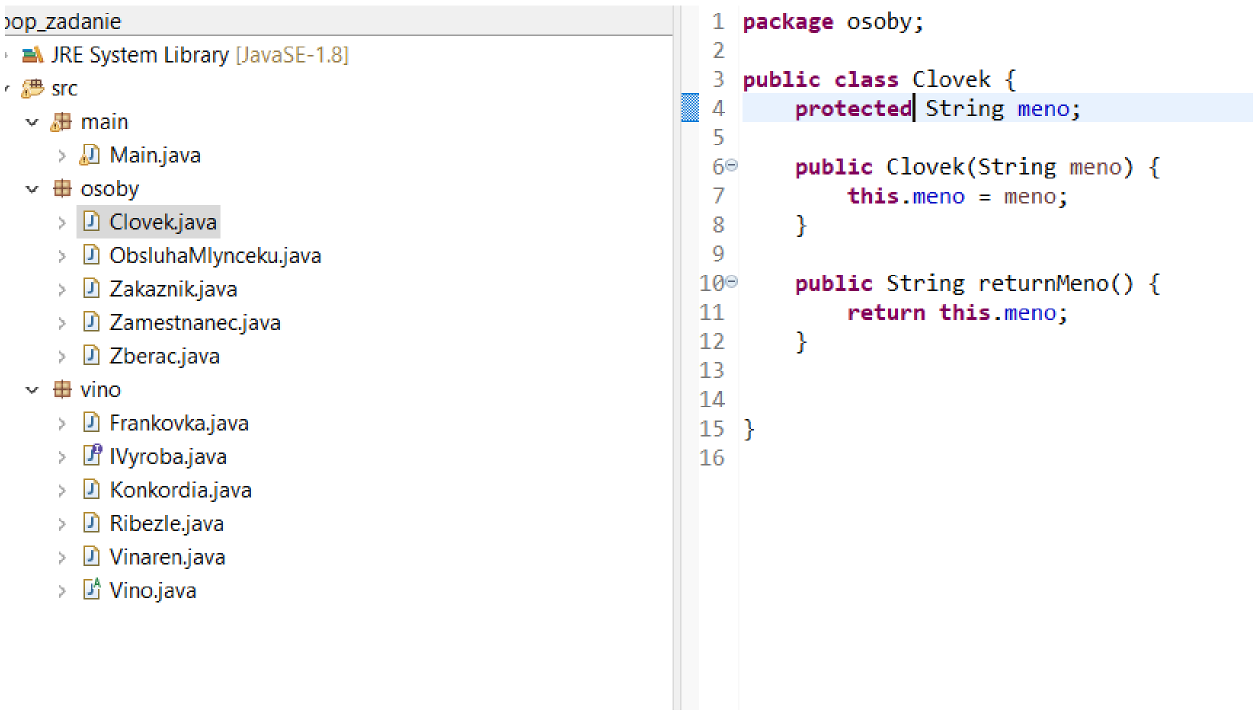
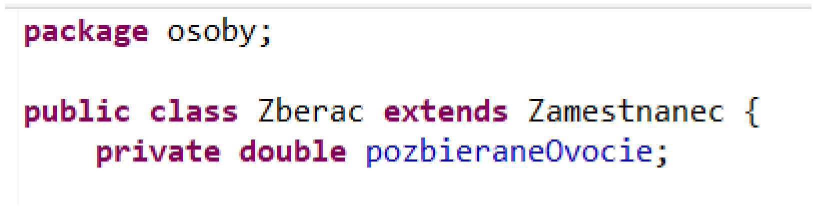
Zapuzdrenie



