

2

Tvorba tried

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 inštalácie
 - vznik
 - správa triede, začiatkový 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

- založené na triedach – najpočetnejšia skupina
 - triedy ako inštancie metatriedy
 - Smalltalk, Python, ...
 - čisté objektové jazyky
 - triedy ako priamo definované objekty
 - (Object) Pascal, C++, C#, Java, ...
 - hybridné objektové jazyky
 - najviac jazykov
- založené na objektoch
 - podporované v málo jazykoch
 - JavaScript, Self, ...

Delenie programovacích jazykov

- založené na triedach – najpočetnejšia skupina
 - triedy ako inštancie metatriedy
 - Smalltalk, Python, ...
 - čisté objektové jazyky
 - triedy ako priamo definované objekty
 - (Object) Pascal, C++, C#, Java, ...
 - hybridné objektové jazyky
 - najviac jazykov
- založené na objektoch
 - podporované v málo jazykoch
 - JavaScript, Self, ...

Trieda ako šablóna

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

Postup pri vytváraní tried

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

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 vhodných 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()`

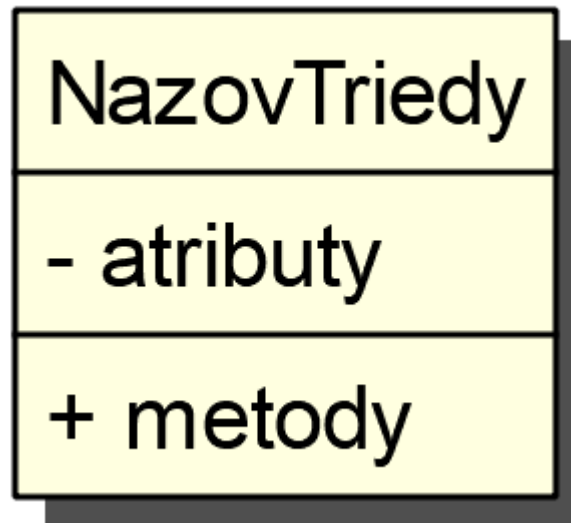
AutomatMHD – vlastnosti

- 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

- + new(cenaListka: int): AutomatMHD
- + getCenaListka(): int
- + getVlozenaCiastka(): int
- + vložMincu(hodnotaMince: int): void
- + tlačListok()

AutomatMHD – vnútorný pohľad

AutomatMHD

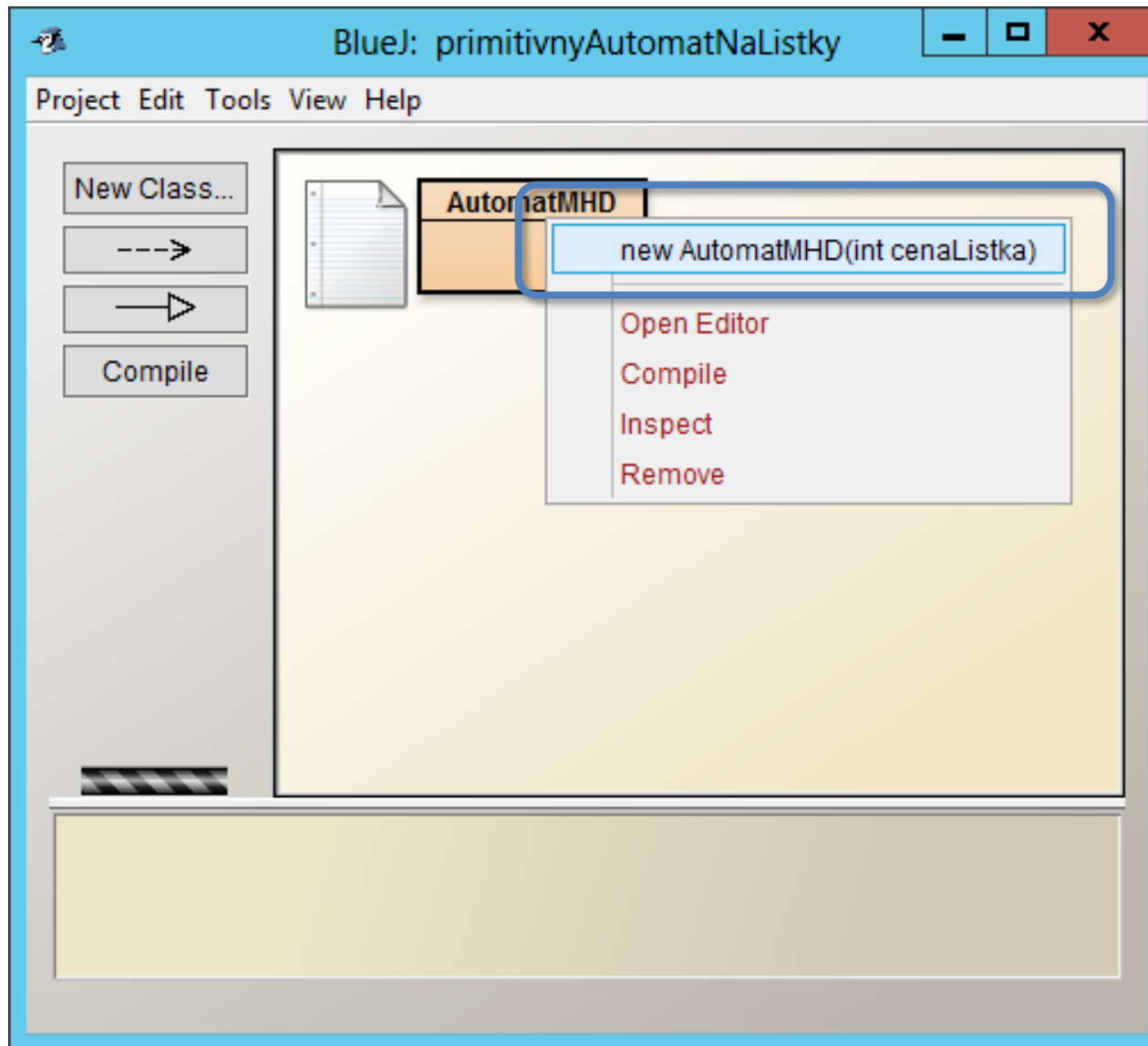
- cenaListka: int
- vlozenaCiastka: int
- trzba: int

