Laborator 4. Design Patterns

Magazin de cadouri - Jucării

Aplicația pe care o veți realiza ar putea deservi un magazin de cadouri jucării.
Fiecare jucărie are un ***preț*** și se recomandă să fie împachetată într-un anumit **tip de cutie** cadou cu **dimensiuni** potrivite cu mărimea jucăriei. Desigur, se consideră că toate jucăriile din magazin nu sunt ambalate în cutii de producător.

Clientul poate cere ca jucăria pe care o cumpără să fie sau nu împachetată într-o cutie cadou. De asemenea mai poate solicita ca respectiva cutie să fie înfășurată într-o panglica sau nu. Cutia și panglica au costuri funcție de dimensiuni.

Jucăriile

Așa cum s-a amintit orice jucărie este descrisă prin atributele: preț, tip cutie în care se poate împacheta, dimensiunile cutiei. Nu se pot instanția jucării în general, ci jucării specifice. Veți implementa doar următoarele trei tipuri de jucării:

- Minge (preț = 50, cutia recomandată *cubică*, de dimensiuni funcție de mărimea mingii)
- Racheta (preț = 120, cutia recomandată *cilindrică*, de dimensiuni funcție de jucărie)
- Avion (preț = 100, cutia recomandată *paralelipipedică*, de dimensiuni funcție de jucărie)

În pachetul `jucarii`, pentru a asigura polimorfismul, puteți utiliza următoarea clasă abstractă ca supertip al tuturor claselor care vor reprezenta jucăriile Minge, Avion, Racheta (pachetul cutii

```
package jucarii;

import cutii.TipCutie;

public abstract class Jucarie {
        private double dimensiuni[] = new double[3];

        public abstract TipCutie getTipCutie();
        public abstract double getPret();

        public Jucarie(double l1, double l2, double l3) {
            this.dimensiuni[0] = l1;
            this.dimensiuni[1] = l2;
            this.dimensiuni[2] = l3;
        }
        public double[] getDimensiuni() {
            return dimensiuni;
        }
}
```

TipCutie este prezentată în continuare. Pentru `toString()` veți decide dacă implementați o metodă generală în clasa abstarctă sau metode specifice în clasele concrete.

Cutiile

Cutiile în care se pot împacheta jucăriile sunt de trei tipuri specificate de următorul ***enum***

```
package cutii;
public enum TipCutie {
    PARALELIPIPED, CUB, CILINDRU;
}
```

Un exemplu de referire la aceste valori TipCutie.CUB

Cutiile de împachetat jucăriile implementează următoarea interfață:

```
package cutii;

public interface ICutie {
    public static double pretUnitateDeSuprafata = 0.05;
    public static double lnod=20;

    abstract public double getSuprafataTotala();
    abstract public double getLungimePanglica();

    public default double pret(){
        return getSuprafataTotala()*pretUnitateDeSuprafata;
    }
}
```

Toate dimensiunile se consideră în cm, iar prețurile unitare sunt pe cm^2 pentru costul cutiei sau pe cm pentru costul panglicii.

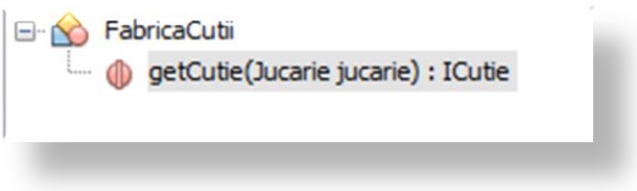
Lungimea panglicii care ar putea orna cutia se calculează după dimensiunile cutiei tinând cont de următoarele:

- pentru o cutie cilindrică – lungimea este cea necesara pentru a înfășura de 2 ori cutia în jurul suprafeței laterale; data fiind prezenta constantei 3.14159… lungimea calculată a panglicii pentru cutiile cilindrice va fi *rotunjită* la o valoare fără zecimale;
- pentru cutiile de forma paralelipipedică lungimea este 2*(L+h) + 2*(l+L), unde l, L, h sunt cele 3 dimensiuni date prin constructorul paralelipipedului; similar și pentru cutiile cubice.