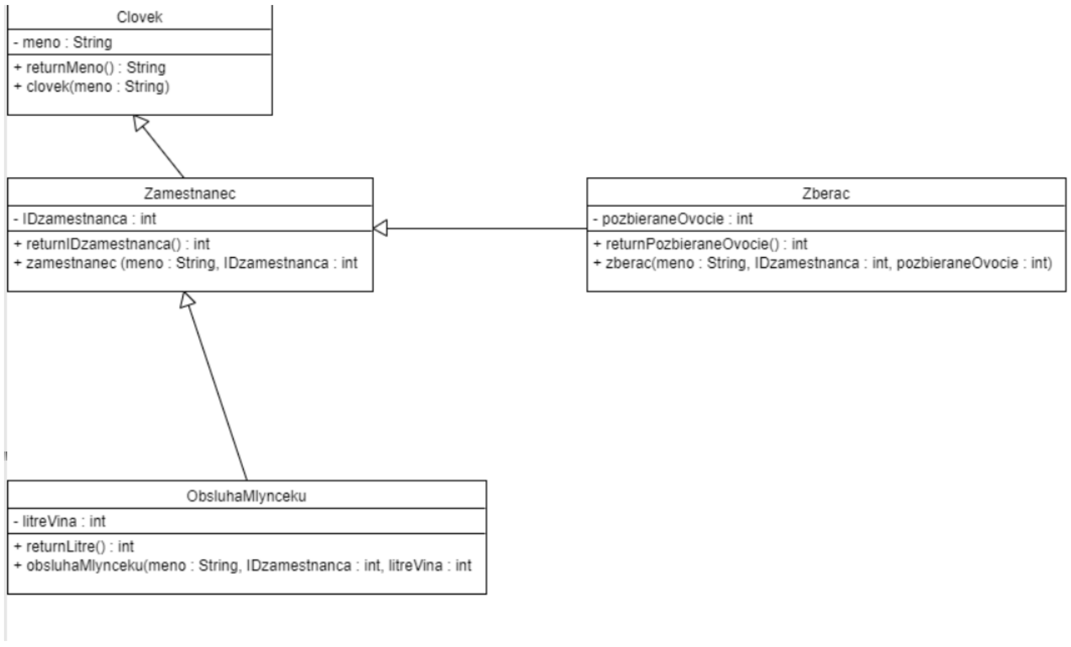
```
op_zadanie
  JRE System Library [JavaSE-1.8]
  src
    main
      Main.java
    osoby
      Clovek.java
      ObsluhaMlynceku.java
      Zakaznik.java
      Zamestnanec.java
      Zberac.java
    vino
      Frankovka.java
      IVyroba.java
      Konkordia.java
      Ribezle.java
      Vinaren.java
      Vino.java
```

```java
package osoby;

public class Clovek {
    protected String meno;

    public Clovek(String meno) {
        this.meno = meno;
    }

    public String returnMeno() {
        return this.meno;
    }


}

```

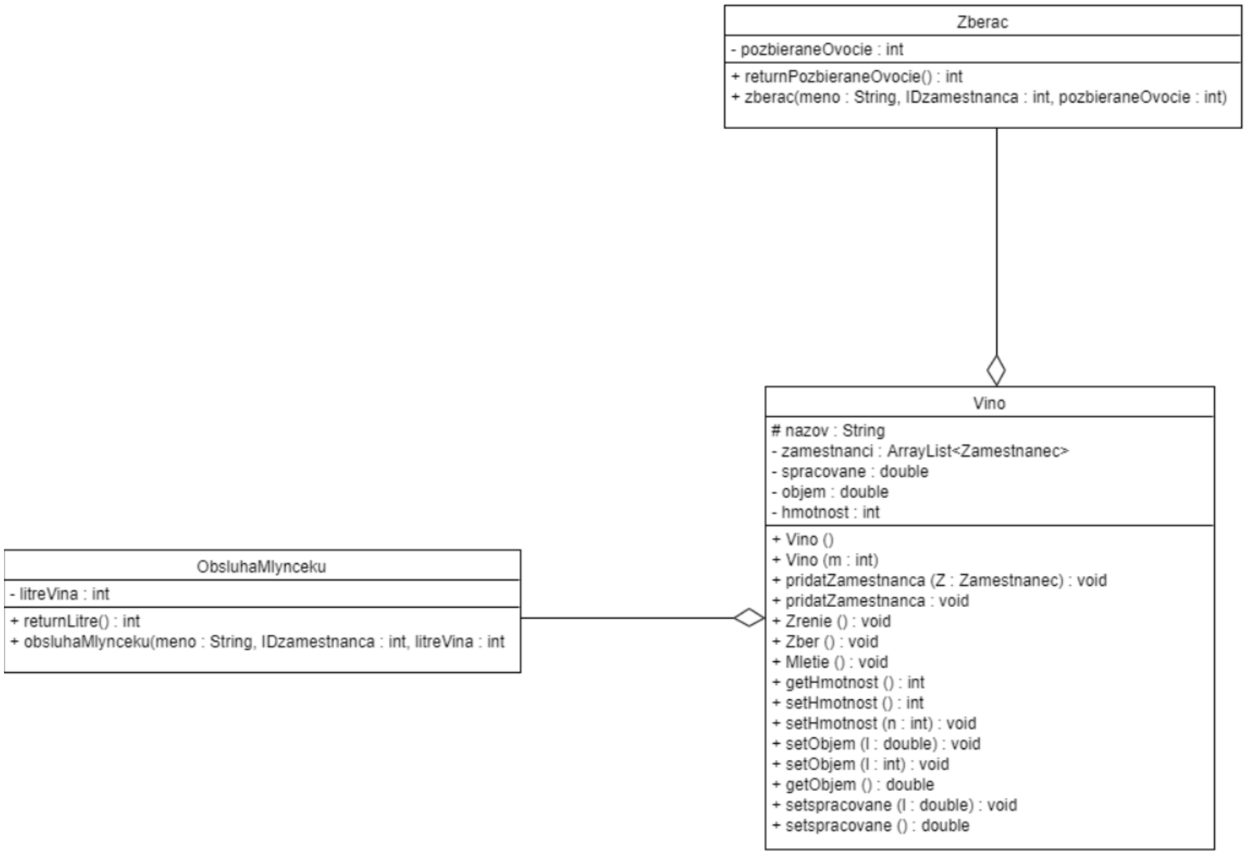


```java
package osoby;

public class Zberac extends Zamestnanec {
    private double pozbieraneOvocie;
```

