

# 6

## Kontajnery

# Pojmy zavedené v 5. prednáške<sub>(1)</sub>

- logické výrazy
  - operátory &&, ||, !
- referencie
  - operatory ==, !=
  - hodnota null
- zánik inštancie
  - zánik pri kompozícii
  - zánik pri asociácii

# Pojmy zavedené v 5. prednáške<sub>(2)</sub>

- reťazce
  - trieda String
  - reťazcové literály
  - spájanie reťazcov
  - reťazcové výrazy
  - rovnosť reťazcov

# Cieľ prednášky

- skupiny objektov – kontajnery
- nový prvok algoritmu – cyklus
- príklad: poznámkový blok

# Jednoduché a zložené objekty

- jednoduché objekty
  - atribúty, parametre – primitívne typy
- zložené objekty
  - kompozícia – celok a časť
    - digitálne hodiny
  - kontajnery – vytváranie skupín prvkov jedného druhu
    - môže obsahovať ľubovoľný počet prvkov

# Príklady kontajnerov

- zoznam študentov v študijnej skupine
- katalóg kníh v knižnici
- cestujúci v trolejbuse
- index – zoznam zapísaných predmetov
- zoznam klientov banky
- diár – zoznam poznámok

# Poznámkový blok – Diár

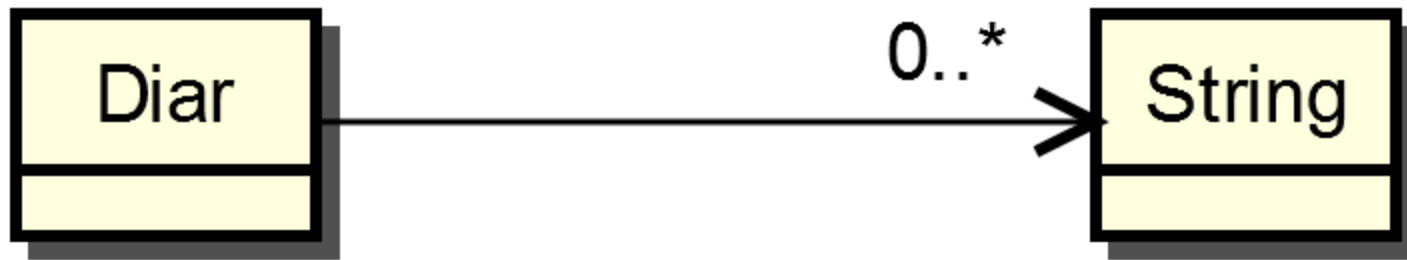
- kontajner na poznámky
  - poznámka – text (reťazec)
- služby:
  - nové poznámky – doplnenie
  - nepotrebné poznámky – škrtanie
  - vypísanie zvolenej poznámky
  - počet všetkých poznámok

## Diár

- + new(): Diar
- + vlozPoznamku(poznamka: String): void
- + vypisPoznamku(poradoveCislo: int): void
- + zmazPoznamku(poradoveCislo: int): void
- + getPocetPoznamok(): int



# Diár – vzťah s poznámkou



# Diár – s doterajšími prostriedkami

- každá poznámka – atribút
- pevne daný počet poznámok
- voľná poznámka == null

# Prázdný diár – objektový diagram

: Diar

- poznamka1: String = null
- poznamka2: String = null
- poznamka3: String = null
- poznamka4: String = null

# Diár s 1. poznámkou

: Diar

- poznamka1: String = "SI - prezentácie"
- poznamka2: String = null
- poznamka3: String = null
- poznamka4: String = null

# Diár so 4 poznámkami - plný

: Diar

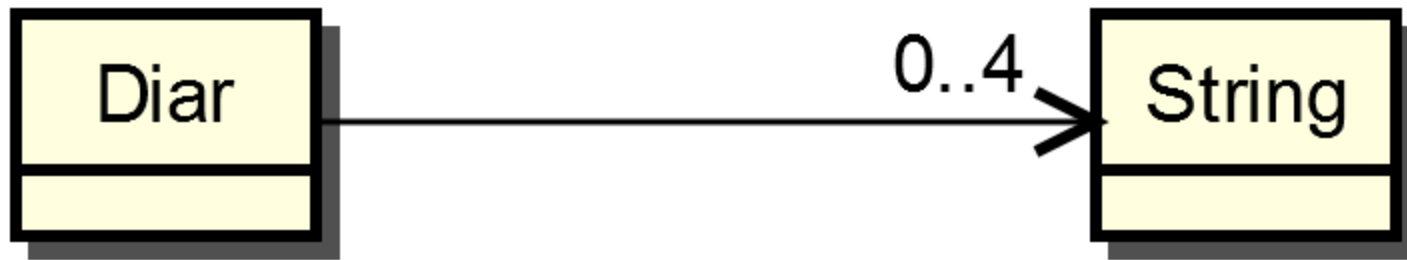
- poznamka1: String = "SI - prezentácie"
- poznamka2: String = "INF1 - prednáška"
- poznamka3: String = "nákup - chlieb"
- poznamka4: String = "INF1 - cvičenie"

# Diár po vymazaní tretej poznámky

: Diar

- poznamka1: String = "SI - prezentácie"
- poznamka2: String = "INF1 - prednáška"
- poznamka3: String = null
- poznamka4: String = "INF1 - cvičenie"

