FIȘA LABORATOR 5

Obiective

Polimorfism compartamental/funcţional: model de lucru pentru *strategii-subclase de obiecte*
Procesare:

Concepte [Sub-teme ţintă]

Separare definire operaţii-servicii în interfeţe sau clase abstracte
Separare implementare operaţii-servicii în clase concrete
Furnizare strategii de implementare multiple pentru aceleaşi operaţii-servicii

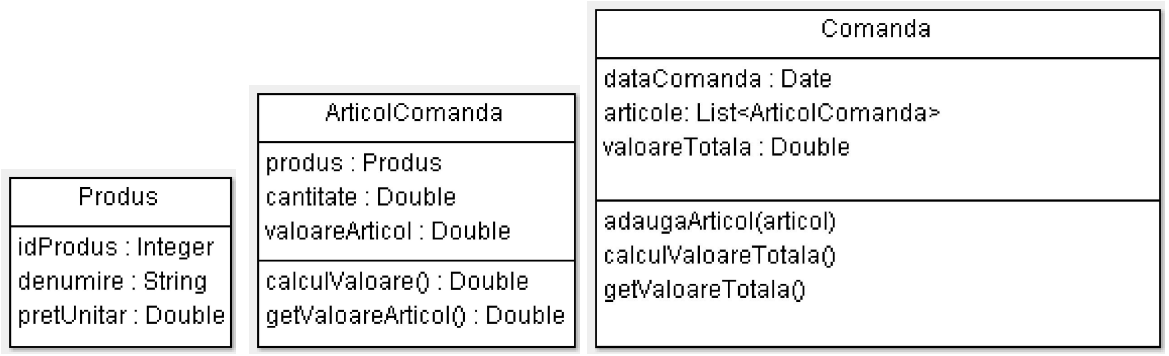
Desfășurare-Repere

Exemplu de predare

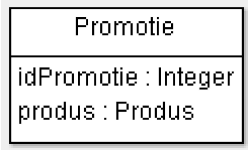
P1. Se lanseaza o campanie de promovare in care pentru calculatoare, cu pret standard 1850: la cel putin 5 calculatoare achizitionate 1 este gratuit, pentru imprimante, cu pret standard 450: la cel putin 6 imprimante achizitionate 2 sunt gratuite. Se face o comanda de 10 calculatoare si 8 imprimante. Se cere valoare cu discount a comenzii.

P2. Se lanseaza o campanie de promovare in care pentru calculatoare, cu pret standard 1850: la o valoare de cel putin 3000 se da o reducere de 20 procente, pentru imprimante, cu pret standard 450: la o valoare de cel putin 2000 se da o reducere de 33 procente. Se face o comanda de 10 calculatoare si 8 imprimante. Se cere valoare cu discount a comenzii.

(i) Creați (dacă nu există deja) clasele **Produs** cu structura *idProdus*, *denumire*, *pretUnitar*. **ArticolComanda** cu structura *produs* și *cantitate*, **Comanda** cu structura *dataComanda* și *articole*. (vezi L2).



(ii) Creați clasa **Promotie** cu atributele *idPromotie*, de tip Integer, și *produs*. Adăugați metodele get/set pentru fiecare atribut precum și constructorul cu parametri și pe cel fără parametri.



(iii) Creați clasa **CampaniePromotionala** cu atributele *id* și *promotii* (listă de promoții). Adăugați metodele get/set pentru fiecare atribut precum și constructorul cu parametri și pe cel fără parametri.

Adăgați, de asemenea, metoda *adaugaPromotie()* pentru "manevrarea" mai ușoară a instanțelor Promotie:

```
public void adaugaPromotie(Promotie promotie){
        this.promotii.add(promotie);
}
```

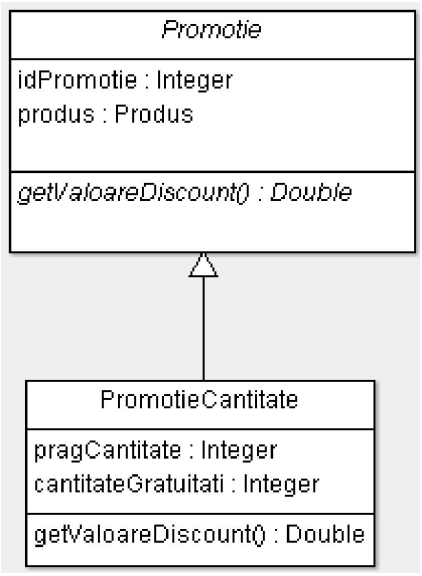
(iv) Există mai multe tipuri de *strategii promoționale*. Scopul fiecărei strategii este calculul unei sume discount care se va deduce din valoarea totală a comenzii. Modul în care se obține acest discount este specific fiecărui tip de strategie în parte. Prin urmare, în clasa *Promotie* se va introduce operația *getValoareDiscount()* care va returna o valoare de tip Double și care *nu va avea o metodă de implementare asociată*, în consecință atât operația cât și clasa vor fi declarate abstracte:

```
public abstract class Promotie {

        public abstract Double getValoareDiscount(Double cantitate);

}
```

(v) Pentru a *implementa* o strategie concretă de calcul al discountului, se va specializa clasa Promotie cu ajutorul unei noi clase **PromotieCantitate** având atributele *pragCantitate*, de tip Integer, și *cantitateGratuități* (prin urmare strategie de calcul a promotiei relativ la cantitatea cumpărată).



Adăugați metodele get/set pentru fiecare atribut precum și constructorul cu parametri și pe cel fără parametri.