- + AutomatMHD(cenaListka: int)
- + getCenaListka(): int
- + getVlozenaCiastka(): int
- + vložMincu(hodnotaMince: int): void
- + tlačListok()

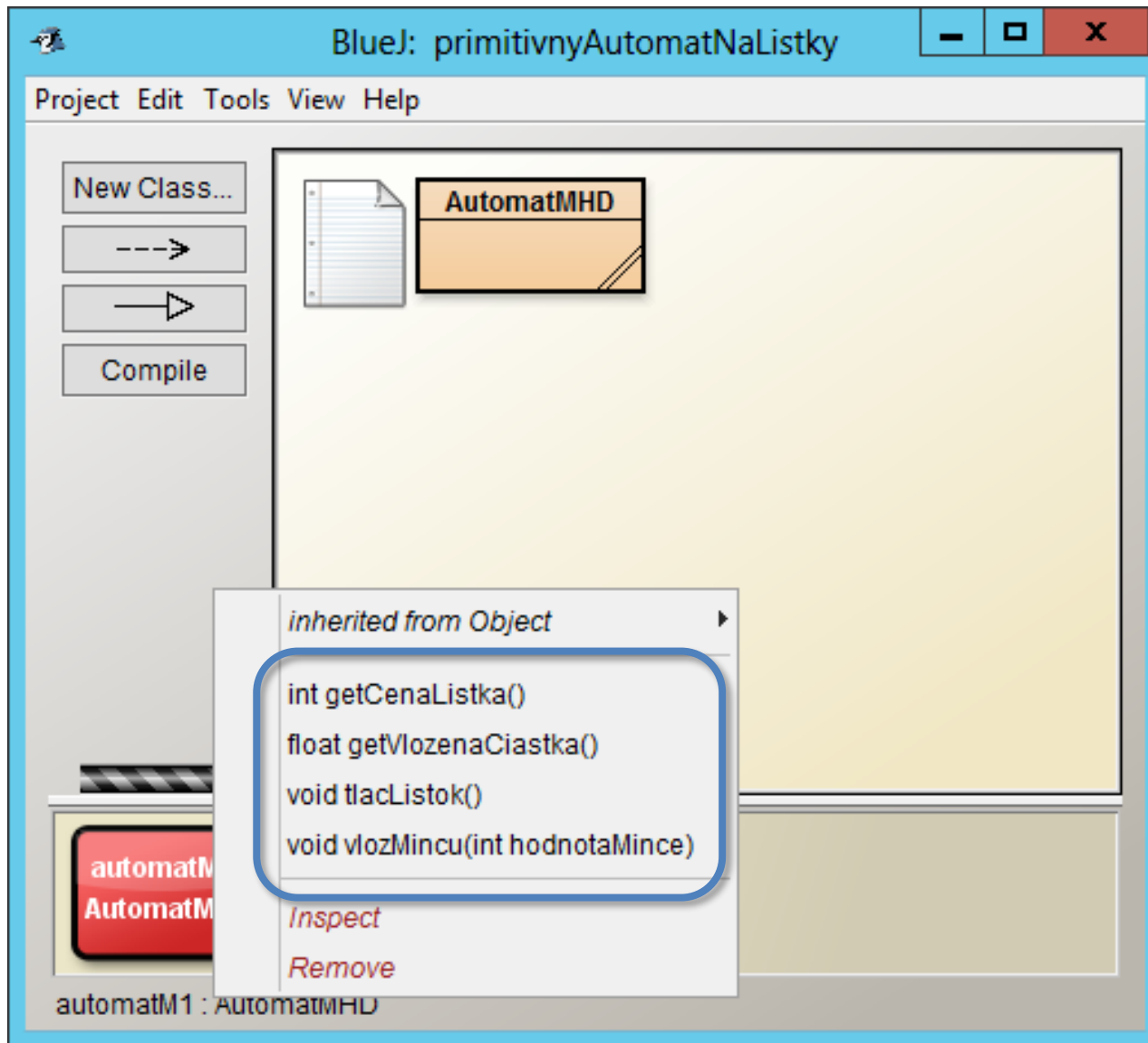


AutomatMHD

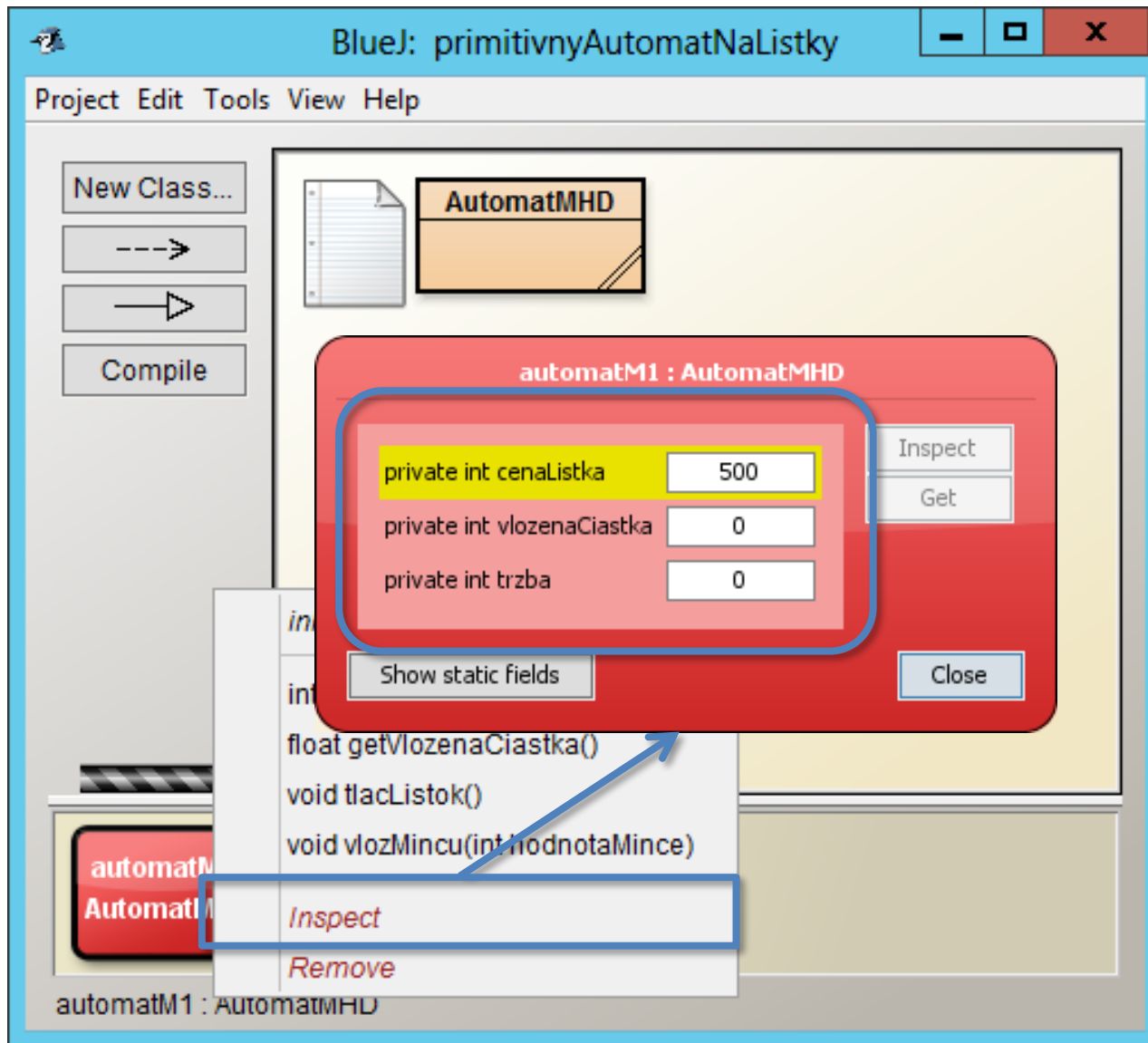
BlueJ – rozhranie triedy



BlueJ – rozhranie inštancie



BlueJ – vnútorný pohľad



UML – objektový diagram

nazov : Trieda

- atribut = hodnota

Objektový diagram automatu

automatM1 : AutomatMHD

- cenaListka = 500
- vlozenaCiastka = 0
- trzba = 0

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

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 uchovávanéj 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 $\langle -2^7, 2^7 - 1 \rangle$, $2^7 = 128$
 - short $\langle -2^{15}, 2^{15} - 1 \rangle$, $2^{15} = 32.768$
 - int $\langle -2^{31}, 2^{31} - 1 \rangle$, $2^{31} = 2.147.483.648$
 - long $\langle -2^{63}, 2^{63} - 1 \rangle$, $2^{63} = 9.223.372.036.854.775.808$

Primitívne dátové typy – pre reálne čísla

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

Obmedzenia číselných typov

- neplatia vždy zákony asociatívny, komutatívny a distributívny pre aritmetické operácie!
- $a = 2^{31} - 1$
- $a + a = -2$

