# Двумерные статические массивы

# Двумерные статические массивы

Двумерный массив — это одномерный **массив** одномерных **массивов**

```
// объявление двумерного массива:
int a[3][4];

// объявление и  инициализация двумерного массива:
int b[2][3] = { { 1, 2, 3 },{ 4, 5, 6 } };
```

| | | | |
|---|---|---|---|
| a[0][0] | a[0][1] | a[0][2] | a[0][3] |
| a[1][0] | a[1][1] | a[1][2] | a[1][3] |
| a[2][0] | a[2][1] | a[2][2] | a[1][2] |

# Двумерные статические массивы

```
#include <iostream>
#include <iomanip>
using namespace std;
void main() {
// объявление и  инициализация двумерного массива:
int a[2][3] = { { 1, 2, 3 },{ 11, 22, 33 } };

// строки
for (int i = 0; i < 2; i++)
{
        // столбцы
        for (int j = 0; j < 3; j++)// переключение по столбцам
        cout << setw(4)<<right<<a[i][j];

cout << endl;
}
}
```

# Двумерные статические массивы. Расположение в памяти

Для двумерного С-массива выделяется единый блок памяти необходимого размера:

```
размер_массива1 * размер_массива2 * sizeof(тип_элемента_массива))
```

- Значения располагаются последовательно.
- Самый левый индекс изменяется медленнее всего.
- Имя (идентификатор) двумерного С-массива является указателем на первый элемент массива

# Двумерные статические массивы. Примеры

Работа с указателями

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

void main() {

int arr[2][3] = { {1, 2, 3},{4, 5, 6} };

        int *ptr = (int *)arr;

        for (int i = 0; i < 2; i++) {
                for (int j = 0; j < 3; j++) {
                        cout << setw(3) << *(ptr + i * 3 + j);
                }
                cout << endl;
        }

}
```

```
// Результат
// 1 2 3
// 4 5 6
```

# Двумерные статические массивы. Примеры

Работа с указателями

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

void main() {

int arr[2][3] = { { 1, 2, 3 },{ 4, 5, 6 } };

        int *ptr = (int *)arr;

                for (int i = 0; i < 2 * 3; i++) {
                cout << setw(4) << ptr[i];
        }

        cout << endl;
}
```

```
// Результат
// 1 2 3 4 5 6
```

# Двумерные статические массивы. Передача в функцию

```
const unsigned int D1 = 3;
const unsigned int D2 = 5;

void func(int arr[D1][D2]) { ... }    // (1)

//  При передаче многомерного С-массива в функцию можно
//  не указывать длину самого левого измерения.

void func(int arr[][D2]) { ... }      // (2)
```

# Двумерные статические массивы. Передача в функцию

Пример 1

```
#include <iostream>
#include <iomanip>
using namespace std;
const unsigned int D1 = 2;
const unsigned int D2 = 3;

void printArr(int a[D1][D2]) {
	for (int i = 0; i < D1; i++) {
		for (int j = 0; j < D2; j++)
			cout << setw(4) << right << a[i][j];
		cout << endl;
}}

void main() {
		int a[D1][D2] = { { 1, 2, 3 },{ 11, 22, 33 } };
		printArr(a);
}
```

# Двумерные статические массивы. Передача в функцию

Пример 2

```
#include <iostream>
#include <iomanip>
using namespace std;
const unsigned int D1 = 2;
const unsigned int D2 = 3;
// ... код для printArr

void changeArr(int a[][D2]) {
	for (int i = 0; i < D1; i++)
		for (int j = 0; j < D2; j++)
			a[i][j]*=10;
}
void main() {
	int a[][D2] = { { 1, 2, 3 },{ 11, 22, 33 } };
	changeArr(a);
	printArr(a);
}
```

```
//  Результат
//  10  20  30
// 111 220 330
```

# Определение типов

# Определение типов

Синонимы (псевдонимы)типов определяются с помощью **typedef**

```
typedef тип новое_имя[размерность];
```

```
typedef int arrayInt[3];      // arrayInt имя типа
typedef int matrixInt[2][3]; // matrixInt имя типа

// Использование matrixInt
// Объявление переменной matrixInt a будет заменено на int a[2][3]
void printArr(matrixInt a, int m, int n);
```

# Определение типов

Пример. matrixInt

```
#include <iostream>
#include <iomanip>
using namespace std;
const unsigned int D1 = 2;
const unsigned int D2 = 3;

typedef int matrixInt[2][3]; // matrixInt имя типа

void printArr(matrixInt a, int D1, int D2) {
	for (int i = 0; i < D1; i++) {
		for (int j = 0; j < D2; j++)
			cout << setw(4) << right << a[i][j];
	cout << endl;}}

void main() {
		matrixInt a = { { 1, 2, 3 },{ 11, 22, 33 } };
		printArr(a, D1, D2);
}
```