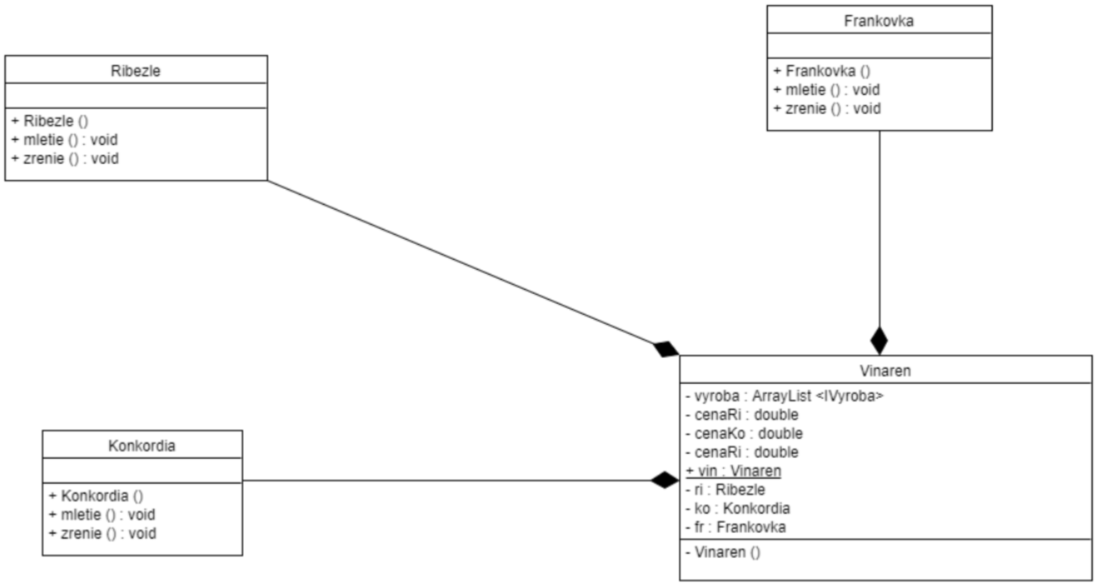
Dedenie + viacnásobne dedenie


Clovek
- meno : String
+ returnMeno() : String
+ clovek(meno : String)
Zamestnanec
- IDzamestnanca : int
+ returnIDzamestnanca() : int
+ zamestnanec (meno : String, IDzamestnanca : int
Zberac
- pozbieraneOvocie : int
+ returnPozbieraneOvocie() : int
+ zberac(meno : String, IDzamestnanca : int, pozbieraneOvocie : int)
ObsluhaMlynceku
- litreVina : int
+ returnLitre() : int
+ obsluhaMlynceku(meno : String, IDzamestnanca : int, litreVina : int