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

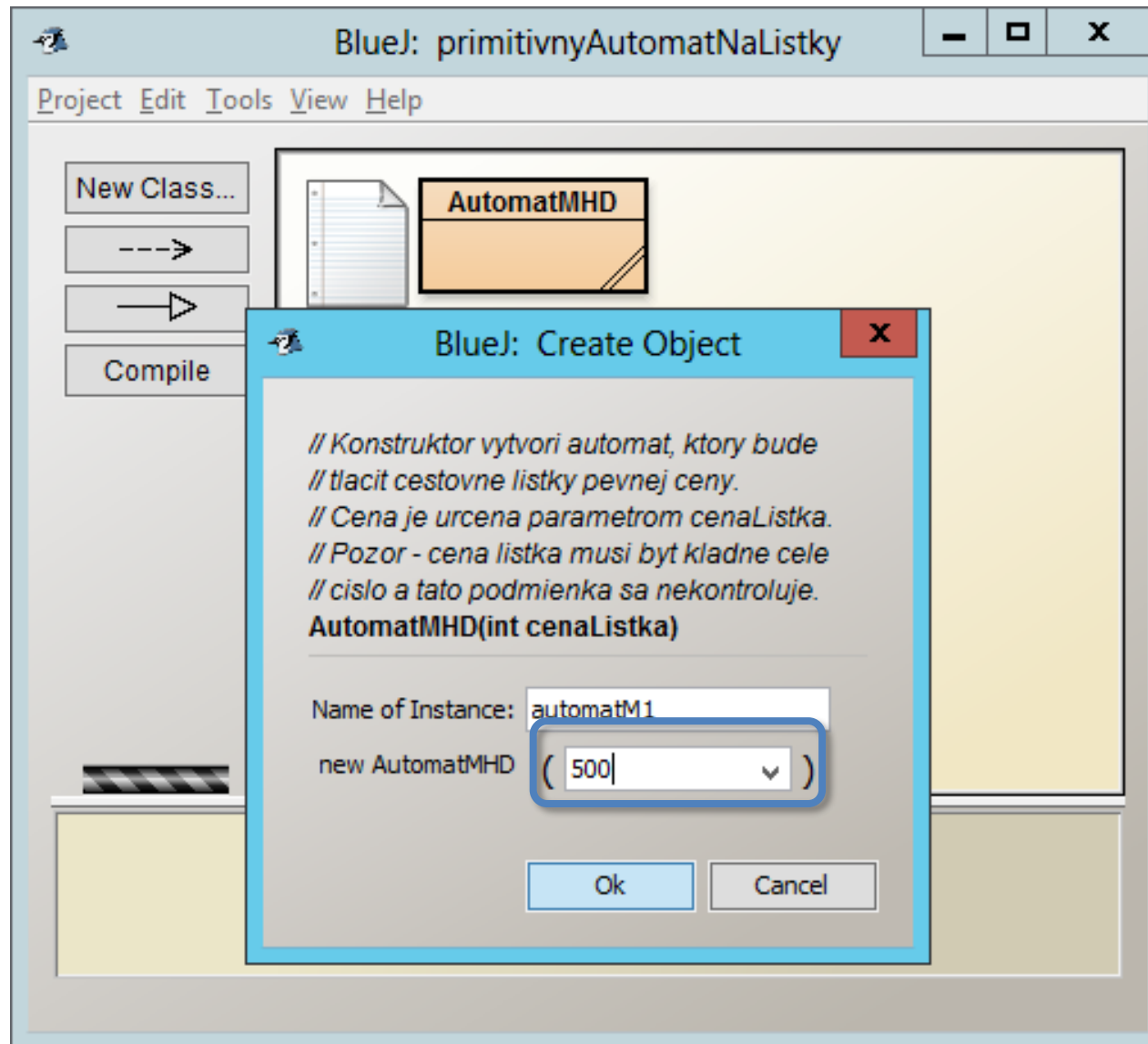
Formálne parametre

- 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;
```

: AutomatMHD

- cenaListka = ???
- vlozenaCiastka = ???
- trzba = ???

BlueJ: Create Object

// Konstruktory vytvori automat, ktorý bude
// tlačit cestovné lístky pevnej ceny.
// Cena je určená parametrom cenaListka.
// Pozor - cena lístka musí byť kladné celé
// číslo a táto podmienka sa nekontroluje.
AutomatMHD(int cenaListka)

Name of Instance: automatM1

new AutomatMHD (500)

Ok Cancel

cenaListka

500

Prirad'ovací príkaz

- príklad:

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

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

this

- klíčové slovo v jazyku Java
- vyjadruje přístup k objektu
- `this.nazovAtributu`
 - přístup 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

