Informatika 1

Tvorba tried

doc. Ing. Ján Janech, PhD.
2. 10. 2023

ŽILINSKÁ UNIVERZITA V ŽILINE
Fakulta riadenia
a informatiky

Pojmy zavedené v 1. prednáške (1)

- objekt
  - verejná časť
    - rozhranie – správy
  - neverejná časť
    - atribúty – stav objektu
    - metódy – chovanie objektu

Pojmy zavedené v 1. prednáške (2)

- delenie objektov
  - trieda
  - inštancia

- životný cyklus inštancie
  - vznik – správa triede, začiatočný stav
  - život objektu – poskytovanie služieb
  - zánik – garbage collector, „recyklácia“

Pojmy zavedené v 1. prednáške (3)

- správa
  - adresát
  - selektor – pomenovanie správy
  - parametre
  - návratová hodnota
  - príklad správy
    - kruhModry.posunVodorovne(100)

Cieľ prednášky

- životný cyklus triedy
- tvorba triedy v jazyku Java
- základné príkazy jazyka Java

- Príklad: automat na cestovné lístky

Objekt, trieda a inštancia

Trieda

Inštancia

Inštancia

Vytváranie tried – metatrieda

- „továreň“ na objekty – trieda
- trieda je tiež objekt

- Aká „továreň“ vyrobí triedu?
- Jedno riešenie – metatrieda, továreň na triedy.
  - Smalltalk, Python

Priamo definovaný objekt

- vytvorenie objektu bez triedy, jednorazovo
- celý objekt priamo nadefinujeme

- priamo definovaný objekt.

Delenie programovacích jazykov

- trieda ako inštancia metatriedy
  - Smalltalk, Python, ...
  - čisté objektové jazyky
- trieda ako priamo definovaný objekt
  - (Object) Pascal, C++, C#, Java, ...
  - hybridné objektové jazyky
  - najviac jazykov
- prototypy objektov
  - JavaScript, Self, ...
  - čisté objektové jazyky
  - neexistuje trieda

Trieda ako šablóna

- súhrn informácií na vytvorenie inštancie
  - rozhranie
  - atribúty
  - metódy

Postup pri vytváraní tried

- abstrakcia (zjednodušenie) reálneho objektu
  - slovná – textová charakteristika inštancií triedy
- grafické znázornenie pomocou diagramu tried v UML – UML .FRI
- definícia triedy v programovacom jazyku Java
  - zdrojový kód – v nástroji Editor BlueJ

Automat na cestovné lístky

Automat na cestovné lístky – služby

- prvá verzia – primitívny automat

- lístky majú jednu cenu (napr. jednopásmové lístky na MHD v Žiline)
- automat prijíma peniaze (mince)
- automat zobrazuje vloženú čiastku – súčet vhodených mincí pred vydaním lístka
- automat tlačí lístok

AutomatMHD – rozhranie triedy

- správa – žiadosť o vytvorenie automatu
  - AutomatMHD.vytvorAutomat(cenaListka)
  - AutomatMHD.new(cenaListka)

AutomatMHD – rozhranie inštancie

- zobrazenie ceny lístka
  - automat.getCenaListka()
- zobrazenie doteraz vloženej čiastky
  - automat.getVlozenaCiastka()
- vloženie mince
  - automat.vlozMincu(hodnotaMince)
- vytlačenie lístka
  - automat.tlacListok()

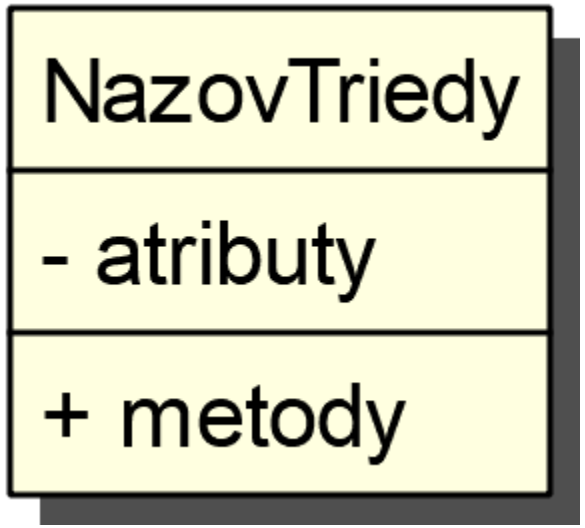
AutomatMHD – vlastnosti

- cena lístka v centoch
  - celé číslo
- aktuálne vložená čiastka v centoch
  - celé číslo
- celková tržba od posledného vybrania peňazí zamestnancom
  - celé číslo

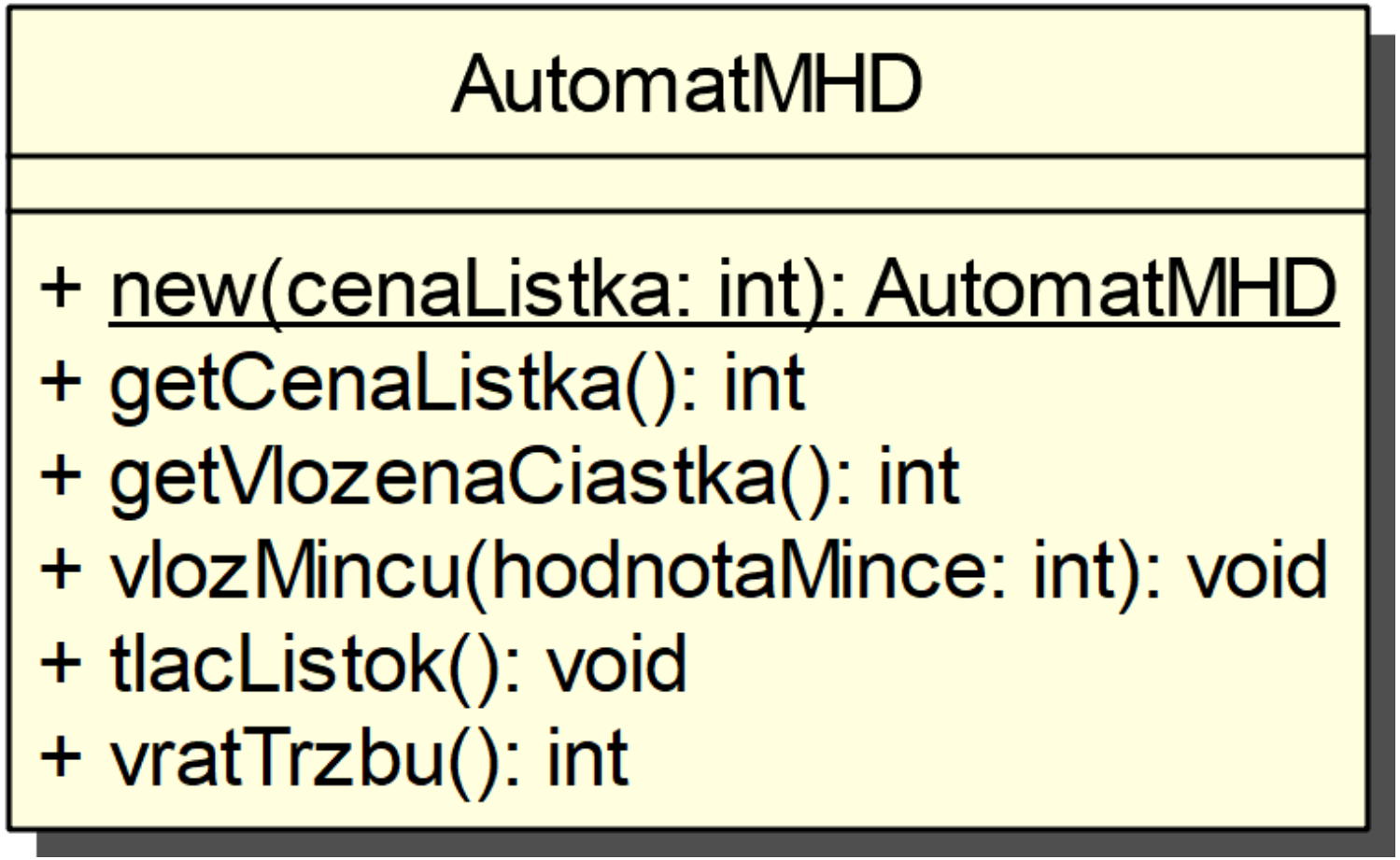
Postup pri vytváraní tried

- abstrakcia (zjednodušenie) reálneho objektu
  - slovná – textová charakteristika inštancií triedy
- grafické znázornenie pomocou diagramu tried v UML – UML .FRI
- definícia triedy v programovacom jazyku Java
  - zdrojový kód – v nástroji Editor BlueJ

