Algoritmizace a programování

Terminálový vstup a výstup

Verze pro akademický rok 2012/2013

Výpis hodnot

- Terminálový vstup a výstup budeme používat jako základní způsob interakce programu s uživatelem
- Výpis hodnot

```
System.out.print(<co>);
System.out.println(<co>);
```

- `System` – třída jazyka Java – tato třída je umístěna v balíčku `java.lang` – tento balíček je implicitně importován, není nutné zapisovat import explicitně
- `out` – členská proměnná třídy `System` typu `PrintStream`, umožňuje práci se standardním (terminálovým. konzolovým) výstupem
- `print(), println(), format()` – metody třídy `PrintStream`

- `println()` – provede výpis a ukončí aktuální řádek vstupu

Formátování výstupu

- Metody `print()`, `println()` mají omezené možnosti formátování výpisu
- Komfortnější formátování hodnot do textových řetězců umožňuje metoda `format()`, opět se jedná o metodu výstupního proudu `out` třídy `System`

```
System.out.format(<formatovaci retezec>, <hodnoty>)
```

- `<formatovaci retezec>` je konstantní textový řetězec, který může obsahovat parametry, parametry začínají znakem %, pro každý parametr (až na výjimky) musí být za formátovacím textovým řetězcem uvedena vypisovaná hodnota, která se při výpisu naformátuje dle specifikovaného formátu do vypisovaného textového řetězce. Hodnoty v seznamu `<hodnoty>` jsou navzájem odděleny čárkou, pořadí a typ musí odpovídat pořadí a typu parametrů ve formátovacím řetězci. Každá hodnota je v obecném případě dána výrazem (proměnná, konstanta, výraz), jehož hodnota musí být typově kompatibilní s typem určeným příslušným parametrem.

Formátování celých čísel

```
int i = 1483;
```

`System.out.format("Cislo: %d",i);` – celé číslo v desítkové soustavě

`System.out.format("Cislo: %o",i);` – celé číslo v osmičkové soustavě

`System.out.format("Cislo: %x",i);` – celé číslo v šestnáctkové soustavě

`System.out.format("Cislo: %X",i);` – celé číslo v šestnáctkové soustavě

`System.out.format("Cislo: %6d",i);` – celé číslo s určením počtu min. pozic

`System.out.format("Cislo: %-6d",i);` – určení počtu pozic, zarovnání vlevo

`System.out.format("Cislo: %+d",i);` – s výpisem kladného znaménka

`System.out.format("Cislo: %06d",i);` – s výpisem nevýznamných nul

`System.out.format("Cislo: %,6d",i);` – s oddělovačem řádů

Formátování reálných čísel

`double d = 1960.00467;`

`System.out.format("Cislo: %f",d);` – reálné číslo, běžný výpis, oddělovač desetinných míst závisí na lokalizaci

`System.out.format("Cislo: %g",d);` – výpis buď ve vědecké notaci nebo jako běžné reálné číslo, oddělovač desetinných míst je vždy tečka

`System.out.format("Cislo: %e",d);` – výpis reálného čísla ve vědecké notaci

Při výpisu reálných čísel lze určit celkový počet pozic a počet desetinných míst

`System.out.format("Cislo: %15.4f",d);` – počet pozic, desetinných míst

Obdobně – zarovnání vlevo, vynucení znaménka, nevýznamných nul, oddělovač řádů

```
System.out.format(" %-15.4f", d);
System.out.format(" %+15.4f", d);
System.out.format(" %015.4f", d);
System.out.format(" %,15.4f", d);
```

Další formátování

- Formátování znakové hodnoty – formátovací parametr %c
- Formátování textového řetězce – formátovací parametr %s

- Obdobně jako u formátování číselných hodnot lze zadávat počet pozic a zarovnání formátované hodnoty vlevo v rámci příslušného počtu pozic

- Ukončení aktuální řádky výstupního proudu – formátovací parametr %n

- Pořadí hodnoty `<poradi>$` – `%2$X`

- Reálné číslo představující datum a čas a jeho formátování

Terminálový vstup

- Pro terminálový vstup budeme používat instanci třídy `Scanner`
- Tato třída je umístěna v baličku `java.util`
- Baliček a jednotlivé prostředky v něm obsažené nejsou implicitně importovány. Abychom mohli používat třídu a její prostředky, je třeba zajistit import této třídy zařazením `import java.util.Scanner` před deklaraci třídy, blíže bude import balíčků probírán později
- Instanci třídy `Scanner` musíme nejdříve vytvořit voláním metody konstruktoru, parametrem je vstupní proud, ze kterého budeme číst – tímto vstupním proudem je v případě terminálového vstupu `System.in`
- Deklarace a inicializace proměnné pro načítání ze standardního vstupu bude potom vypadat následovně

```
Scanner sc = new Scanner(System.in);
```

- Poté je možné volat metody třídy `Scanner` pro načítání hodnot různých základních typů z příslušného proudu.

Terminálový vstup

- Čtení celého čísla typu `int` – metoda
  `sc.nextInt()`
- Čtení reálného čísla typu `double`
  `sc.nextDouble()`
- Čtení znaku – prvního znak zadaného textového řetězce
  `sc.nextLine().charAt(0)`
- Čtení textového řetězce do prvního prázdného znaku
  `sc.next()`
- Čtení celé řádky do textového řetězce
  `sc.nextLine()`

- Metody vrací načtenou hodnotu svým jménem
- Obdobné metody jsou pro načítání hodnot ostatních primitivních typů
- Kombinace načítání čísel a znaků
- Lokalizace – `sc.useLocale(locale.US)`

Příklad

```
package ja001vstupvystup;

import java.util.Scanner;

public class Main {

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);

        int n;
        float x;

        System.out.println("Zadej cele cislo");
        n = sc.nextInt();
        System.out.println("Zadej realne cislo");
        x = sc.nextFloat();

        System.out.println("Zadana cisla jsou");
        System.out.println("Prvni cislo cele " + n);
        System.out.println("Druhe cislo realne " + x);
        System.out.println();

        System.out.format("%nVypis metodou format%nPrvni cislo %6d%nDruhe cislo %6.2f%n", n, x);

        System.out.format("%nKonec programu");
    }
}
```

Inicializace terminálového vstupu.

Deklarace proměnných

Načtení hodnot proměnných ze standardního (terminálového) vstupu

Komentovaný výpis hodnot proměnných na terminálový výstup pomocí metody `println()`

Komentovaný výpis hodnot proměnných na terminálový výstup pomocí metody `format()`

Všimněte si oddělovače reálných čísel – při načítání, při výpisu pomocí `println()`, `format()`