Algoritmizace a programování

Typ class

- Deklarace typu, data a metody třídy
- Použití typu class, vytvoření instance
- Přístup k položkám
- Rušení instance třídy
- Metody, parametry a návratová hodnota
- Pole hodnot typu class

Verze pro akademický rok 2012/2013

Referenční datový typ

- ***Pole*** a ***třídy*** v jazyce Java používají ***referenční model***

- Doposud probrané datové typy – primitivní datové typy – nepoužívají referenční model tj.:
  - Deklarace proměnné primitivního datového typu – přidělení paměti potřebné pro uchování příslušné hodnoty – paměťové místo obsahuje konkrétní hodnotu

- ***Referenční model***
  - Deklarace proměnné – přidělení paměti pro uchování reference (odkazu, ukazatele), deklarací proměnné ještě **není alokována paměť pro hodnotu** příslušného typu, rozumnou číselnou hodnotu reference neumíme získat

    ```
    Scanner sc;
    ```

  - Přidělení paměti v jazyce Java zajistíme operátorem **`new`** – použití závisí na konkrétním referenčním typu – zda se jedná o pole nebo třídu

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

  - Hodnota neplatné (neexistující) reference (odkazu) je hodnota konstanty **`null`**

Rozsah platnosti identifikátorů

- Identifikátor je použitelný
  - v bloku, ve kterém je deklarován
  - od místa, na kterém je deklarován

Typ `class`

- Jazyk Java je objektový
- Základem jazyka jsou třídy a objekty – typy deklarované jako `class` a jejich instance

- Objekty v programu, třídy jako šablony pro vytváření objektů

- Součástí třídy jsou
  - Data – proměnné deklarované uvnitř třídy – členské proměnné třídy, datové položky třídy
  - Metody – podprogramy (funkce) deklarované uvnitř nějaké třídy
    - obsahují výkonný kód – příkazy, které se postupně provedou po zavolání metody
    - pracují nad datovými položkami instance třídy
    - parametry metody – hodnoty, pro které se má příslušný kód provést
    - vrací hodnotu svým jménem – návratová hodnota, specifikace typu návratové hodnoty (použití typu `void` – metoda nevrací hodnotu (vrací prázdnou hodnotu))

Deklarace nové veřejné třídy

- Deklarace nové třídy
  - Umístění v novém souboru
  - Každá třída patří do nějakého balíčku `package`
  - Nově deklarované třídy můžeme umístit do stejného balíčku jako je hlavní třída naší aplikace – třída s metodou `main` obsahující hlavní program
  - V prostředí Netbeans
    - „Aplikace“
    - „Source packages“ – zde je jediný balíček – kliknutí pravým tlačítkem myši – položka „New“ a dále vybrat „Java class“
    - Po výběru položky je zobrazeno dialogové okno pro zadání jména nově vytvářeného typu
    - Do aplikace je přidán nový soubor – v něm je deklarována prozatím prázdná třída zadaného jména, třída je ve stejném balíčku jako hlavní třída aplikace
    - Nově vytvořené třídě přidáme data a metody
    - Deklarací nové třídy jsme vytvořily nový typ – nyní můžeme typ používat – deklarovat proměnné tohoto typu, vytvářet instance (jedná se o referenční typ)

Nově vytvořená prázdná třída

```
package mypackage;

/**
*
* @author Jiřina Královcová
*/
public class Bod {

}
```

Třída Bod

```
package mypackage;

public class Bod {
    public double x, y;

    public double vzd(){
        double v = Math.sqrt(x*x + y*y);
        return v;
    }

}
```

Použití deklarovaného typu

- Deklarace proměnné

```
Bod b;
```

- Vytvoření instance – použití konstruktoru (implicitní konstruktor) – stejné jméno jako třída, volá se jako metoda, tedy jménem následují okrouhlé závorky, implicitní konstruktor nemá parametry tedy závorky jsou prázdné

```
b = new Bod();
```

- Použití vytvořené instance

```
b.x = 7.5; b.y =-2.4; System.out.println(b.vzd());
```

- Rušení instance – Java nemá prostředky pro rušení instancí, které by mohl programátor použít v kódu – rušení instancí je prováděno automaticky – *Garbage collector* – pro zrušení je třeba zajistit, aby žádná proměnná neobsahovala referenci na danou instanci (platí i pro pole).

```
b = null;                    b = new Bod();
```

Deklarace třídy

- Deklarace třídy
  `<specifikátory> **class** <identifikator> { ... }`
- Specifikátory
  `public, final, abstract`
- Identifikátor
  - Deklarace třídy, třída ~ typ, jméno třídy používáme jako typ při deklaraci proměnných
- Blok deklarace třídy
  - Členské proměnné
  - Metody
- Všechny třídy budeme deklarovat s modifikátorem `public` – taková třída musí být umístěna v samostatném souboru stejného jména jako má třída

