Curs 9 POO

© Conf. univ dr. Crenguta M. Puchianu

- Interfete. Definitie
- Relatia de realizare
- Utilizarea interfetelor
- Interfete Java
- Mostenire multipla
- Clase si interfete sealed

Interfete

- Declaraţia unei interfeţe:

[*ModificatorVizibilitate*] **interface** *Identificator* [**extends** *Interfata1*{**,** *Interfata2*}] **{**

Tip Identificator=Expresie

[*ModificatorVizibilitate*] *DeclaratieMetoda***; . . .**

[*ModificatorVizibilitate*] **default** *DeclaratieMetoda* **Bloc**

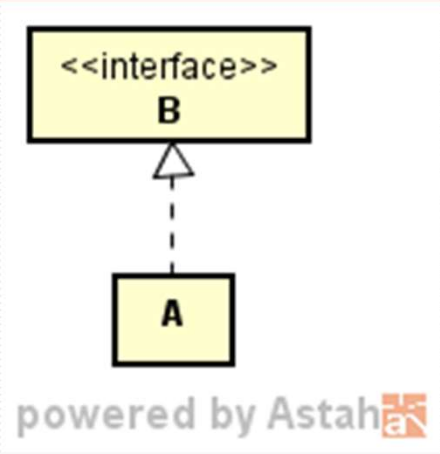
[*ModificatorVizibilitate*] **static** *DeclaratieMetoda* **Bloc**

}

```
public interface OperatiiTablouri{
 public void adaugaElement(Object o);
 public Object getElement(int pozitie);
 public Object stergeElement(int pozitie);
 public boolean cautaElement(Object o);
 public void afiseazaElemente();
}
```

OperatiiTablouri.java => OperatiiTablouri.class

Relatia UML de realizare



- Sintaxa:

class IdentificatorA implements *NumeInterfataB*[**,** *NumeInterfataC*]***{**
//CorpClasa
}

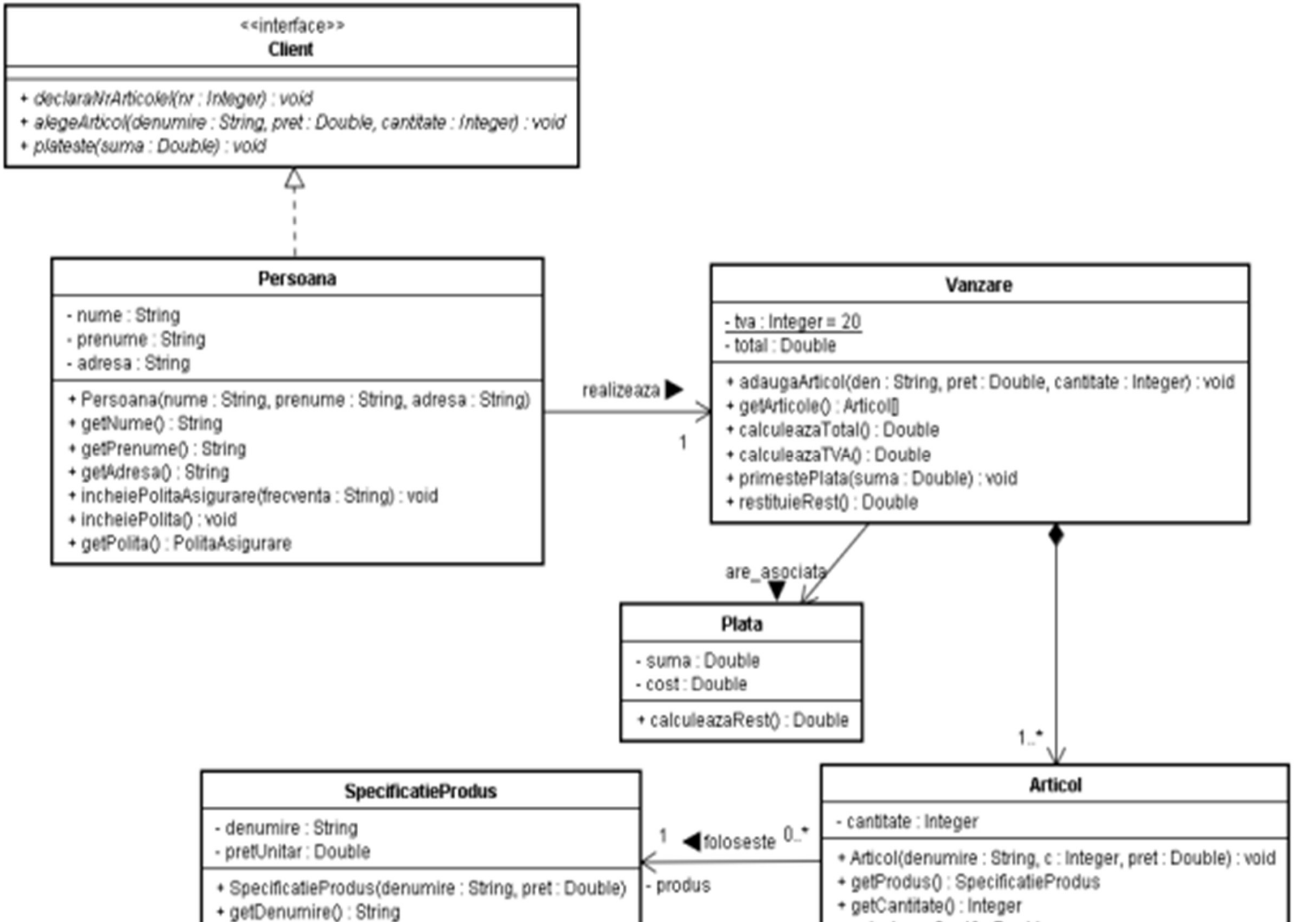
- **Regula**. O clasă care doreşte să implementeze o interfaţă, trebuie să implementeze toate metodele implicit abstracte ale interfeţei. In caz contrar, clasa trebuie sa fie declarata abstracta si toate subclasele ei trebuie sa defineasca metodele mostenite care au ramas nedefinite.