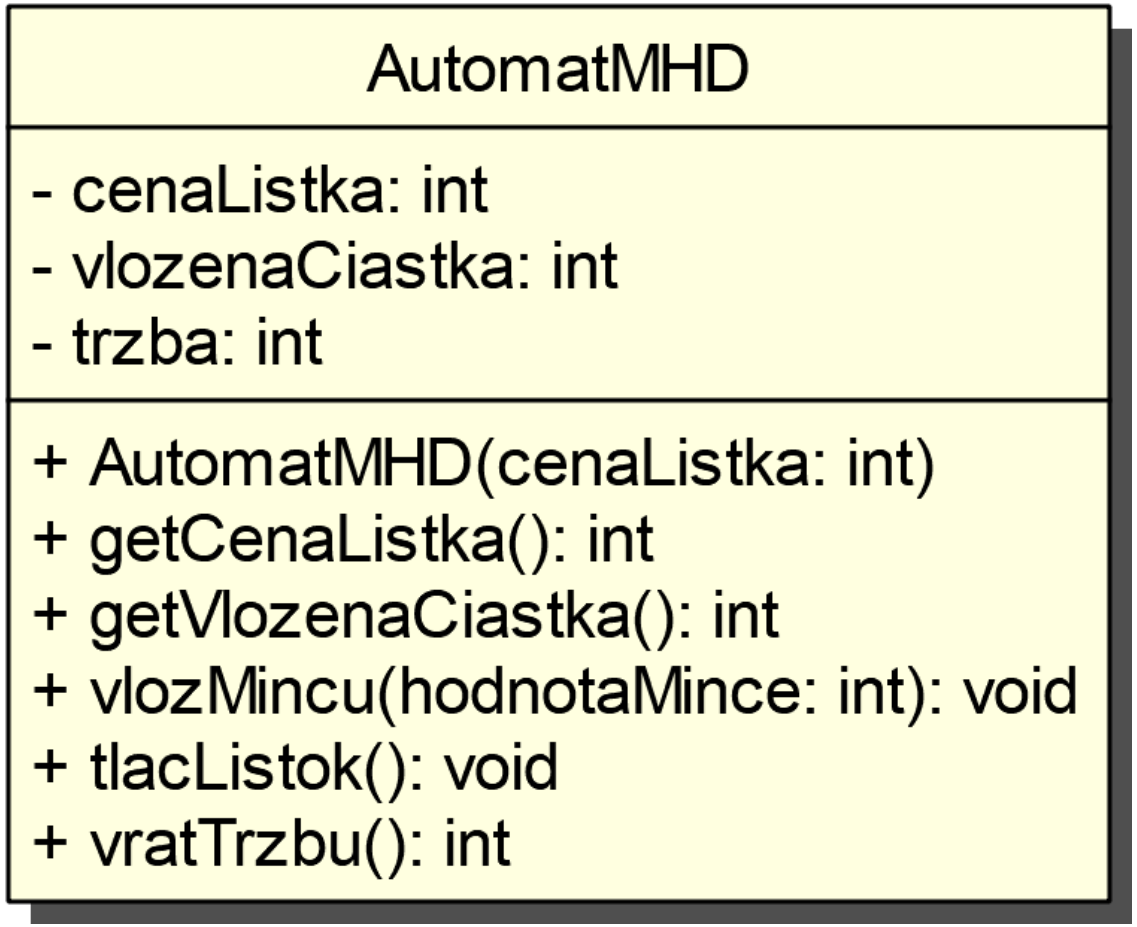
Model triedy v UML



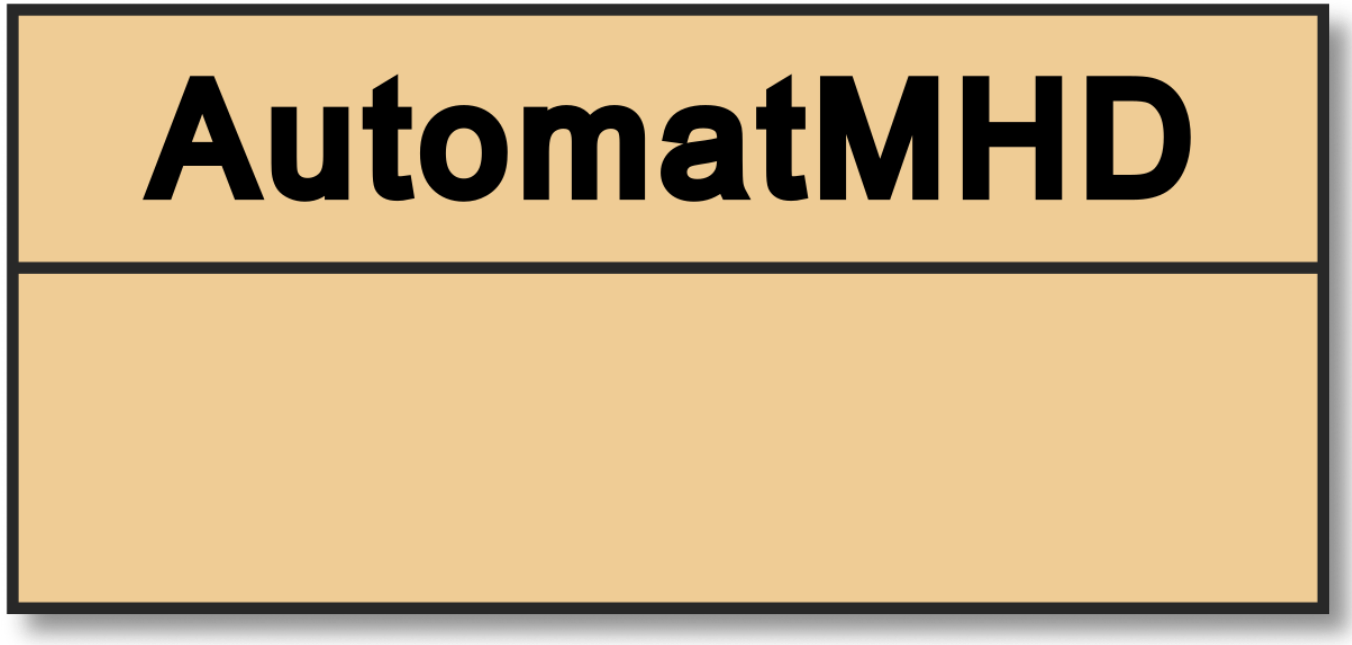
AutomatMHD – vonkajší pohľad



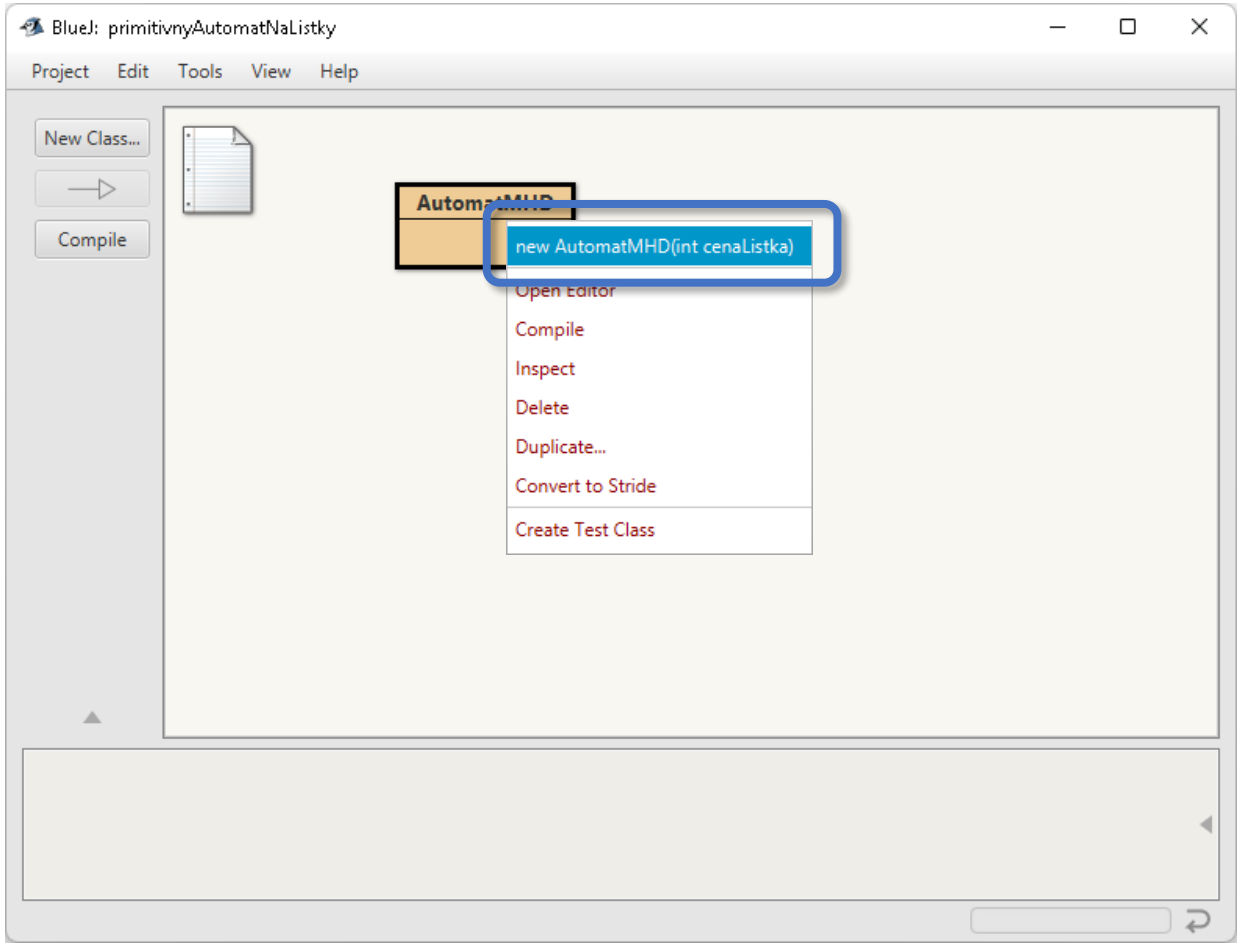
AutomatMHD – vnútorný pohľad



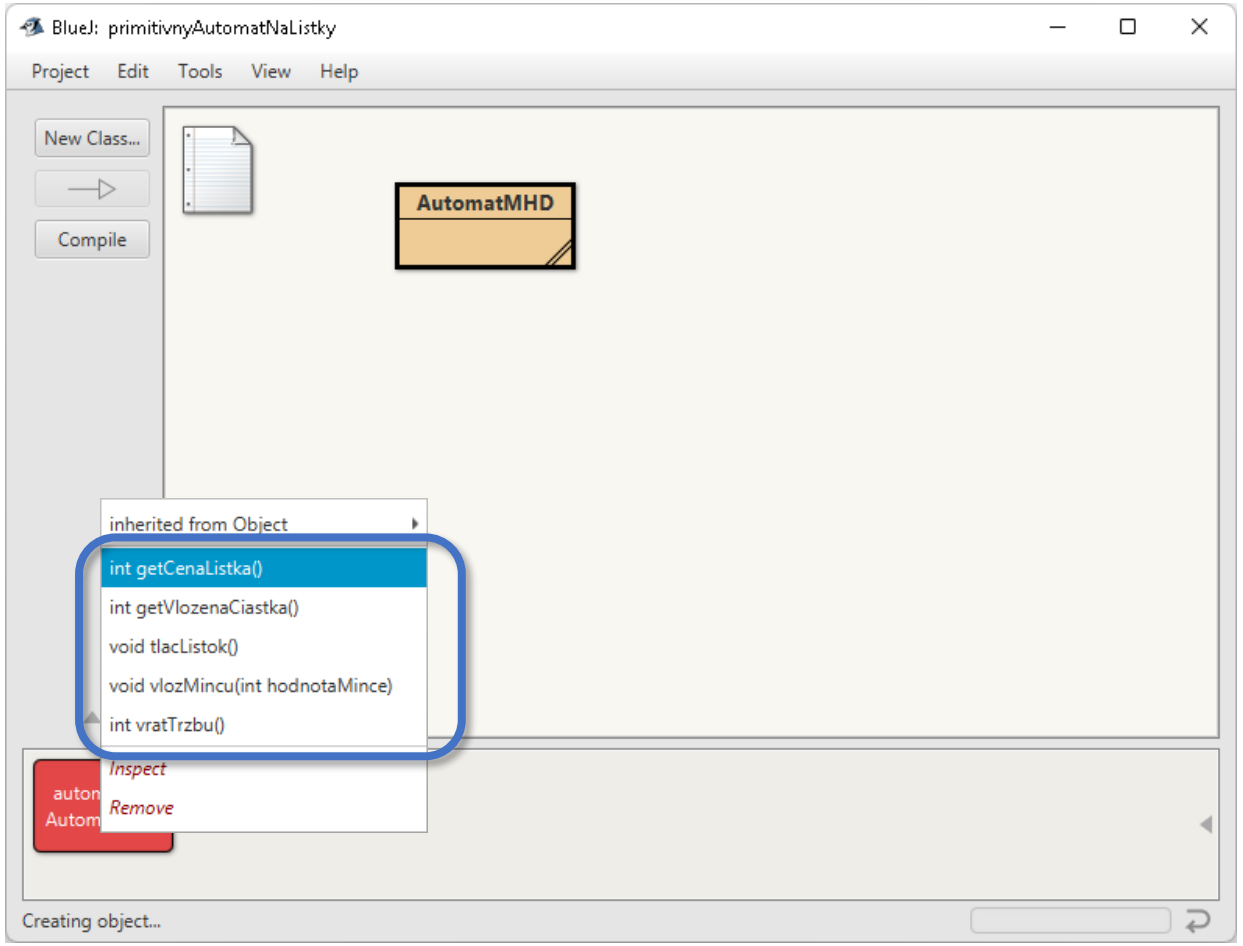
AutomatMHD – BlueJ



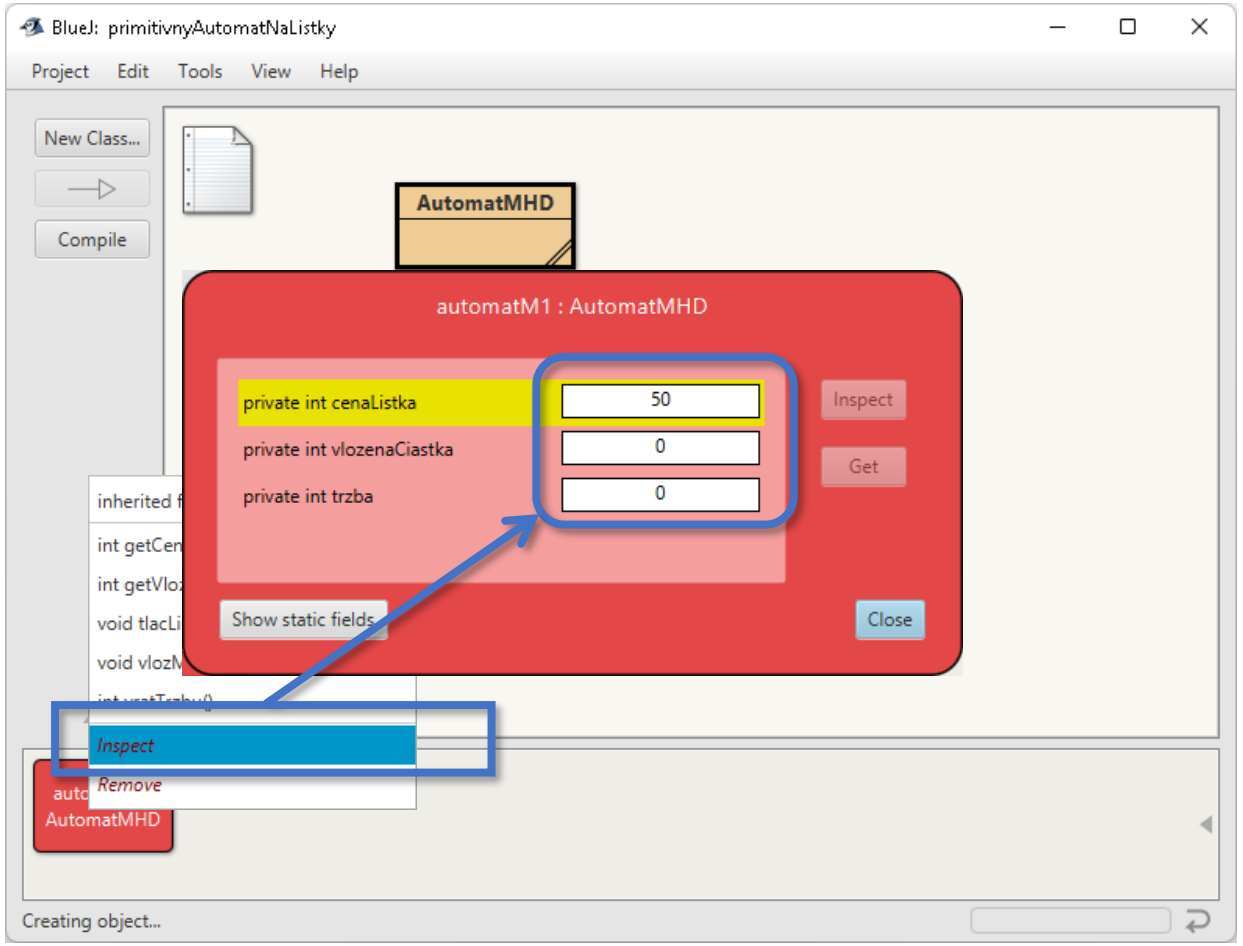
BlueJ – rozhranie triedy



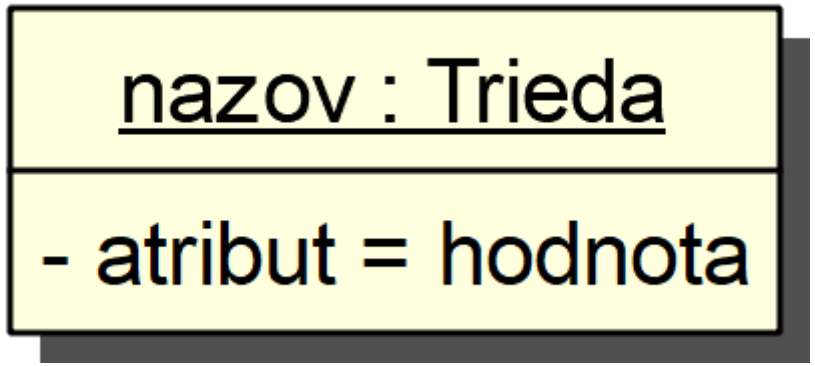
BlueJ – rozhranie inštancie



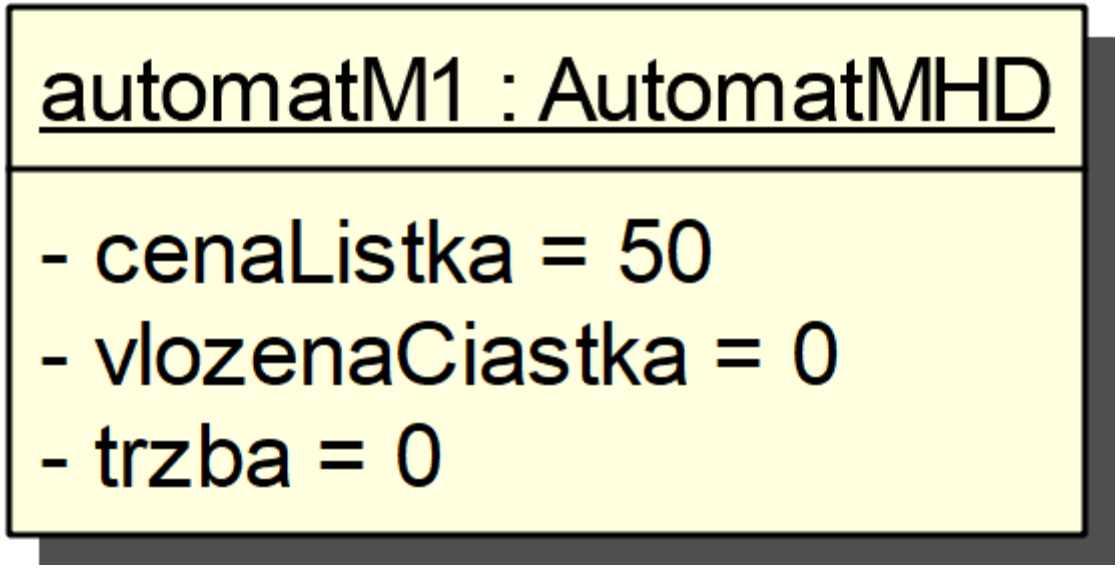
BlueJ – vnútorný pohľad



UML – objektový diagram



Objektový diagram automatu



Postup pri vytváraní tried

- abstrakcia (zjednodušenie) reálneho objektu
  - slovná – textová charakteristika inštancií triedy
- grafické znázornenie pomocou diagramu tried v UML – UML .FRI
- definícia triedy v programovacom jazyku Java
  - zdrojový kód – v nástroji Editor BlueJ

Definícia triedy – Java

```
hlavička triedy {
    telo triedy
}
```

Definícia triedy – hlavička

- príklad:

```
public class AutomatMHD
```

- public – kľúčové slovo, označuje verejný prístup
- class – kľúčové slovo, označuje triedu
- AutomatMHD – identifikátor, názov triedy

Identifikátor

- názov, pomenovanie inštancie, triedy, správy, metódy a atribútu sa nazýva identifikátor

- identifikátor je jedno alebo viacslovné pomenovanie

Návrh identifikátorov (1)

- identifikátory navrhujeme, volíme tak, aby vyjadrovali význam, zmysel pojmu
- nepoužívame neznáme skratky
- programy sú síce určené na to, aby ich vykonávali počítače, ale čítajú ich aj ľudia
- úspešný program sa stále nejako mení

Návrh identifikátorov (2)

- pre tvorbu identifikátorov platia
