Programski jezik Java

Interno gradivo za predmet

Algoritmi in programski jeziki (4. letnik)

Naštevni podatkovni tipi

(neprečiščeno besedilo)

bostjan.vouk@tsc.si

Naštevni podatkovni tip

- ***Uporaba***
  - kadar nek podatek zavzame omejeno število različnih vrednosti
  - prevajalnik lahko preveri pravilnost pri prirejanju vrednosti
- ***Deklaracija***
  - ključna beseda enum
  - ime naštevnega tipa
  - seznam vrednosti, ki jih lahko zavzame

  public enum Ocena {A, B, C, D, NI_PISAL};
- Naštevni tip je razred
  - omogoča deklaracijo spremenljivk/atributov
  - vsaka spremenljivka/atribut je "objekt" naštevnega tipa
    - zavzame lahko le eno izmed naštetih vrednosti
    - kontrola tipa se izvaja že v času prevajanja

bostjan.vouk@tsc.si

Naštevni podatkovni tip

- Ostale značilnosti
  - naštevni tip je razširitev tipa java.lang.Enum
    - Enum je razred , ki pa sam ni naštevni tip
  - za naštevni tip ne obstaja javni konstruktor
    - s tem je preprečeno kreiranje dodatnih primerkov, ki niso bili deklarirani v času prevajanja
  - vrednosti naštevnega tipa so
    - public, static in final
  - vrednosti naštevnega tipa lahko primerjamo z ==, equals() in compareTo()
    - metoda compareTo() upošteva vrstni red pri naštevanju
  - na voljo sta metodi toString() in valueOf()
    - Ocena.NI_PISAL.toString() → vrne "NI_PISAL"
    - Ocena.valueOf("NI_PISAL") → vrne Ocena.NI_PISAL

bostjan.vouk@tsc.si

Naštevni podatkovni tip

- Zanka po vseh možnih vrednostih
  - metoda values() vrne tabelo vseh vrednosti naštevnega tipa

```
public void izpisiVse(){
for (Ocena oc:Ocena.values())
System.out.println(oc)
}
```

- Uporaba v stavku switch

```
switch (k.vrniOceno()){
case A: System.out.println("zelo primeren"); break;
case B: System.out.println("primeren"); break;
case C: System.out.println("manj primeren"); break;
case D: System.out.println("neprimeren"); break;
case NI_PISAL:
System.out.println(k.vrniOceno().toString());
}
```

bostjan.vouk@tsc.si

Naštevni podatkovni tip - primer

Primer Avtomobili:

-
```
enum IzdelovalciAvto{TOYOTA, HONDA, NISSAN, MITSUBISHI};
public class EnumPrimer{
public static void main(String[] args) {
    IzdelovalciAvto izd = IzdelovalciAvto.TOYOTA;
    System.out.println("Izbral si " + izd);
}
}
```
- Izpis: Izbral si TOYOTA
- Primer napačne prireditve -> ~~IzdelovalciAvto izd = TOYOTA;~~

bostjan.vouk@tsc.si

Naštevni podatkovni tip

- Deklaracija brez poznavanja naštevnega podatkovnega tipa
  - Konstante, ki predstavljajo poljubne like
    - public static final int PRAVOKOTNIK = 0;
    - public static final int KROG = 1;
    - public static final int PREMICA = 2;
    - ...
  - int risanje = PRAVOKOTNIK;
    - Lahko priredimo poljubno celo število
  - risanje = 99;
    - Nelogično vendar prevajalnik to dovoljuje

bostjan.vouk@tsc.si

Definiranje naštevnega tipa

- Deklariranje naštevnega tipa
  - public enum Lik { PRAVOKOTNIK,KROG,PREMICA}
  - Deklariranje spremenljivke naštevnega tipa
    - Lik risanje = Lik.PRAVOKOTNIK;
      - priredimo lahko samo vrednost oblike
  - Izpis vrednosti naštevnega tipa
    - System.out.println(risanje);
      - Izpiše pravokotnik

Naštevni podatkovni tip

- Izpis vseh naštevnih tipov s pomočjo for-each zanke
- for (Lik shp : Lik.values()) {
  System.out.println(shp);
  }

- switch (risanje) {
  case PRAVOKOTNIK:
  g.drawRect(x, y, width, height);
  break;
  case KROG :
  g.drawOval(x, y, width, height);
  break;
  case PREMICA:
  g.drawLine(x, y, x+width,y+ height);
  break;
  }

Naštevni podatkovni tip

- Izpis ordinalne vrednosti naštevnega tipa
- System.out.println(risanje.ordinal());
  - Izpiše 0 za pravokotnik.
- valueOf()
  - Metodo valueOf() lahko uporabimo za pretvorbo niza v naštevni podatkovni tip.
  - Primer branja v spremenljivko tipa Lik s pomočjo Scanner-ja
  - risanje = Lik.valueOf(vhod.next());

Iz naštevnega / v naštevni tip

- Deklaracija
  - *public enum Pasma {SAMOJED, DALMATINEC, LABRADOREC, DOBERMAN};*
- Iz naštevnega tipa v ordinalno vrednost
  - Pasma pes= Pasma.LABRADOREC;
  - int x = pes.ordinal();
    - X->2
- Iz ordinalne vrednosti v naštevni tip
  - int y = 2;
  - Pasma pes = Pasma.values()[y];
    - pes->LABRADOREC
    - System.out.println(pes);

bostjan.vouk@tsc.si

Primer

```
public class PrimerPasme{
public enum Pasma {SAMOJED,
        DALMATINEC, LABRADOREC,
        DOBERMAN};
public static boolean lap(Pasma p) {
switch (p)
{
case SAMOJED: return true;
case DALMATINEC:
case LABRADOREC:
default: return false;
}
}
```

```
public static void main(String[] args) {
Pasma pes = Pasma.LABRADOREC;
System.out.println(pes);                //izpis LABRADOREC
System.out.println(lap(pes));           // false
System.out.println(pes.ordinal());      //izpis 2
System.out.println(Pasma.SAMOJED);      //izpis SAMOJED
if ( pes.compareTo(Pasma.DALMATINEC)>0){
System.out.println("LABRADOREC je za DALMATINCEM"); }
else { System.out.println("DALMATINEC je za LABRADOR");}
pes = null;
Pasma najPasma = Pasma.valueOf("Samojed".toUpperCase());
System.out.println(najPasma);
int i = 1;
Pasma mojaPasma = Pasma.values()[i]; // izpis DALMATINEC
System.out.println(mojaPasma);
System.out.println("Vse pasme");
for (Pasma p : Pasma.values())
        System.out.println(p);
}
}
```

bostjan.vouk@tsc.si

Advantages

- The advantages of enums over an array of Strings are:
  - You can use enums as case labels.
  - Methods come built in with enums to do do things like convert enums names to ordinals, and combos with enumset.
  - You can attach additional fields and code to enum constants.
  - enums are type safe. With Strings all your items in all categories are the same type. There is nothing to stop you from feeding a fruit category to an animal parameter.
  - You can compare enums quickly with ==. You don't need to use equals as you do with String.

bostjan.vouk@tsc.si

Disadvantages

- The disadvantages of enums over an array of Strings are:
  - You can't create new enum constants at run time. You must recompile. In contrast, it is easy to add another String to an array or ArrayList.
  - You have to create en entire new class for each enum.
  - You can't create derived enum classes with a few extra enum constants. You can fairly easily add a few more Strings to a List.

bostjan.vouk@tsc.si