Uvod u programiranje
dr Ninoslava Savić

Obeležene petlje

Obeležene petlje

- Obeležavanje petlje nekim identifikatorom omogućava da se programski tok usmeri na mesto u programu specificirano tim identifikatorom.
- Ako se iza naredbi break ili continue koristi identifikator tj. obeležje petlje, može se postići direktan izlazak izvan ugnježdene petlje na obeleženo mesto u programu
- Obeležena petlja ima sledeću sintaksu:
  ime_petlje: petlja

Obeležene petlje

Primer:

```
        ....
        int m = 20;
izvan:  for ( int i = 0; i < 10; i++) {
         while ( m < 50 ) {
              if (i * m++ > 100 )
                   break izvan;   // izlaz iz obe petlje!
              } // unutrašnji ciklus
        } // spoljašnji ciklus
        System.out.println("m = " + m );
        ....
```

Uvod u programiranje
dr Ninoslava Savić

Rad sa tekstom

- Klase String i StringBuffer

Rad sa tekstom klase String i StringBuffer

- Promenljive koje sadrže proizvoljni tekst (niz karaktera) zovemo string promenljive ili stringovi
- U Javi ne postoji prosti tip podatka za opis tekstualnih promenljivih
- Za deklaraciju takvih promenljivih služe klase String i StringBuffer, obe iz paketa java.lang
- Svi string literali se u Javi implementiraju kao objekti klase String
- String objekti su konstante i nakon kreiranja se ne mogu menjati

5

Tekstualne promenljive – klase String i StringBuffer

- Klasa StringBuffer je slična klasi String, ali se njeni objekti mogu menjati
- Objekti klase StringBuffer sadrže niz karaktera, ali se njegova dužina i sadržaj može menjati određenim metodama ove klase (metod *append(arg)*)

6

Deklaracija i inicijalizacija tekstualne promenljive – tipa String

- Deklaracija stringa
  String *ime_promenljive;*
- Deklaracija i inicijalizacija stringa
  String *ime_promenljive* = "vrednost";
- Vrednost string konstante
  piše se između navodnika

Primer:

```
String pozdrav = "Zdravo!";
```

7

Kreiranje stringa – 2. način

- Moguće je kreirati String objekat i uz pomoć operatora *new* i metoda konstruktora klase String, što je pravilo za kreiranje svih objekata u Javi

Primer :

```
String grad = new String("Novi Sad");
```

Primer :

```
char[] nizSlova = {'z','d','r','a','v','o'};
String pozdrav = new String(nizSlova);
.....
System.out.println(pozdrav);
```

8

String: dužina, učitavanje

- Dužina stringa dobija se pozivom metoda **length()** klase String:
  s.length( )

Primer:

```
String palindrom = "Ana voli milovana";
int duzina = palindrom.length();  //  17
```

- Učitavanje stringa može se izvršiti pomoću metoda nextLine() klase Scanner:

Primer:

```
Scanner ulaz = new Scanner(System.in);
System.out.println("Unesite rečenicu: ");
tekst = ulaz.nextLine();
```

9

Neke metode klase String

Sledeći metodi klase String vraćaju:

s.length() – dužinu stringa s
s1.concat(s2) – konkatenaciju stringova s1 i s2
s1+s2 – konkatenaciju stringova s1 i s2
s.charAt(poz) – znak na zadatoj poziciji poz
s.indexOf(s1) – poziciju podstringa s1 u stringu s
s.substring(p1,p2) – podstring od pozicije p1 do p2
s.substring(p1) – podstring od pozicije p1 do kraja s
s.compareTo(s1) – upoređivanje s i s1, metod vraća ceo broj (negativan, 0 ili pozitivan)

10

Neke metode klase String

s.lastIndexOf(s1) – poziciju poslednje pojave s1
s.replace(z1, z2) – zamena z1 sa z2 u celom s
s.startsWith(s1) – *true* ako s počinje znakom ili podstringom s1
String.valueOf(num) – prevodi broj u string
s.equals(s1) – upoređivanje sadržaja stringova s i s1
s.equalsIgnoreCase(s1) – upoređivanje sadržaja stringova s i s1 bez obzira na veličinu slova

11

Primer – konkatenacija (spajanje) stringova

```
String prefiks = "Moje godište je: ";
int godiste = 1985;
String tekst = prefiks + godiste;
                // + je znak za konkatenaciju
System.out.println(tekst);
```

Rezultat izvođenja je ispis:

```
Moje godište je: 1985
```

12

Primer – konkatenacija (spajanje) stringova

```
String ime = "Ana ";
String prezime = "Petrović";

String spoj = ime.concat(prezime);
```

ili:

```
String spoj = ime + prezime;
```

// Rezultujući tekst u stringu **spoj** je isti

```
System.out.print(ime);
System.out.println(prezime);
```

ili:

```
System.out.println(spoj);
```

/* Rezultat pri izvođenju prve dve naredbe ili samo treće je isti */

13

Primer – neke metode klase String

String rec = "Svi smo mi studenti";

- Pronalaženje pozicije podstringa:

```
int pozicija = rec.indexOf("smo");
```

// pozicija dobija vrednost 4
// prvi znak u stingu ima indeks 0, kao i kod nizova

