[НОУ ИНТУИТ | Лекция | Двумерные динамические массивы (intuit.ru)](https://intuit.ru/studies/courses/648/504/lecture/11452?page=1)

Лекция 27:

**Двумерные динамические массивы**

**A**

 |

[версия для печати](https://intuit.ru/intuit?destination=studies%2Fcourses%2F648%2F504%2Fprint_lecture%2F11452)

[< Лекция 26](https://intuit.ru/studies/courses/648/504/lecture/11451) || **Лекция 27**: **1**[2](https://intuit.ru/studies/courses/648/504/lecture/11452?page=2)[3](https://intuit.ru/studies/courses/648/504/lecture/11452?page=3) || [Лекция 28 >](https://intuit.ru/studies/courses/648/504/lecture/11453)

**Аннотация:**В лекции рассматриваются определение, объявления, инициализация двумерных динамических массивов, способы и этапы выделения и освобождения динамической памяти под двумерный массив, доступ к элементам двумерного динамического массива.

**Ключевые слова:**[освобождение памяти](https://intuit.ru/studies/courses/648/504/lecture/11452?page=1" \l "keyword1), [динамические массивы](https://intuit.ru/studies/courses/648/504/lecture/11452?page=1#keyword2), [одномерный массив](https://intuit.ru/studies/courses/648/504/lecture/11452?page=1#keyword3), [доступ](https://intuit.ru/studies/courses/648/504/lecture/11452?page=1#keyword4), [двумерный массив](https://intuit.ru/studies/courses/648/504/lecture/11452?page=1#keyword5), [массив](https://intuit.ru/studies/courses/648/504/lecture/11452?page=1#keyword6), [время выполнения](https://intuit.ru/studies/courses/648/504/lecture/11452?page=1#keyword7), [память](https://intuit.ru/studies/courses/648/504/lecture/11452?page=1#keyword9), [динамическое распределение памяти](https://intuit.ru/studies/courses/648/504/lecture/11452?page=1#keyword11), [указатель](https://intuit.ru/studies/courses/648/504/lecture/11452?page=1#keyword12), [значение](https://intuit.ru/studies/courses/648/504/lecture/11452?page=1#keyword13), [синтаксис](https://intuit.ru/studies/courses/648/504/lecture/11452?page=1#keyword16), [идентификатор](https://intuit.ru/studies/courses/648/504/lecture/11452?page=1#keyword17), [блок памяти](https://intuit.ru/studies/courses/648/504/lecture/11452?page=1#keyword18), [динамический массив](https://intuit.ru/studies/courses/648/504/lecture/11452?page=1#keyword25), [размерность массива](https://intuit.ru/studies/courses/648/504/lecture/11452?page=1#keyword32), [выражение](https://intuit.ru/studies/courses/648/504/lecture/11452?page=1#keyword33), [функция](https://intuit.ru/studies/courses/648/504/lecture/11452?page=1#keyword40), [нетипизированный указатель](https://intuit.ru/studies/courses/648/504/lecture/11452?page=1#keyword41), [адресация](https://intuit.ru/studies/courses/648/504/lecture/11452?page=2#keyword63), [MAS](https://intuit.ru/studies/courses/648/504/lecture/11452?page=2#keyword66), [индекс](https://intuit.ru/studies/courses/648/504/lecture/11452?page=2#keyword67), [SL](https://intuit.ru/studies/courses/648/504/lecture/11452?page=2#keyword68), [переменная](https://intuit.ru/studies/courses/648/504/lecture/11452?page=2#keyword70), [Единичная матрица](https://intuit.ru/studies/courses/648/504/lecture/11452?page=2#keyword73), [четность](https://intuit.ru/studies/courses/648/504/lecture/11452?page=2#keyword74), [mass](https://intuit.ru/studies/courses/648/504/lecture/11452?page=2#keyword75), [адрес](https://intuit.ru/studies/courses/648/504/lecture/11452?page=2#keyword76), [фактический параметр](https://intuit.ru/studies/courses/648/504/lecture/11452?page=2#keyword78), [определение](https://intuit.ru/studies/courses/648/504/lecture/11452?page=2#keyword79), [вывод](https://intuit.ru/studies/courses/648/504/lecture/11452?page=3#keyword97), [алгоритм](https://intuit.ru/studies/courses/648/504/lecture/11452?page=3#keyword98), [входные данные](https://intuit.ru/studies/courses/648/504/lecture/11452?page=3#keyword99), [диапазон](https://intuit.ru/studies/courses/648/504/lecture/11452?page=3#keyword100), [Алгебраическим дополнением](https://intuit.ru/studies/courses/648/504/lecture/11452?page=3#keyword102)

