

Mátrixok kezelése mutatókkal

Kormányos Andor

Komplex Rendszerek Fizikája Tanszék

2022 október 11.

Adott egy $M \times N$ méretű mátrix, amit tárolni szeretnénk. Ehhez gyakran kényelmes vektorokat és mutatókat használni.

Adattípus definiálásához szükséges:

- két változó a méret tárolására: `cols`, `rows`
- egy $M \times N$ `double` tárolására elegendő memóriaterület
- a mátrixot tároló memória területre mutató `double *matr` pointer
- indexeljük a mátrix sorait az `i`, oszlopait a `j` változókkal

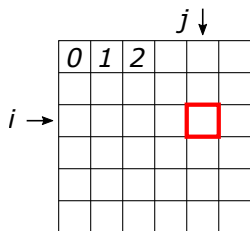
Mátrix tárolása vektorban

Adott egy $M \times N$ méretű mátrix, amit tárolni szeretnénk. Ehhez gyakran kényelmes vektorokat és mutatókat használni.

Adattípus definiálásához szükséges:

- két változó a méret tárolására: `cols`, `rows`
- egy $M \times N$ `double` tárolására elegendő memóriaterület
- a mátrixot tároló memória területre mutató `double *matr` pointer
- indexeljük a mátrix sorait az `i`, oszlopait a `j` változókkal

Mit kell `matr[...]` indexelésekor a zárójelek közé írni?



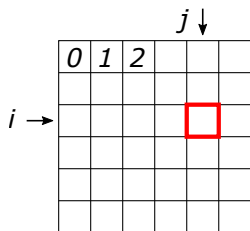
Mátrix tárolása vektorban

Adott egy $M \times N$ méretű mátrix, amit tárolni szeretnénk. Ehhez gyakran kényelmes vektorokat és mutatókat használni.

Adattípus definiálásához szükséges:

- két változó a méret tárolására: `cols`, `rows`
- egy $M \times N$ `double` tárolására elegendő memóriaterület
- a mátrixot tároló memória területre mutató `double *matr` pointer
- indexeljük a mátrix sorait az `i`, oszlopait a `j` változókkal

Mit kell `matr[...]` indexelésekor a zárójelek közé írni?



A mátrix soronkénti (row-major) tárolása esetén az i, j elem:

`matr[i * cols + j]`

A vektorok kezelésére használt függvényekhez hasonlóan:

```
1 double *alloc_matrix(int cols, int rows) { //memoria foglaló függvény
2     double *matr = (double*)malloc(cols * rows * sizeof(double));
3     if (matr == 0) {
4         printf("Memory allocation error.\n");
5         exit(-1);
6     }
7     return matr;
8 }
9
10 void read_matrix(FILE* f, double *m, int cols, int rows) { //matrix elemeit beolvasó függvény
11     for (int i = 0; i < rows; i++) {
12         for (int j = 0; j < cols; j++) {
13             fscanf(f, "%lf", &m[i * cols + j]); //egyszeru beolvasas fscanf-fel
14         }
15     }
16 }
17
18 void write_matrix(FILE* f, double *m, int cols, int rows) { //matrix kiirasa
19     for (int i = 0; i < rows; i++) {
20         for (int j = 0; j < cols; j++) {
21             fprintf(f, "%f ", m[i * cols + j]);
22         }
23         fprintf(f, "\n");
24     }
25 }
```