Exercícios:

1) Dadas as classes abaixo, refatore a classe Customer

```
public class DomainObject {

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

        public DomainObject () {};

        public String name ()  {
                return _name;
        };

        public String toString() {
                return _name;
        };

        protected String _name = "no name";
}

public class Movie extends DomainObject {
    public static final int  CHILDRENS = 2;
    public static final int  REGULAR = 0;
    public static final int  NEW_RELEASE = 1;


        private int _priceCode;

        public Movie(String name, int priceCode) {
                _name = name;
                _priceCode = priceCode;
        }

        public int priceCode() {
                return _priceCode;
        }

        public void persist() {
                Registrar.add ("Movies", this);
        }

        public static Movie get(String name) {
                return (Movie) Registrar.get ("Movies", name);
        }
}

class Tape extends DomainObject
    {
    public Movie movie() {
        return _movie;
    }
    public Tape(String serialNumber, Movie movie) {
        _serialNumber = serialNumber;
        _movie = movie;
    }
    private String _serialNumber;
    private Movie _movie;
    }

class Rental extends DomainObject
    {
    public int daysRented() {
        return _daysRented;
    }
    public Tape tape() {
        return _tape;
    }
    private Tape _tape;
    public Rental(Tape tape, int daysRented) {
        _tape = tape;
        _daysRented = daysRented;
    }
    private int _daysRented;
    }

class Customer extends DomainObject
    {
    public Customer(String name) {
        _name = name;
    }
    public String statement() {
        double totalAmount = 0;
        int frequentRenterPoints = 0;
        Enumeration rentals = _rentals.elements();
        String result = "Rental Record for " + name() + "\n";
        while (rentals.hasMoreElements()) {
            double thisAmount = 0;
            Rental each = (Rental) rentals.nextElement();

            //determine amounts for each line
            switch (each.tape().movie().priceCode()) {
                case Movie.REGULAR:
                    thisAmount += 2;
                    if (each.daysRented() > 2)
                        thisAmount += (each.daysRented() - 2) * 1.5;
                    break;
                case Movie.NEW_RELEASE:
                    thisAmount += each.daysRented() * 3;
                    break;
                case Movie.CHILDRENS:
                    thisAmount += 1.5;
                    if (each.daysRented() > 3)
                        thisAmount += (each.daysRented() - 3) * 1.5;
                    break;

            }
            totalAmount += thisAmount;

            // add frequent renter points
            frequentRenterPoints ++;
            // add bonus for a two day new release rental
            if ((each.tape().movie().priceCode() == Movie.NEW_RELEASE)
&& each.daysRented() > 1) frequentRenterPoints ++;

            //show figures for this rental
            result += "\t" + each.tape().movie().name()+ "\t" +
String.valueOf(thisAmount) + "\n";
```

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

    }
    public void addRental(Rental arg) {
        _rentals.addElement(arg);
    }
    public static Customer get(String name) {
        return (Customer) Registrar.get("Customers", name);
    }
    public void persist() {
        Registrar.add("Customers", this);
    }
    private Vector _rentals = new Vector();
    }
```