PB162 Programování v jazyce Java

5. cvičenie

2. úloha

- otázky
- deadline dnes!!
- javadoc – upresnenie
  - v angličtine komentovať triedu a verejné metódy
  - v prípade implementácie rozhrania sa JavaDoc dedí – nie je nutné ho kopírovať (Ctrl + C, Ctrl + V)
  - **ale**: ak by ste napísali JavaDoc inak, je to indícia, že ho treba prepísať
    - vedľajší efekt
    - dôležitý implementačný detail

...nielen 2. úloha

- konvencie
  - http://java.sun.com/docs/codeconv/
  - https://is.muni.cz/auth/diskuse/diskusni_forum_predn
    - 10–18, 22, 24, 27, **29**, 33, posledný príspevok
    - Príspevok Martina Hincu + správne riešenie
- testík/rozcvička na budúcom cvičení

dedičnosť *(inheritance)*

- 2. pilier OOP
- spôsob vyjadrenia vzťahu nadtyp–podtyp = generalizácia–špecializácia
- vzťah **ISA** (**is a** special case of)
- dnes
  - hierarchie typov
  - abstraktné triedy
  - dedičnosť medzi bežnými triedami

príklad

- poznáte?

príklad

- poznáte?
- do akej triedy patria?

Tatra 600 „Tatraplán“ (1947)

Škoda Octavia (1959)

príklad

- poznáte?

príklad

- do akej triedy patria?

Škoda Liaz 706 RT (1955)

Tatra 148 (1972)

príklad

- do akej triedy patria?
- a čo tie pred tým?

Škoda Liaz 706 RT (1955)

Tatra 148 (1972)

príklad

- spoločné rysy

=> spoločné dáta

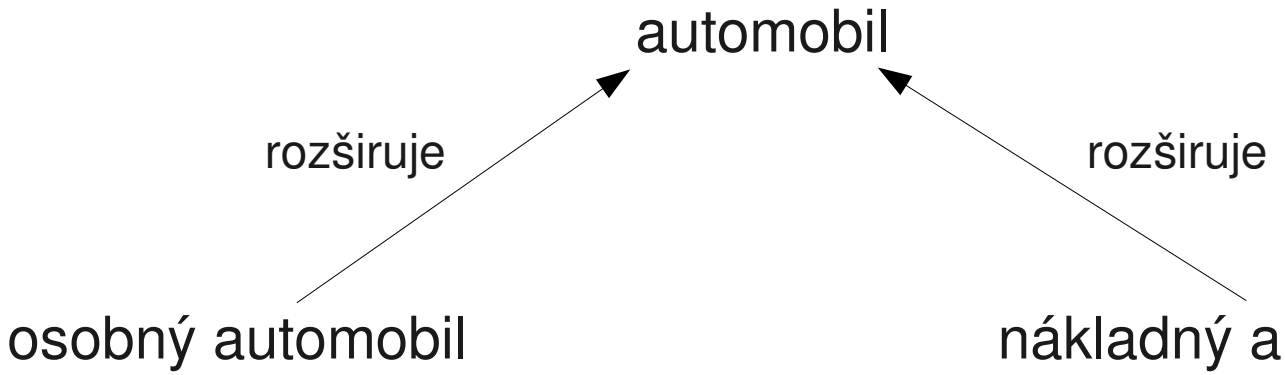
  - objem, výkon motora
  - maximálna rýchlosť

osobný automobil

nákladný automobil

príklad

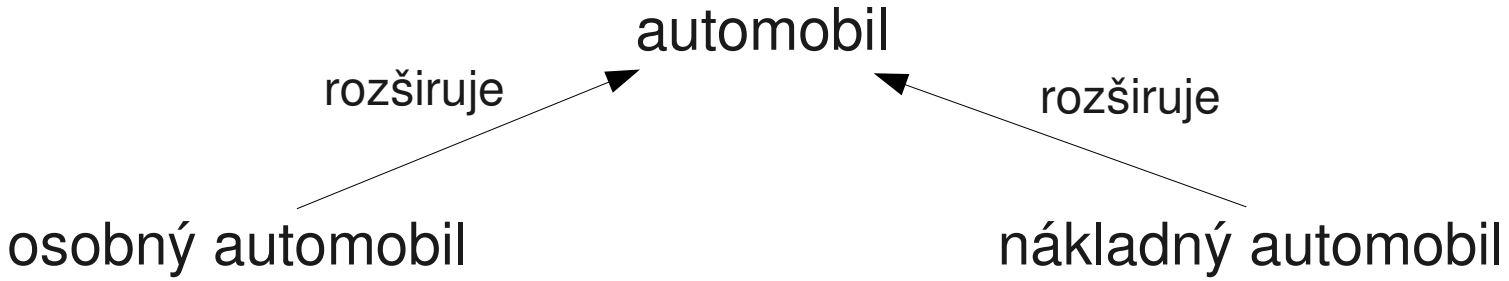
- vyčlenením týchto rysov do spoločného predka vytvoríme jednoduchú **hierarchiu tried**



príklad

- neexistuje automobil, ktorý nie je osobný ani nákladny (:-))

=> spoločný predok je **abstraktný**



príklad

- poznáte?

príklad

- do akej triedy patria?
- kam patria do predchádzajúcej hierarchie?

