NombreEvento>
No se puede usar la entrada porque el evento <NombreEvento> no está en curso.
```

Cuando el evento está en curso y el cliente puede pasar:

```
Usando entrada con cliente <NombreCliente> (<RutCliente>) para evento
<NombreEvento>
Entrada encontrada, <NombreCliente> puede pasar.
```

Cuando el evento está en curso pero la entrada ya se usó

```
Usando entrada con cliente Alex Tapia (19.000.019-4) para evento Fiesta
Entrada para rut 19.000.019-4 ya fue usada, no puede pasar.
```

Cambio de estado del evento (En curso o no en curso)

```
El evento <NombreEvento> se ha cambiado: <EnCurso>.
```

Cuando se muestra la cantidad de entradas vendidas por un vendedor:

```
El vendedor <NombreVendedor> ha vendido:
<CantidadEntradaNormal> entrada/s normal/es
<CantidadEntradaVip> entrada/s VIP
```

Listado de requerimientos

Requerimiento
Clase Entrada con atributos
Clase Entrada VIP con atributos
Clase Cliente con atributos
Clase Vendedor con atributos
Clase Evento con atributos
Herencia con Persona (o alguna similar) como SuperClase de cliente y vendedor
Herencia de Entrada VIP como subclase de Entrada
Encapsulamiento de clase Entrada
Encapsulamiento de clase Cliente
Encapsulamiento de clase Vendedor
Encapsulamiento de clase Evento
Encapsulamiento de clase Entrada VIP
Derecho de admision a eventos según edad de la persona
Calcular edad de la persona en base a la fecha de nacimiento
Calcular edad de la persona en base a la fecha de nacimiento con un método
Prevenir venta de entrada si el cliente tiene una edad incorrecta
Prevenir que una entrada se use más de una vez
Mantener la cantidad de entradas vendidas por los vendedores