Polje podataka (niz, poredak, *array*)

Polje podataka

- **podatkovna struktura**
- skup **istovrsnih** podataka sa **zajedničkim** simboličkim **imenom**
- svaki podatak – **element ili član** polja
- pristup podacima u polju moguć pomoću cjelobrojnog **indeksa** koji pokazuje redni broj člana
  - lakše pišemo izraze za izračunavanje
  - olakšavamo rukovanje većim brojem podataka
- kod obrade podataka u polju koristimo **petlju** (indeks za pristup članu polja mijenja se sa svakom iteracijom-ponavljanjem petlje)

Jednodimenzionalna polja

- članovi polja su u linearnom slijedu memorijskih lokacija, a indeks pojedinog člana polja odgovara udaljenosti od početnog člana
- **Deklaracija polja**
  - pokazuje tip, ime, dimenziju i broj članova u polju
  - rezervira prostor za sve članove polja:

```
float x[5];   //eksplicitna deklaracija
```

- broj članova u polju mora biti pozitivna cjelobrojna konstanta

Pridruživanje vrijednosti članovima polja

članovima polja vrijednosti se mogu pridružiti

- **<u>inicijalizacijom</u>**, npr. sa:

```
float x[5] = {1.2, 3., 0.1,-1.2e9,-1.};
```

- naredbama pridruživanja, npr.

```
x[0]=1.2;,
```

- učitavanjem jednog po jednog člana
- popunjavanjem polja petljom:

```
float x[5];
for (int i=0;i<=4;i++) x[i]=i*100;
```

kod inicijalizacije može se navesti i duljina polja, ako je članova manje od duljine polja svi preostali članovi postaju jednaki nuli

prvi član polja ima indeks 0

Višedimenzionalna polja

- 2D polja (dvodimenzionalna polja):
  - primjer: matrice u matematici
- 3Dpolja:
  - indeksi mogu biti dan, mjesec i godina (vizualno predstavljamo blokom)...
- višedimenzionalna polja
  - nema potrebe za vizualizacijom (podaci su ionako u nizu memorijskih lokacija)
- za obradu se koriste višestruke ugniježđene petlje
- Primjer deklaracije: *int Tablica[3][5];*
- članove 2D polja (matrice) dohvaćamo preko dva indeksa (za redak i za stupac)

Deklaracija, inicijalizacija, ispis

- Članovi se mogu inicijalizirati prilikom deklaracije:

  int tablica[3][5] = {{11,12,13,14,15},

  {21,22,23,24,25}}; //po retcima
- *Ispis:cout << tablica[0][0] << endl;*
- Nenavedeni članovi implicitno se inicijaliziraju na nulu. Implicitno se inicijaliziraju i nenavedene vrijednosti za broj redaka/stupaca
- potrebna veličina memorije jednaka je produktu dimenzija (preveliki zahtjevi uzrokuju pogrešku kod izvođenja – ne kod kompilacije)
- preporuka: deklarirana veličina polja mora biti malo veća od najveće očekivane veličine polja

Napomene

- Za fleksibilniji program maksimalna duljina polja obično se definira kao const (simbolička konstanta), koja mora biti poznata prije deklaracije polja, npr. sa

  const int N=10; double b[N];

- **Navede li se preveliki ili negativni indeks, prevoditelj neće javiti pogrešku i pristupit će se memorijskoj adresi koja je izvan područja rezerviranog za polje**