Agregácia


Zberac
- pozbieraneOvocie : int
+ returnPozbieraneOvocie() : int
+ zberac(meno : String, IDzamestnanca : int, pozbieraneOvocie : int)
Vino
# nazov : String
- zamestnanci : ArrayList<Zamestnanec>
- spracovane : double
- objem : double
- hmotnost : int
+ Vino ()
+ Vino (m : int)
+ pridatZamestnanca (Z : Zamestnanec) : void
+ pridatZamestnanca : void
+ Zrenie () : void
+ Zber () : void
+ Mletie () : void
+ getHmotnost () : int
+ setHmotnost () : int
+ setHmotnost (n : int) : void
+ setObjem (l : double) : void
+ setObjem (l : int) : void
+ getObjem () : double
+ setspracovane (l : double) : void
+ setspracovane () : double
ObsluhaMlynceku
- litreVina : int
+ returnLitre() : int
+ obsluhaMlynceku(meno : String, IDzamestnanca : int, litreVina : int

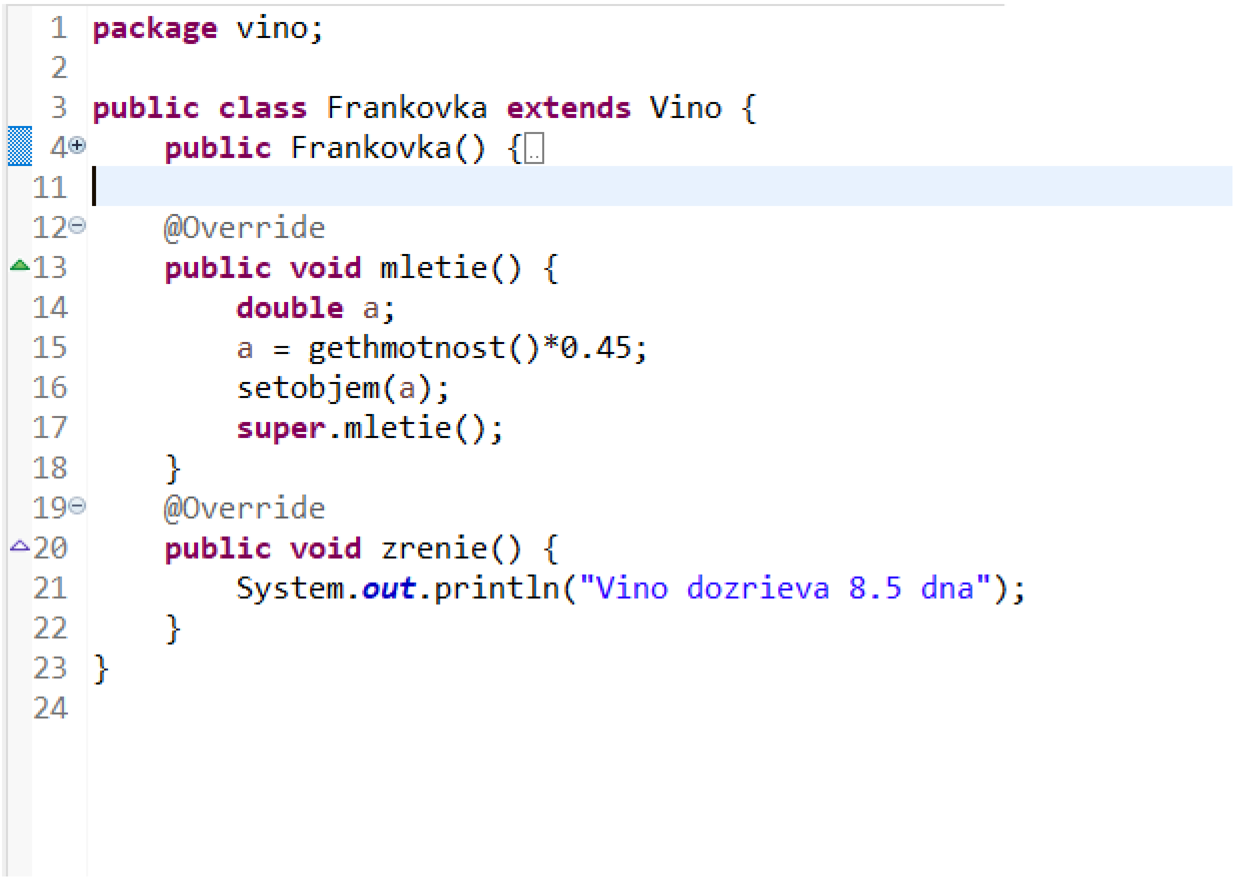
Kompozícia



Overloading

