Многомерни масиви

доц. д-р Нора Ангелова

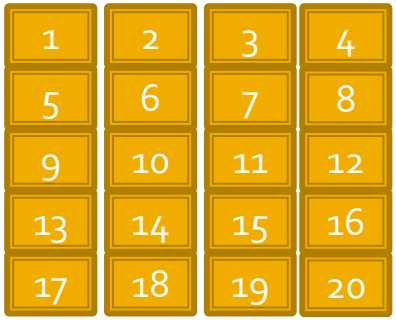
Масиви

- Едномерни масиви – редица от елементи от един и същ тип
- Двумерни масиви – масиви, чийто елементи са масиви от елементи

Многомерни масиви

- Дефиниция на двумерен масив

```
<T> <array_name>[<ROX_SIZE>][<COLUMN_SIZE>] [= {
  {<елемент1>, … },
  {<елемент2>, … },
  {…}
}; ]опц;
```



<T> ::= дефиниран тип
<array_name> ::= име на масив (идентификатор)
<ROW_SIZE>, <COLUMN_SIZE> ::= предварително дефинирани **константи**

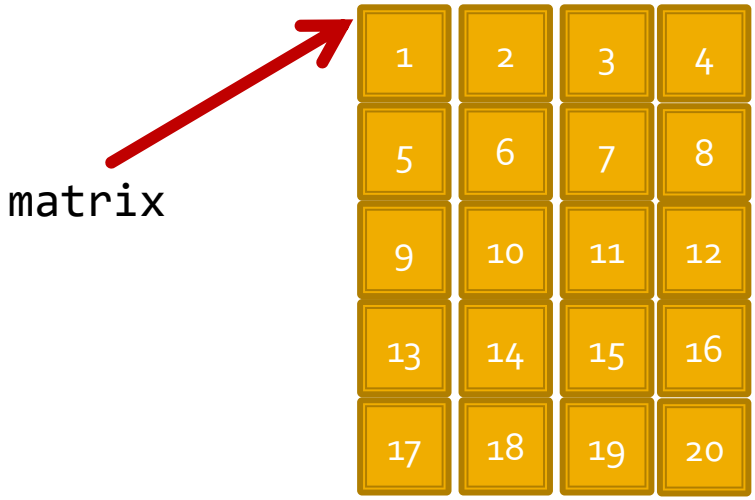
<елемент1>, <елемент2> ::= елементи от тип T

Многомерни масиви

- Пример

```
const int ROW_SIZE = 5;
const int COLUMN_SIZE = 4;

int matrix[ROW_SIZE][COLUMN_SIZE] = {
  { 1,  2,  3,  4},
  { 5,  6,  7,  8},
  { 9, 10, 11, 12},
  {13, 14, 15, 16},
  {17, 18, 19, 20}
};
```



Указатели и едномерни масиви

```
int arr[40] = {1, 2, 3, 4};
arr – указател към първи елемент на масив.
arr – съдържа адрес на arr[0].
```

```
arr == &arr[0]
```

Какъв е типът на arr?
int *

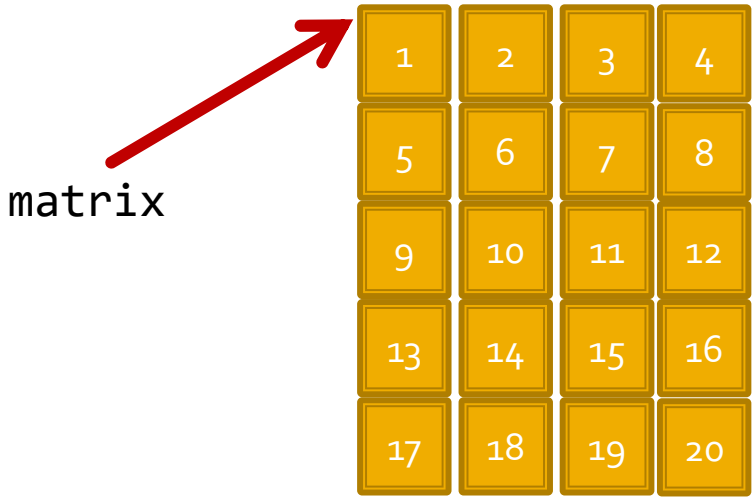
* Указателят е константен

Указатели и двумерни масиви

```
const int ROW_SIZE = 5;
const int COLUMN_SIZE = 4;
int matrix[ROW_SIZE][COLUMN_SIZE] = {
  { 1,  2,  3,  4},
  { 5,  6,  7,  8},
  { 9, 10, 11, 12},
  {13, 14, 15, 16},
  {17, 18, 19, 20}
};
```

matrix – указател към първи елемент на масива, който също е масив.

matrix – съдържа адреса на matrix[0].



