12. Večdimenzionalna polja

1

MATRIKE
VEČDIMENZIONALNA POLJA

12. Večdimenzionalna polja

2

- C++ dovoljuje polja elementov poljubnega tipa
- **večdimenzionalna polja**
  - polje, katerega elementi so polja
- *statična* večdimenzionalna polja deklariramo na naslednji način:
  - ob deklaraciji podamo velikost za vsako dimenzijo polja
    - torej podobno kot pri enodimenzionalnih poljih

SINTAKSA DEKLARACIJE:

$$\texttt{tip ime[}d_1\texttt{][}d_2\texttt{]..[}d_n\texttt{];}$$

Dostop do elementov

3

- do posameznega elementa dostopamo preko indeksov
- za vsako dimenzijo v oglatih oklepajih zapišemo indeks

PRIMER:

$ime[i_1][i_2]\ldots[i_n] = \ldots$

Primer

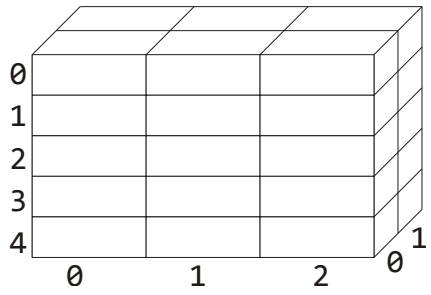
4

```
// dvodimenzionalno polje (matrika) realnih
// števil dimenzij 6x4, elementi so indeksirani
// od (0,0) do(5,3)

double matrika[6][4];

// element v vrstici z indeksom 3 in stolpcu z
// indeksom 2 dobi vrednost 0.56
matrika[3][2] = 0.56;
```

Primer večdimenzionalnega polja

5

```
int polje[5][3][2];
```



Primer: delo z matriko (1/4)

6

```
/*
    Program napolni matriko dimenzije 8x8
    po vrsti z vrednostmi od 1 do 64. Nato
    izpiše vrednosti iz matrike po vrsticah.
    Na koncu izpiše vrednosti iz matrike
    na naslednji način, glede na ostanek pri
    deljenju s 3:
    * če je ostanek enak 0, izpiše $
    * če je ostanek enak 1, izpiše +
    * če je ostanek enak 2, izpiše &
*/
#include <iostream>
#include <iomanip>
using namespace std;
```

Primer: delo z matriko (2/4)

7

```
void napolni(int mat[8][8]) {
  for(int v=0; v<8; v++)   // po vrsticah
    for(int s=0; s<8; s++) // po stolpcih
      mat[v][s] = v*8 + s + 1;
}

void izpis(int mat[8][8]) {
  for(int v=0; v<8; v++) {
    for(int s=0; s<8; s++)
      cout << setw(2) << mat[v][s] << " ";
    cout << endl;
  }
}
```

Primer: delo z matriko (3/4)

8

```
void kripticni_izpis (int mat[8][8]) {
  for (int v = 0; v < 8; v++) {
    for (int s = 0; s < 8; s++)
      switch (mat[v][s]%3 ) {
        case 0: cout << "$ "; break;
        case 1: cout << "+ "; break;
        case 2: cout << "& "; break;
      }
    cout << endl;
  }
}
```

Primer: delo z matriko (4/4)

9

```
int main() {
  int sahovnica[8][8];
  napolni(sahovnica);
  izpis(sahovnica);
  kripticni_izpis(sahovnica);

   return 0;
}
```

Matrike kot parametri funkcij

10

- kadar večdimenzionalno polje uporabimo za formalni parameter funkcije
  - podamo vse dimenzije (kot prejšnji primer)
  - podamo vse dimenzije, razen morda **prve** (naslednji primer)
    - v tem primeru podamo prvo dimenzijo kot parameter

Primer: matrika kot formalni parameter (brez prve dimenzije)


```
// funkcija vrne vsoto vrednosti elementov matrike
int vsota (int aM[][3], int aV) {
  int vs = 0
  for (int i = 0; i < aV; i++)
    for (int j = 0; j < 3; j++)
      vs += aM [i][j];
  return vs;
}

int main () {
  // pri deklaraciji lahko inicializiramo elemente
  int matrika [2][3] = {{1,2,3},{1,2,3}};
  cout << "Vsota je: " << vsota (matrika,2);
  ...
```