Utilizarea interfetelor

- Rol = descriere a unui set de proprietăţi şi acţiuni pe care le au, respectiv, le executa, unul sau mai multe obiecte într-un anumit context.

- Rolurile sunt definite de responsabilităţile pe care trebuie să le îndeplinească instanţele altor clase care joacă rolul respectiv. Atunci, putem folosi interfeţele pentru a implementa responsabilităţile unui rol.

- Proiectul Magazin: conceptul de client este un rol al unei persoane care cumpără unul sau mai multe articole.
- Rolul de client este descris prin următoarele responsabilităţi:
  - să declare numărul de articole cumpărate;
  - să aleagă fiecare articol pe care vrea să-l cumpere, şi
  - să plătească cu o sumă cel puţin egală cu costul total al articolelor cumpărate.


«interface»
Client
+ declaraNrArticole(nr : Integer) : void
+ alegeArticol(denumire : String, pret : Double, cantitate : Integer) : void
+ plateste(suma : Double) : void
Persoana
- nume : String
- prenume : String
- adresa : String
+ Persoana(nume : String, prenume : String, adresa : String)
+ getNume() : String
+ getPrenume() : String
+ getAdresa() : String
+ incheiePolitaAsigurare(frecventa : String) : void
+ incheiePolita() : void
+ getPolita() : PolitaAsigurare
realizeaza
1
Vanzare
- tva : Integer = 20
- total : Double
+ adaugaArticol(den : String, pret : Double, cantitate : Integer) : void
+ getArticole() : Articol[]
+ calculeazaTotal() : Double
+ calculeazaTVA() : Double
+ primestePlata(suma : Double) : void
+ restituieRest() : Double
are_asociata
Plata
- suma : Double
- cost : Double
+ calculeazaRest() : Double
1..*
SpecificatieProdus
- denumire : String
- pretUnitar : Double
+ SpecificatieProdus(denumire : String, pret : Double)
+ getDenumire() : String
1
foloseste
0..*
- produs
Articol
- cantitate : Integer
+ Articol(denumire : String, c : Integer, pret : Double) : void
+ getProdus() : SpecificatieProdus
+ getCantitate() : Integer

Interfete cu metode default sau statice

- Observatii despre metodele default:
  - pot apela numai metode ale interfeței în care sunt definite;
  - sunt metode instanță;
  - trebuie sa aiba corp de definitie;
  - nu pot fi statice, finale sau synchronized

- Exemplu: interfata Cautare

- Observatii despre metodele statice:
- Sunt apelate cu **NumeInterfata.metodaStatica([parametri actuali])**
- Nu pot fi redefinite in clasele care realizeaza interfata