# Diár – diagram tried



# Diár – ďalšie podmienky

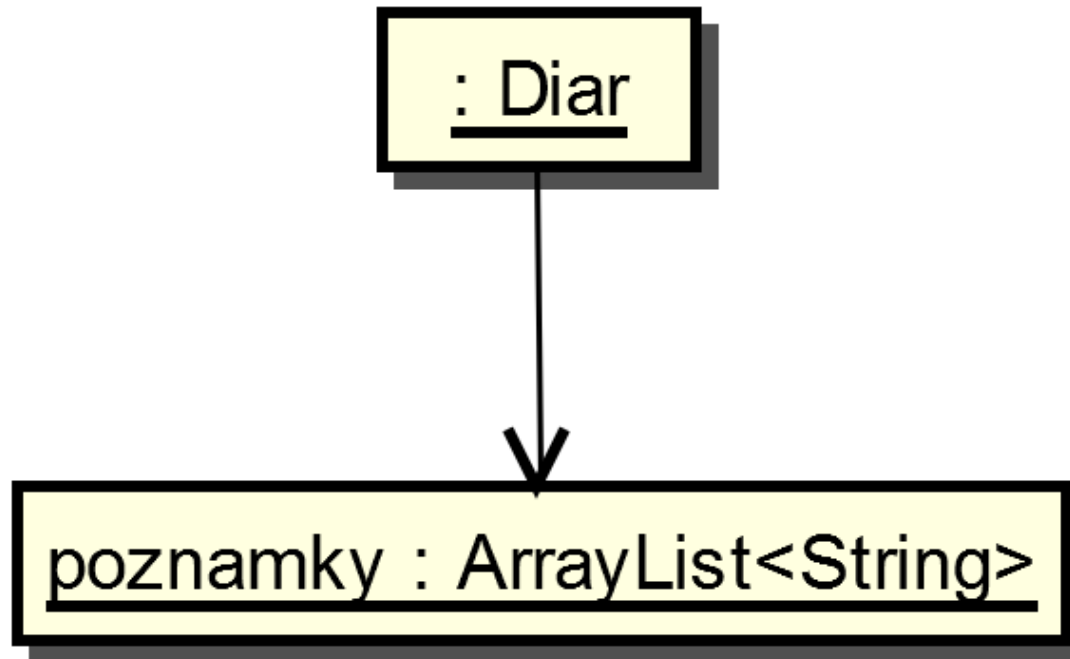
- neurčená horná hranica
- počet poznámok sa mení
- počet je ľubovoľný – aj žiadna



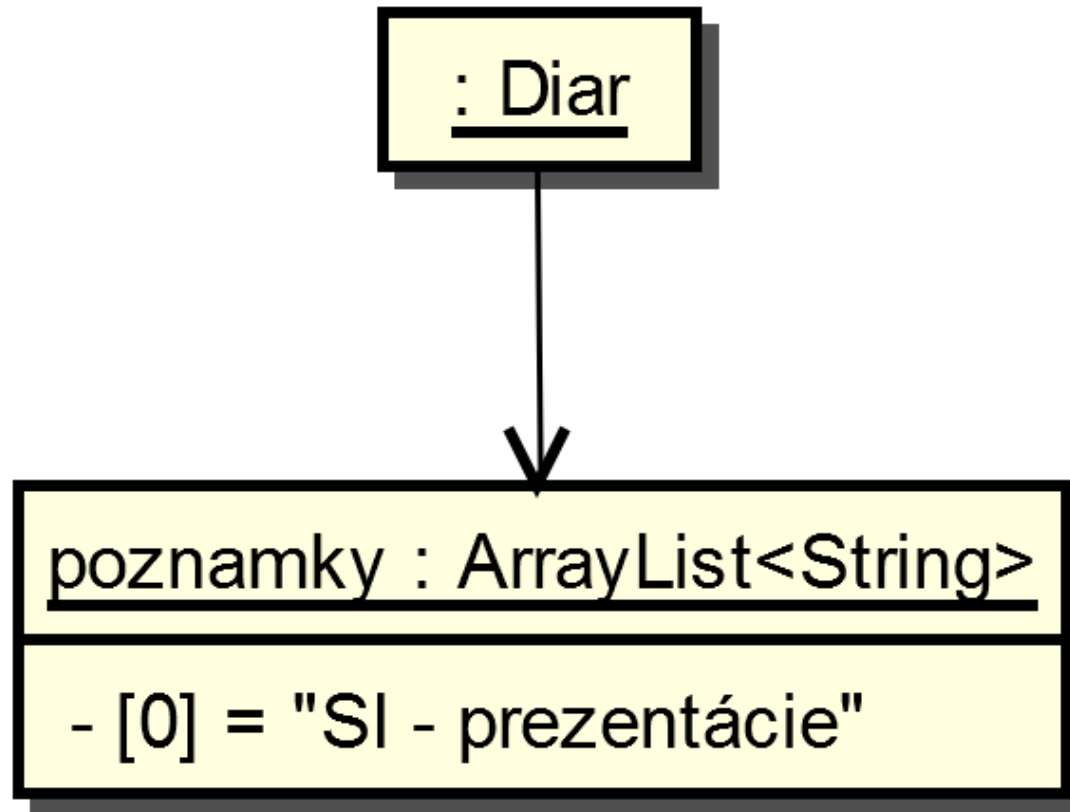
# ArrayList – Java

- `ArrayList<TypPrvkov>`
  - `TypPrvkov` – ľubovoľný objektový typ
- kontajner pre zvolený typ prvkov
- môže obsahovať ľubovoľný počet prvkov
- ktorýkoľvek prvok sa dá vymazať
- ku prvkom sa dá pristupovať pomocou poradového čísla – indexu
  - číslovanie začína od 0