  - pravidlá – určuje programovací jazyk
  - konvencie – určuje programátorská komunita

Pravidlá tvorby identifikátorov

- programovací jazyk definuje pravidlá, ktoré musí spĺňať každý identifikátor
- Java definuje tieto pravidlá
  - môže sa skladať z písmen, číslic a znakov „_“ (podčiarkovník) a „$“ (dolár)
  - nesmie začínať číslicou
  - rozlišujú sa malé a VEĽKÉ písmená
  - nesmie sa zhodovať so žiadnym kľúčovým slovom
    - kľúčové slovo – slovo alebo identifikátor so špecifickým významom v programovacom jazyku

Konvencie pre tvorbu identifikátorov

- jednotlivé slová sa píšu bez medzier
- prvé slovo sa píše malým písmenom
- druhé a ďalšie slová začínajú veľkým písmenom
- výnimka: identifikátor triedy vždy začína veľkým písmenom

- príklady
  - trieda: AutomatMHD
  - metóda: tlacListok

Definícia triedy – telo

- definície atribútov
- definície konštruktorov
- definície metód

- poradie je konvenciou

Definícia triedy – atribúty

- príklad:

```
private int cenaListka;
private int vlozenaCiastka;
private int trzba;
```

- private – kľúčové slovo, označuje súkromnú zložku inštancie triedy
- int – typ, určuje druh uchovávanej informácie
- cenaListka – identifikátor, názov atribútu
- ; (bodkočiarka) – ukončenie definície atribútu

Typ – dátový typ

- typ určuje druh dát
  - príklad: int – označuje informáciu, ktorá je vyjadrená celým číslom
- dátové typy v jazyku Java – primitívne a objektové

Primitívne dátové typy – pre celé čísla

- celočíselné typy – pre rôzne konečné podmnožiny množiny celých čísel
  - byte $<-2^7, 2^7 - 1>$, 2^7 = 128
