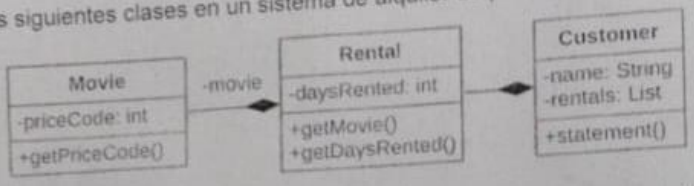
FIUBA Departamento de Computación
Nombre Completo:
Padrón/Legajo :
Fecha: 11/12/2024

Tiempo: 120 minutos

1. Explique que muestra un diagrama Burndown chart utilizado por un equipo Scrum. Realice uno de ejemplo visto a mitad del sprint de 2 semanas y explique lo que podría estar pasando en dicho ejemplo.

2. Se cuenta con las siguientes clases en un sistema de alquiler de películas.



En el cual el costo del alquiler o renta de la película depende del tipo de película y de la cantidad de días que se alquila. También posee un sistema de puntos con beneficios, los mismos se obtienen con los alquileres.
Las películas están categorizadas en estrenos (NEW_RELEASE), normales o no estreno (REGULAR)

El costo del alquiler actualmente está definido como:
Para las películas REGULAR, el costo es $2 por los primeros dos días + $1,5 por día adicional
Para las películas NEW_RELEASE, el costo es $3 por día

Los puntos que acumula el alquiler actualmente está definido como:
Todo alquiler suma un punto y si es estreno y se alquila mas de un día suma un punto adicional.

Tener en cuenta que una película deja de ser estreno luego de un período de tiempo que define el administrador según cada película, y así sucesivamente con el paso del tiempo cuando el administrador lo defina.

Se muestran fragmentos del código actual:

```
public class Movie {

 public static final int NEW_RELEASE = 1;
 public static final int REGULAR = 0;

 private String _title;
 private int _priceCode;

 public Movie(String title, int priceCode) {
  _title = title;
  _priceCode = priceCode;
 }

 public int getPriceCode() {
  return _priceCode;
 }

 public void setPriceCode(int arg) {
  _priceCode = arg;
 }
 public String getTitle() {
  return _title;
 }
}
```

```
public class Rental {

 private Movie _movie;
 private int _daysRented;

 public Rental(Movie movie, int
daysRented) {
  _movie = movie;
  _daysRented = daysRented;
 }

 public int getDaysRented() {
  return _daysRented;
 }

 public Movie getMovie() {
  return _movie;
 }
}
```

```
public class Customer {

 private String _name;
 private Vector _rentals = new
Vector();

 public Customer(String
name) {
  _name = name;
 }

 public void addRental(Rental
arg) {
  _rentals.addElement(arg);
 }

 public String getName() {
  return _name;
 }

 public String statement() {
 .....
 }
```

Donde Customer genera un reporte o listado con el método statement que muestra el listado de alquileres del cliente con el costo total y los puntos que acumuló:

```
public String statement() {
        double totalAmount = 0;
        int frequentRenterPoints = 0;
        Enumeration rentals = _rentals.elements();
        String result = "Rental Record for " + getName() + "\n";

        while (rentals.hasMoreElements()) {
                double thisAmount = 0;
                Rental each = (Rental) rentals.nextElement();

                //determine amounts for each line
                switch (each.getMovie() getPriceCode()) {
                        case Movie.REGULAR:
                                thisAmount += 2;
                                if (each.getDaysRented() > 2)
                                        thisAmount += (each.getDaysRented() - 2)* 1.5;
                                break;
                        case Movie.NEW_RELEASE:
                                thisAmount += each.getDaysRented()* 3;
                                break;
                }
                // add frequent renter points
                frequentRenterPoints++;
                // add bonus for a two day new release rental
                if ((each.getMovie().getPriceCode() == Movie.NEW_RELEASE) && each.getDaysRented() > 1)
                        frequentRenterPoints++;

                // show figures for this rental
                result += "\t" + each.getMovie().getTitle() + "\t" + String.valueOf(thisAmount) + "\n";
                totalAmount += thisAmount;
        }

        // add footer lines
        result += "Amount owed is " + String.valueOf(totalAmount) + "\n";
        result += "You earned " + String.valueOf(frequentRenterPoints) + " frequent renter points";

        return result;
}
```

1. ¿Siendo este una muestra representativa del código del sistema, qué críticas se le puede hacer al dicho código y diseño?
2. El usuario del sistema, nos plantea que necesita algunos cambios:
   - Poder generar un reporte con todos los clientes con la cantidad de puntos ganados hasta la fecha ordenados por cantidad de puntos en forma descendente.
   - Se quiere categorizar a las películas como: NEW RELEASE a los estrenos, RECENT HITS para cuando ya dejan de ser estrenos pero siguen siendo recientes, REGULAR para el resto que ya lleva más años y OLDS, para las más viejas. Cada tipo acumulará puntos adicionales, aún no se sabe bien cuántos puntos cada una y con qué lógica se obtendrán en cada caso.

Proponga un nuevo diseño que permita agregar los nuevos requerimientos respetando los principios y buenas prácticas vistos en la materia. Utilice diagramas UML para expresar el nuevo diseño.