T06. Klasių paveldėjimas

Paveldėjimas

Paveldėjimas – tai procesas, kuriame palikuonių klasė įgauna visas protėvio (tėvo) klasės charakteristikas ir elgesį.

Paveldėjimo ryšys kartais vadinamas **specializacija** (“Trikampis” specializuoja “Figurą”) arba **generalizacija** (“Figura” generalizuoja “Trikampį”)

Terminai:

paveldimoji klasė / superklasė / bazinė klasė / tėvo klasė

paveldinčioji klasė / subklasė / vaiko klasė

Klasių paveldėjimas

Šios naujos klasės (nauji tipai) gali praplėsti protėvių charakteristikas ir veiksmus, kad geriau atitiktų naujas sąlygas ar paskirtį.
Tai leidžia kurti sudėtingesnius objektus iš paprastesnių.

Kadangi kiekvienas objektas yra atskiras egzempliorius, galintis turėti skirtingus duomenis ir veikti skirtingoje aplinkoje, paveldėti metodai dažnai pakoreguojami (kad geriau atitiktų naują objekto aplinką).
Tai padaryti įgalina antra **poliformizmo** rūšis – **metodų užklotis.**

Užklotis – tai galimybė vienodai pavadintą veiksmą realizuoti skirtingais būdais **skirtingoms** objektų klasėms.

Klasių paveldėjimas – tesinys

Paveldėjimas – tai galimybė **inkapsuliacijos** atveju praplėsti duomenų apdorojimo funkcijas (papildyti kapsulę naujais metodais).
Kompozicijai kapsulės duomenys gali būti nepasiekiami (jei nėra tam numatyta reikaikalingų metodų "*getXxx*()",...)

Vaikas paveldi visus "matomus" tėvo klasės kintamuosius (*protected; public* pažeistų inkapsuliacija) ir "matomus" metodus (*public*, *protected*) ir gali savo klasę pasipildyti savais metodais ir kintamaisiais.

Tai vienos krypties papildymas – tėvo klasės objektams vaiko kintamieji ir metodai **nepasiekiami** (nematomi).

** **final** tipo klasės paveldėti negalima.

Klasių paveldėjimo schemos

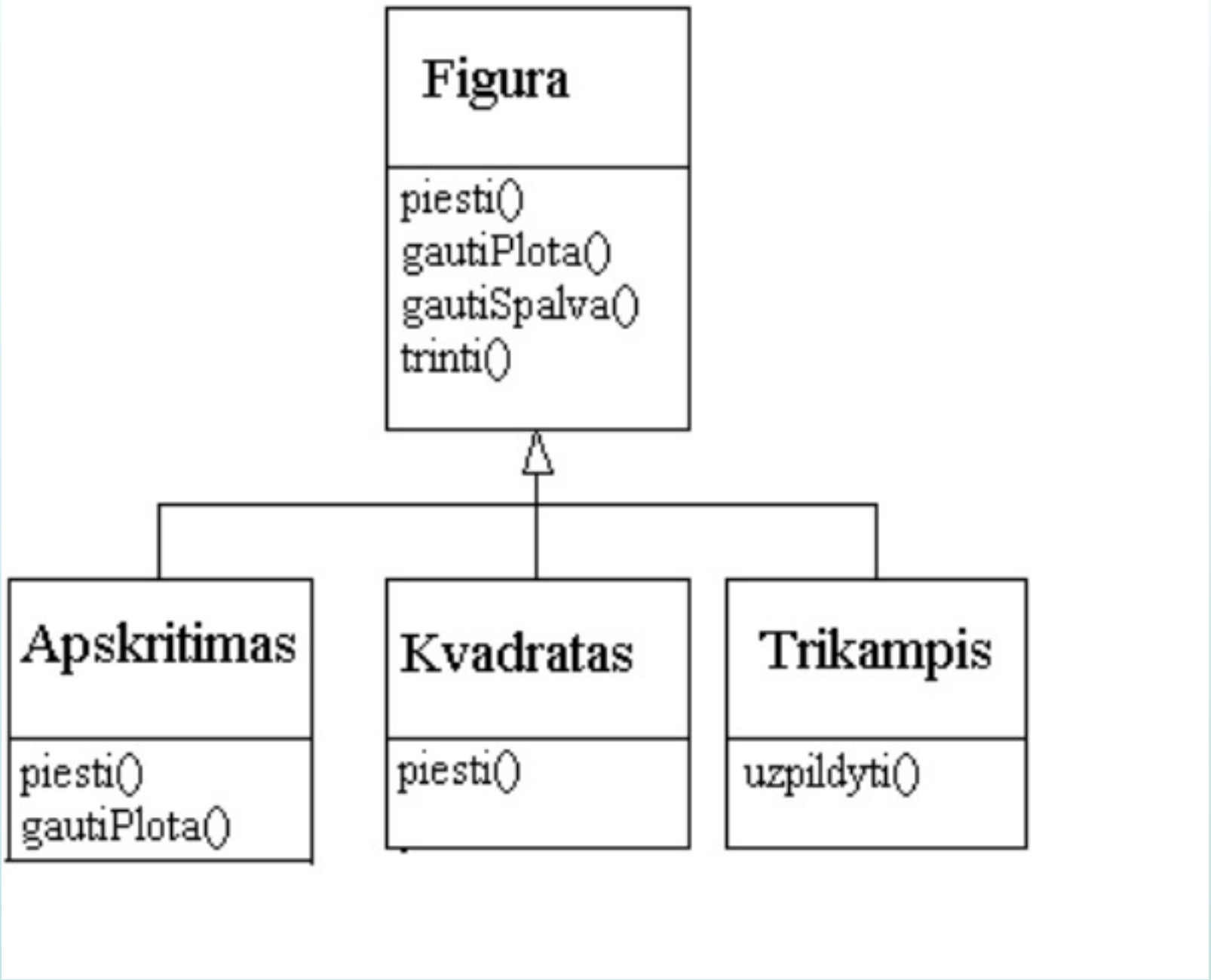
Galimos paveldėjimo schemos:

1. **Vaiko** klasė tik pakeičia ("užkloja") kai kuriuos **tėvo** metodus - vaiko klasė paveldi visą tėvo klasės struktūrą.
   Tai atras polimorfizmo atvejis – **metodų užklotis** (apie jį vėliau).

2. **Vaiko** klasė papildoma naujais metodais.

3. Mišrus variantas –> 1 + 2

Klasių paveldėjimo schemos (grafiškai)



Konstruktoriai ir jų paveldėjimas

Nesant konstruktorių, kompiliatorius **tėvo** ir **vaiko** klasėms sukurs ir iškvies “bevardžius” konstruktorius (be parametrų);

Konstruktorių vykdymas pradedamas **nuo vyriausios** klasės ir eina žemyn pagal hierarchiją;

** Norint pačiam iškviesti **tėvo** klasės konstruktorių, naudojamas žodelis

super(parametrai) ;

Skliaustuose gali būti pateikiami perduodami parametrai, skiriant juos kableliais;

Metodų užklotis (overriding)

Tai situacija kai:

- ✓ **tėvo** ir **vaiko** klasė turi metodus **vienodais** vardais;
- ✓ metodų **antraštės** absoliučiai sutampa (net grąžinamas tipas).

Metodų užklotis leidžia praplėsti sistemos galimybes su mažais bazinės programos kodo pakeitimais.

Metodo pasirinkimas atliekamas pagal tipą objekto, kuriam šis metodas kviečiamas. Tai atliekana vykdymo metu, todėl pailgina programos vykdymą (“vėlyvas susiejimas” /“runtime binding”).

** Neužsikloja:

- konstruktoriai;
- **final** tipo metodai (***final*** tipo klasėje visi metodai tampa ***final***);
- **statiniai** metodai

Paveldėjimas ir kompozicija

Skirtumai tarp sąvokų „**kompozicija**“ ir „**paveldėjimas**“:

Kompozicija – tai kitos klasės objekto panaudojimas naujai projektuojamoje klasėje.

Kompozicija naudojama tada, kai kuriamoji klasė tiesiog naudoja kitos klasės metodus.

Paveldėjimas naudojamas tada, kai kuriamoji klasė naudoja kitos klasės struktūra (interfeisą).

Tėvo klasė iš pirmos paskaitos (modifikuota)

```
public class Klientas {
private  String kodas;
private  int amžius;
private  double indėlis;

public  Klientas(String kodas, int amžius, double indėlis) {
        this.kodas = kodas;
        this.amžius = amžius;
        this.indėlis = indėlis;
}
//  Grąžina indelio sumą (be palūkanų)
public  double getIndėlis() {
return   indėlis;
}
public  void setIndėlis (double indėlis) {
   this.indėlis = indėlis > 0 ? indėlis : 0;
}
```

Tėvo klasės pabaiga

```
//  Grąžina būsimą indėlį metų gale (su palūkanomis)
public  double getIndėlisMetuGale() {
   // Čia turėtų būti abstraktus metodas arba interfeiso realizacija
   System.out.println("Nežinomas kliento statusas");
   return indelis;
}
```

```
//  Perrašomas (užklojamas) Object metodas toString,
public  String toString() {
   return String.format(
“kodas     %7s, amzius %3d, indelis %9.2f”
,     kodas, amzius, indelis);
    }

} // klasės Klientas pabaiga
```

