12 Úvod do práce s reťazcami.

12.1 Reťazce

Doteraz sme v našich program pracovali s najmä s číselnými primitívnymi údajovými typmi, menej so znakovými a logickými typmi. V Jave je však pomerne rozšírené aj používanie **reťazcov**.

Reťazec je vlastne postupnosť znakov znakovej sady UNICODE, na rozdiel od C++ však nie je ukončený tzv. nulovým znakom \0.

Reťazce nepatria medzi primitívne údajové typy, nie sú to ani znakové polia, sú to samostatné **objekty** triedy **String** (presnejšie inštancie triedy *String*). Trieda *String* je súčasťou balíka *java.lang*, ktorý je automaticky importovaný – viď. predchádzajúcu kapitolu. Všetky objekty, vrátane reťazcov, sú v Jave vytvárané na tzv. halde (nazývanej tiež voľné úložište), primitívne typy sú vytvárané vždy v zásobníku. Pojmy trieda, objekt inštancia, zásobník a halda budú neskôr podrobnejšie vysvetlené.

V Jave sú reťazce konštantné – nemeniteľné. To znamená, že vytvorený reťazec nie je možné zmeniť. Akákoľvek požadovaná zmena vedie vždy k nahradeniu pôvodného reťazca novým (pôvodný reťazec je z pamäte odstránený).

12.2 Vytvorenie reťazca

Java ponúka množstvo spôsobov (v objektovom programovaní budeme hovoriť o použití tzv. konštruktorov) ako vytvoriť reťazec. Pre jednoduchosť budeme zatiaľ používať zápis veľmi podobný zápisu, ktorý používame pri deklarácii a inicializácii jednoduchej premennej.

Vytvorenie reťazca a jeho inicializácia:

```
String identifikatorRetazca = hodnota;
```

- `String` – názov triedy
- `identifikatorRetazca` - meno reťazca
- `hodnota` – hodnota; môže to byť reťazcový literál, iný reťazec, metóda, ktorá vráti reťazec

Príklady:

```
String slovo = "Ahoj";
String noveSlovo = slovo;
```

Poznámka pre programátorov v C++: *slovo* (rovnako aj *noveSlovo*) je v skutočnosti ukazovateľ (pointer) v zásobníku, veľkosti 4B, v ktorom je uložená adresa bajtu prvého znaku reťazca. Samotný reťazec `"Ahoj"` je uložený na halde. V Jave je pojem ukazovateľ nahradený pojmom odkaz.

12.3 Vstupy a výstupy

Do reťazca možno načítať aj postupnosť znakov z klávesnice. Využijeme pritom metódu **nextLine()** triedy *Scanner,* ktorú už čiastočne poznáme:

```
celaVeta = konzlovyVstup.nextLine();
```

Načíta sa celý riadok až po Enter.

Na výstup použijeme už klasicky metódu *println()*

```
System.out.println("Zadali ste znaky" + celaVeta);
```

12.4 Metódy pre prácu s reťazcami

Java ponúka viac než 60 metód pre prácu s reťazcami. Jednotlivé metódy aj s podrobným popisom možno nájsť napr. na stránke:

http://download.oracle.com/javase/7/docs/api/java/lang/String.html

Teraz sa zoznámime s niektorými užitočnými metódami.

Upozornenie! Žiadna z ďalej uvedených metód nemení pôvodný reťazec.

Metóda length() – dĺžka reťazca

Metóda vráti dĺžku reťazca.

```
String veta = "Ucime sa programovat v Jave";
veta.length() //metóda vráti hodnotu 27
```

Metóda equals(String r) – zhoda reťazcov

Metóda zisťuje, či sú dva reťazce zhodné. Ak sú, vráti *true*, inak vráti *false*

```
String slovo1 = "Dnes je pondelok";
String slovo2 = "Dnes je pondelok";
slovo1.equals(slovo2);                    //vráti true
slovo1.equals("Zajtra bude utorok");      //vráti false
```

Je možné použiť aj metódu **`equalsIgnoreCase(r)`** – nerozlišuje veľké a malé znaky.

