PB162 Programování v jazyce Java

10. cvičenie

úvodom...

- 4. úloha
  - použitie **`super`**
  - komentáre
    - komentáre tried
    - `@param`, `@return`
  - `StringBuilder`
- 5. úloha
  - deadline dnes

systém MHD

- zastávky (trieda `Stop`)
- linky (trieda `Line`)
  - priame (1)
  - okružné (44)
- sieť je zložitá a rozmanitá
  - zjednodušenie
    - žiadne nesymetrické a polokružné linky

systém MHD

- zásadnejšia otázka: ako modelovať sieť liniek
  - zoznam zastávok v každej linke
  - atribút `beltLine`: priama/okružná

rozhranie `java.util.List` = zoznam

- lineárna dátová štruktúra
- explicitne (zvonka) usporiadané dáta
- metódy
  - `add`, `insert`
  - `contains`, `indexOf`, `get`
  - `remove`
- implementované triedami
  - `ArrayList` – pomocou poľa
  - `LinkedList` – pomocou previazaných objektov

`java.util.List` – použitie

- import balíčkov

```
import java.util.List;
import java.util.ArrayList;
```

- vytvorenie inštancie

```
private final List<Stop> stops = new ArrayList<Stop>();
```

- pridanie zastávky
  - `stops.add(stop);` (na koniec)
  - `stops.add(4, stop);` (medzi 4. a 5.)
- získanie 5. zastávky
  - `stops.get(4);` (zoznam číslovaný od 0)

`java.util.List` – použitie

- odobratie zastávky 5. zastávky
  - `stops.remove(4);`
- odobratie 1. výskytu zastávky
  - `stops.remove(stop);`
- vyprázdnenie zoznamu
  - `stops.clear()`
- iterácia – for-each

```
for (Stop stop: stops) {
    System.out.println(stop);
}
```

sieť

- vieme modelovať jednotlivé linky a pracovať s nimi
- čo je vlastne sieť?
  - zoznam liniek?
    - zastávky majú zmysluplné poradie, linky nie
- sieť je **množina** liniek
  - každá linka len raz
  - na poradí nezáleží

rozhranie `java.util.Set` = množina

- matematické poňatie množiny
  - neusporiadané dáta
  - objekt nanajvyš raz
- metódy
  - `add`
  - `contains`
  - `remove`
- implementované triedami
  - `HashSet` – pomocou hashovania
  - `TreeSet` – pomocou čierno-bieleho stromu

`java.util.Set` – použitie

- použitie `HashSet`
  - ak prekyjeme `equals`, musíme i `hashCode`
  - lepšia časová zložitosť
    - elementárne operácie (`add`, `contains`, `remove`) v *O(1)*
- použitie `TreeSet`
  - typ množiny musí byť `Comparable`
  - automaticky a priebežne sa usporiadava
  - cena: časová zložitosť je horšia
    - elementárne operácie (`add`... `remove`) v *O(log n)*

`java.util.Set` – použitie

- import balíčkov

```
import java.util.Set;
import java.util.HashSet;
```

- vytvorenie inštancie

```
private final Set<Line> lines = new HashSet<Line>();
```

- pridanie linky
  - `lines.add(line);`
- dopyt na prítomnosť
  - `lines.contains(line);`

`java.util.Set` – použitie

- odobratie linky
  - `lines.remove(line);`
- vyprázdnenie množiny
  - `stops.clear()`
- iterácia – for-each

```
for (Line line: lines) {
    System.out.println(lines);
}
```

rozdiely – množina vs. zoznam

- poradie prvkov
  - zoznam je (explicitne) usporiadaný,
  - množina nemá definované poradie prvkov
    - nemá metódy na prácu s prvkami pomocou indexov (`insert`, `add(4, stop)`, `get(4)`, `remove(4)`)
- zložitosť operácií `contains`, `remove`
  - v zozname lineárna
  - v množine konštantná (`HashSet`)
- každé sa hodí na niečo iné!