  - short $<-2^{15}, 2^{15} - 1>$, 2^{15} = 32.768
  - int $<-2^{31}, 2^{31} - 1>$, 2^{31} = 2.147.483.648
  - long $<-2^{63}, 2^{63} - 1>$, 2^{63} = 9.223.372.036.854.775.808

Primitívne dátové typy – pre reálne čísla

- pohyblivá rádová čiarka – pre rôzne konečné podmnožiny racionálnych čísel
  - float – (±) ~1,4 × 10^{-45} – (±) ~3,4 × 10^{38}
    – ±1.4e-45 – ±3.4e38
  - double – (±) ~4,9 × 10^{-324} – (±) ~1,8 × 10^{308}
    – ±4.9e-324 – ±1.8e308

Obmedzenia číselných typov

- aritmetické operácie sa môžu správať zvláštne

- $a = 2^{31} - 1$
  $a + a = -2$
- $12 \times 5 \div 10 = 6$
  $12 \div 10 \times 5 = 5$
- $b = 0.1$
  $b + b + b + b + b + b + b + b + b + b = ?$

10x

Primitívne dátové typy – ostatné

- boolean – typ pre logické hodnoty, môže nadobúdať jednu z dvoch hodnôt
  - true – pravda, áno, ...
  - false – nepravda, nie, ...

- char – typ pre jeden znak z množiny znakov Unicode; písanie textov zvyčajným spôsobom v rôznych jazykoch
  - môže obsahovať ľubovoľný znak z 65,535 znakov Unicode

Objektový typ String

- String – reťazec znakov
  - obsahuje ľubovoľnú postupnosť znakov unicode

Definícia triedy – konštruktor

```
hlavička konštruktora {
  telo konštruktora
}
```

Definícia triedy – hlavička konštruktora

- príklad:

```
public AutomatMHD(int cenaListka)
```

- public – kľúčové slovo, označuje verejnú zložku triedy => rozhranie triedy obsahuje správu – žiadosť o vytvorenie inštancie
- AutomatMHD – identifikátor, názov konštruktora – musí byť zhodný s názvom triedy
- (int cenaListka) – zoznam formálnych parametrov konštruktora