- Određivanje podstringa na osnovu početne i krajnje pozicije

```
String sub1 = rec.substring(8,9);
```

// sub1 dobija vrednost "mi"

14

Primer – metode klase String

```
class PrimerStringa {
  public static void main (String [ ] args) {
    String a = "prirodno-matematicki";
    String b = new String("fakultet");       // moze i ovako
    System.out.println(a.length());          // 20 (duzina stringa)
    if  ( a.compareTo( b ) < 0 ) {
      System.out.println( a + " < " + b );
    }
    else if  ( a.compareTo( b ) == 0 ) {
       System.out.println( a + " == " + b );
    }
    else {
       System.out.println( a + " > " + b );      // ovo će biti
    } } }
```

15

Primer - metode klase String

```
System.out.println(b.substring(2));          // "kultet"
System.out.println(b.substring(2, 5));       // "kul"
System.out.println(a.charAt(4));             // 'o'
System.out.println(a.concat(b));  // "prirodno-matematickifakultet"
System.out.println(a + b);                   // isto sto i malopre
System.out.println(a.indexOf('o'));          // 4
System.out.println(a.indexOf("ma"));         // 9
System.out.println(a.indexOf("ma", 11));     // 13
System.out.println(b.lastIndexOf('t'));      // 7
System.out.println(b.replace('t', 'k'));     // "fakulkek"
System.out.println(a.startsWith("priro"));   // true
double d = 874.45 / 22.2;
System.out.println(String.valueOf(d));       // "39.38963963963964"
System.out.println("abc".length());          // 3
 }
}
```

16

Primer – Odštampati tekst sa obrnutim redosledom slova u odnosu na ucitani tekst

```
public class StringDemo {
    public static void main(String[ ] args) {
        String tekst = "Danas je lepo vreme";
        int duz = tekst.length( );
        char[ ] pomNiz = new char[duz];    // niz slova
        char[ ] noviNiz = new char[duz];

        for (int i = 0; i < duz; i++) {
            pomNiz[i] = tekst.charAt(i);
        }
        for (int i = 0; i < duz; i++) {
            noviNiz[i] = pomNiz[duz - 1 - i];
        }
        String obrnutiTekst = new String(noviNiz);
        System.out.println(obrnutiTekst);
    }
}
```

17

Zadatak za vežbu: Čitanje programskog koda

Šta radi sledeći deo programa?
Koje značenje imaju promenljive suma i broj ?

```
.....
int i = 0, suma = 0, broj = 0;
while ( i < 101) {
    if (( i%2 == 0) || ( i%3 == 0)) {
            suma += i;
            broj++;
    }
     i++;
}
```

Rešenje

Odgovor je u nastavku koda:

```
......
System.out.println
 ("Suma prvih 100 prirodnih brojeva koji su
   deljivi sa 2 ili sa 3 je " + suma);
System.out.println
 ("Ukupan broj takvih brojeva je " + broj);
....
```

Uvod u programiranje
dr Ninoslava Savić

Obrada izuzetaka

Obrada izuzetaka

- Greške u toku izvođenja Java programa (koje se ne mogu prepoznati prilikom prevođenja) su izuzeci
- Naziv potiče od imena klase Exception koja u Java biblioteci služi za upravljanje run-time greškama - neželjenim situacijama kao što su:
  - deljenje nulom
  - iznenadni prekid mrežne komunikacije
  - pristup nepostojećem elementu niza
  - nemoguća eksplicitna konverzija
  - ...

21

Sintaksa `try-catch` naredbe

- Upravljanje i obrada izuzetaka u Javi sprovodi se uz pomoć `try-catch` i `throw` naredbe
- Sintaksa `try-catch` naredbe

```
try {
   // deo koda kod koga je moguće da dođe do izuzetka
} catch(NazivKlaseIzuzetka NazivPromenjive) {
   // obrada prvog izuzetka
} catch(NazivKlaseIzuzetka NazivPromenjive) {
   // obrada drugog izuzetka
}
...
```

22

Primer `try-catch` naredbe

```
int i = 30;
try {

   int niz[] = new int[10];
   niz[i] = 7;

catch ( ArrayIndexOutOfBoundsException e ) {
      System.out.println
      ("Index niza je pogrešan!");
      e.printStackTrace();
}
```

Metod `printStackTrace()` ispisuje kod i opis konkretne greške pri izvođenju programa – izuzetka `e`

23

Primer

```
import java.util.Scanner;
public class Deljenje {
   public static void main(String[ ] args) {
         int brojilac, imenilac, kolicnik;
         Scanner consoleIn = new Scanner(System.in);
         System.out.println("Unesite brojilac: ");
         brojilac = consoleIn.nextInt( );
    System.out.println("Unesite imenilac: ");
    imenilac = consoleIn.nextInt( );
    try {
        kolicnik = brojilac/imenilac;
      System.out.print("kolicnik je: " + kolicnik);
        } catch (ArithmeticException e) {
          System.out.println
                ("Deljenje nulom nije dozvoljeno!");
      e.printStackTrace( ) }
     }
}
```

24