[НОУ ИНТУИТ | Лекция | Одномерные динамические массивы (intuit.ru)](https://intuit.ru/studies/courses/648/504/lecture/11451?page=1)

Лекция 26:

**Одномерные динамические массивы**

**A**

 |

[версия для печати](https://intuit.ru/intuit?destination=studies%2Fcourses%2F648%2F504%2Fprint_lecture%2F11451)

[< Лекция 25](https://intuit.ru/studies/courses/648/504/lecture/11450) || **Лекция 26**: **1**[2](https://intuit.ru/studies/courses/648/504/lecture/11451?page=2)[3](https://intuit.ru/studies/courses/648/504/lecture/11451?page=3) || [Лекция 27 >](https://intuit.ru/studies/courses/648/504/lecture/11452)

**Аннотация:**В лекции рассматриваются определение, объявления, инициализация одномерных динамических массивов, способы выделения и освобождения динамической памяти под одномерный массив, доступ к элементам одномерного динамического массива.

**Ключевые слова:**[время выполнения](https://intuit.ru/studies/courses/648/504/lecture/11451?page=1" \l "keyword1), [динамическое выделение памяти](https://intuit.ru/studies/courses/648/504/lecture/11451?page=1#keyword2), [стандарт языка](https://intuit.ru/studies/courses/648/504/lecture/11451?page=1#keyword3), [массив](https://intuit.ru/studies/courses/648/504/lecture/11451?page=1#keyword4), [память](https://intuit.ru/studies/courses/648/504/lecture/11451?page=1#keyword5), [динамическое распределение памяти](https://intuit.ru/studies/courses/648/504/lecture/11451?page=1#keyword6), [динамические массивы](https://intuit.ru/studies/courses/648/504/lecture/11451?page=1#keyword8), [язык программирования](https://intuit.ru/studies/courses/648/504/lecture/11451?page=1#keyword9), [встроенная функция](https://intuit.ru/studies/courses/648/504/lecture/11451?page=1#keyword10), [операции](https://intuit.ru/studies/courses/648/504/lecture/11451?page=1#keyword11), [динамический массив](https://intuit.ru/studies/courses/648/504/lecture/11451?page=1#keyword14), [синтаксис](https://intuit.ru/studies/courses/648/504/lecture/11451?page=1#keyword15), [идентификатор](https://intuit.ru/studies/courses/648/504/lecture/11451?page=1#keyword16), [блок памяти](https://intuit.ru/studies/courses/648/504/lecture/11451?page=1#keyword17), [значение](https://intuit.ru/studies/courses/648/504/lecture/11451?page=1#keyword20), [указатель](https://intuit.ru/studies/courses/648/504/lecture/11451?page=1#keyword21), [размерность массива](https://intuit.ru/studies/courses/648/504/lecture/11451?page=1#keyword29), [выражение](https://intuit.ru/studies/courses/648/504/lecture/11451?page=1#keyword30), [функция](https://intuit.ru/studies/courses/648/504/lecture/11451?page=1#keyword35), [нетипизированный указатель](https://intuit.ru/studies/courses/648/504/lecture/11451?page=1#keyword36), [освобождение памяти](https://intuit.ru/studies/courses/648/504/lecture/11451?page=1#keyword39), [адресация](https://intuit.ru/studies/courses/648/504/lecture/11451?page=2#keyword48), [статические массивы](https://intuit.ru/studies/courses/648/504/lecture/11451?page=2#keyword50), [MAS](https://intuit.ru/studies/courses/648/504/lecture/11451?page=2#keyword52), [индекс](https://intuit.ru/studies/courses/648/504/lecture/11451?page=2#keyword53), [SL](https://intuit.ru/studies/courses/648/504/lecture/11451?page=2#keyword54), [переменная](https://intuit.ru/studies/courses/648/504/lecture/11451?page=2#keyword56), [одномерный массив](https://intuit.ru/studies/courses/648/504/lecture/11451?page=2#keyword59), [адрес](https://intuit.ru/studies/courses/648/504/lecture/11451?page=2#keyword65), [вывод](https://intuit.ru/studies/courses/648/504/lecture/11451?page=3#keyword75), [входные данные](https://intuit.ru/studies/courses/648/504/lecture/11451?page=3#keyword76), [диапазон](https://intuit.ru/studies/courses/648/504/lecture/11451?page=3#keyword77), [число Фибоначчи](https://intuit.ru/studies/courses/648/504/lecture/11451?page=3#keyword78), [mass](https://intuit.ru/studies/courses/648/504/lecture/11451?page=3#keyword86)

**Цель лекции**: изучить объявления, выделения и освобождения памяти для одномерных динамических массивов, обращения к элементам, научиться решать задачи с использованием одномерных динамических массивов в языке C++.