Škoda 706 RTO LUX (1956)

Škoda ŠD (1963)

príklad

- poznáte?

príklad

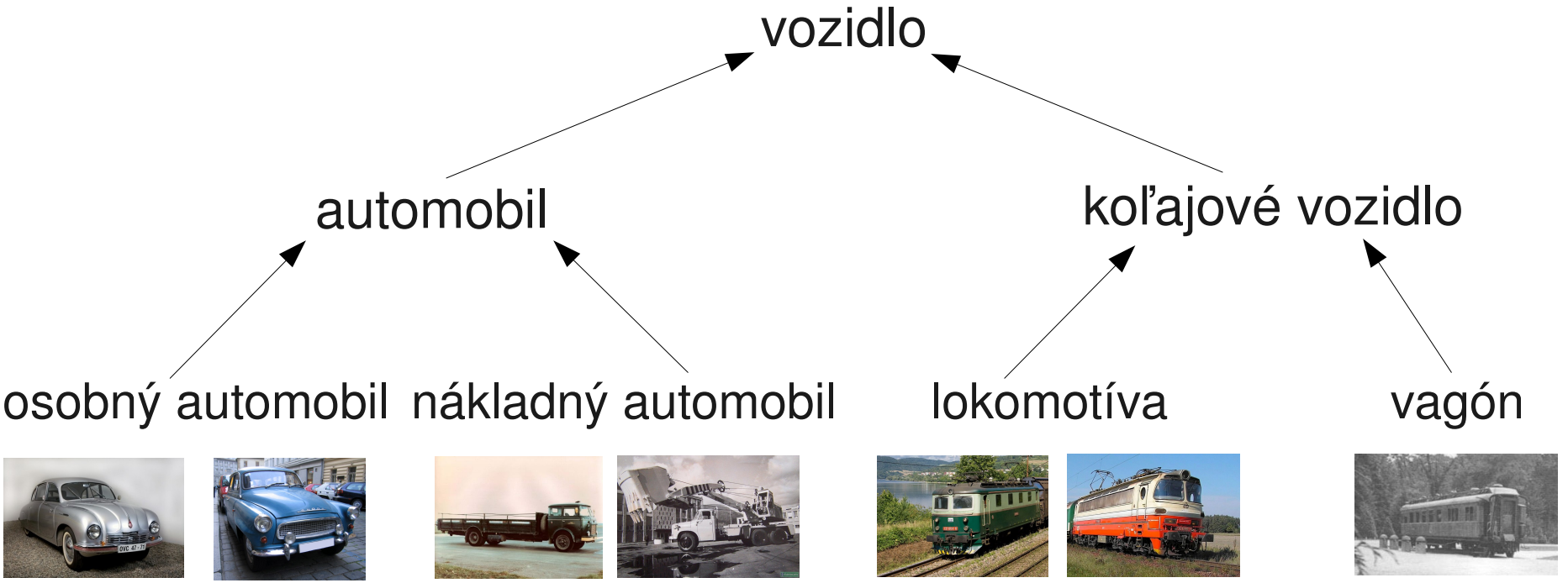
- do akej triedy patria?
  - viem, každá do inej :-)

E 469.1 121 004 „Bobina“ (1960)

S 489.0 230 099 „Laminátka“ (1967)

príklad

- hierarchia tried
  - neúplná, ako vždy



dedičnosť

1. implementácia rozhrania
2. rozšírenie abstraktnej triedy
3. rozšírenie triedy
   - hrozí zneužitie dedičnosti, lebo „sa to hodí“, aj keď nejde o vzťah ISA
     - Circle extends Point
       - spoločné x, y; kružnica pridá r
     - District extends Municipality
       - spoločné: počet obyvateľov, rozloha...
   - **NEPOUŽÍVAŤ** – ide o vzťah **PART OF**, NIE **ISA**!

abstraktná trieda

- definuje stav i správanie
  - rozhranie nedefinovalo stav ani implementáciu správania
- **ale:** nemožno vytvárať inštancie
- trieda určená na rozširovanie
- **použitie** – ak existuje nadtyp
  - spoločné atribúty a metódy pre podtypy
  - nikdy neexistuje samostatne, vždy len v nejakom podtype

dedičnosť – syntax

```
abstract class Vehicle {
    protected double weight; //lepsie pouzit private!!
    public abstract void unload();
    public getWeight() {
        return weight;
    }
}

abstract class Automobile extends Vehicle {
    protected String spz; //lepsie pouzit private!!
}

class Truck extends Automobile{
    protected double payload; //lepsie pouzit private!!
    public void unload() {
        …
    }
    …
}
```

dedičnosť – volanie metód predka

```
abstract class Vehicle {
    protected double weight; //lepsie pouzit private!!
    public Vehicle (double weight) {
        this.weight = weight;
    }
}

abstract class Automobile extends Vehicle {
    protected String spz; //lepsie pouzit private!!
    public Automobile (String spz, double weight) {
        super(weight);
        this.spz = spz;
    }
}

class Truck extends Automobile {
    protected double payload; //lepsie pouzit private!!
    public void unload() {
        super.openDoor(...) {
        …
    }
}
```

viditeľnosť – zhrnutie

- trieda
  - **public** – verejná
    - viditeľná celému „svetu“
    - „svet“ – pre jednoduchosť objekty vo VM
  - bez modifikátora – *package-private*
    - viditeľná v rámci balíčka

viditeľnosť – zhrnutie

- člen triedy (trieda, atribút, metóda)
  - **public** – verejný
  - bez modifikátora – *package-private*
  - **protected** – viditeľný z triedy a jej potomkov
  - **private** – viditeľný len z triedy

- http://java.sun.com/docs/books/tutorial/java/javaOO/accesscontrol.html

zadanie 3. úlohy

- deadline 22. 10. 2008 o 24.00
- najprv uvažovať, potom programovať
  - platí aj pre písomky
    - ceruzka, papier
    - prečítajte **celé** (naozaj celé...) zadanie
    - značte si, kreslite diagramy tried, **uvažujte** (*návrh*)
    - programujte (*kódovanie*)
- hlavná činnosť „programátora“ je **premýšľanie**