Indiferent de forma cutiei la lungimea calculată se adaugă lnod=20 cm pentru realizarea nodului decorativ.

Confecționarea cutiilor

Magazinul are un dispozitiv special de confecționat cutii la comanda de forma și dimensiunile cutiilor recomandate pentru fiecare jucărie. Acest dispozitiv va fi modelat în program prin clasa **FabricaCutii** care conține o singură metodă statică ce primește o referință către o jucărie și returnează cutia recomandată pentru acea jucărie:



Ce *design pattern* veți utiliza pentru realizarea acestei clase ?

Panglica

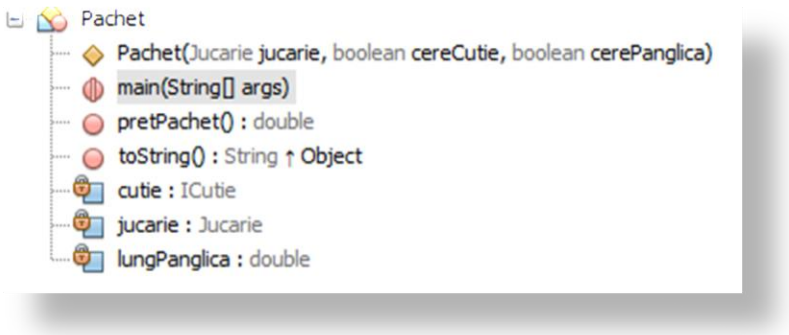
In magazin există operațională o **singură** rolă de panglică care inițial are 10000 cm și din care taie toți vânzătorii. Rola este caracterizată de **costUnitateLungime** (0.01 lei/cm) și **disponibil** (lungimea care a rămas în rola; inițial 10000).

Atunci când se ***cumpără*** o anumită lungime de panglică se verifică dacă lungimea cerută este mai mică sau egală cu **disponibil**. Dacă este mai mare nu se dă curs comenzii de cumpărare (nu se poate cumpăra o lungime mai mică decât cea necesară pentru a orna o anumită cutie).

Ce design pattern veți folosi pentru clasa **RolaPanglica** ?

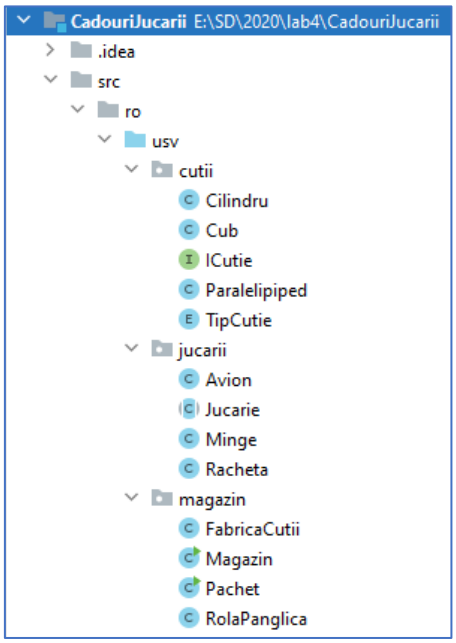
Cumpărarea unui cadou

Această acțiune presupune realizarea unui ***Pachet*** constituit din **jucărie** și, funcție de ce a cerut cumpărătorul, **cutie** și **panglică**. Clasa Pachet o veți realiza potrivit structurii următoare:



TEMA DE REALIZAT

1. **(1p)** Diagrama UML (**0.5p** dacă este întocmită în timpul laboratorului)
2. **(7p)** Veți realiza aplicația sub forma a 3 pachete ca în figura următoare:



Se recomandă dezvoltarea în etape succesive a claselor (“code a little, test a little”). Punctajul de 7 p. va fi acordat astfel:

- pentru realizarea pachetului **cutii** veți primi 2p.,
- pentru pachetul **jucării** 2p,
- clasa **FabricaCutii** – 1p,
- clasa **RolaPanglica** – 1p
- clasa **Pachet** – 1p