**Цель лекции**: изучить объявление, выделение и *освобождение памяти* под двумерные *динамические массивы*, обращение к элементам, научиться решать задачи с использованием двумерных динамических массивов на языке C++.

Под *двумерным массивом* понимается *одномерный массив*, элементами которого являются одномерные массивы. Другими словами, это набор однотипных данных, имеющий общее имя, и *доступ* к элементам которого осуществляется по двум индексам. Иногда *двумерный массив* также называют *матрицей*.

**Динамическим массивом** называют *массив* с переменным размером, то есть количество элементов может изменяться во *время выполнения* программы.

Для создания двумерного *динамического массива* вначале нужно распределить *память* для массива указателей на одномерные массивы, а затем выделить *память* для одномерных массивов. При *динамическом распределении памяти* для массивов следует описать соответствующий *указатель*, которому будет присвоено *значение* адреса начала области выделенной памяти.

**Объявление двумерных динамических массивов**

Под объявлением двумерного *динамического массива* понимают объявление двойного указателя, то есть объявление указателя на *указатель*.

*Синтаксис*:

Тип \*\* ИмяМассива;

ИмяМассива – *идентификатор* массива, то есть имя двойного указателя для выделяемого *блока памяти*.

Тип – тип элементов объявляемого *динамического массива*. Элементами *динамического массива* не могут быть функции и элементы типа void.

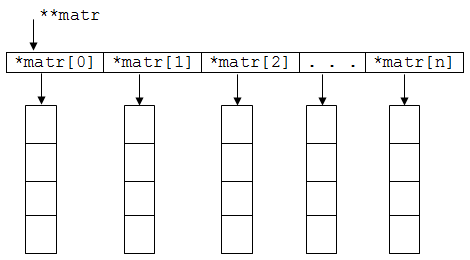
Например:

int \*\*a;

float \*\*m;

**Выделение памяти под двумерный динамический массив**

При формировании двумерного *динамического массива* сначала выделяется *память* для массива указателей на одномерные массивы, а затем в цикле с параметром выделяется *память* под одномерные массивы. На [рис. 26.1](https://intuit.ru/studies/courses/648/504/lecture/11452?page=1" \l "image.26.1) представлена схема динамической области памяти, выделенной под *двумерный массив*.



**Рис. 26.1.**Выделение памяти под двумерный массив

При работе с динамической памятью в языке С++ существует 2 способа выделения памяти под двумерный *динамический массив*.

1) *при помощи операции* new, которая позволяет выделить в динамической памяти участок для размещения массива соответствующего типа, но не позволяет его инициализировать.

*Синтаксис* выделения памяти под *массив* указателей:

ИмяМассива = new Тип \* [ВыражениеТипаКонстанты];

*Синтаксис* выделения памяти для массива значений:

ИмяМассива[ЗначениеИндекса] = new Тип [ВыражениеТипа

Константы];

ИмяМассива – *идентификатор* массива, то есть имя двойного указателя для выделяемого *блока памяти*.

Тип – тип указателя на *массив*.

ВыражениеТипаКонстанты – задает количество элементов (*размерность) массива*. *Выражение* константного типа вычисляется на этапе компиляции.

Например:

int n, m;//n и m – количество строк и столбцов матрицы

float \*\*matr; //указатель для массива указателей

matr = new float \* [n]; //выделение динамической памяти

под массив указателей

for (int i=0; i<n; i++)

matr[i] = new float [m]; //выделение динамической памяти

для массива значений

При выделении динамической памяти размеры массивов должны быть полностью определены.

2) *при помощи библиотечной функции* malloc (calloc), которая предназначена для выделения динамической памяти.

