13. enum in struct

1

NAŠTEVNI TIP
STRUKTURIRANI TIP

13. 1. Stavek enum

2

- naštevni podatkovni tip
  - vrednosti domene podatkovnega tipa naštejemo
- stavek enum omogoča, da ustvarimo nov naštevni tip
- na ta način lahko izboljšamo preglednost programov

enum – primer 01

3

PRIMER:

```
enum dnevi {pon, tor, sre, cet, pet, sob, ned};

dnevi dan = pon;
if (dan == pon)
    cout << "Danes imamo ORI." << endl;

if (dan>=pon && dan<=pet)
    cout << "delovni dan" << endl;
```

vsaki vrednosti domene tipa dnevi, se priredi celoštevilska vrednost:
`pon = 0, tor = 1, …, ned = 6`

še o enum – primer 02

4

- v splošnem, lahko vsaki vrednosti domene določimo ustrezen "indeks", ali pa samo nekaterim

PRIMER:

```
enum meseci {jan=1, feb, mar, apr, maj, jun,
             jul, avg, sep, okt, nov, dece};
meseci m1, m2;
// m1 dobi vrednost jan
m1 = jan;
// m2 dobi vrednost dec
m2 = dece;

if (m1 == jan)
    cout << "Zacetek leta!" << endl;
```

13. 2. Zapisi

5

- zapis je sestavljeni oz. strukturirani podatkovni tip
  - podobno kot polje
  - vsebuje več vrednosti **poljubnega** tipa
    - polja vsebujejo več vrednosti istega tipa
  - posameznim vrednostim v zapisu pravimo komponente ali člani

SINTAKSA:

```
struct imeTipa {
  // deklaracije komponent
  tip1 ime1;
  tip2 ime2;
  …
};
```

ustvarili smo nov podatkovni tip!

(ne spremenljivke)

Zapisi

6

- spremenljivko tipa zapis deklariramo, kot smo vajeni:
  - imeTipa imeSpremenljivke;
- do posamezne komponente dostopamo preko operatorja . (pika) in imena komponente, kot smo ga določili v definiciji tipa
- posamezna komponenta predstavlja spremenljivko danega tipa komponente

Zapisi – primer 03 (1/2)

7

```
struct student {
    string ime;
    int starost;
    string vpisnaSt;
};


int main()
{
    student s1;
```

Zapisi – primer 03 (2/2)

8

```
// preberimo podatke o studentu s1
cout << "Vnesi ime studenta: ";
getline(cin, s1.ime);
cout << "Vnesi starost studenta: ";
cin >> s1.starost;
cout << "Vnesi pisno stevilko studenta: ";
cin >> s1.vpisnaSt;

cout << "Podatki o studentu:" << endl;
cout << "Ime: " << s1.ime << endl;
cout << "Starost: " << s1.starost << endl;
cout << "Vpisna stevilka: " << s1.vpisnaSt << endl;
```

Še o zapisih

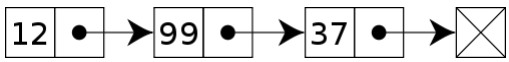
9

- zapise pogosto uporabljamo za hranjenje podatkov v podatkovnih bazah
- posplošitev koncepta zapisov so razredi
  - spomni se razreda string
  - komponente v razredih so spremenljivke in funkcije
  - v C++ je minimalna razlika med zapisi in razredi
- zapise/razrede uporabljamo za dinamične podatkovne strukture
  - strukture, ki spreminjajo svojo velikost po potrebi
  - pomagamo si s kazalci
  - primeri takih struktur: sklad, vrsta, seznam , drevo

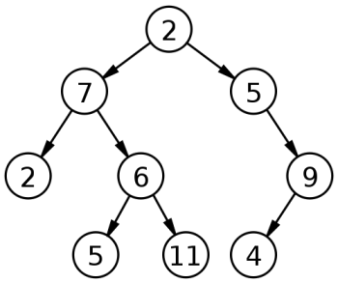
Primera dinamičnih struktur

10

- seznam



- drevo



Enojno povezan seznam – primer 04

- na kratko bomo predstavili enojno povezan seznam
  - več lahko izveste pri predmetu Podatkovne strukture

```
struct vozlisce { // tip vozlišča seznama
   int podatek;
   vozlisce *naslednji; //kazalec na naslednje vozlišče
};

vozlisce* prvi,* drugi;    // dva kazalca na vozlišče
prvi = new vozlisce;       // ustvari novo vozlišče, na katero kaže kazalec prvi
prvi -> podatek = 10 ;     // v komponento podatek shrani vrednost 10.
                           // enakovredno stavku (*prvi).podatek = 10;

drugi = new vozlisce;      // ustvari novo vozlišče, nanj kaže kazalec drugi
drugi -> podatek = 20;     // v komponento podatek shrani vrednost 20.
drugi -> naslednji = 0;    // kazalec iz 2. vozlišča kaže na 0.
prvi -> naslednji = drugi; // kazalec iz 1. vozlišča kaže na drugo vozlišče
```

Enojno povezan seznam – primer 05 (1/2)


```
#include <iostream>
using namespace std;

// tip vozlišča seznama
struct vozlisce {
   int podatek;
   vozlisce *naslednji; //kazalec na naslednje vozlišče
};

void vstavi (vozlisce* &seznam, int pod) {
   vozlisce* zac = new vozlisce;    //zac kaže na pomožno vozlišče
   zac->podatek = pod;              // vstavimo podatek v pomožno vozlišče

   // zac iz pomožnega vozlišča usmerimo na prvo vozlišče
   zac->naslednji = seznam;

   // kazalec začetka seznama usmerimo na novo vozlišče
   seznam = zac;
}
```

Enojno povezan seznam – primer 05 (2/2)


```
void izpis (vozlisce* seznam) {
   while (seznam != 0) {
         cout << seznam->podatek << " ";
         seznam = seznam->naslednji;
   }
   cout << endl;
}

int main() {
   vozlisce * prvi = 0;
   for (int  i = 0; i < 5; i++)
         vstavi (prvi, i);

    izpis (prvi);

   return 0;
}
```