In clasa **Magazin** veți scrie o metodă ***main***() care va testa clasele și pachetele amintite anterior. Spre exemplu, pentru testarea clasei **Pachet** veți insera secvența următoare:

```
System.out.println("\n===== Demo Pachet (1p) =====");
Pachet p1 = new Pachet(new Minge(10), true, true);
System.out.println(p1);
System.out.println("Pret="+p1.pretPachet());
```

Pentru a primi punctajul trebuie ca rezultatul produs de program să fie următorul

```
===== Demo Cutii (2p) =====
[Cutie - Paralelipiped{l=10.0, L=20.0, h=30.0}, Cutie - Cub{l=40.0},
Cutie-Cilindru{r=50.0, h=60.0}]

===== Demo Jucarii (2p) =====
[{Avion, cutie recomandata:PARALELIPIPED, l1=15.0, l2=25.0, l3=35.0},
{Minge, cutie recomandata:CUB, l1=45.0}, {Racheta, cutie
recomandata:CILINDRU, l1=55.0, l2=65.0}]

===== Demo Fabrica de cutii (1p) =====
Pentru jucaria:{Avion, cutie recomandata:PARALELIPIPED, l1=15.0, l2=25.0,
l3=35.0}
      cutia:Cutie - Paralelipiped{l=15.0, L=25.0, h=35.0}
Pentru jucaria:{Minge, cutie recomandata:CUB, l1=45.0}
      cutia:Cutie - Cub{l=45.0}
Pentru jucaria:{Racheta, cutie recomandata:CILINDRU, l1=55.0, l2=65.0}
      cutia:Cutie-Cilindru{r=55.0, h=65.0}

===== Demo Panglica (1p) =====
Pentru cutia:Cutie - Paralelipiped{l=10.0, L=20.0, h=30.0} necesar
lung_panglica=:180.0
     dupa cumparare: {RolaPanglica, disponibil=9820.0}
Pentru cutia:Cutie - Cub{l=40.0} necesar lung_panglica=:340.0
     dupa cumparare: {RolaPanglica, disponibil=9480.0}
Pentru cutia:Cutie-Cilindru{r=50.0, h=60.0} necesar lung_panglica=:648.0
     dupa cumparare: {RolaPanglica, disponibil=8832.0}

===== Demo Pachet (1p) =====
Pachet{jucarie={Minge, cutie recomandata:CUB, l1=10.0}, impachetat in
Cutie - Cub{l=10.0}, lungPanglica=100.0}
Pret=81.0
```

Pentru `Demo cutii` și `Demo jucării` *se pot* utiliza tablouri care să fie apoi afișate cu metoda **Arrays.toString (** tablou **).**

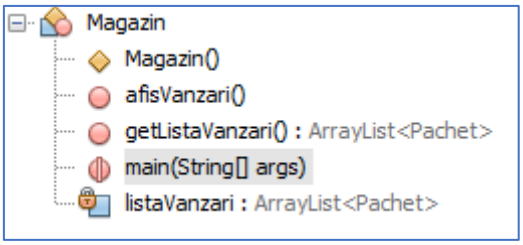
IMPORTANT

1. *Metodele toString() utilizate trebuie să producă **obligatoriu** același format de ieșire ca cel exemplificat mai sus. Nestructurarea aplicației pe 3 pachete se depunctează cu 1p. Codul redundant se depunctează cu pana la 2p.*

3. **(2p.)** Completați clasa **Magazin** astfel încât să înregistreze toate Pachetele vândute într-o lista de vânzări, să le afișeze și să calculeze valoarea totală încasată.