*Синтаксис* выделения памяти под *массив* указателей:

ИмяМассива = (Тип \*\*) malloc(N\*sizeof(Тип \*));

или

ИмяМассива = (Тип \*\*) calloc(N, sizeof(Тип \*));

*Синтаксис* выделения памяти для массива значений:

ИмяМассива[ЗначениеИндекса]=(Тип\*)malloc(M\*sizeof(Тип));

или

ИмяМассива[ЗначениеИндекса]=(Тип\*)calloc(M,sizeof(Тип));

ИмяМассива – *идентификатор* массива, то есть имя двойного указателя для выделяемого *блока памяти*.

Тип – тип указателя на *массив*.

N – количество строк массива;

M – количество столбцов массива.

Например:

int n, m;//n и m – количество строк и столбцов матрицы

float \*\*matr; //указатель для массива указателей

matr = (float \*\*) malloc(n\*sizeof(float \*));

//выделение динамической памяти под массив указателей

for (int i=0; i<n; i++)

matr[i] = (float \*) malloc(m\*sizeof(float));

//выделение динамической памяти для массива значений

Так как *функция* malloc (calloc) возвращает *нетипизированный указатель* void \*, то необходимо выполнять его преобразование в *указатель* объявленного типа.

**Освобождение памяти, выделенной под двумерный динамический массив**

Удаление из динамической памяти двумерного массива осуществляется в порядке, обратном его созданию, то есть сначала освобождается *память*, выделенная под одномерные массивы с данными, а затем *память*, выделенная под *одномерные массив* указателей.

*Освобождение памяти*, выделенной под двумерный *динамический массив*, также осуществляется 2 способами.

1) *при помощи операции* delete, которая освобождает участок памяти ранее выделенной операцией new.

*Синтаксис* *освобождения памяти*, выделенной для массива значений:

delete ИмяМассива [ЗначениеИндекса];

*Синтаксис* *освобождения памяти*, выделенной под *массив* указателей:

delete [] ИмяМассива;

ИмяМассива – *идентификатор* массива, то есть имя двойного указателя для выделяемого *блока памяти*.

Например:

for (int i=0; i<n; i++)

delete matr [i];

//освобождает память, выделенную для массива значений

delete [] matr;

//освобождает память, выделенную под массив указателей

Квадратные скобки [] означают, что освобождается *память*, занятая всеми элементами массива, а не только первым.

2) *при помощи библиотечной функции* free, которая предназначена для освобождения динамической памяти.

*Синтаксис* *освобождения памяти*, выделенной для массива значений:

free (ИмяМассива[ЗначениеИндекса]);

*Синтаксис* *освобождения памяти*, выделенной под *массив* указателей:

free (ИмяМассива);

ИмяМассива – *идентификатор* массива, то есть имя двойного указателя для выделяемого *блока памяти*.

Например:

for (int i=0; i<n; i++)

free (matr[i]);

//освобождает память, выделенную для массива значений

free (matr);

//освобождает память, выделенную под массив указателей

**[Дальше >>](https://intuit.ru/studies/courses/648/504/lecture/11452?page=2)**

**Обращение к элементам двумерного динамического массива**

*Адресация* элементов *динамического массива* осуществляется с помощью индексированного имени.

*Синтаксис*:

ИмяМассива[ВыражениеТипаКонстанты][ВыражениеТипаКонстанты];

или

ИмяМассива[ЗначениеИндекса][ЗначениеИндекса];

Например:

*mas*[5][7] – *индекс* задается как константа,

*sl*[i][j] – *индекс* задается как *переменная*,

array[4\*p][p+5] – *индекс* задается как *выражение*.

*Пример 1*. Сформируйте и выведите на экран *единичную матрицу* с целыми элементами, вводя ее порядок с клавиатуры.