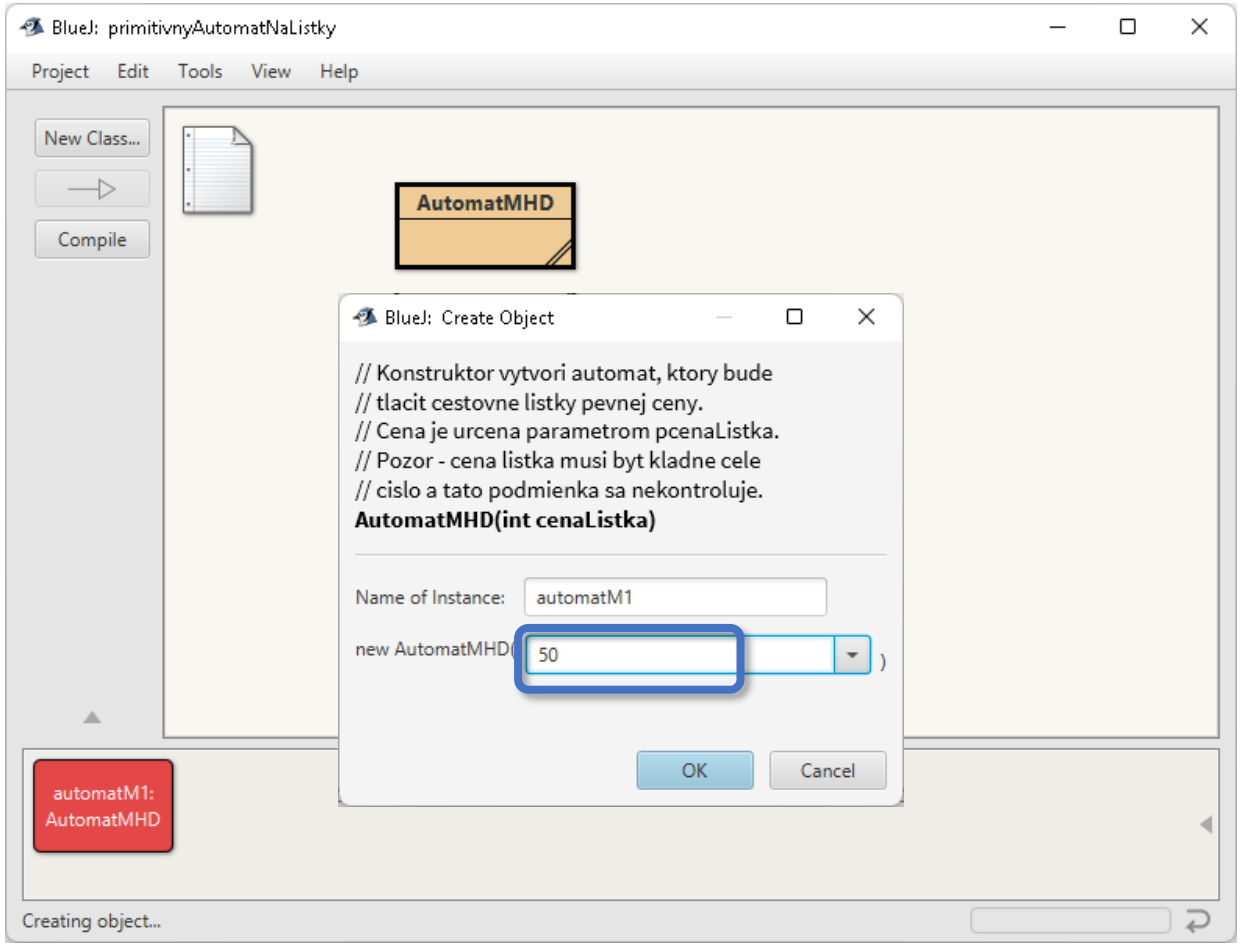
Formálne parametre

- konštruktory a metódy
- príklad: jednoprvkový zoznam

```
public AutomatMHD(int cenaListka)
```

- () – ohraničenie zoznamu
- dvojica int cenaListka – parameter
  - int – typ hodnoty prenášanej parametrom
  - cenaListka – identifikátor, názov parametra

Skutočné parametre



Definícia triedy – telo konštruktora

- príklad:

```
this.cenaListka = cenaListka;
this.vlozenaCiastka = 0;
this.trzba = 0;
```

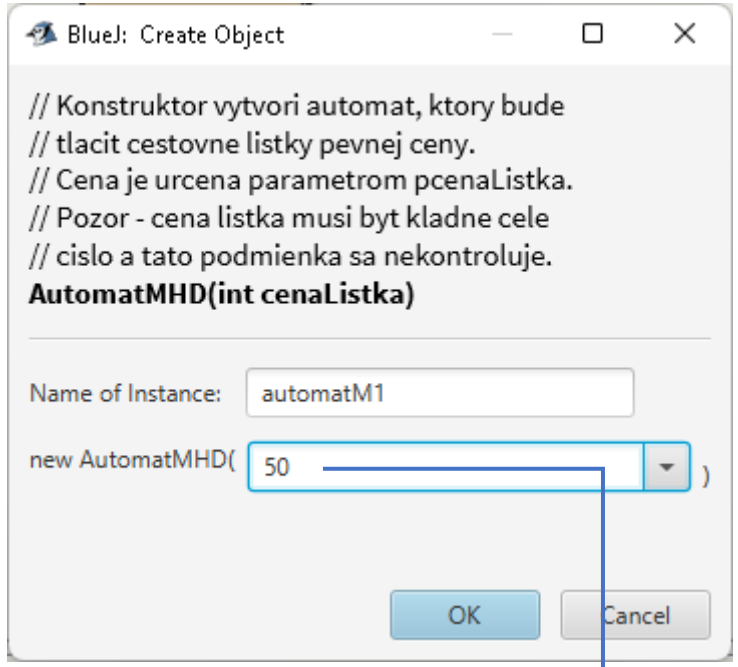
- postupnosť príkazov oddelených bodkočiarkou
- inicializácia inštancie – definícia hodnôt atribútov = uvedenie inštancie do začiatočného stavu

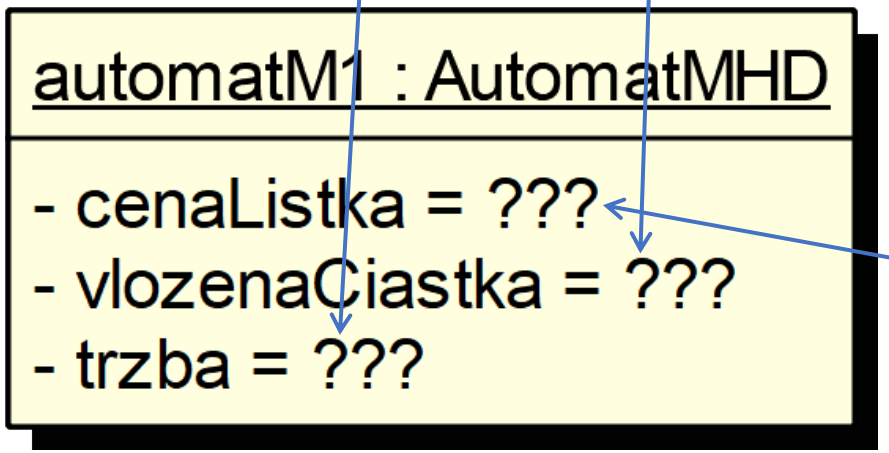
Definícia triedy – konštruktor

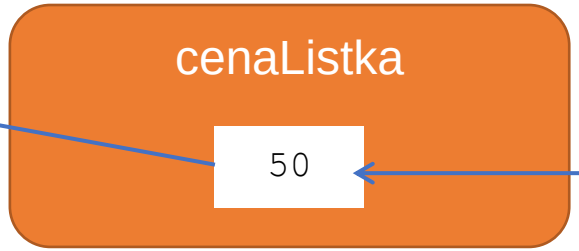
```
public AutomatMHD(int cenaListka) {
    this.cenaListka = cenaListka;
    this.vlozenaCiastka = 0;
    this.trzba = 0;
}
```

Vytvorenie a inicializácia inštancie

```
this.cenaListka = cenaListka;
this.vlozenaCiastka = 0;
this.trzba = 0;
```







Priraďovací príkaz

- príklad:

```
this.trzba = 0;
```

- nie je to rovnica!!!
- this.trzba – ľavá strana priraďovacieho príkazu
- = (rovná sa) – operátor priradenia
- 0 – výraz
- ; (bodkočiarka) – ukončenie príkazu

this

- kľúčové slovo v jazyku Java
- vyjadruje prístup k objektu, ktorý vykonáva aktuálnu metódu
- this.nazovAtributu
  - prístup k atribútu nazovAtributu

Literál

- zápis konštantných hodnôt v jazyku
- priame vyjadrenie hodnoty v zdrojovom kóde
- nepotrebuje žiadny výpočet
- príklady:
  - this.trzba = 0;
  - int i = 100000;
  - boolean vysledok = true;
  - char velkeC = 'C';
  - double d1 = 123.4;
  - String nazov = "Školská linka FRI";

Definícia triedy – metóda

```
hlavička metódy {
   telo metódy
}
```

Definícia triedy – hlavička metódy

- príklad:

```
public int getCenaListka()
```

- public – kľúčové slovo, označuje verejnú zložku inštancie triedy => rozhranie triedy obsahuje správu
- int – typ návratovej hodnoty
- getCenaListka – identifikátor, názov metódy
- () – zoznam formálnych parametrov – môže byť aj prázdny, zátvorky však musia byť

Typ návratovej hodnoty

- príklad:

```
public int getCenaListka()
```

- void – kľúčové slovo, označenie typu návratovej hodnoty, ak metóda neposkytuje žiadnu výstupnú informáciu (správa nemá návratovú hodnotu)

Definícia triedy – telo metódy

- príklad:

```
return this.cenaListka;
```

- Telo metódy sa skladá z postupnosti príkazov oddelených bodkočiarkou rovnako ako telo konštruktora.