# Prázdný diár



# Diár s jednou poznámkou



# Diár so 4 poznámkami

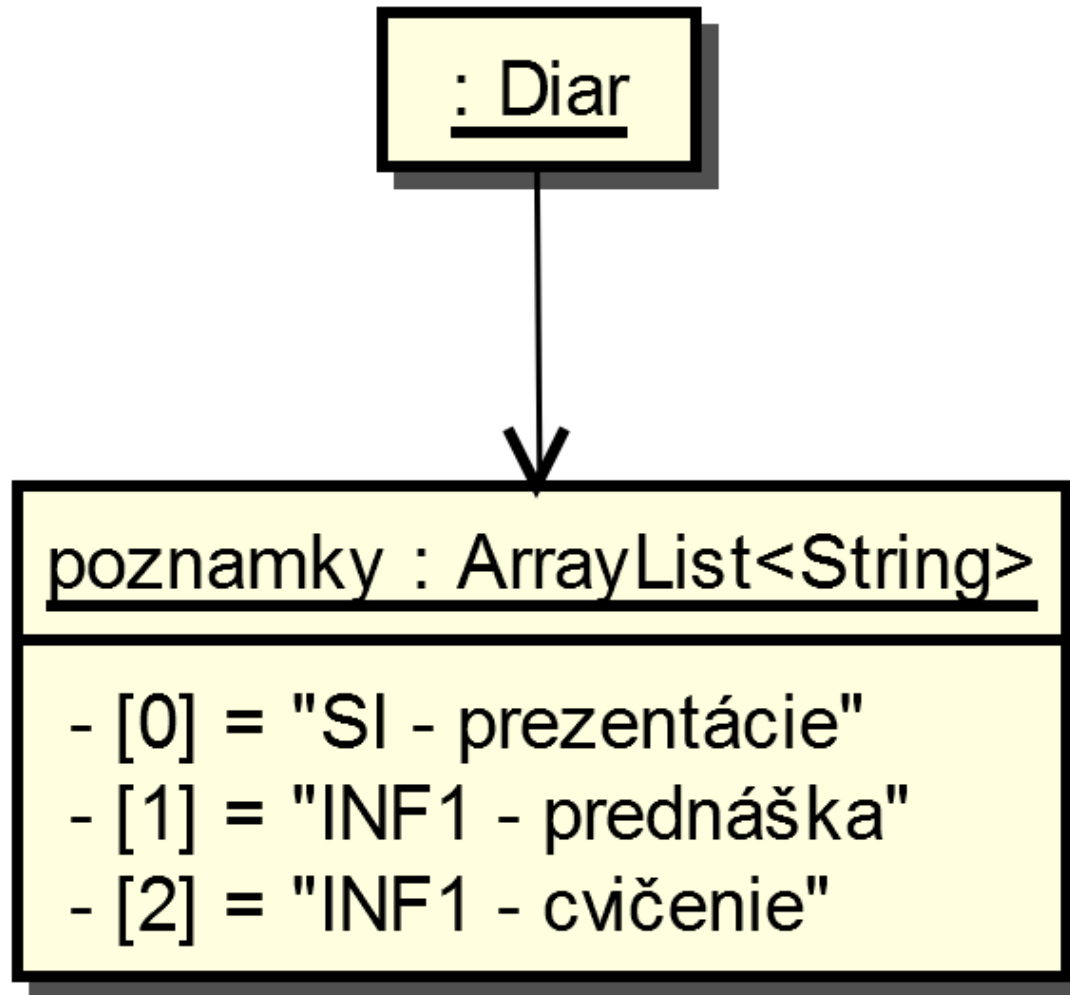
: Diar



poznamky : ArrayList<String>

- [0] = "SI - prezentácie"
- [1] = "INF1 - prednáška"
- [2] = "nákup - chlieb"
- [3] = "INF1 - cvičenie"

# Diár po vymazaní tretej poznámky



# ArrayList<String> – rozhranie

ArrayList<String>

- + new(): ArrayList<String>
- + add(prvok: String): void
- + get(index: int): String
- + remove(index: int): String
- + size(): int

# Generické triedy – Java

- ArrayList je jedna z generických tried

NazovTriedy<zoznamParametrov>

- parametrami musia byť objektové typy
  - typové parametre

# Generické triedy – Java

- deklarácia premennej:

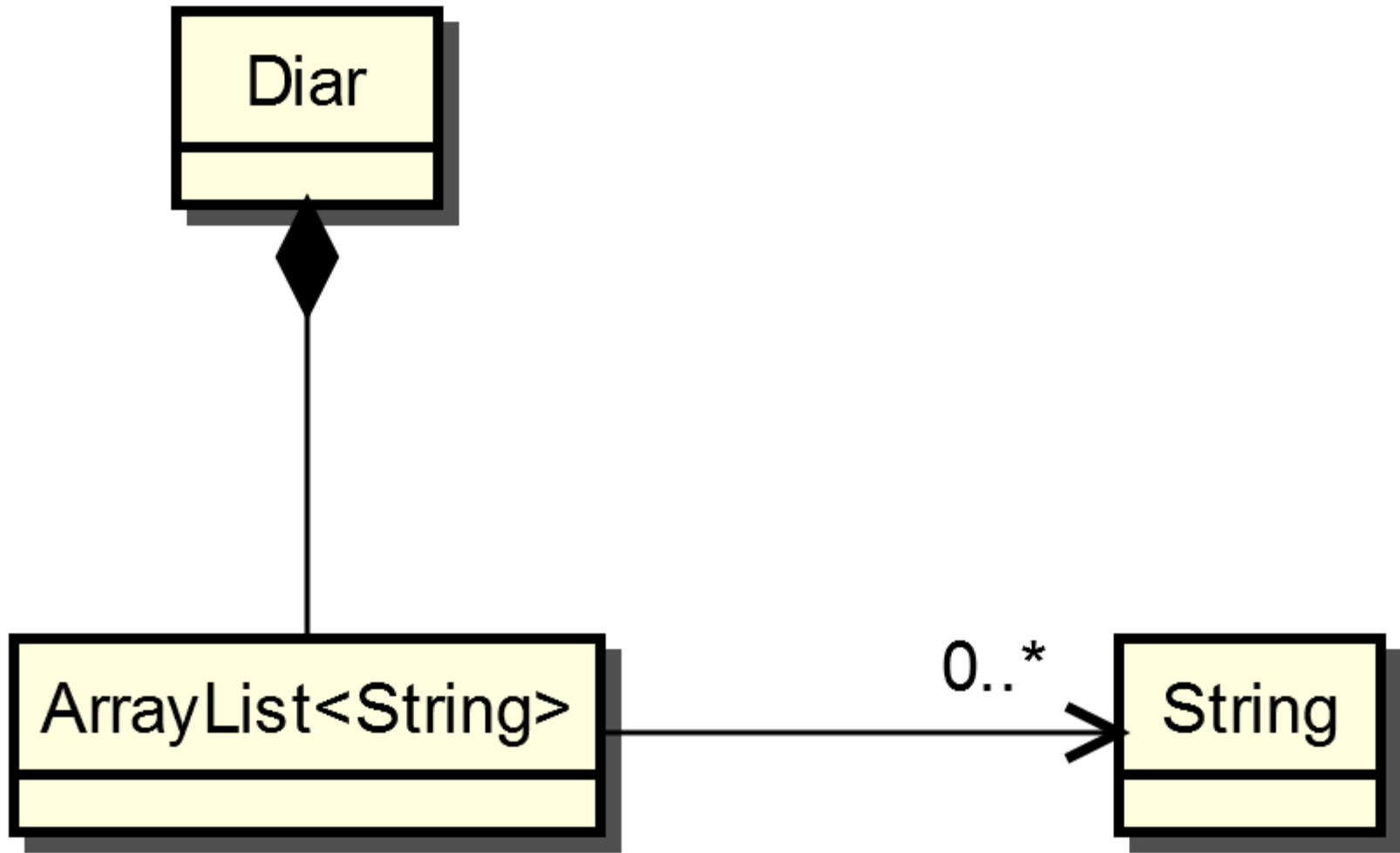
```
NazovTriedy<zoznamParametrov> premenna;
```

- vytvorenie inštancie:

```
new NazovTriedy<zoznamParametrov>  
                                (parametreKonstruktora)
```



# Diar – diagram tried



# Diár – vnútorný pohľad

## Diár

- poznamky: `ArrayList<String>`

- + `Diar()`
- + `vlozPoznamku(poznamka: String): void`
- + `vypisPoznamku(poradoveCislo: int): void`
- + `zmazPoznamku(poradoveCislo: int): void`
- + `getPocetPoznamok(): int`

# Diár – definícia triedy, atribút

```
public class Diar {  
    private ArrayList<String> poznamky;  
  
    ...  
}
```

# Diár – konštruktor

```
public Diar() {  
    this.poznamky = new ArrayList<String>();  
}
```

# Diár – vloženie poznámky

```
public void vložPoznamku(String poznamka) {  
    this.poznamky.add(poznamka);  
}
```

# Diár – zobrazenie poznámky

```
public void vypisPoznamku(int porCislo) {  
    int pocet = this.poznamky.size();  
    if (porCislo >= 0 && porCislo < pocet) {  
        System.out.println(this.poznamky.get(porCislo));  
    }  
}
```

# Diár – počet poznámok

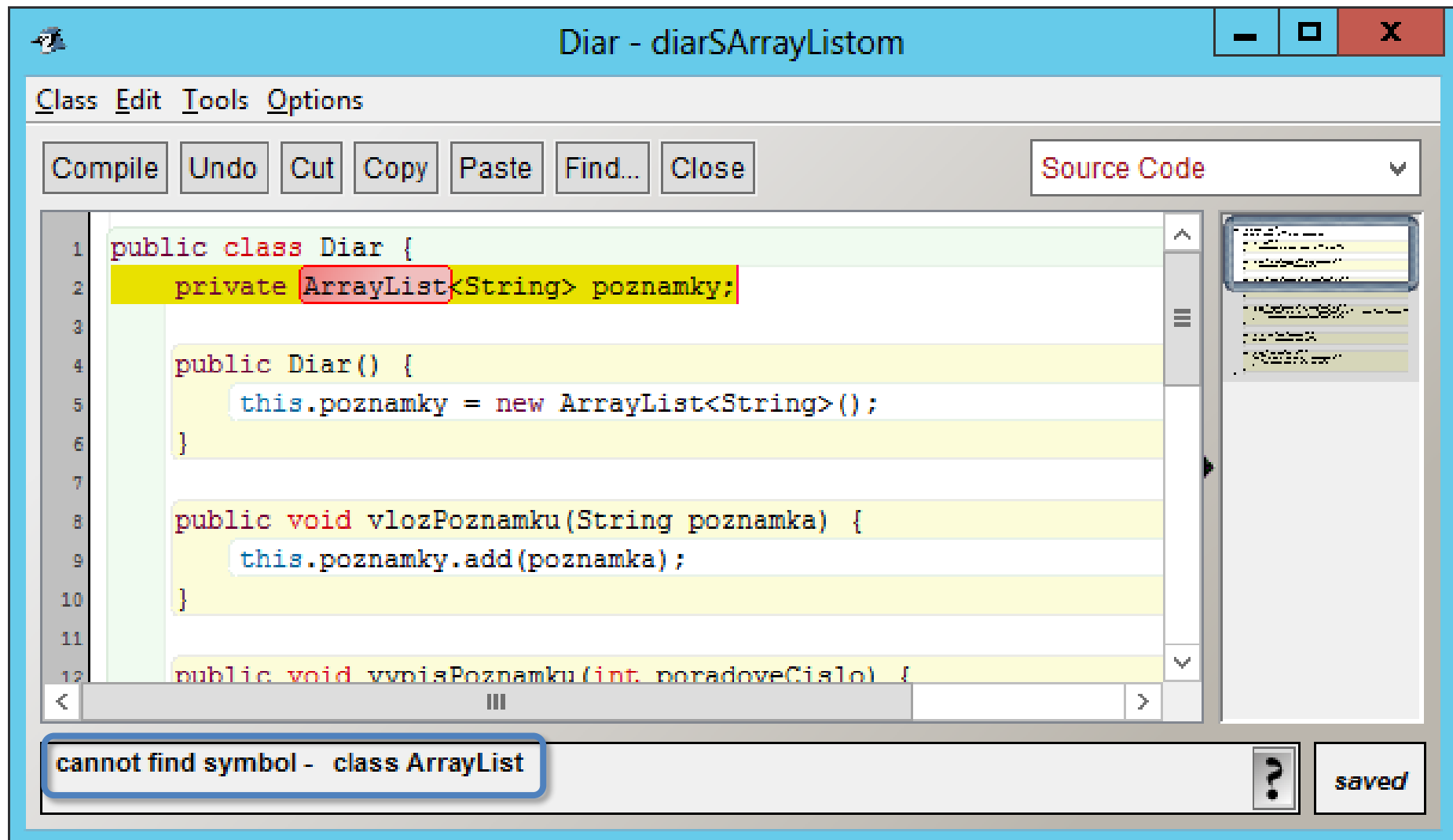
```
public int getPocetPoznamok() {  
    return this.poznamky.size();  
}
```