При использовании многих структур данных достаточно часто бывает, что они должны иметь переменный размер во *время выполнения* программы. В этих случаях необходимо применять *динамическое выделение памяти*. Одной из самых распространенных таких структур данных являются массивы, в которых изначально размер не определен и не зафиксирован.

В соответствии со *стандартом языка* *массив* представляет собой совокупность элементов, каждый из которых имеет одни и те же атрибуты. Все эти элементы размещаются в смежных участках памяти подряд, начиная с адреса, соответствующего началу массива. То есть общее количество элементов массива и размер памяти, выделяемой для него, получаются полностью и однозначно заданными определением массива. Но это не всегда удобно. Иногда требуется, чтобы выделяемая *память* для массива имела размеры для решения конкретной задачи, причем ее объем заранее не известен и не может быть фиксирован. Формирование массивов с переменными размерами (динамических массивов) можно организовать с помощью указателей и средств *динамического распределения памяти*.

**Динамический массив** – это *массив*, размер которого заранее не фиксирован и может меняться во время исполнения программы. Для изменения размера *динамического массива* *язык программирования* С++, поддерживающий такие массивы, предоставляет специальные *встроенные функции* или *операции*. *Динамические массивы* дают возможность более гибкой работы с данными, так как позволяют не прогнозировать хранимые объемы данных, а регулировать размер массива в соответствии с реально необходимыми объемами.

**Объявление одномерных динамических массивов**

Под объявлением одномерного *динамического массива* понимают объявление указателя на переменную заданного типа для того, чтобы данную переменную можно было использовать как *динамический массив*.

*Синтаксис*:

Тип \* ИмяМассива;

ИмяМассива – *идентификатор* массива, то есть имя 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;

double \*d;

В данных примерах a и d являются указателями на начало выделяемого участка памяти. Указатели принимают *значение* адреса выделяемой области памяти для значений типа int и типа double соответственно.

Таким образом, при динамическом распределении памяти для динамических массивов следует описать соответствующий *указатель*, которому будет присвоено *значение* адреса начала области выделенной памяти.

**Выделение памяти под одномерный динамический массив**

Для того чтобы выделить *память* под одномерный *динамический массив* в языке С++ существует 2 способа.

1) *при помощи операции* new, которая выделяет для размещения массива участок динамической памяти соответствующего размера и не позволяет инициализировать элементы массива.

*Синтаксис*:

ИмяМассива = new Тип [ВыражениеТипаКонстанты];

ИмяМассива – *идентификатор* массива, то есть имя 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 \*mas;

mas = new int [100]; /\*выделение динамической памяти

размером 100\*sizeof(int) байтов\*/

double \*m = new double [n]; /\*выделение динамической

памяти размером n\*sizeof(double) байтов\*/

long (\*lm)[4];

lm = new long [2] [4]; /\*выделение динамической памяти

размером 2\*4\*sizeof(long) байтов\*/

При выделении динамической памяти размеры массива должны быть полностью определены.

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

*Синтаксис*:

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

или

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

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

Тип – тип указателя на *массив*.

N – количество элементов массива.

Например:

float \*a;

a=(float \*)malloc(10\*sizeof(float));

// или

a=(float \*)calloc(10,sizeof(float));

/\*выделение динамической памяти размером 10\*sizeof(float) байтов\*/

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

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

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

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

*Синтаксис*:

delete [] ИмяМассива;

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

Например:

delete [] mas; /\*освобождает память, выделенную под

массив, если mas адресует его начало\*/

delete [] m;

delete [] lm;

Квадратные скобки [] сообщают оператору, что требуется освободить *память*, занятую всеми элементами, а не только первым.

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

*Синтаксис*:

free (ИмяМассива);

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

Например:

free (a); //освобождение динамической памяти

**[Дальше >>](https://intuit.ru/studies/courses/648/504/lecture/11451?page=2)**

**Обращение к элементам одномерного динамического массива**

*Адресация* элементов *динамического массива* осуществляется аналогично адресации элементов *статического массива*, то есть с помощью индексированного имени.

*Синтаксис*:

ИмяМассива[ВыражениеТипаКонстанты];

или

ИмяМассива[ЗначениеИндекса];

Например:

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

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

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

*Пример 1*. Сформировать динамический *одномерный массив*, заполнить его случайными числами. Преобразовать *массив* таким образом, чтобы в первой его половине располагались элементы, стоявшие в чётных позициях, а во второй половине – элементы, стоявшие в нечётных позициях.

