1. Companiile, retailer-ii si persoanle fizice sunt partiile implicate in comercializarea de componente electronice. Fiecare poate fi si cumparator si vanzator. In functie de tipul cumparatorului si tipul vanzatorului, se aplica diferite discount-uri (vezi tabel).

	Cumparator		
Vanzator	Companie	Retailer	Pers. fizica
Companie	20%	15%	5%
Retailer	15%	10%	5%
Pers. fizica	X	X	0%

Explicatie tabel: daca o companie vinde unui retailer atunci ii ofera acestuia un discount de 15%. Daca un retailer vinde unei companie, atunci ii ofera un discount de 5%. Conform tabelului de mai sus, persoanele fizice nu pot vinde componente unei companii sau unui retailer. Orice incercare de acest fel ar trebui sa genereze o exceptie.

Se cere sa se implementeze sistemul descris mai sus: calcularea discount-ului pentru o vanzare, in functie de tipul vanzatorului si a cumparatorului, astfel incat implementarea sa nu incalce nici una din principiile programarii orientate pe obiecte. Nu se recomanda sub nici o forma verificari de tipuri de clase (instanceof). De asemenea se precizeaza ca pe viitor s-ar putea sa mai apara tipuri noi de vanzatori/cumparatori cu diferite rate de discount la vanzare/cumpare, sau al caror pret se va calcula in mod diferit.

Observatii:

- Prezentarea solutiei implica in mod necesar: cod si resp. diagrame de clase UML. Suplimentar puteti insoti aceste elemente cu explicatii sau alte tipuri de diagrame.
- Codul nu trebuie sa fie complet, dar trebuie sa evidentieze in mod complet mecanismele folosite pentru solutionarea problemei.
- Acolo unde faceti uz de un tipar de proiectare mentionati specific ce tipar ati folosit si motivul optiunii pentru acel pattern.

2. In secventa de cod de mai jos este incalcat un anumit principiu de proiectare studiat.
   a. Identificati principiul incalcat si prezentati care este dezavantajul incalcarii principiului din perspectiva evolutiei potentiale ale unui sistem din care clasele de mai jos ar putea face parte.
   b. Ce alt principiu de proiectare studiat ar fi incalcat daca am proceda la extinderea interfetei definite de clasa **Car** cu metoda **ShiftGears(char gear)**? care este efectul negativ dat de incalcarea principiului pe care l-ati identificat? Care ar fi efectul pozitiv al acestei decizii (din perspectiva codului din functia **main**)?

```
public abstract class Car  {
    public abstract void Drive();
}
public class AutomaticTransmissionCar extends Car  {
    public void Drive() { /* blah */ }
}

public class ManualTransmissionCar extends Car  {
    public void ShiftGears(char gear) { /* blah */ }
    public void Drive() { /* blah */ }
}

public class CarProgram  {
    private static Car CreateCarInstance() {
        // something that gets us a car...
    }
}
public static void main(String[] args)  {
        Car theCar = CreateCarInstance();
        if (theCar instanceof ManualTransmissionCar)  {
            ((ManualTransmissionCar)theCar).ShiftGears('1');
        }
```

```
            theCar.Drive();
        }
    }
```