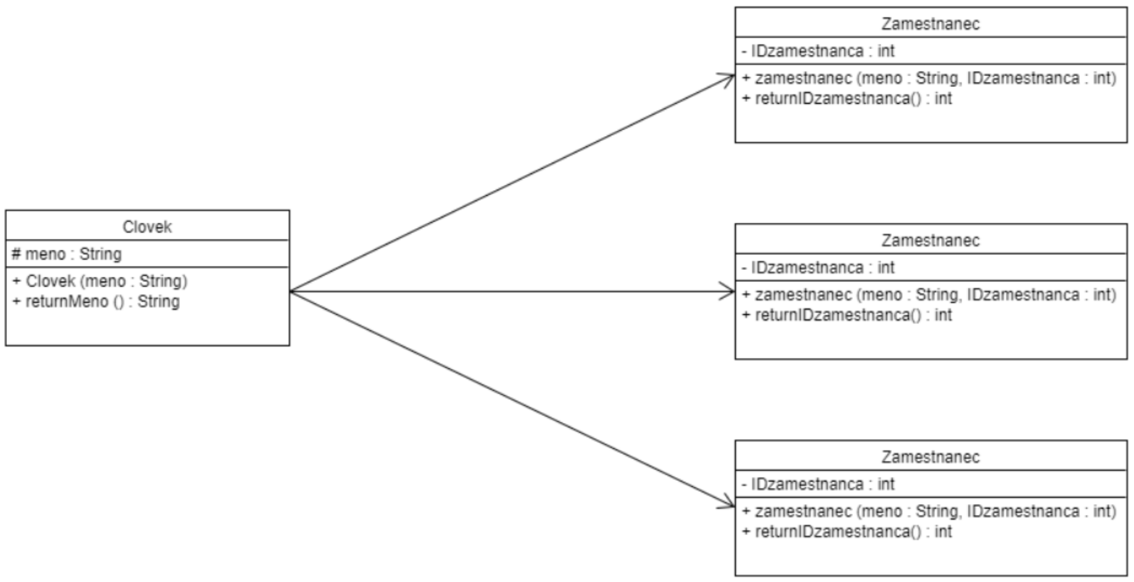
```java
public Vino () {
    zamestnanci = new ArrayList<Zamestnanec>();
}

public Vino (int m) {
    this();                                                            //Overload
    hmotnost = m;
}

public void pridatZamestnanca(Zamestnanec Z)  {
    zamestnanci.add(Z);
}

public void pridatZamestnanca()  {
    Random rng = new Random();
    zamestnanci.add(new Zberac("-",rng.nextInt(100),rng.nextDouble()+1));      //Overload
}
```

Overriding

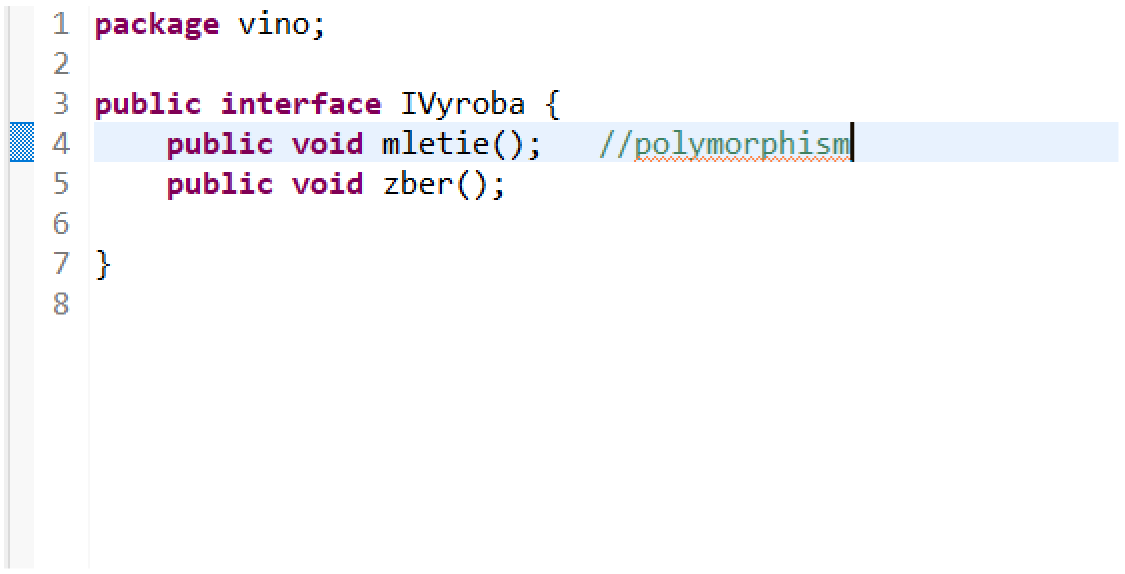


```java
package vino;

public class Frankovka extends Vino {
    public Frankovka() {

    @Override
    public void mletie() {
        double a;
        a = gethmotnost()*0.45;
        setobjem(a);
        super.mletie();
    }
    @Override
    public void zrenie() {
        System.out.println("Vino dozrieva 8.5 dna");
    }
}
```

