22 Prístupové metódy

22.1 Súkromné členské premenné. Zapuzdrenie objektov.

V aplikácii *21.5 EvidenciaVozidiel* sme si vytvorili tri členské premenné, do ktorých sme potom zapisovali hodnoty tak, že sme uviedli názov inštancie, zaň sme dali bodku, názov členskej premennej a priradili sme nejakú hodnotu, prípadne sme ju načítali z klávesnice:

```
...
automobil.pocetKolies = 4;
automobil.rokVyroby = conIN.nextInt();
automobil.najazdeneKM = 56820.8;
...
```

Tomuto prístupu k premenným hovoríme **priamy prístup**. Priamy preto, lebo môžeme priamo pristupovať k členským premenným, bez akýchkoľvek obmedzení. Je to vďaka tomu, že členské premenné boli v triede deklarované ako tzv. **verejné (public)** premenné – ak uvedieme *public* alebo neuvedieme nič, tak sú členské premenné brané ako verejné.

Priamym prístupom k premenným však obchádzame jednu veľmi dôležitú vlastnosť OOP a to je tzv. **zapuzdrenie (encapsulation)**. Zapuzdrenie je princíp, ktorým objekt chránime voči ostatným objektom. V OOP sa snažíme o to, aby objekty so svojím okolím (s inými objektmi a ďalšou časťou kódu) komunikovali iba tak, ako to my dovolíme. Tým dosiahneme väčšiu bezpečnosť pri práci s objektmi, čo je jedným z hlavných cieľov OOP.

Ochranu objektu pred svojim okolím zabezpečíme tak, že niektorú jeho časť urobíme súkromnou a niektorú ponecháme verejnou. Na to musíme myslieť už pri návrhu triedy. V prevažnej väčšine prípadov sú v triede chránené členské premenné a metódy triedy sú verejne prístupné. Členské premenné chránime tak, že z nich urobíme tzv. **súkromné (private)** členské premenné. K takto deklarovaným premenným má priamy prístup len trieda, v ktorej sú deklarované (aj zdedené triedy, ak je pôvodná trieda *public* alebo *protected* – toto si vysvetlíme pri dedičnosti).

Aby sme zabezpečili prístup súkromným členským premenným aj mimo triedy, musíme na to v danej triede vytvoriť príslušné metódy.

22.2 Prístupové metódy.

Metódam, ktoré umožňujú prístup k členským premenným aj mimo triedy, v ktorej sú deklarované, hovoríme **prístupové metódy**. Úlohou prístupových metód je umožniť zápis údajov do súkromnej členskej premennej a rovnako zistenie jej hodnoty.

V Jave je zaužívané pravidlo, že názvy prístupových metód, ktoré umožňujú zápis do členských premenných, začínajú slovom **set**. Názvy prístupových metód, ktoré umožňujú zistiť hodnotu začínajú

slovom **get** – hovoríme im aj tzv. **gettre** a **settre**. Deklaráciu súkromných členských premenných a prístup k týmto premenným cez prístupové metódy si ukážeme na konkrétnej aplikácii.

22.3 Aplikácia NovaEvidenciaVozidiel.

Vytvoríme si novú aplikáciu s názvom **NovaEvidenciaVozidiel**. Bude podobná už spomínanej aplikácii *21.5 EvidenciaVozidiel*. Všetky členské premenné triedy Vozidlo deklarujeme ako *private* a vytvoríme k nim prístupové metódy. V nasledovnej časti kódu sú prístupové metódy iba pre premennú pocetKolies

```
public class Vozidlo
{
 //deklarácia členských premenných
 private int pocetKolies;
 private int rokVyroby;
 private double najazdeneKM;

 //implementácia prístupových metód
 public void setPocetKolies(int p)
 {
     pocetKolies = p;          //pocetKolies môžeme použiť, lebo k nej má
 }                             //metóda priamy prístup

 public int getPocetKolies()
 {
     return pocetKolies;
 }
 ...                           //podobne by boli vytvorené prístupové metódy
}                              //k ďalším premenným
```

členské premenné – všetky sú súkromné

V metóde main() teraz vytvoríme inštanciu, teda objekt triedy *Vozidlo* a nazveme ho *automobil*. Ďalej pomocou prístupových metód zapíšeme počet kolies a prečítame:

```
...
Vozidlo automobil = new Vozidlo();
automobil.setPocetKolies(4);
int kolesa = automobil.getPocetKolies();
System.out.print("Automobil má" + kolesa + "kolesá.");
...
```

Výhodou použitia prístupových metód je aj možnosť ošetrenia vstupov. Napríklad v metóda *setPocetKolies()* by nemala umožniť zadať počet kolies ako záporné číslo. Tento problém teda neriešime v metóde, ktorá *setPocetKolies()* volá, ale v samotnej metóde *setPocetKolies()* – toto je typický OOP prístup. Kompletná aplikácia aj s komentármi je na internete.

22.4 Ostatné členské metódy.

Okrem prístupových metód sú v triedach implementované aj bežné metódy, ktoré umožňujú rôzne manipulovať s objektom, napríklad umožňujú vykonávať s členskými premennými rôzne výpočty, porovnania a pod.

22.5 Aplikácia Stvoruholniky

Vytvoríme aplikáciu s názvom **Stvoruholniky**. V nej deklarujeme triedu s názvom **Stvoruholnik** s tromi súkromnými členskými premennými *stranaA*, *stranaB* a *obvod*. Implementujeme prístupové metódy pre strany a vytvoríme tiež metódy na výpočet obvodu štvoruholníka. Ďalej v metóde main() vytvoríme dve inštancie: stvorec a obdlznik. Načítame a vypíšeme dĺžky strán a vypočítame obvody oboch štvoruholníkov.

22.6 Cvičenie

1. Vytvorte aplikáciu s názvom **BodyVRovine**. Deklarujte triedu s názvom **Bod** s dvomi súkromnými členskými premennými *suradnicaX* a *suradnicaY*. Implementujte prístupové metódy a tiež metódu na výpočet vzdialenosti od bodu [0,0]. Vytvorte niekoľko inštancií triedy Bod a pre každú vypočítajte vzdialenosť od bodu [0,0].
2. Vytvorte aplikáciu s názvom **Vyrobky**. Deklarujte triedu **Vyrobok** s tromi súkromnými členskými premennými *nazovVyrobku*, *pocetKusov* a *cenaVyrobku*. Implementujte prístupové metódy s ošetrením vstupov, vytvorte 5 inštancií, uskutočnite vstupy a výstupy.
3. Vytvorte aplikáciu s názvom **StudijneVysledky.** Program nech umožní zadať meno, priezvisko a triedu žiaka, ďalej známky z troch predmetov a počet vymeškaných hodín. Po zadaní všetkých hodnôt ich program vypíše spolu s priemerom známok. Riešte využitím OOP, pričom priemer známok nech je tiež členskou premennou.

22.7 Otázky

1. Aký je rozdiel medzi súkromnými a verejnými členskými premennými?
2. Vysvetlite pojem zapuzdrenie objektu.
3. Na čo slúžia prístupové metódy?
4. Akými slovami začínajú identifikátory prístupových metód?