#include "stdafx.h"

#include <iostream>

using namespace std;

int \_tmain(int argc, \_TCHAR\* argv[]){

int n,i,j;

int \*\*matr;//указатель для массива указателей

cout << "Input matrix order:";

cin >> n;

matr = new int \*[n];

//выделение памяти под массив указателей

for(i=0; i<n; i++){

matr[i] = new int[n];

//выделение памяти для массива значений

for (j=0; j<n; j++) //заполнение матрицы

matr[i][j] = (i==j ? 1 : 0);

}

cout << "Result: ";

for(i=0; i<n; i++){

cout << "\n";

for (j=0; j<n; j++)

cout << " " << matr[i][j];

delete matr[i];

//освобождение памяти из-под массива значений

}

delete [] matr;

//освобождение памяти из-под массива указателей

system("pause");

return 0;

}

*Пример 2*. Вычислить сумму элементов, лежащих на диагоналях матрицы N x N (обратить внимание на *четность*-нечетность числа N ). Размер массива должен задаваться пользователем с клавиатуры.

#include "stdafx.h"

#include <iostream>

using namespace std;

#include <time.h>

void gen (int nn,int a, int b,int \*\*\*mas);

//объявление функции генерации массива

int summa(int nn, int \*\*mas);

/\*объявление функции вычисления суммы заданных элементов массива\*/

void out (int nn,int \*\*mas);

//объявление функции вывода массива

int \_tmain(int argc, \_TCHAR\* argv[]){

int \*\*mass, n;

int s;

printf("Введите n: ");

scanf("%d",&n);

printf("\nГенерация массива \n");

gen(n,0,10,&mass);

s=summa(n,mass);

out(n,mass);

printf("\nСумма элементов = %d",s);

system("pause");

return 0;

}

void gen(int nn, int a, int b, int \*\*\*mas){

//функция генерации массива

int i,j;

srand(time(NULL)\*1000);

\*mas=(int\*\*)malloc(nn\*sizeof(int\*));

for (i=0;i<nn;i++){

(\*mas)[i]=(int\*)malloc(nn\*sizeof(int));

for (j=0;j<nn;j++)

(\*mas)[i][j]=rand()%(b-a)+a;

}

}

int summa(int nn, int \*\*mas) {

//функция вычисления суммы элементов диагоналей

int i,j, sum=0;

for (i=0;i<nn;i++)

for (j=0;j<nn;j++) {

if ((i==j) || (i==nn-j-1)) {

//нахождение элементов диагоналей

sum+=mas[i][j];

//суммирование элементов диагоналей

}

}

return sum;

}

void out (int nn,int \*\*mas){

//функция вывода массива

int i,j;

for (i=0;i<nn;i++) {

for (j=0;j<nn;j++)

printf("%4d",mas[i][j]);

printf("\n");

free (mas[i]);

}

free (mas);

}

В языке С++ предусмотрено использование указателя вида \*\*\**mass*. В данном примере в функцию генерации массива передается не *адрес* указателя, а его *значение*. Передача *фактического параметра* при вызове функции осуществляется через *определение* адреса указателя \*\**mass*.

**Ключевые термины**

**Двумерный динамический массив** – это *двумерный массив*, расположенный в динамической памяти.

**Динамический массив** – это *массив*, размер которого заранее не фиксирован и может меняться во время исполнения программы.

**Значение указателя на двумерный динамический массив** – это *адрес* массива указателей на одномерные массивы или *адрес* выделяемой области динамической памяти, если *двумерный массив* представляется как одномерный.

**Тип двумерного динамического массива** – это тип элементов массива.

**Указатель на двумерный динамический массив** – это *указатель* на *массив* указателей на одномерные массивы или на начало выделяемого участка динамической памяти, если *двумерный массив* представляется как одномерный.

**Краткие итоги**

1. При работе с массивами, размер которых заранее не известен, используют *динамические массивы*.
2. Под объявлением двумерного *динамического массива* понимают объявление двойного указателя, то есть объявление указателя на указатель. Если *двумерный массив* рассматривается как одномерный, то при объявлении такого массива объявляется указатель на переменную соответствующего типа.
3. Работа с двумерными динамическими массивами начинается с выделения участка памяти, а завершается освобождением ранее выделенного участка.
4. Выделение и *освобождение памяти* под двумерный *динамический массив* выполняется с помощью операций или функций для работы с динамической памятью.
5. Адресация элементов *динамического массива* осуществляется с помощью индексированного имени.