Pirma vaiko klasė

```
public class ProcentinisKlientas extends Klientas {
private  double procentas; // palūkanos % (intervale  [0,100] )

ProcentinisKlientas(String  kodas, int amzius, double indelis,
        double procentas) {
super(kodas,   amzius, indelis); // kviečia tėvo konstruktorių
this.procentas   = procentas;
}
```

```
//  Grąžina metų palūkanas
public  double getProcentas() {
    return procentas;
}

//  Pakeičia metų palūkanas
public  void setProcentas(double procentas) {
   this.procentas = procentas > 0 ? procentas : 0;
}
```

Pirma vaiko klasė (pabaiga)

```
@Override
//  Grąžina būsimą indėlį metų gale (su palūkanomis)
public  double getIndėlisMetuGale() {
    double suma = getIndelis(); // indėlio suma iš tėvo klasės
    return suma + (suma * getProcentas() / 100);
}

@Override
public  String toString() {
return   String.format("%s, metu palukanos procentais %.2f",
super.toString(),     procentas);
//      super tam, kad iškviesti tėvo klasės metodą
    }


} // klasės ProcentinisKlientas pabaiga
```

Antra vaiko (anūko) klasė

```
public class VipinisKlientas extends ProcentinisKlientas {

private  double priedas;      // koeficientas (intervale [0, 1] )
  VipinisKlientas (String kodas, int amzius, double indelis,
double      procentas, double priedas) {
super(kodas,   amzius, indelis, procentas);
setPriedas(priedas);
}

//  Grąžina priedo koeficientą
public  double getPriedas() {
    return priedas;
}

//  Pakeičia priedo koeficientą
public  void setPriedas(double priedas ) {
   this. priedas = priedas > 0 ? priedas : 0;
}
```

Antra vaiko klasė (tęsinys)

```
@Override
//  Grąžina būsimą indėlį metų gale (su palūkanomis)
public  double getIndėlisMetuGale() {
    return super.getIndelisMetuGale() +
(getIndelis()      * getPriedas());
}
@Override
public  String toString() {
    return String.format("%s, priedo koeficientas %5.2f",
super.toString(),    getPriedas());
}

} // klasės VipinisKlientas pabaiga
```

Antra vaiko klasė (komentarai)

** Tam, kad kuriant objektą nebūtų atsitiktinių klaidų (neigiamas priedas ir panašiai), konstruktoriuje vietoje tiesioginio parametro priedas priskirimo objekto kintamajam tikslinga naudoti metodą setPriedas su atitinkamos informacijos kontrole.

** Metode **toString** vietoje objekto kintamojo **priedas** panaudotas metodas getPriedas. Metodų panaudojimas (ne tik *toString*, bet ir kituose metoduose) turi tą privalumą, kad pasikeitus objekto kintamojo vardui reikės pakoreguoti tik ***get*** metodus.

Testas (pradžia)

```
public class Testas {

  public static void main(String[] args) {

Klientas  klientai[ ] = new Klientas[4];

klientai[0]  = new ProcentinisKlientas("P001", 22, 5000, 1.7);
klientai[1]  = new Klientas("K001", 20, 450);
klientai[2]  = new VipinisKlientas("V001", 27, 10000, 2.0, 0.1);
klientai[3]  = new VipinisKlientas("V002", 33, 25000, 2.2, 0.12);
```

Testas (pabaiga)

```
System.out.println("------------  Klientu sarasas ----------\n");

for  (Klientas indelininkas : klientai) {
if   (indelininkas instanceof VipinisKlientas) {
      VipinisKlientas vip = (VipinisKlientas) indelininkas;
      vip.setPriedas(vip.getPriedas() + 0.05); // svenčių proga :-)

     // arba tiesiogiai – be kintamojo vip:
     ((VipinisKlientas)indelininkas).setPriedas(((VipinisKlientas)indelininkas).
getPriedas()      + 0.05);
}

System.out.println(indelininkas   + "\n"); // vykdomas toString
System.out.printf("Suma   metu gale  %9.2f\n\n",
indelininkas.getIndėlisMetuGale());
}
  }
} // Testas pabaiga
```

Testo rezultatai

---------- Klientu sarasas ----------
kodas   P001, amzius 22, indelis   5000,00,
metu palukanos procentais 1,70
Suma metu gale   5085,00

kodas   K001, amzius 20, indelis   450,00
Nezinomas kliento statusas - beprocentinis indelis
Suma metu gale   450,00

kodas   V001, amzius 27, indelis 10000,00,
metu palukanos procentais 2,00, priedo koeficientas 0,15
Suma metu gale   11700,00

kodas   V002, amzius 33, indelis 25000,00,
metu palukanos procentais 2,20, priedo koeficientas 0,17
Suma metu gale   29800,00