Na porovnanie nemožno použiť zápis `slovo1 == slovo2`, lebo v tomto prípade by sa porovnávali odkazy, t.j. zisťovalo by sa, či oba odkazy odkazujú na ten istý objekt!

Metóda toLowerCase() – veľké písmená na malé

Metóda vráti reťazec, v ktorom budú všetky písmená malé.

```
String velkePismena = "VELKE PISMENA ZMENIME NA MALE";
velkePismena.toLowerCase(); //"velke pismena zmenime na male";
```

Metóda toUpperCase() – malé písmená na veľké

Metóda vráti reťazec, v ktorom budú všetky písmená veľké.

```
String malePismena = "male pismena zmenime na velke";
velkePismena.toUpperCase();  //"MALE PISMENA ZMENIME NA VELKE"
```

Metóda concat(String r) – spájanie reťazcov

Metóda spojí dva reťazce. Na spojenie reťazcov možno využiť aj **+** .

```
String ret1 = "Dobry ";
String ret2 = "den.";
String ret3 = ret1.concat(ret2);   //alebo String ret3 = ret1 + ret2;
```

Metóda replace(char z1, char z2) – náhrada znakov v reťazci

Metóda nahradí ľubovoľný znak (každý jeho výskyt) iným znakom.

```
String ret1 = "zlato";
String ret2 = ret1.replace('z','b');    //"blato"
```

Metóda charAt(int n) – vrátenie znaku

Metóda vráti n plus prvý znak reťazca.

```
String ret1 = "Ucime sa programovat v Jave";
char znak = ret1.charAt(6);   //do premennej znak vloží siedmy znak reťazca ret1,
                              //čiže znak 's'
```

Metóda substring(int index1, int index2) – vráti podreťazec

Metóda vráti časť pôvodného reťazca – od pozície index1, po pozíciu index2..

```
String ret1 = "Dnes je utorok";
String ret2 = ret1.substring(5,7); //"je"
```

Neskôr sa zoznámime aj s ďalšími dôležitými metódami, najmä tými, ktoré umožňujú vykonať konverziu medzi primitívnymi údajovými typmi a reťazcami.

Pre prácu s reťazcami existuje ešte ďalšia trieda s názvom **StringBuffer**. Tej sa tiež budeme venovať o niečo neskôr.

12.13 Aplikácia Retazce

Vytvoríme aplikáciu **Retazce**. V nej vytvoríme niekoľko reťazcov priradením, ale aj vstupom z klávesnice. Vyskúšame si tiež hore uvedené metódy.

12.14 Cvičenie

1. Vytvorte aplikáciu s názvom **EvidenciaKnihy.** Program si vyžiada zadanie nasledovných údajov: názov knihy, meno a priezvisko autora, rok vydania, názov vydavateľstva, cena knihy. Program potom zadané údaje vypíše, každý na samostatný riadok.
2. Vytvorte aplikáciu s názvom **Firmy**.
   Program si vyžiada zadanie nasledovných údajov:
   názov firmy
   adresu firmy
   rok založenia
   obrat v roku 2012 v mil. €
   Po načítaní údajov vypíše program tieto údaje pod seba nasledovne:
   Názov firmy Adresa firmy Rok založenia Obrat v roku 2012 Počet rokov
   Počet rokov je doba v rokoch od založenia firmy po dnes.

12.15 Otázky

1. Čo je to reťazec?
2. Ako sa volá trieda, ktorá umožňuje vytvoriť reťazec, a v ktorom balíku sa nachádza?
3. Čo znamená, že reťazce sú v Jave nemeniteľné?
4. Ako vyzerá zápis, ktorým vytvoríme reťazec?
5. Akú metódu používame pre vstup reťazca z klávesnice?
6. Uveďte aspoň tri metódy pre prácu s reťazcami.