- Exemplu: interfata CautareStatica

Interfete Java standard

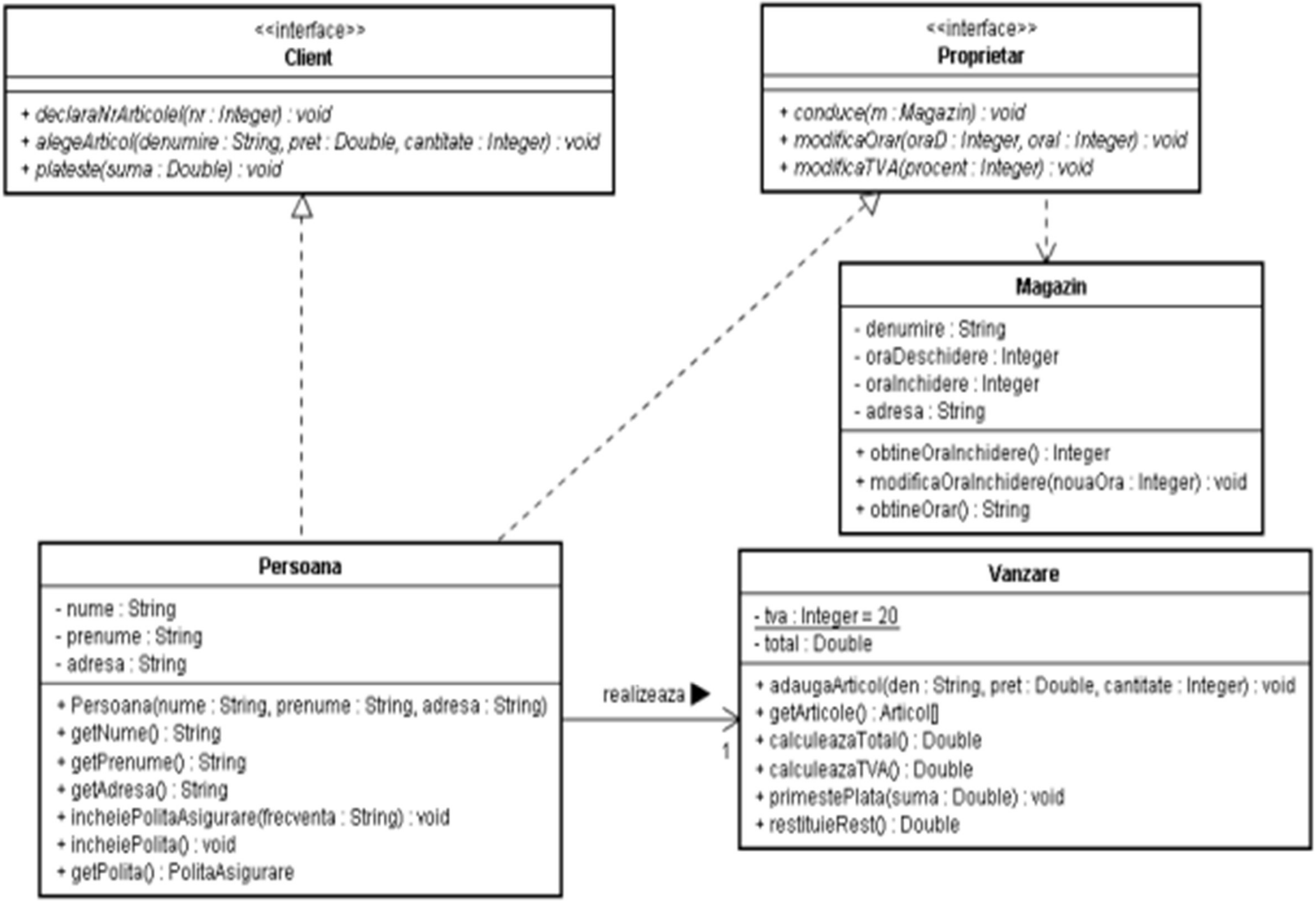
- Interfata Comparable din java.lang

```
public interface Comparable<T>{
  public int compareTo(T o);
}
```

- Interfata CharSequence este realizata de: String, StringBuffer, CharBuffer, StringBuilder şi Segment

Prototip metodă	Efect
`char charAt(int poz)`	Returnează caracterul de pe poziţia `poz` din secvenţă.
`int length()`	Returnează lungimea secvenţei de caractere.
`CharSequence subSequence(int pozIn, int pozSf)`	Creează şi returnează o secvenţă de caractere formată din caracterele de pe poziţiile dintre `pozIn` şi `pozSf` exclusiv.
`String toString()`	Returnează un obiect de tip `String` format prin concatenarea caracterelor din secvenţă.

Mostenirea multipla



Clase si interfete sealed

- O clasa sau o interfata sealed poate fi extinsa sau implementata numai de catre clasele carora li se permite acest lucru.

```
public abstract sealed class ContBancar permits
                                ContCurent,ContEconomii{…}
```

Aceste clase trebuie sa declare la randul lor ce tip de permisiune au: `sealed`, `non-sealed` sau `final`.
Interpretorul recunoaste la executie clasele si interfetele sealed si impiedica celelalte clase sa le extinda sau sa le realizeze/implementeze.

- O clasa sealed poate avea o subclasa abstracta cu conditia ca ea sa nu fie sealed.

- Subclasele pot fi definite in acelasi fisier cu supraclasa lor si, in acest caz, clauza `permits` poate lipsi.

- Beneficiu: putem folosi variabile de tip subclasa ca valori ale case-urilor unui switch