# Diár – zmazanie poznámky

```
public void zmazPoznamku(int porCislo) {  
    int pocet = this.poznamky.size();  
    if (porCislo >= 0 && porCislo < pocet) {  
        this.poznamky.remove(porCislo);  
    }  
}
```



# Chyba pri preklade



# Diár – príkaz import

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

```
public class Diar {  
    private ArrayList<String> poznamky;  
  
    ...  
}
```

# Používanie knižníc<sub>(1)</sub>

- knižnice – rozširujúca funkčnosť
  - delí sa na balíčky
  - balíčky obsahujú triedy
  - poznáme len rozhrania
- štandardná knižnica – súčasť jazyka Java
  - String – balíček java.lang
  - ArrayList – balíček java.util

# Používanie knižníc<sub>(2)</sub>

- príkaz import:

```
import nazovBalicka.NazovTriedy;
```



- príklad:

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

- pre triedy z balíčka java.lang netreba import

# Zobrazenie všetkých poznámok

- dokážeme zobrazit' ktorúkoľvek poznámku
- ďalšia požiadavka:
  - výpis všetkých poznámok do okna terminálu
- výpis musíme teda postupne zopakovať pre každú poznámku v kontajneri

# Diár – zobraz všetky poznámky

```
public void zobrazVsetko() {  
    System.out.println(this.poznamky.get(0));  
    System.out.println(this.poznamky.get(1));  
    System.out.println(this.poznamky.get(2));  
    System.out.println(this.poznamky.get(3));  
}
```

- zopakovanie časti algoritmu podľa zadanych pravidiel
  - vypísanie všetkých poznámok v diári
  - sčítanie čísel od 1 po dané číslo
  - vyhľadanie knihy v knižnici
- rôzne typy cyklov – rôzne pravidlá
- pravidlá sa vyhodnocujú počas vykonávania algoritmu procesorom

# Cyklus foreach

- jeden z cyklov
- pravidlo:
  - vykonaj pre každý prvok kontajnera

```
for (TypPrvku prvok : kontajner) {  
    // telo cyklu  
}
```

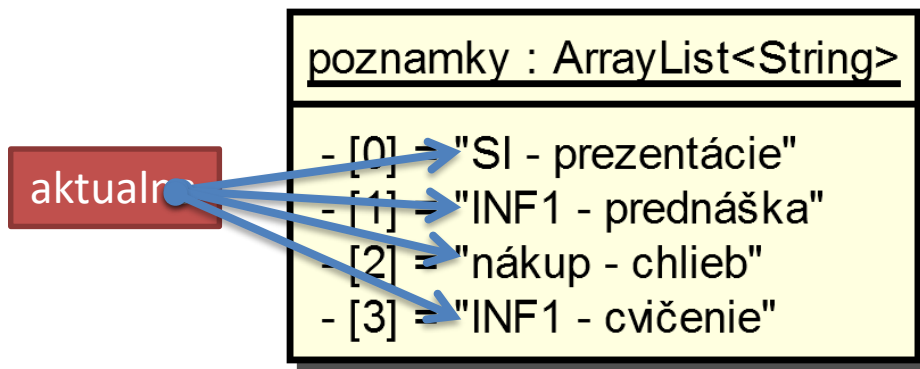
- !!! kontajner sa nesmie v tele cyklu meniť



# Diár – výpis všetkých poznámok

```
public void zobrazVsetko() {  
    for (String aktualna : this.poznamky) {  
        System.out.println(aktualna);  
    }  
}
```

# Cyklus foreach – vykonanie



```
BlueJ: Terminal Window - ...
```

```
Options
```

```
SI - prezentácie
INF1 - prednáška
nákup - chlieb
INF1 - cvičenie
```

```
for (String aktualna : this.poznamky) {  
    System.out.println(aktualna);  
}
```