```
public int getCenaListka()
```

```
return this.cenaListka;
```

```
private int cenaListka;
```

```
return 5; // EUR
```

```
return 5.50; // EUR
```



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 vložMincu(int hodnotaMince) {  
    this.vlozenaCiastka = this.vlozenaCiastka + hodnotaMince;  
}
```

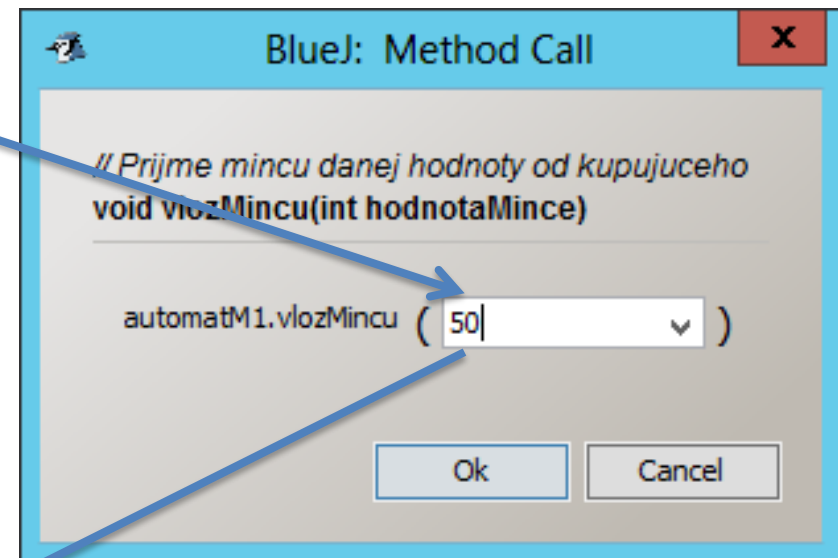
Metódy a stav objektu₍₃₎

: AutomatMHD

- cenaListka = 500
- vlozenaCiastka = 0
- trzba = 0

: AutomatMHD

- cenaListka = 500
- vlozenaCiastka = 50
- trzba = 0



Vykonanie priradovacieho príkazu₍₁₎

priradovací príkaz NIE JE ROVNICA!!!



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

Vykonanie priradovacieho príkazu₍₂₎

priradovací príkaz NIE JE ROVNICA!!!

výraz

```
this.vlozenaCiastka = this.vlozenaCiastka + hodnotaMince;  
this.vlozenaCiastka + hodnotaMince  
0 + 50  
this.vlozenaCiastka = 50;
```

