ПЗ-08: **Работа с двумерными массивами** (2 ч.)

(Двумерные (многомерные) статические массивы)

В **Си**, наряду с одномерными, существуют и многомерные массивы.

Например, двумерный массив: его можно представлять, как массив массивов, или как матрицу.

Размерность массива может быть и больше: трёхмерные, четырёхмерные и т.д.

Синтаксис остаётся прежним, добавляется только новая размерность

**<тип> <имя>[размерность1][размерность2]...;**

Например, двумерный массив: **int a[2][3];**

Трёхмерный массив: **int a[3][4][5];**

Доступ до элементов массива осуществляется также, как и в одномерном массиве.

|  |  |
| --- | --- |
| *#include <conio.h>*  *#include <stdio.h>*  *#define I1 7*  *#define I2 7*  *void main()*  *{*  *int M[I1][I2];*  *unsigned i, j;*  *for (i = 0; i < I1; i++)*  *for (j = 0; j < I2; j++)*  *M[i][j] = i \* j;*  *do {*  *printf("Enter indexes:\n");*  *printf("Exit: enter indexes > 7 \n");*  *scanf("%d", &i);*  *scanf("%d", &j);*  *if (i <= I1 && j <= I2)*  *printf("M[%d][%d] == %d\n", i, j, M[i][j]);*  *else break;*  *} while (1);*  *for (i = 0; i < I1; i++)*  *{*  *for (j = 0; j < I2; j++)*  *printf("\t%d", M[i][j]);*  *printf("\n");*  *} printf("For exit enter any key...\n");*  *getch();*  *}* |  |

Особенностью является то, что по своему строению многомерный массив является обыкновенным, "одномерным", массивом.

Все элементы расположены друг за другом. Доступ до элемента **a[i][j]** – по существу сдвиг на i\*число столбцов + j.

В двумерном массиве, таким образом, элементы расположены "по рядам", в трёхмерном - "по слоям", внутри которых элементы расположены "по рядам" и т.д.

В связи с этим, при начальной инициализации опускать размерность можно только в первых квадратных скобках:

**int a[ ][3] = {0};**

**int b[ ][5][25] = {0};**

Компилятор будет знать в таком случае сдвиг, необходимый для доступа к элементу.

С этим связаны и особенности начальной инициализации. Так как многомерный массив по сути одномерный, то его начальную инициализацию можно провести так

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

Можно опустить первую размерность

**int a[][2][3] = {1, 2, 3, 34, 5, 6, 7, 8, 9, 10, 11, 12};**

Можно с помощью фигурных скобок сделать данные более удобными для чтения

**int a[2][3] = {{1, 2, 3}, {4, 5, 6}}**

или

**int a[][3][4] =**

**{{{1, 2, 3, 4}, {2, 4, 6, 8}, {3, 6, 9, 12}},**

**{{1, 2, 3, 4}, {2, 4, 6, 8}, {3, 6, 9, 12}},**

**{{1, 2, 3, 4}, {2, 4, 6, 8}, {3, 6, 9, 12}}};**

Также, как и в одномерных массивах, если заявлено данных больше, чем указано при инициализации, то оставшиеся заполняются нулями.

Например, единичная матрица 3 на 3: **int zero3[3][3] = {{1}, {0, 1}, {0, 0, 1}};**

Из того, что многомерный массив является одномерным по структуре, вытекают некоторые интересные свойства. Например, доступ до элемента может быть осуществлён через его порядковый номер:

**a[i][j] === a[0][i\*число столбцов + j] и т.д.**

**Пример:**

Пользователь вводит 10 слов.

Вывести слово с максимальной длиной.

Программа внешне совершенно простая, единственная проблема - считывание и вывод слова.

Так как слова хранятся в двумерном массиве, то указатель на words[i][0] – это начало нового слова.

Также не забываем об ограничении на длину при вводе.

|  |  |
| --- | --- |
| *#include <conio.h>*  *#include <stdio.h>*    *#define SIZE 10*  *#define MAX\_LENGTH 128*    *void main() {*  *//Массив хранит 10 слов максимум по 128 символов*  *char words[SIZE][MAX\_LENGTH];*  *unsigned i, j, n, maxLength;*  *//Так как длина слова ограничена 127 символами, то типа char хватит*  *printf("Enter the number of words (indexes <= 10) :\n");*  *scanf("%d", &n);*  *unsigned char counter[SIZE];*    *for (i = 0; i <= n; i++)*  *{*  *//Считываем слова. words[i][0] - это символ, нам нужен*  *//адрес, начиная с которого можно писать в массив*  *fgets(&words[i][0], MAX\_LENGTH - 1, stdin);*  *j = 0;*  *//Считаем длину слова*  *while (words[i][j])*  *{*  *j++;*  *}*  *counter[i] = j;*  *}*    **Задачи (n <=10):**   1. Дан двумерный массив. Найти минимальную сумму элементов строки. 2. Определить координаты (индексы) элемента, наиболее близкого к среднему значению всех элементов массива. 3. Определить, имеются ли в двумерном массиве одинаковые элементы (количество). 4. Дан двумерный массив. Определить максимальный номер строки, состоящей только из элементов, кратных числу **а**. 5. Определить, есть ли в двумерном массиве столбец, в котором равное количество положительных и отрицательных элементов. 6. Определить, является ли квадратный массив симметричным относительно своей главной диагонали. 7. В квадратном массиве выделим четыре четверти, ограниченные главной и побочной диагоналями (без учета элементов, расположенных на диагоналях). Найти сумму элементов верхней четверти. 8. В каждом столбце двумерного массива поменять местами первый отрицательный элемент и последний нулевой. 9. Дан двумерный массив размером **n** х **n**. Сформировать одномерный массив из элементов заданного массива, расположенных на побочной диагонали. 10. Дан двумерный массив. Найти номера двух соседних столбцов, сумма элементов в которых минимальна. 11. Для каждой строки двумерного массива выяснить количество элементов, последняя цифра которых равна **а**. 12. Определить, является ли квадратный массив магическим квадратом, то есть массивом, в котором суммы элементов во всех строках, столбцах и на обеих диагоналях одинаковы.   *//Ищем слово с максимальной длиной*  *maxLength = counter[0];*  *j = 0;*  *for (i = 1; i <= n; i++)*  *{*  *if (counter[i] > maxLength)*  *{*  *maxLength = counter[i];*  *j = i;*  *}*  *}*  *//Выводим слово на печать. При выводе строки*  *//необходимо передавать указатель*  *printf("Max word:\n");*  *printf("%s", &words[j][0]);*  *getch();*  *}* |  |