(vi) De asemenea, *extinderea* clasei **Promotie** obligă la adaugarea metodei *getValoareDiscount()* cu implementare în subclasă. Strategia de calcul a discountului se referă la faptul că atunci când cantitatea cumpărată depășește un prag (minimal) o anumită cantitate (fixată) va fi achiziționată cu titlu gratuit, prin urmare valoarea ei se va deduce.

```
public class PromotieCantitate extends Promotie {
	/*... ... ... ... ... ... ... ... ... ... ... ... ... ... ...*/
	@Override
	public Double getValoareDiscount(Double cantitate) {
		if (cantitate > pragCantitate)
			return (cantitate/pragCantitate) *
				cantitateGratuitati * this.produs.getPretUnitar();
		return 0.0;
	}
	/*... ... ... ... ... ... ... ... ... ... ... ... ... ... ...*/
}
```

(vii) Implementarea logicii de calcul a valorilor reduse ale comenzilor ca urmare a promoțiilor va fi delegată clasei **CampaniePromotionala** prin operația *getValoareComandaCuDiscount()* care va primi ca argument o **Comandă**.

```
public Double getValoareComandaCuDiscount(Comanda comanda){
	Double valoareCuDiscount = 0.0;
	// Pentru fiecare promotie
	for (Promotie promotie: /* ... ? ... */){
		// Pentru fiecare articol din comanda
		for (ArticolComanda articol: /* ... ? ... */){
			// daca produsul promotiei este cel al articolului comenzii
			if (promotie.getProdus().equals(/* ... ? ... */)
				// aplica/afla valoarea discountului pentru cantitatea articolului
				valoareCuDiscount += promotie.getValoareDiscount(/* ... ? ... */);
		}
	}
	return comanda.getValoareTotala() - valoareCuDiscount;
}
```

(viii) Instanțiați obiectele Produs, ArticolComanda, Comanda, Promotie și CampaniePromotionala care descriu problema 1:

```
public class TestPromotii {
/**
 * P1. Se lanseaza o campanie de promovare in care
 * pentru calc., cu pret standard 1850:la cel putin 5 calculatoare achizitionate 1 este gratuit,
 * pentru imprimante, cu pret standard 450: la cel putin 6 impr. achizitionate 2 sunt gratuite.
 * Se face o comanda de 10 calculatoare si 8 imprimante. Se cere valoare cu discount a comenzii.
 *
 * P2. Se lanseaza o campanie de promovare in care
 * pentru calc., cu pret standard 1850: la o valoare de cel putin 3000 se da o reducere de 20 %,
 * pentru imprimante,cu pret standard450:la o valoare de cel putin 2000 se da o reducere de 33%.
 * Se face o comanda de 10 calculatoare si 8 imprimante. Se cere valoare cu discount a comenzii.
*/

	public static void main(String[] args) {
		// produse
		Produs p1 = new Produs(1, "Calculator", 1850.0);
		Produs p2 = new Produs(2, "Imprimanta", 450.0);

		// comanda
		Comanda comanda = new Comanda();
		comanda.adaugaArticol(new ArticolComanda(p1, 10.0));
		comanda.adaugaArticol(new ArticolComanda(p2, 8.0));
		System.out.println("Valoare comanda: " + comanda.getValoareTotala());
```

```
                // reduceri promotii cantitate
                CampaniePromotionala c1 = new CampaniePromotionala();
                c1.adaugaPromotie(new PromotieCantitate(1, p1, 5, 1));
                c1.adaugaPromotie(new PromotieCantitate(2, p2, 6, 2));
                System.out.println("Valoare cu discount cantitate: " +
                        c1.getValoareComandaCuDiscount(comanda));
        }

}
```

(ix) Adăugați o nouă strategie promoțională cu referire la valoarea articolelor cumpărate – subclasa **PromotieValoare**: la o valoare de achiziție peste un prag limită (atributul pragValoare de tip Double) se acordă un discount procentual (atributul procentReducere de tip Double) la valoarea ce depășește acel prag.

```
public class PromotieValoare extends Promotie {
        private Double pragValoare;
        private Double procentReducere;
        /*... ... ... ... ... ... ... ... ... ... ... ... ... ... ...*/

        @Override
        public Double getValoareDiscount(Double cantitate) {
                Double valoareAchizitie = /*... ? ... */;
                if (valoareAchizitie > pragValoare)
                        return /*... ? ... */;
                return 0.0;
        }
        /*... ... ... ... ... ... ... ... ... ... ... ... ... ... ...*/

}
```

(x) Exemplificați valoarea cu discount a comenzii folosind această nouă strategie (care să corespundă problemei 2).

```
public class TestPromotii {
/**
 * P1. Se lanseaza o campanie de promovare in care
 * pentru calc., pret standard 1850: la cel putin 5 calculatoare achizitionate 1 este gratuit,
 * pentru imprimante, cu pret standard 450: la cel putin 6 impr. achizitionate 2 sunt gratuite.
 * Se face o comanda de 10 calculatoare si 8 imprimante. Se cere valoare cu discount a comenzii.
 *
 * P2. Se lanseaza o campanie de promovare in care
 * pentru calc., cu pret standard 1850: la o valoare de cel putin 3000 se da o reducere de 20 %,
 * pentru imprimante pret standard 450:la o valoare de cel putin 2000 se da o reducere de 33 %.
 * Se face o comanda de 10 calculatoare si 8 imprimante. Se cere valoare cu discount a comenzii.
 */
        public static void main(String[] args) {
                /*... ... ... ... ... ... ... ... ... ... ... ... ... ... ...*/

                // reduceri promotii valoare
                CampaniePromotionala c2 = new CampaniePromotionala();
                // creati/adaugati promotii valoare in campania c2
                /*... ?? ... /
                System.out.println("Valoare cu discount valoare: " +
                        c2.getValoareComandaCuDiscount(comanda));
        }

}
```