Tlač z metód – print, println₍₁₎

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

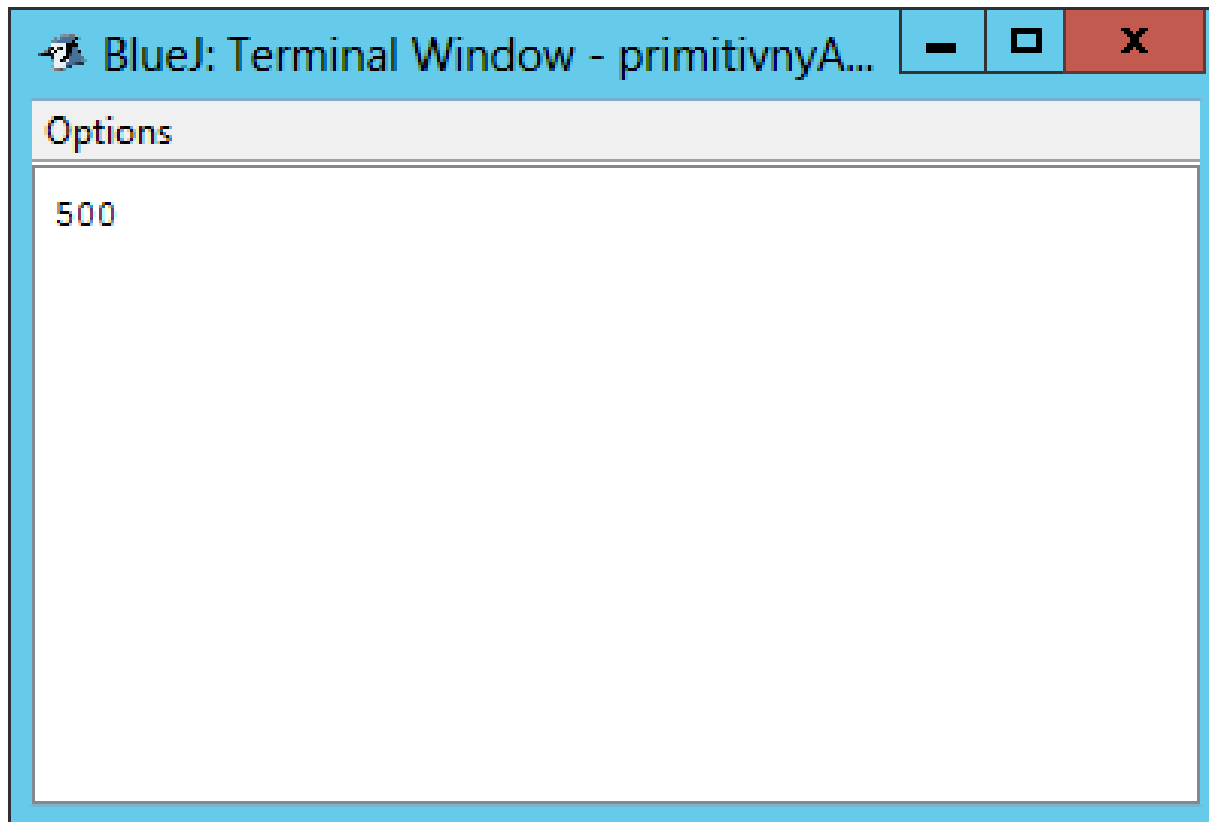
- System.out – zabudovaný objekt pre tlač do okna terminálu
- print(this.cenaListka) – správa – žiadosť o tlač hodnoty atribútu cenaListka
- println(this.cenaListka) – správa – žiadosť o tlač hodnoty atribútu cenaListka a prechod na nový riadok

Tlač z metód – print, println₍₂₎

- [System.out.print\(skutočnýParameter\)](#) – tlač hodnoty skutočného parametra na aktuálny riadok
- [skutočnýParameter](#) – ľubovoľný výraz
- každý typ má definovaný svoj formát pre tlač do okna terminálu
- [println\(\)](#) – správa – žiadosť o ukončenie riadku (prechod na začiatok nového riadku)