# Rozšírenie možností diára

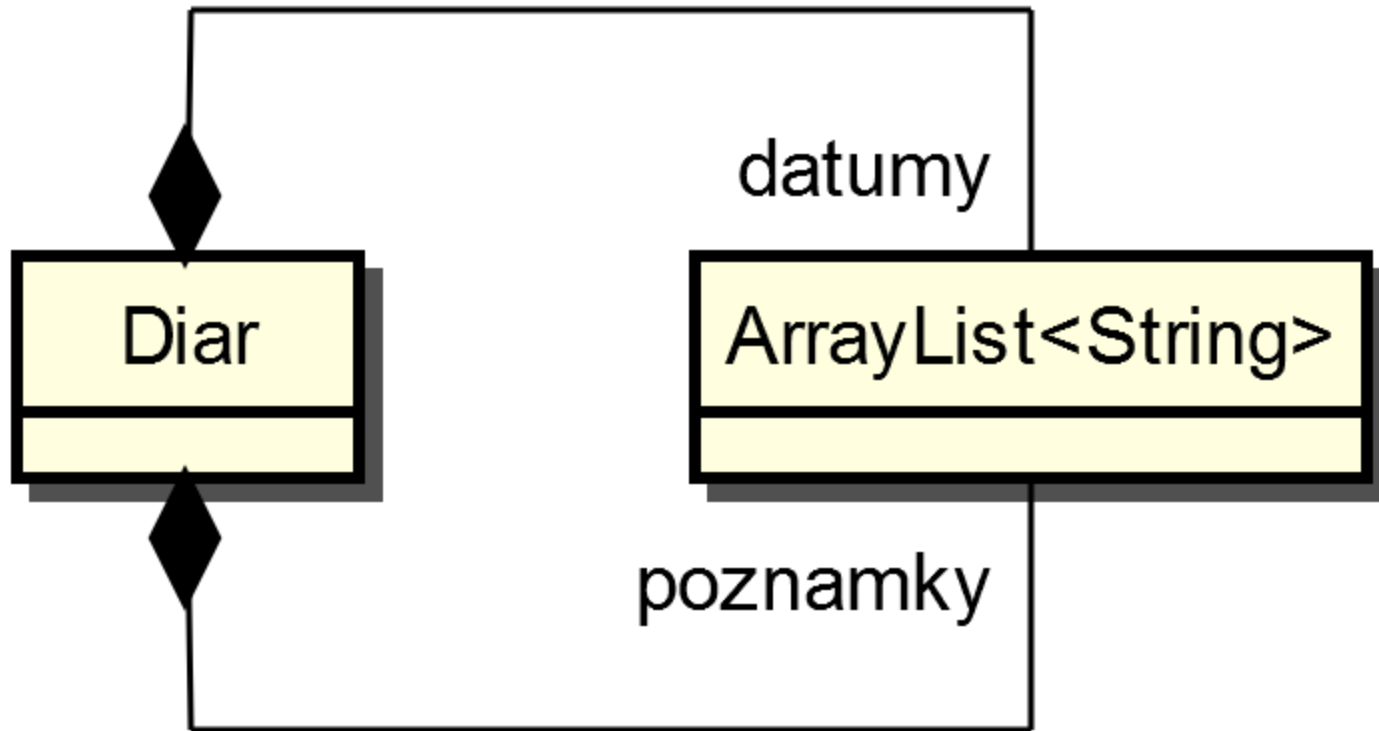
- aktuálna verzia – poznámka = len text
- požiadavka – ukladanie dátumu v poznámke

# Nový diár - rozhranie

## Diar

- + new(): Diar
- + vlozPoznamku(datum: String, poznamka: String): void
- + vypisPoznamku(poradoveCislo: int): void
- + zmazPoznamku(poradoveCislo: int): void
- + getPocetPoznamok(): int
- + zobrazVsetko(): void