* Указателят е константен

Указатели и двумерни масиви

```
const int ROW_SIZE = 5;
const int COLUMN_SIZE = 4;

int matrix[ROW_SIZE][COLUMN_SIZE] = {
  { 1,  2,  3,  4},
  { 5,  6,  7,  8},
  { 9, 10, 11, 12},
  {13, 14, 15, 16},
  {17, 18, 19, 20}
};
```

matrix[i] – масив от цели числа.

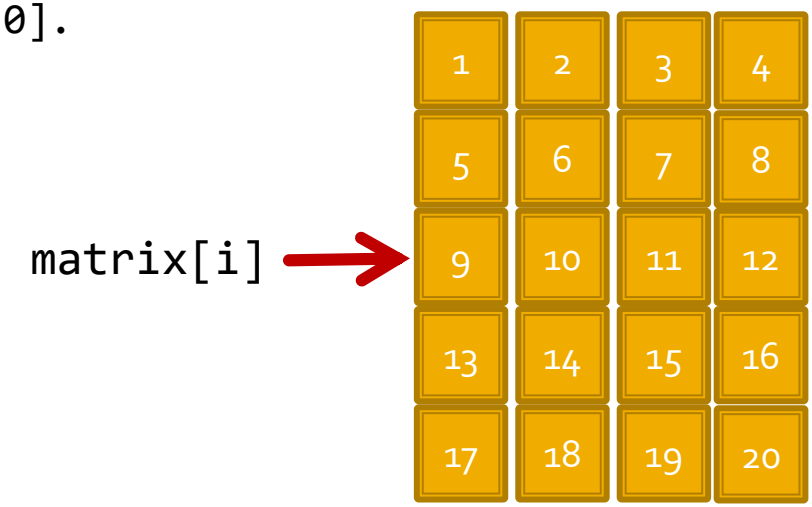
matrix[i] – съдържа адреса на matrix[i][0].

Какъв е типът на matrix?

<първи_елемент_на_масива>*

<масив_от_цели_числа>*

int**



* Указателят е константен

Задача:
Да се изведат стойностите на двумерен масив. Да се използват указателна аритметика.

```
const int ROW_SIZE = 5;
const int COLUMN_SIZE = 4;

int matrix[ROW_SIZE][COLUMN_SIZE] = {
  { 1,  2,  3,  4},
  { 5,  6,  7,  8},
  { 9, 10, 11, 12},
  {13, 14, 15, 16},
  {23, 24, 25, 26}
};

for(int rowIndex = 0; rowIndex < ROW_SIZE; ++rowIndex) {
  for(int colIndex = 0; colIndex < COLUMN_SIZE; ++colIndex) {
    std::cout << matrix[rowIndex][colIndex] << " ";
  }
  std::cout << std::endl;
}
```

Задача:
Да се изведат стойностите на двумерен масив.
Да се използват указателна аритметика.

```
const int ROW_SIZE = 5;
const int COLUMN_SIZE = 4;

int matrix[ROW_SIZE][COLUMN_SIZE] = {
  { 1,  2,  3,  4},
  { 5,  6,  7,  8},
  { 9, 10, 11, 12},
  {13, 14, 15, 16},
  {23, 24, 25, 26}
};

for(int rowIndex = 0; rowIndex < ROW_SIZE; ++rowIndex) {
  for(int colIndex = 0; colIndex < COLUMN_SIZE; ++colIndex) {
    std::cout << *(*(matrix + rowIndex) + colIndex) << " ";
  }
  std::cout << std::endl;
}
```

Указатели и масиви

```
int arr[40] = {1, 2, 3, 4};
int matrix[5][4] = {
  { 1,  2,  3,  4},
  { 5,  6,  7,  8},
  { 9, 10, 11, 12},
  {13, 14, 15, 16},
  {23, 24, 25, 26}
};
```

```
cout << arr << &arr[0];
cout << arr + 39;
cout << *(arr + 4) << arr[4]
cout << matrix;
cout << matrix + 2;
cout << *(matrix+3) + 1;
cout << *(*(matrix+4) + 3)
     << matrix[4][3];
```

Да се изведе на стандартния изход:

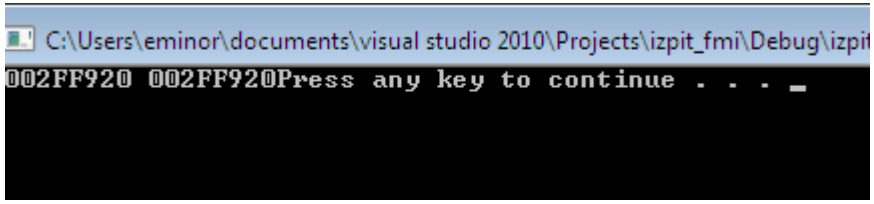
1. Адресът на първия елемент на arr;
2. Адресът на последния елемент на arr;
3. Петият елемент на arr;
4. Адресът на matrix;
5. Адресът на третия ред на матрицата matrix;
6. Адресът на елемента на четвърти ред и втори стълб на matrix;
7. Последният елемент на matrix;