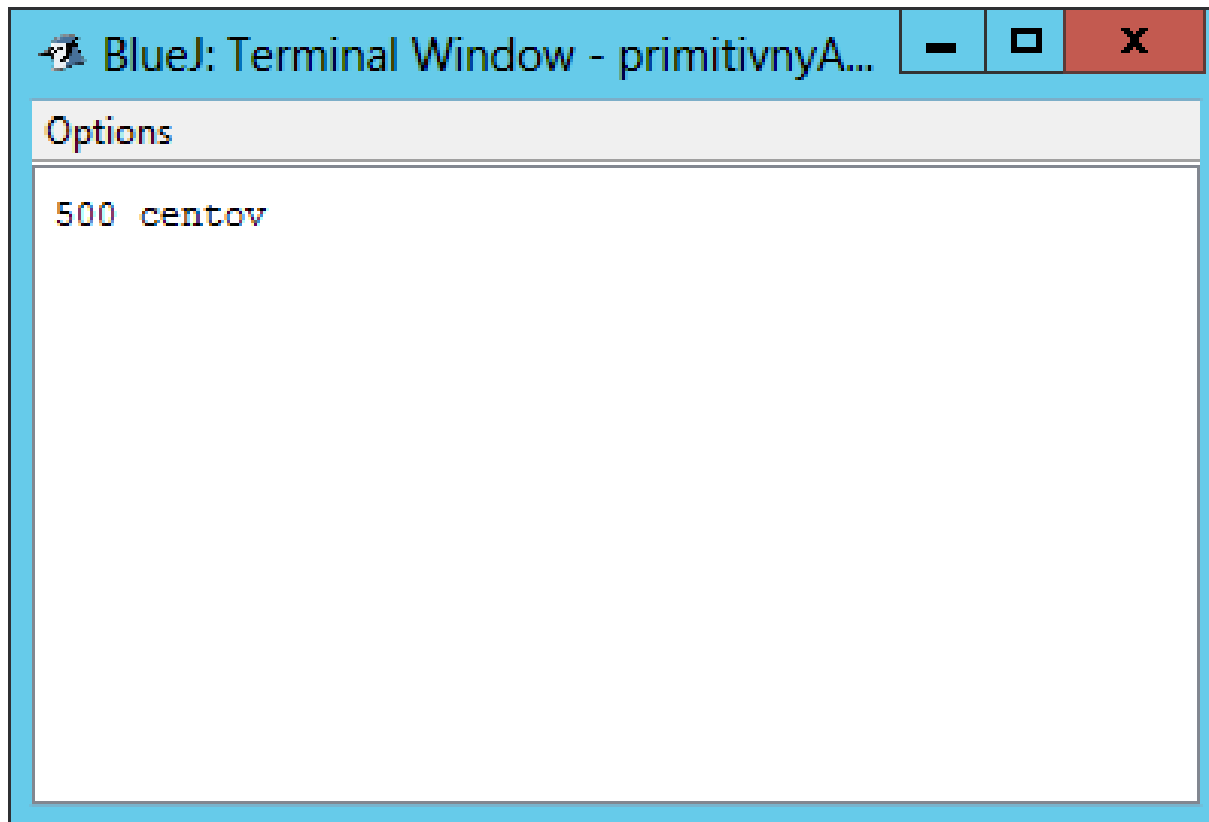
Tlač z metód – print, println₍₃₎

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



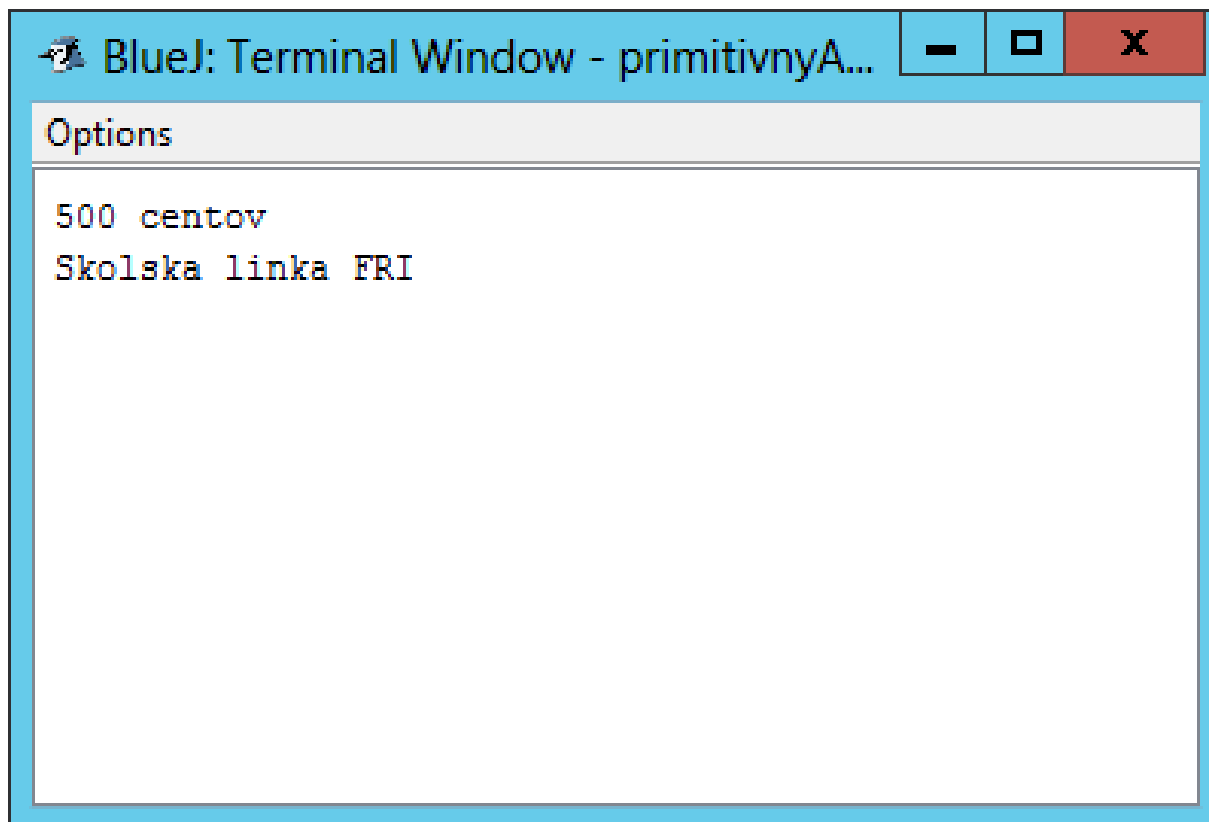
Tlač z metód – print, println₍₄₎

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



Tlač z metód – print, println₍₄₎

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



The image shows a screenshot of a BlueJ Terminal Window. The title bar reads "BlueJ: Terminal Window - primitivnyA...". Below the title bar is a tab labeled "Options". The main area of the terminal displays two lines of text: "500 centov" and "Skolska linka FRI".