Proměnné deklarované ve třídě

- Deklarace proměnné na úrovni třídy

```
<specifikatory> <typ> <identifikátor>;
<specifikatory> <typ> <identifikátor> = <hodnota>;
```

- Specifikátory

```
public, private, final, static, protected
```

  - Rozsahu platnosti identifikátoru

- Typ

- Identifikátor

Metody deklarované ve třídě

- Třídy mohou obsahovat metody – výkonný kód
- Členění programu do menších programových jednotek

- Metoda je „podprogram“, který je součástí nějaké třídy – všechny podprogramy v jazyce Java musí být v nějaké třídě (tedy metody)

- Struktura metody
  - Záhlaví – modifikátory, typ návratové hodnoty, jméno, parametry v okrouhlé závorce
  - Kód metody – tvořen blokem příkazů, tento blok příkazů se provede po volání metody

- Volání metody, použití metody
  - Statické metody (modifikátor `static`) – metody třídy
  - Metody, které nejsou deklarovány jako statické – metody instance

Deklarace metody

- Deklarace metody

```
<specifikatory> <typ> <jmeno> ( <formalni parametry>) {
    <telo metody>
}
```

- Specifikátory – `static, public, final, private, protected`
- Typ – typ hodnoty, kterou bude metoda vracet svým jménem, pokud metoda nevrací hodnotu použijeme typ `void`
- Jméno – identifikátor
- Formální parametry
  - Deklarace formálních parametrů, jejichž hodnota bude používána při výpočtu - viz dále
- Tělo metody
  - Je tvořeno příkazy, které se postupně provedou po volání metody
  - Deklarace proměnných
  - Příkaz `return` – okamžité ukončení vykonávání metody

    `return <hodnota>;` – ukončení metody, hodnota je vrácena jménem metody, hodnota musí být kompatibilní vzhledem k přiřazení s typem metody

Parametry metod

- Formální parametry – v deklaraci
  - Deklarace formálních parametrů v záhlaví metody – v okrouhlých závorkách za jménem metody
  - Jednotlivé deklarace od sebe odděleny čárkou
  - V každé deklaraci je uveden typ a identifikátor parametru
  - Identifikátor používáme v kódu pro přístup k příslušné hodnotě
  - Typ parametru
    - Parametry primitivních datových typů
    - Parametry typu pole (popřípadě vícerozměrné pole)
    - Parametry objektového typu (typu, který je deklarován jako třída)

- Skutečné parametry – při volání
  - Volání metody – jménem
  - Parametry, pro které má být proveden aktuální výpočet
  - Pořadí, počet a typ skutečných parametrů musí být shodný s pořadím, počtem a typem formálních parametrů uvedených při deklaraci metody

Metoda konstruktoru

- Metoda stejného jména jako je třída
- Nemá určený návratový typ

Třída Bod

```
package mypackage;

public class Bod {
     private double x, y;

     public Bod(double xa; double ya) {
             x = xa;
             y = ya;
     }

     public double vzd(){
             return Math.sqrt(x*x + y*y);
     }

     public double getX() {return x;}
     public double getY() {return y;}
}
```

Přístup k položkám třídy

- Pro přístup k položkám třídy/objektu se používá tečková notace
  - Přístup k položkám nižší hierarchické úrovně – identifikátor nižší hierarchické úrovně oddělujeme tečkou

```
Math.PI
Math.abs(…);
```

- Úplné jméno – včetně jména balíčku
  - ***Plně kvalifikované jméno*** = ***fully qualified name***

```
java.lang.Math.PI
```

Různé použití tříd

- Třída jako **vstupní bod programu**
  - Metoda `main`
- Třída jako **typ** – šablona pro odvozování konkrétních objektů
  - Veřejný konstruktor – metoda stejného jména jako je název třídy
- Třída jako **knihovna funkcí**
  - Metody deklarované s modifikátorem `static`
- … uvedené možnosti lze kombinovat v jediné třídě

Různě strukturované typy

- Různě strukturované typy v jazyce Java
  - Pole polí – vícerozměrná pole
  - Pole objektů
  - Třída může obsahovat členské proměnné jiných než primitivních datových typů
  - Třída s členskou proměnnou typu pole
  - Třída s členskou proměnnou typu deklarovaného jako třída

Pole objektů

```
int n = sc.nextInt();
// deklarace promenne pro  pole bodu, prideleni pameti poli
Bod[] body = new Bod[n+1];

// nacteni souradnic bodu
// prideleni pameti objektum – instancim tridy Bod
for (int i = 0; i < body.length-1; i++) {
   body[i] = new Bod(sc.nextDouble(); sc.nextDouble());
}

// vypocet obvodu n-uhelnika
body[body.length-1] = body[0];
double obvod = 0;
for (int i = 1; i < body.length; i++) {
   double dx = body[i].x – body[i-1].x;
   double dy = body[i].y – body[i-1].y;
   obvod += Math.sqrt(dx*dx + dy*dy);
}
System.out.format("Obvod: %f ",obvod);
```