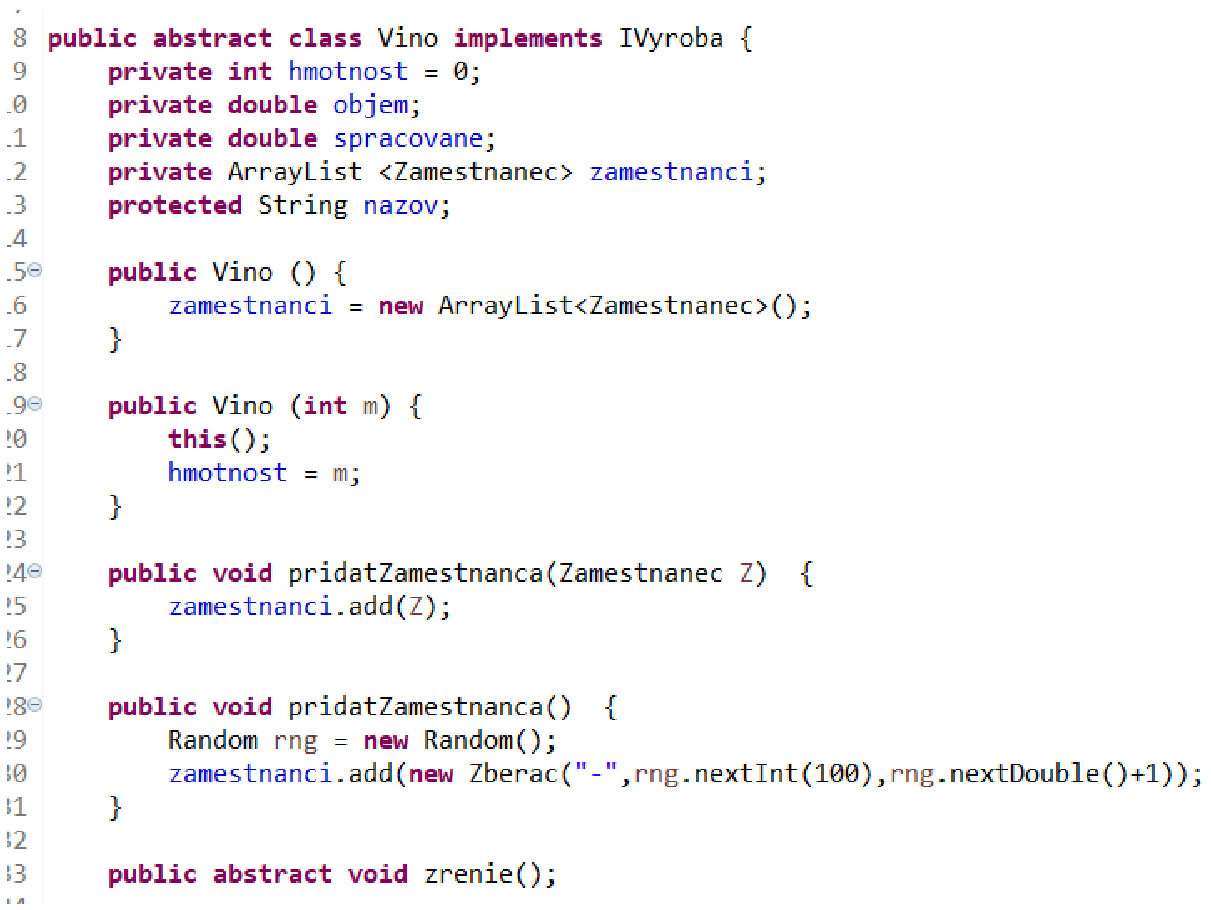
Jednoduchá Asociácia


Clovek
# meno : String
+ Clovek (meno : String)
+ returnMeno () : String
Zamestnanec
- IDzamestnanca : int
+ zamestnanec (meno : String, IDzamestnanca : int)
+ returnIDzamestnanca() : int
Zamestnanec
- IDzamestnanca : int
+ zamestnanec (meno : String, IDzamestnanca : int)
+ returnIDzamestnanca() : int
Zamestnanec
- IDzamestnanca : int
+ zamestnanec (meno : String, IDzamestnanca : int)
+ returnIDzamestnanca() : int