# Nový diár – možné riešenie



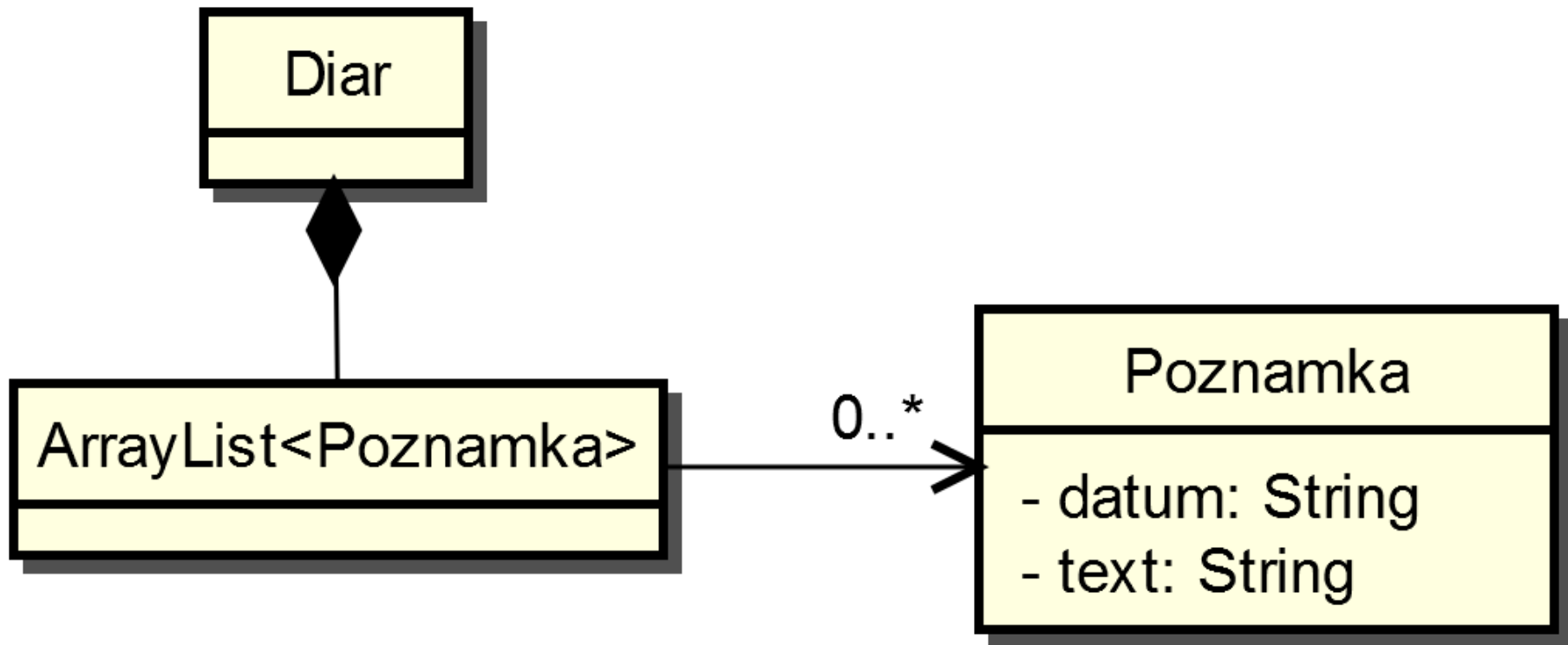
# Nový diár – problémy

- samostatné dátumy + samostatné poznámky
- nie je možné jednoducho spárovať dátum a poznámku
- komplikované vypisovanie dátumov a poznámok
- ...

# Riešenie – nová trieda Poznamka

- iba jeden zoznam – zoznam poznámok
- každá poznámka obsahuje
  - dátum
  - text

# Nový diár – lepšie riešenie





# Nový diár – vnútorný pohľad

## Diar

- poznamky: `ArrayList<Poznamka>`

- + `Diar()`
- + `vlozPoznamku(datum: String, poznamka: String): void`
- + `vypisPoznamku(poradoveCislo: int): void`
- + `zmazPoznamku(poradoveCislo: int): void`
- + `getPocetPoznamok(): int`
- + `zobrazVsetko(): void`

# ArrayList<Poznamka> – rozhranie

ArrayList<Poznamka>

- + new(): ArrayList<Poznamka>
- + add(prvok: Poznamka): void
- + get(index: int): Poznamka
- + remove(index: int): Poznamka
- + size(): int

# Trieda Poznamka – rozhranie

## Poznamka

- + new(datum: String, text: String): Poznamka
- + getDate(): String
- + getText(): String
- + zobrazPoznamku(): void

# Nový diár – definícia triedy, atribút

```
public class Diar {  
    private ArrayList<Poznamka> poznamky;  
  
    ...  
}
```

# Nový diár – konštruktor

```
public Diar() {  
    this.poznamky = new ArrayList<Poznamka>();  
}
```

# Nový diár – vloženie poznámky

```
public void vložPoznamku(String datum, String text) {  
    Poznamka nova = new Poznamka(datum, text);  
    this.poznamky.add(poznamka);  
}
```

# Nový diár – vloženie poznámky

```
public void vložPoznamku(String datum, String text) {  
    this.poznamky.add(new Poznamka(datum, text));  
}
```

anonymný objekt



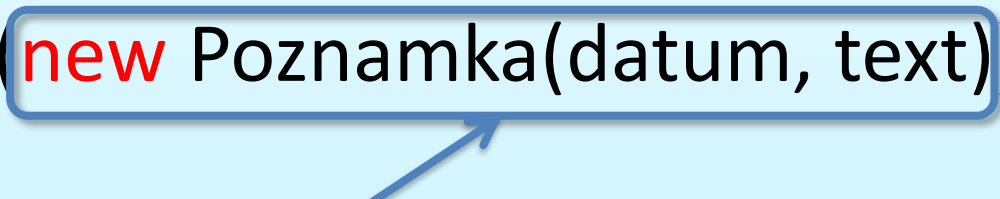
# Anonymný objekt

- anonymná inštancia
  - nemá názov  $\Rightarrow$  nie je priradená do premennej
- Použitie:
  - ako parameter správy
  - ako adresát správy



# Anonymný objekt

```
public void vložPoznamku(String datum, String text) {  
    this.poznamky.add(new Poznamka(datum, text));  
}
```



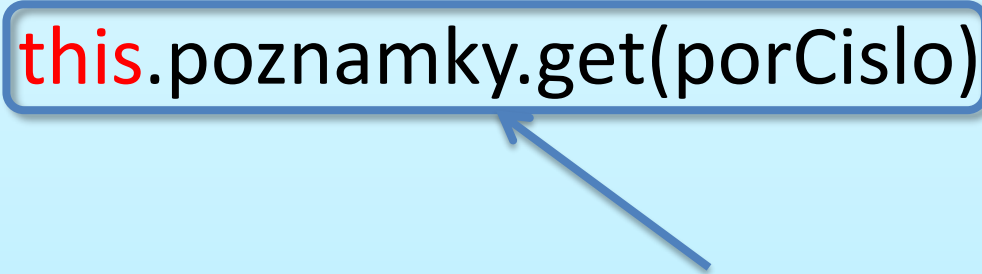
anonymný objekt ako parameter správy

# Nový diár – zobrazenie poznámky

```
public void vypisPoznamku(int porCislo) {  
    int pocet = this.poznamky.size();  
    if (porCislo >= 0 && porCislo < pocet) {  
        this.poznamky.get(porCislo).zobrazPoznamku();  
    }  
}
```