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

Metóda tlacListok₍₁₎

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

Metóda tlačListok₍₂₎

// tlač 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();
```

...

Metóda tlačListok₍₃₎

...

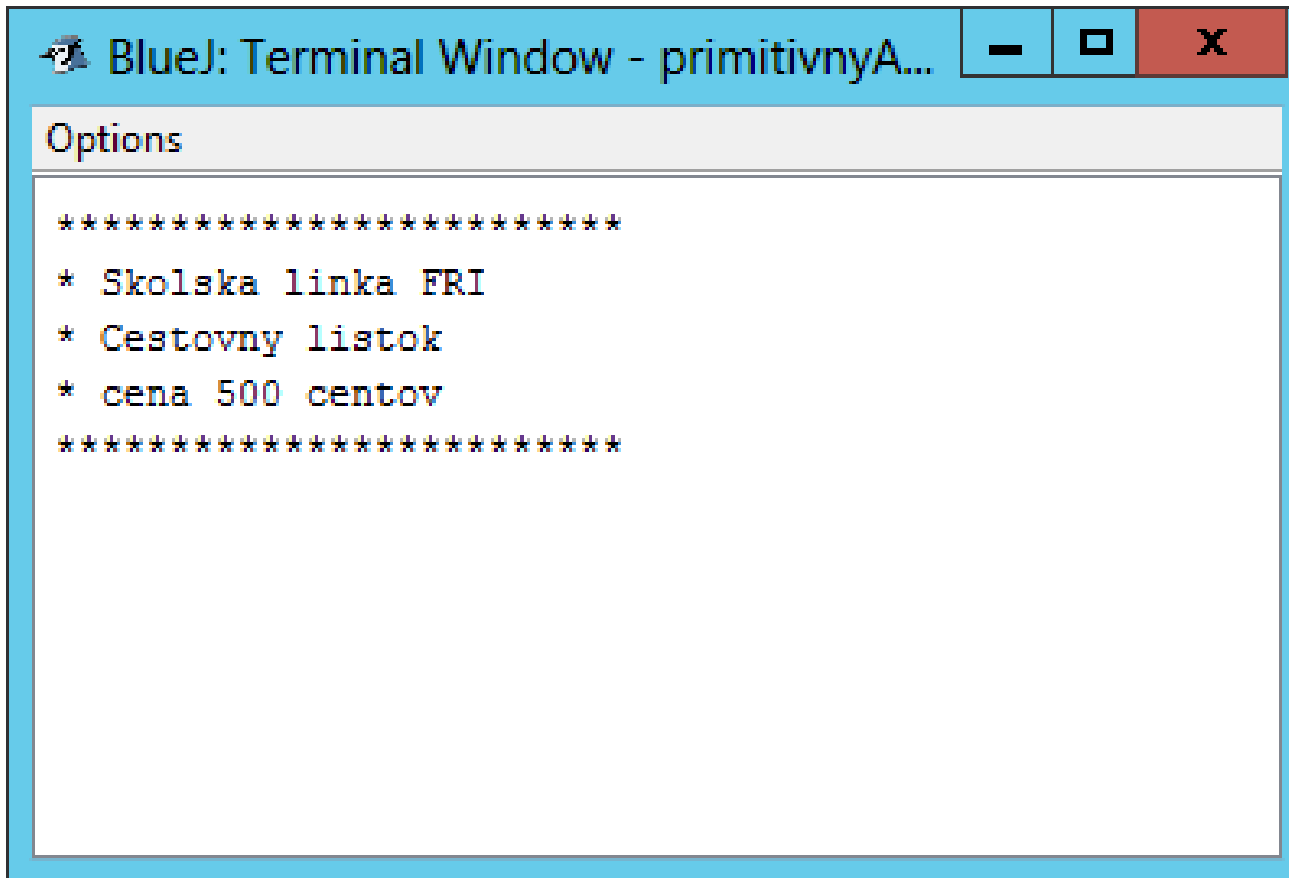
// pripocitaj vlozenu ciastku k trzbe

this.trzba = **this**.trzba + **this**.vlozenaCiastka;

// nuluj vlozenu ciastku

this.vlozenaCiastka = 0;

Výstup metódy tlačListok



A screenshot of a BlueJ Terminal Window titled "BlueJ: Terminal Window - primitivnyA...". The window has a light blue border and standard window controls (minimize, maximize, close). Inside, there is a tab labeled "Options". The terminal output is as follows:

```
*****  
* Skolska linka FRI  
* Cestovny listok  
* cena 500 centov  
*****
```

Vďaka za pozornosť