Definícia triedy – metóda

```
public int getCenaListka() {
    return this.cenaListka;
}
```

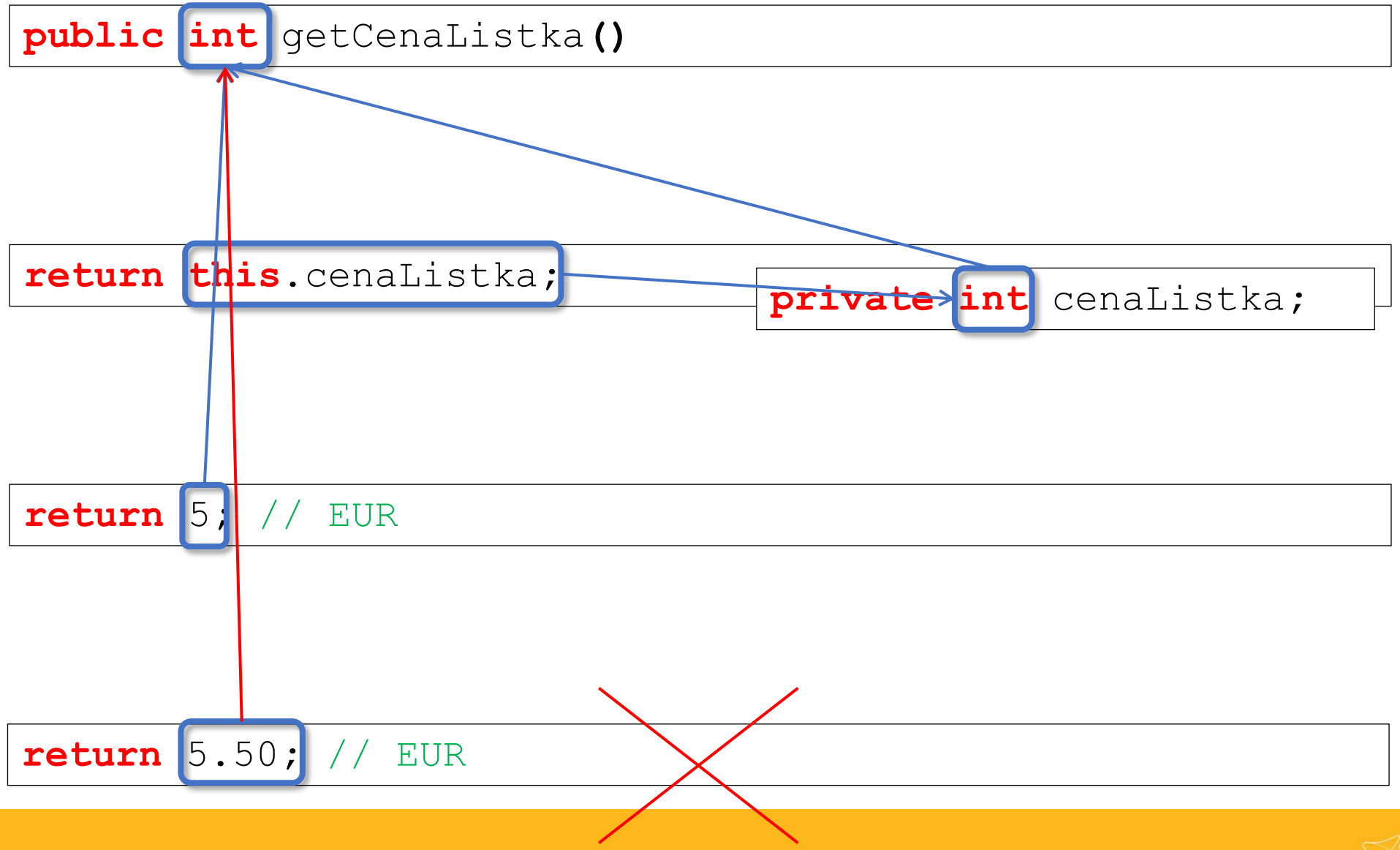
Príkaz návratu/príkaz return

- príklad:

```
return this.cenaListka;
```

- return – kľúčové slovo; uvádza príkaz, ktorým metóda vracia hodnotu
- this.cenaListka – určuje návratovú hodnotu
  - typ musí sedieť s typom návratovej hodnoty metódy
- ; (bodkočiarka) – koniec príkazu
- posledný vykonaný príkaz v tele metódy

Návratový typ a príkaz návratu



Konštruktor a metóda – rozdiely

- konštruktor:
  - identifikátor konštruktora je vždy zhodný s identifikátorom triedy
  - nikdy nemá typ návratovej hodnoty
- metóda:
  - identifikátor metódy je vždy zhodný so selektorom správy
  - má vždy typ návratovej hodnoty
  - môže mať návratovú hodnotu

Metódy a stav objektu (1)

- prístupové – poskytujú informáciu súvisiacu so stavom objektu

```
public int getCenaListka() {
    return this.cenaListka;
}
```

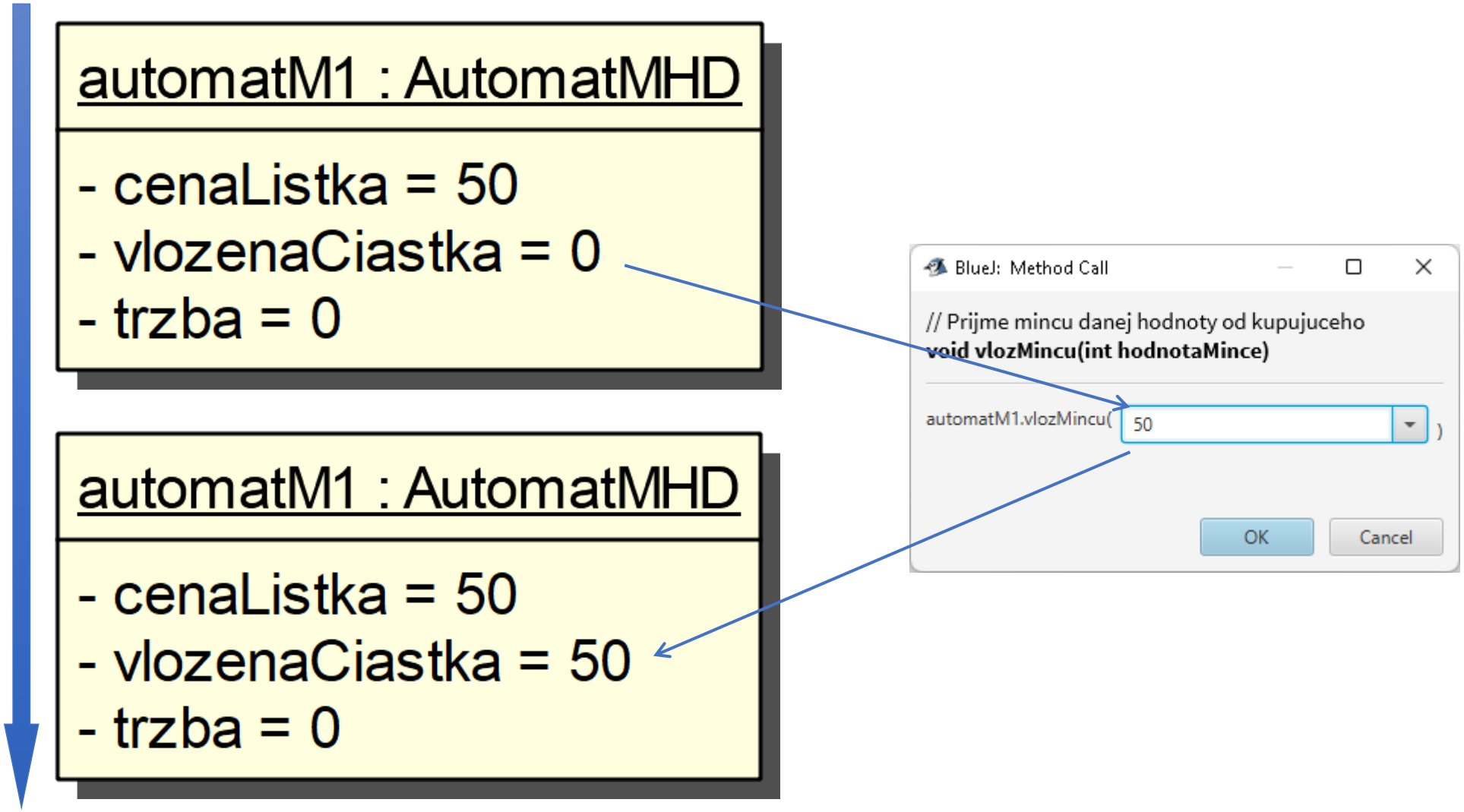
- najčastejšie prostredníctvom návratovej hodnoty