Указатели и двумерни масиви

```
int arr[40] = {1, 2, 3, 4};
int matrix[5][4] = {
  { 1,  2,  3,  4},
  { 5,  6,  7,  8},
  { 9, 10, 11, 12},
  {13, 14, 15, 16},
  {23, 24, 25, 26}
};
```

1. Адресът на третия ред на матрицата matrix;

```
std::cout << matrix + 2;   std::cout << *(matrix + 2) + 0;
```



Указатели и двумерни масиви

```
int arr[40] = {1, 2, 3, 4};
int matrix[5][4] = {
  { 1,  2,  3,  4},
  { 5,  6,  7,  8},
  { 9, 10, 11, 12},
  {13, 14, 15, 16},
  {23, 24, 25, 26}
};
```

// Можем ли да отместим само с броя на елементите

1. Адресът на третия ред на матрицата matrix;

```
std::cout << matrix + 8;
```
ERROR

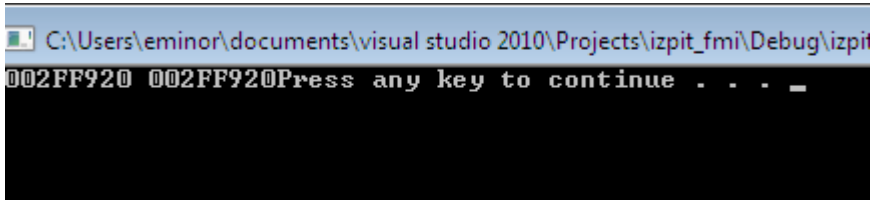
```
matrix + 8*sizeof(int*)
```
(not int)

Указатели и двумерни масиви

```
int arr[40] = {1, 2, 3, 4};
int matrix[5][4] = {
  { 1,  2,  3,  4},
  { 5,  6,  7,  8},
  { 9, 10, 11, 12},
  {13, 14, 15, 16},
  {23, 24, 25, 26}
};
```

1. Адресът на третия ред на матрицата matrix;

std::cout << *matrix + 8; std::cout << matrix + 2;



Масиви

- Едномерни масиви

- Двумерни масиви – масиви, чийто елементи са масиви от елементи
- Многомерни масиви – тримерни, четимерни, …

Многомерни масиви и функции

Въвеждане на елементите на двумерен масив (N x M)

```
for(int i = 0; i < N; i++) {
  for(int j = 0; j < M; j++) {
     std::cout << "matrix[" << i << "]["<< j << "]=";
     std::cin >> matrix[i][j];
  }
}
```

Задача

Дадена е квадратна матрица от низове.
Да се напише програмен фрагмент, който намира броя на палиндромите под главния диагонал (заедно с него).

Задача

Дадена е квадратна матрица от низове.
Да се напише програмен фрагмент, който намира броя на палиндромите под главния диагонал (заедно с него).

```
// Вариант 1

int br = 0;

char strMatrix[MAX_SIZE][MAX_SIZE][MAX_SIZE2];

//TODO Въвеждане на елементите на матрицата
for(int rowIndex = 0; rowIndex < MAX_SIZE; ++rowIndex){
  for(int colIndex = 0; colIndex <= rowIndex; ++colIndex) {

    // Дължина на низа
    int len = strlen(strMatrix[rowIndex][colIndex]);

    // Създаване на обърнат низ
    char revStr[MAX_SIZE2];
    for(int strIndex = len-1; strIndex >= 0; strIndex--) {
      revStr[len - strIndex - 1] = strMatrix[rowIndex][colIndex][strIndex];
    }

    revStr[len] = '\0';

    // Сравнение с обърнатия низ
    if (!strcmp(strMatrix[rowIndex][colIndex], revStr)) {
      br++;
    }
  }
}
```

Задача

Дадена е квадратна матрица от низове.
Да се напише програмен фрагмент, който намира броя на палиндромите под главния диагонал (заедно с него).

```
// Вариант 2

int br = 0;

char strMatrix[MAX_SIZE][MAX_SIZE][MAX_SIZE2];

// Въвеждане на елементите на матрицата

for(int rowIndex = 0; rowIndex < MAX_SIZE; rowIndex++){
  for(int colIndex = 0; colIndex <= rowIndex; colIndex++) {

    // Дължина на низа
    int len = strlen(strMatrix[rowIndex][colIndex]);

    // Директно сравнение
    …
  }
}
```

Задача

Дадена е квадратна матрица от низове.
Да се напише програмен фрагмент, който намира броя на палиндромите под главния диагонал (заедно с него).

```
// Вариант 3

int br = 0;

char strMatrix[MAX_SIZE][MAX_SIZE][MAX_SIZE2];

// Въвеждане на елементите на матрицата

for(int rowIndex = 0; rowIndex < MAX_SIZE; rowIndex++){
  for(int colIndex = 0; colIndex <= rowIndex; colIndex++) {

    // Дължина на низа
    int len = strlen(strMatrix[rowIndex][colIndex]);

    // Проверка с функция
    …
  }
}
```

Следва продължение...