Polymorfizmus + Rozhranie

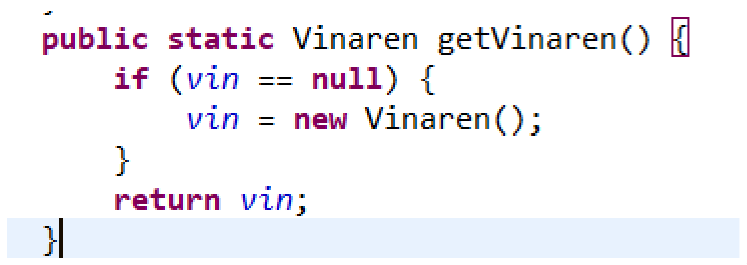

```java
package vino;

public interface IVyroba {
    public void mletie();    //polymorphism
    public void zber();

}
```

Abstraktná trieda



```
public abstract class Vino implements IVyroba {
    private int hmotnost = 0;
    private double objem;
    private double spracovane;
    private ArrayList <Zamestnanec> zamestnanci;
    protected String nazov;

    public Vino () {
        zamestnanci = new ArrayList<Zamestnanec>();
    }

    public Vino (int m) {
        this();
        hmotnost = m;
    }

    public void pridatZamestnanca(Zamestnanec Z)  {
        zamestnanci.add(Z);
    }

    public void pridatZamestnanca()  {
        Random rng = new Random();
        zamestnanci.add(new Zberac("-",rng.nextInt(100),rng.nextDouble()+1));
    }

    public abstract void zrenie();
```

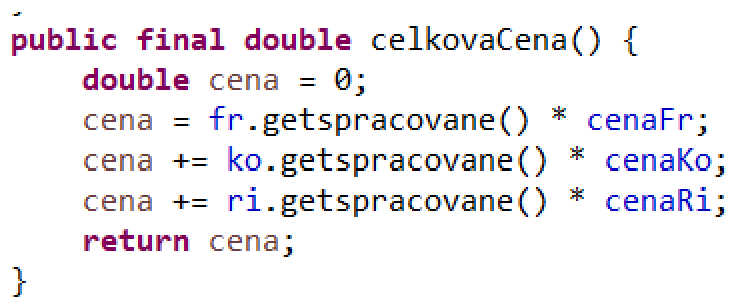
Statický atribút a metóda



```
public static Vinaren getVinaren() {
    if (vin == null) {
        vin = new Vinaren();
    }
    return vin;
}
```

//vin je statický atribút

Finálny atribút a metóda

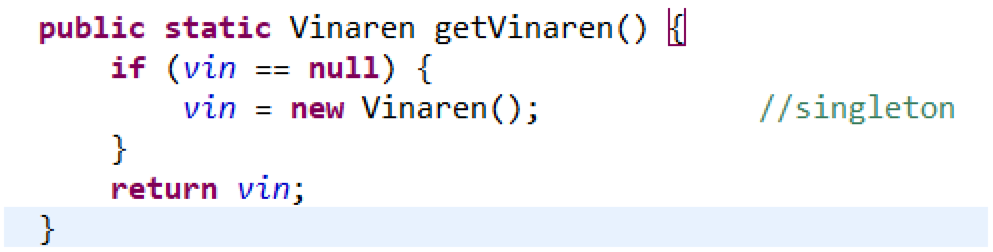


```
public final double celkovaCena() {
    double cena = 0;
    cena = fr.getspracovane() * cenaFr;
    cena += ko.getspracovane() * cenaKo;
    cena += ri.getspracovane() * cenaRi;
    return cena;
}
```

//cenaFr,cenaKo,cenaRi

sú finálne atribúty

Privátny constructor/Singleton



```
public static Vinaren getVinaren() {
    if (vin == null) {
        vin = new Vinaren();            //singleton
    }
    return vin;
}
```