Metódy a stav objektu (2)

- zmenové – menia stav objektu
- stav objektu je iný pred vykonaním zmenovej metódy a iný po jej vykonaní.

```
public void vlozMincu(int hodnotaMince) {
    this.vlozenaCiastka = this.vlozenaCiastka + hodnotaMince;
}
```

Metódy a stav objektu (3)



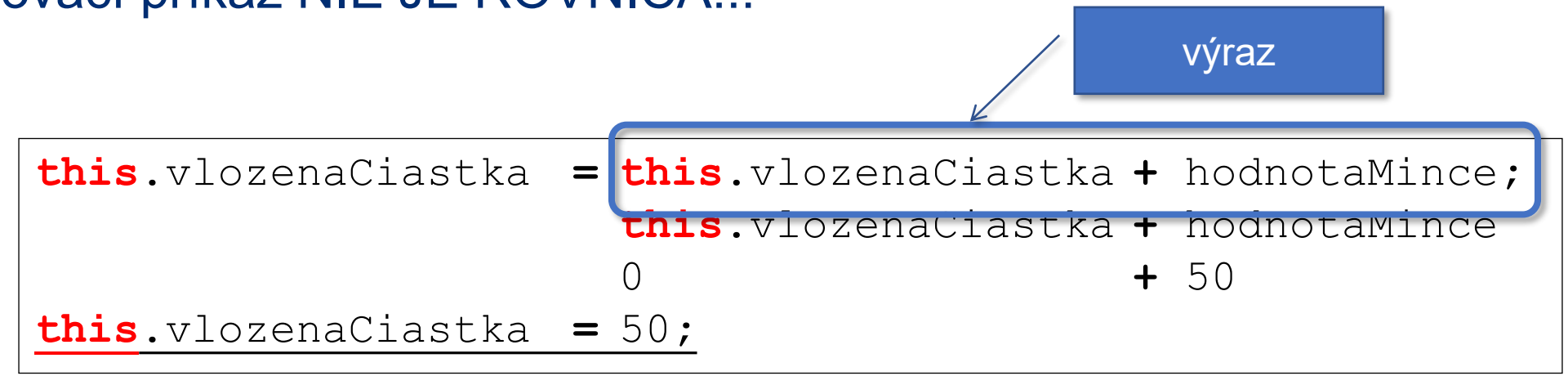
Vykonanie priraďovacieho príkazu (1)

- priraďovací príkaz NIE JE ROVNICA!!!

```
this.vlozenaCiastka  = this.vlozenaCiastka + hodnotaMince;
this.vlozenaCiastka  = this.vlozenaCiastka + hodnotaMince
                   0 = hodnotaMince
```

Vykonanie priraďovacieho príkazu (2)

- priraďovací príkaz NIE JE ROVNICA!!!



Hlavička metódy tlacListok

```
/**
 * Vytlaci cestovny listok,
 * pripocita vlozenu ciastku k trzbe a
 * vynuluje vlozenu ciastku
 */
public void tlacListok() {
    …
}
```

Komentáre

- komentár – vysvetľujúci text pre ľudí čítajúcich zdrojový text programu.
- jednoriadkový komentár – od // do konca riadku

```
// suma vlozenych minci pred tlacou listka
private int vlozenaCiastka;
// celkova suma penazi za vsetky listky
private int trzba;
```

Viacriadkový komentár

- príklad

```
/* Trieda modeluje primitívny automat
 * na predaj cestovných lístkov.
 * Model predpokladá, že kupujúci vloží
 * čiastku presne podľa ceny lístka.
 */
```

- /* – úvodné znaky viacriadkového komentára
- /** – úvodné znaky dokumentačného komentára
- */ – ukončujúce znaky komentára

Použitie komentárov

- nad komentovaný riadok
- uvádzame:
  - informácie, ktoré nie sú priamo zrejmé zo zdrojového textu
  - dôvody pre niektoré riešenia

- dokumentačný komentár slúži na automatické generovanie dokumentácie.
  - popisuje chovanie triedy/metódy/konštruktora

Telo metódy tlacListok (1)

```
// tlac listka do okna konzoly
System.out.println("**************************");
System.out.println("* Skolska linka FRI");
System.out.println("* Cestovny listok");
System.out.print("* cena ");
System.out.print(this.cenaListka);
System.out.println(" centov");


System.out.println("**************************");
System.out.println();
…
```

Telo metódy tlacListok (2)

```
…
// pripocitaj vlozenu ciastku k trzbe
this.trzba = this.trzba + this.vlozenaCiastka;
// nuluj vlozenu ciastku
this.vlozenaCiastka = 0;
```

Tlač z metód – print, println (1)

```
System.out.print(this.cenaListka);
```

- System.out – zabudovaný objekt pre tlač do okna terminálu
- print(this.cenaListka) – správa – žiadosť o tlač hodnoty atribútu cenaListka
- println(this.cenaListka) – správa – žiadosť o tlač hodnoty atribútu cenaListka a prechod na nový riadok

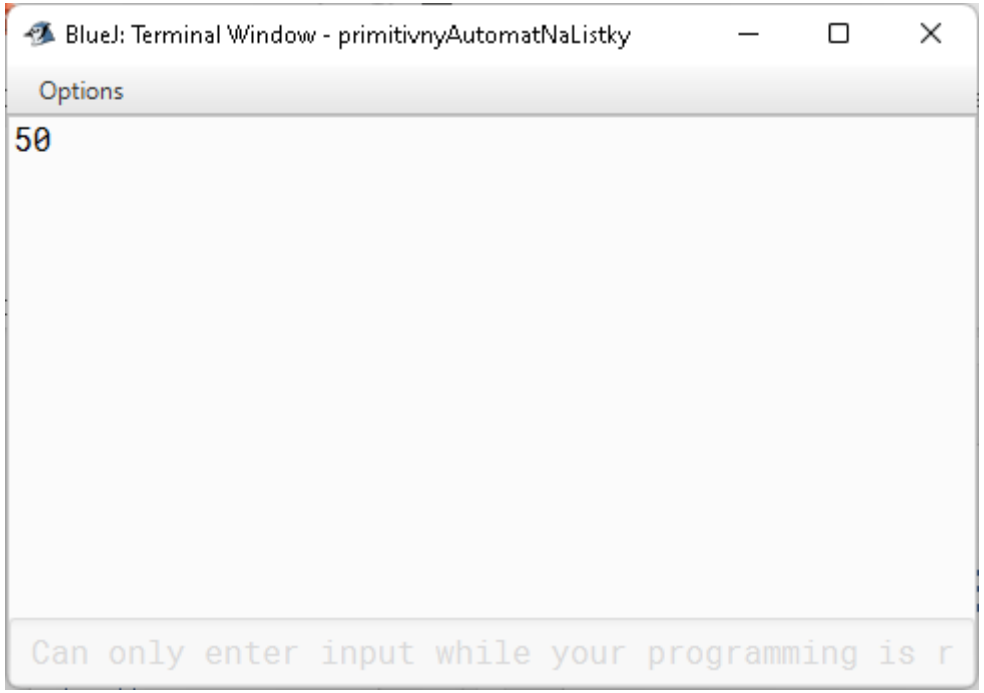
Tlač z metód – print, println (2)

- System.out.print(skutocnyParameter) – tlač hodnoty skutočného parametra na aktuálny riadok
- skutocnyParameter – ľubovoľný výraz
- každý typ má definovaný svoj formát pre tlač do okna terminálu

- println() – správa – žiadosť o ukončenie riadku (prechod na začiatok nového riadku)