# Anonymný objekt

```
public void vypisPoznamku(int porCislo) {  
    int pocet = this.poznamky.size();  
    if (porCislo >= 0 && porCislo < pocet) {  
        this.poznamky.get(porCislo).zobrazPoznamku();  
    }  
}
```



anonymný objekt ako adresát správy

# Nový diár – výpis všetkých poznámok

```
public void zobrazVsetko() {  
    for (Poznamka aktualna : this.poznamky) {  
        aktualna.zobrazPoznamku();  
    }  
}
```

# Ďalšia požiadavka

- teraz môžeme škrtiť záznamy po jednom
- vymazanie celého obsahu diára
  - vytrhnutie popísaných strán
  - práca pokračuje s prázdny diárom

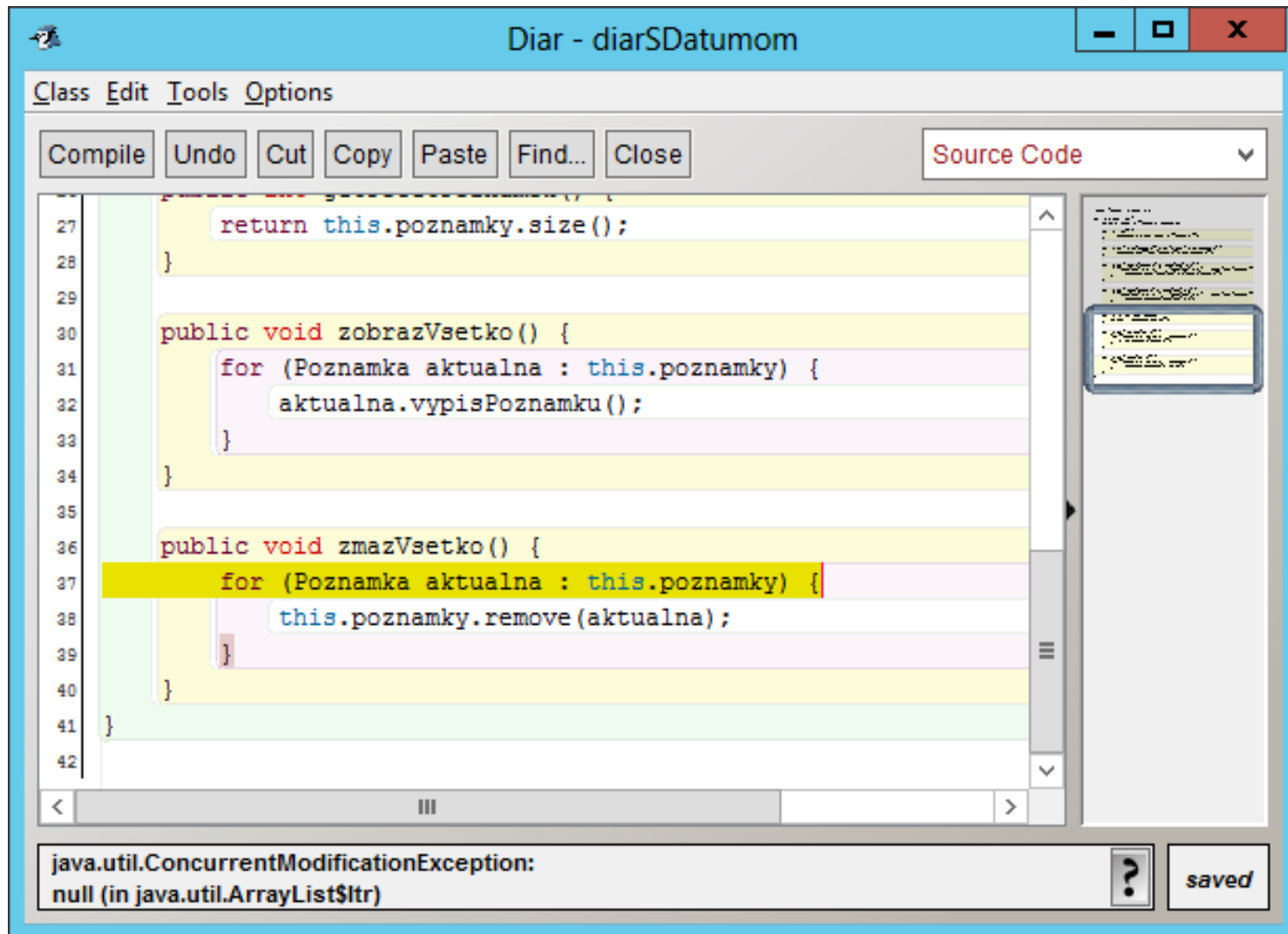
# Postup

- pre všetky položky diára (poznámky):
  - vymaž položku zo zoznamu

# Nový diár – zmaž všetky poznámky

```
public void zmazVsetko() {  
    for (Poznamka aktualna : this.poznamky) {  
        this.poznamky.remove(aktualna);  
    }  
}
```

# Chyba pri mazaní



kontajner sa nesmie v tele cyklu meniť



# Cyklus while

- Další z cyklov
- Pravidlo:
  - vykonávaj telo cyklu kým platí podmienka

```
while (podmienka) {  
    // telo cyklu  
}
```

# Nový diár – zmaž všetky poznámky

```
public void zmazVsetko()
{
    while (!this.poznamky.isEmpty()) {
        this.poznamky.remove(0);
    }
}
```

# Vďaka za pozornosť