Sugestie. în clasa **Magazin** vânzările vor fi înregistrate în

ArrayList<Pachet> listaVanzari;



Funcția ***main()*** din clasa **Magazin** se va completa cu secvența următoare:

```
System.out.println("\n===== Demo Magazin (2p) =====");
Magazin m = new Magazin();
List<Pachet> vanzari = m.getListaVanzari();
vanzari.add( new Pachet(new Minge(10), true, true) );
vanzari.add( new Pachet(new Minge(10), true, false) );
vanzari.add( new Pachet(new Minge(10), true, true) );
vanzari.add( new Pachet(new Minge(10), false, false) );
vanzari.add( new Pachet(new Racheta(10,20), false, false) );
vanzari.add( new Pachet(new Avion(10,20, 30), false, false) );
vanzari.add( new Pachet(new Avion(10,20, 30), true, false) );
m.afisVanzari();
System.out.println("In rola au mai ramas "+
                RolaPanglica.getRola().getDisponibil()+" cm");
```

Rezultatul execuției trebuie să fie următorul:

```
===== Demo Magazin (2p) =====
Vanzari efectuate
1. Pachet{jucarie={Minge, cutie recomandata:CUB, l1=10.0}, impachetat in
Cutie - Cub{l=10.0}, lungPanglica=100.0}
Pret=81.0
2. Pachet{jucarie={Minge, cutie recomandata:CUB, l1=10.0}, impachetat in
Cutie - Cub{l=10.0}}
Pret=80.0
3. Pachet{jucarie={Minge, cutie recomandata:CUB, l1=10.0}, impachetat in
Cutie - Cub{l=10.0}, lungPanglica=100.0}
Pret=81.0
4. Pachet{jucarie={Minge, cutie recomandata:CUB, l1=10.0}}
Pret=50.0
5. Pachet{jucarie={Racheta, cutie recomandata:CILINDRU, l1=10.0,
l2=20.0}}
Pret=120.0
```

```
6. Pachet{jucarie={Avion, cutie recomandata:PARALELIPIPED, l1=10.0,
l2=20.0, l3=30.0}}
Pret=100.0
7. Pachet{jucarie={Avion, cutie recomandata:PARALELIPIPED, l1=10.0,
l2=20.0, l3=30.0}, impachetat in Cutie - Paralelipiped{l=10.0, L=20.0,
h=30.0}}
Pret=210.0

Suma totala incasata=722.0
In rola au mai ramas 8532.0 cm
```

TEMA ACASA

Dacă aplicația este complet funcțională atunci în clasa **Magazin** scrieți următoarele metode:

1. **`afisPacheteDupaPreț( boolean descrescator)`** - care afișează toate pachetele în ordinea descrescătoare (dacă *descrescator* =true) sau crescătoare (dacă *descrescator* =false) a prețului lor (***2p***)
2. **`cautaPachet(Pachet x)`** care caută în lista de vânzări dacă exista un pachet cu același conținut ca și pachetul **x**. Metoda va întoarce indexul pachetului din lista de vânzări sau -1 dacă nu există un astfel de pachet (***2p***)
3. **`citesteVanzari(String numeFisier)`** – care preia dintr-un fișier text comenzile efectuate într-o anumită perioadă (putem presupune o aplicație de comerț electronic care preia fișierul cu toate comenzile făcute online) pregătește pachetele cu jucării, cutiile și panglicile care se cer conform comenzii făcute. La sfârșit se afișează vanzările potrivit informațiilor din fișierul citit. Fișierul are una sau mai multe linii, pe fiecare linie fiind trei șiruri:

   `numele_jucăriei DA_sau_NU_cutie DA_sau_NU_panglica`

   Exemplu

   ```
   Minge da DA
   Minge da nu
   Minge da   da
   Minge   NU   nu
   RachetA     nu nu
   AVION NU  NU
   Avion da   nu
   ```

   Se va considera că dimensiunile jucăriilor sunt cele utilizate pentru secvența de exemplificare a vânzărilor din *===Demo Magazin===.* (***2p)***.

4. Implementați următoarea strategie în clasa **RolaPanglica**: atunci când rola s-a epuizat (când la prima cerere lungimea panglicii solicitate este mai mică decât disponibilul) să fie adusă una nouă (i.e. să se creeze un nou obiect unic). ***(1p)***