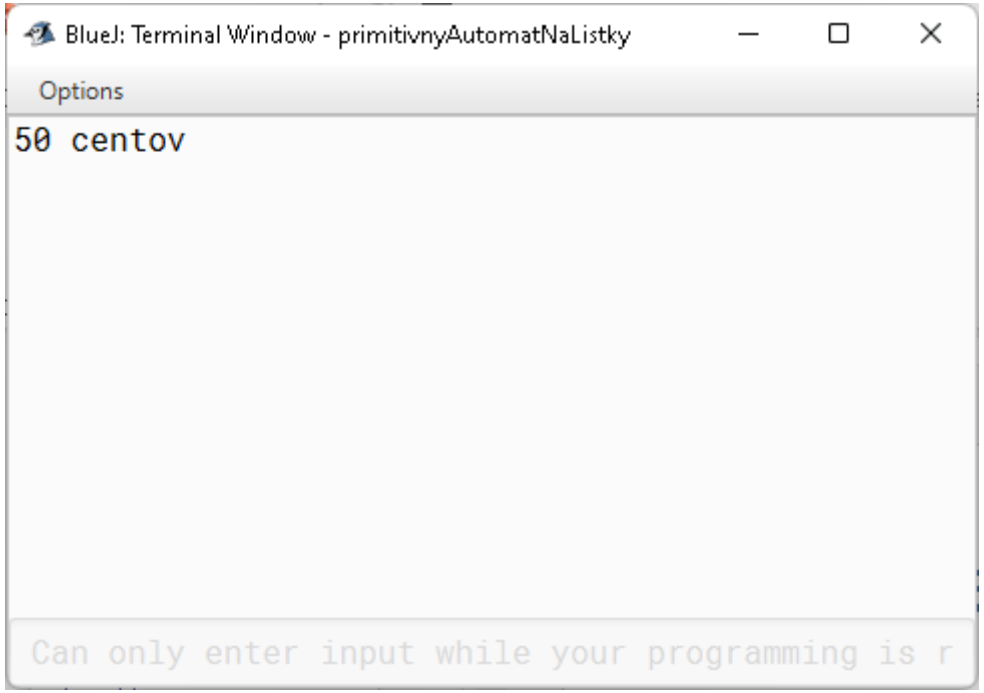
Tlač z metód – print, println (3)

```
System.out.print(this.cenaListka);
```



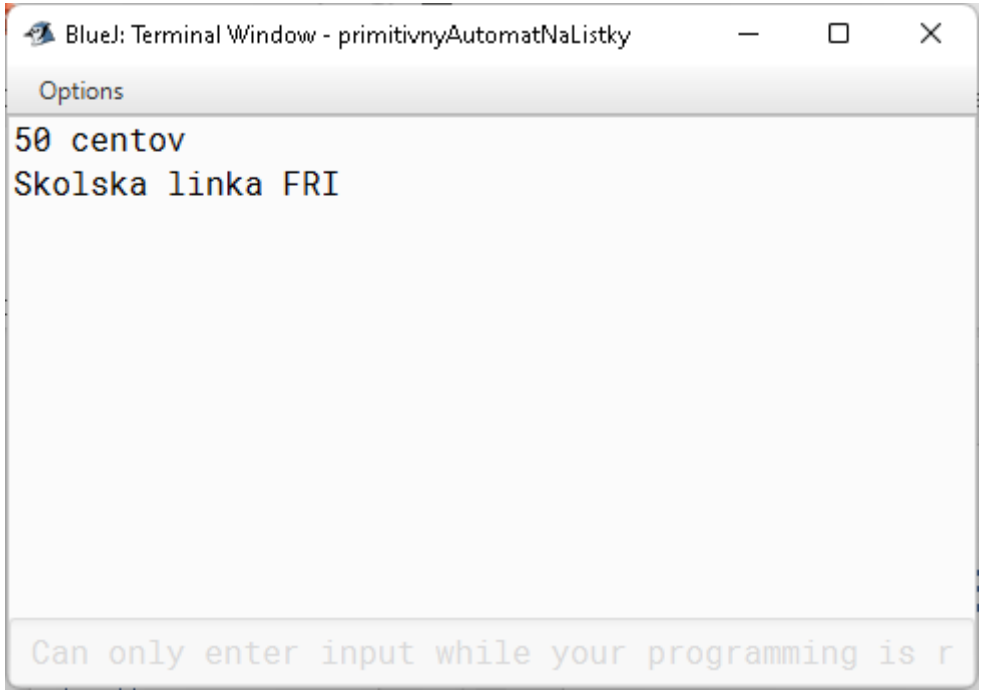
Tlač z metód – print, println (4)

```
System.out.println(" centov");
```

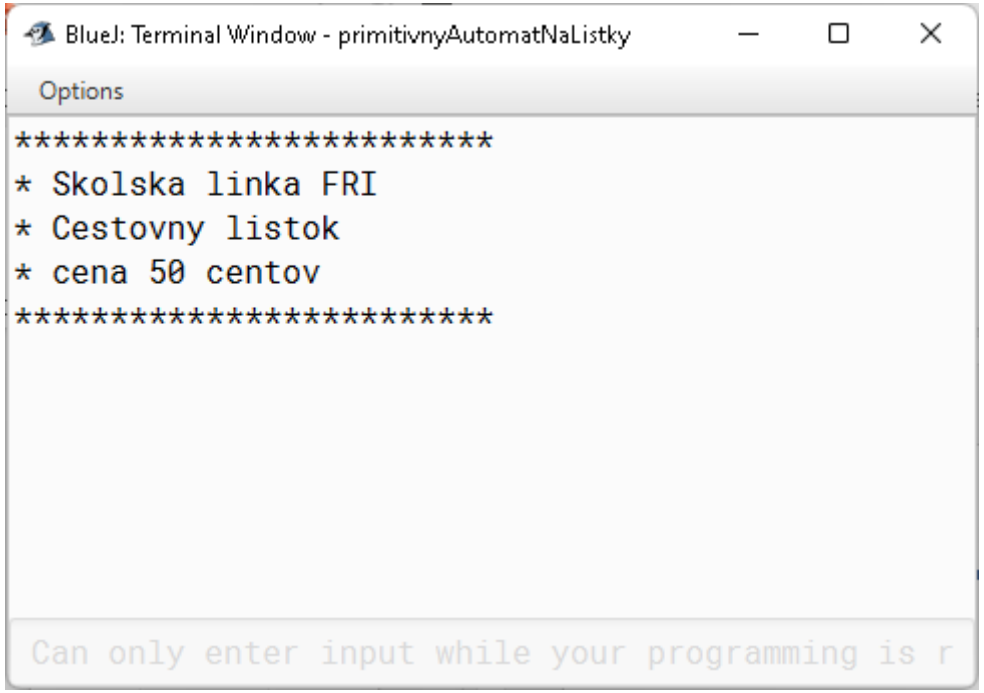


Tlač z metód – print, println (5)

```
System.out.println("Skolska linka FRI");
```



Výstup metódy tlacListok



Získanie tržby z automatu (1)

- ako automat pozná, koľko ma vrátiť?
- this.trzba
- je metóda správna?

```
public int vratTrzbu() {
    return this.trzba;
}
```

- treba nulovať trzbu.

Získanie tržby z automatu (2)

- je teraz metóda správna?

```
public int vratTrzbu() {
     return this.trzba;
     this.trzba = 0;
}
```

- preklad je s chybou
- nulovanie sa nemôže vykonať

Získanie tržby z automatu (3)

- je teraz metóda správna?

```
public int vratTrzbu() {
     this.trzba = 0;
     return this.trzba;
}
```

- preklad je bez chýb
- automat nevráti nič
  - metóda vráti nulu

Automat vráti zostatok(4)

```
public int vratTrzbu() {
    int povodnaTrzba;
    povodnaTrzba = this.trzba;
    this.trzba = 0;
    return povodnaTrzba;
}
```

Lokálna premenná

- definícia lokálnej premennej

```
int povodnaTrzba;
```

- definícia a inicializácia lokálnej premennej

```
int povodnaTrzba = this.trzba;
```

- rovnako ako

```
int povodnaTrzba;
povodnaTrzba = this.trzba;
```

Premenná – spoločné vlastnosti

- atribúty, parametre a lokálne premenné sú miesta v pamäti, v ktorých sú uložené hodnoty
- atribúty, parametre a lokálne premenné majú vždy definovaný typ hodnoty, ktorú uchovávajú
- pokiaľ budeme o atribútoch, parametroch a lokálnych premenných hovoriť všeobecne, budeme hovoriť o premenných

Atribúty, parametre a lokálne premenné (1)

- účel, poslanie
  - atribút – uchováva stav objektu
  - parameter – prenos spresňujúcej informácie
  - lokálna premenná – dočasné uloženie určitej hodnoty v rámci bloku

Atribúty, parametre a lokálne premenné (2)

- definícia
  - atribút – telo triedy, mimo konštruktorov a metód
  - parameter – hlavička konštruktora alebo metódy
  - lokálna premenná – v bloku (v tele konštruktora alebo metódy)

Atribúty, parametre a lokálne premenné (3)

- inicializácia
  - atribút – v konštruktore v čase vytvárania inštancie
  - parameter – v odosielanej správe ako skutočný parameter
  - lokálna premenná – v bloku (v tele konštruktora alebo metódy)

Atribúty, parametre a lokálne premenné (4)

- rozsah platnosti (viditeľnosť, použiteľnosť)
  - atribút – v každom konštruktore alebo metóde
  - parameter – v tele daného konštruktora alebo metódy
  - lokálna premenná – v bloku od miesta definície po koniec bloku definície aj vo všetkých vnorených blokoch.

Atribúty, parametre a lokálne premenné (5)

- životný cyklus – existencia
  - atribút – životný cyklus inštancie
  - parameter – v čase vykonávania konštruktora alebo metódy
  - lokálna premenná – v čase vykonávania bloku od miesta definície po koniec bloku, v ktorom bola definovaná