#include "stdafx.h"

#include <iostream>

using namespace std;

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

int \*a, n, i;

cout << "Введите n: ";

cin >> n;

cout << ' ';

a = new int [n]; //Выделение памяти под массив

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

cout << "Введите a[" << i << "]: ";

cin >> a[i];

cout << ' ';

}

int \*buf = new int [n];

//Выделение памяти под вспомогательный массив

int j = 0; //Индекс вспомогательного массива

for (i=0; i<n; i+=2) {

//Переписываем элементы с чётным индексом в новый массив

buf[j] = a[i];

j++;

}

for (i=1; i<n; i+=2) {

//Переписываем элементы с нечётным индексом в новый массив

buf[j] = a[i];

j++;

}

cout << "Преобразованный: " << ' ';

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

cout << buf[i] << ' ';

delete [] a; //Освобождаем память

delete [] buf;

system("pause");

return 0;

}

Отметим, что *указатель* на *массив* не обязательно должен показывать на начальный элемент некоторого массива. Он может быть сдвинут так, что начальный элемент будет иметь *индекс*, отличный от нуля, причем он может быть как положительным, так и отрицательным.

*Пример 2*.

#include "stdafx.h"

#include <iostream>

using namespace std;

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

float \*mas;

int m;

scanf("%d",&m);

mas=(float \*)calloc(m,sizeof(float));

//сейчас указатель q показывает на начало массива

mas[0]=22.3;

mas-=5;

/\*теперь начальный элемент массива имеет индекс 5,

а конечный элемент индекс n-5\*/

mas[5]=1.5;

/\*сдвиг индекса не приводит к перераспределению массива

в памяти и изменится начальный элемент\*/

mas[6]=2.5; // это второй элемент

mas[7]=3.5; // это третий элемент

mas+=5;

/\*теперь начальный элемент вновь имеет индекс 0,

а значения элементов q[0], q[1], q[2] равны

соответственно 1.5, 2.5, 3.5\*/

mas+=2;

/\*теперь начальный элемент имеет индекс -2, следующий -1,

затем 0 и т.д. по порядку\*/

mas[-2]=8.2;

mas[-1]=4.5;

mas-=2;

/\*возвращаем начальную индексацию, три первых элемента

массива q[0],q[1],q[2], имеют значения 8.2, 4.5, 3.5\*/

mas--;

/\*вновь изменим индексацию. Для освобождения области

памяти, в которой размещен массив q используется

функция free(q), но поскольку значение указателя q

смещено, то выполнение функции free(q) приведет к

непредсказуемым последствиям.

Для правильного выполнения этой функции указатель q

должен быть возвращен в первоначальное положение \*/

free(++mas);

system("pause");

return 0;

}

*Пример 3*. Задача Иосифа Флавия или считалка Джозефуса.

Задача в своей основе имеет легенду. Отряд из 41-го сикария, защищавший галилейскую крепость Массада, не пожелал сдаваться в плен блокировавшим его превосходящим силам римлян. Сикарии стали в круг и договорились, что каждые два воина будут убивать третьего, пока не погибнут все. Самоубийство – тяжкий грех, но тот, кто в конце концов останется последним, должен будет его совершить. Иосиф Флавий, командовавший этим отрядом, якобы быстро рассчитал, где нужно стать ему и его другу, чтобы остаться последними. Но не для того, чтобы убить друг друга, а чтобы сдать крепость римлянам. В современной формулировке задачи участвует n воинов и убивают каждого k -го. Требуется определить номера m и t начальных позиций двоих воинов, которые должны будут остаться последними.

#include "stdafx.h"

#include <iostream>

using namespace std;

void kill(int \*mass,int n,int i);

void krug(int \*mass,int n,int k, int i=0);

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

int n,k,\*mass,i;

FILE \*f;

f=fopen("input.txt","r");

fscanf(f,"%d %d",&n,&k);

fclose(f);

mass=(int \*)malloc(n\*sizeof(int));

for (i=0;i<n;i++) mass[i]=i+1;

f=fopen("output.txt","w");

fprintf(f,"Исходная нумерация: \n");

for (i=0;i<n;i++) fprintf(f,"%d ",mass[i]);

fclose(f);

krug(mass,n,k);

f=fopen("output.txt","a+");

fprintf(f,"\nОставшиеся в живых: \n");

for (i=0;i<k;i++) fprintf(f,"%d ",mass[i]);

fclose(f);

free(mass);

system("pause");

return 0;

}

void kill(int \*mass,int n,int i) {

int j;

for (j=i;j<n-1;j++)

mass[j]=mass[j+1];

}

void krug(int \*mass,int n,int k,int i) {

int ii;

if (n>k) {

ii=i+k-1;

if (ii>=n) ii=ii%n;

kill(mass,n,ii);

krug(mass,n-1,k,ii);

}

}

**Ключевые термины**

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

**Значение указателя на одномерный динамический массив** – это *адрес* выделяемой области динамической памяти.

**Одномерный динамический массив** – это *одномерный массив*, расположенный в динамической памяти.

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

**Указатель на одномерный динамический массив** – это *указатель* на начало выделяемого участка динамической памяти.

**Краткие итоги**

1. При работе с массивами, размер которых заранее не известен, используют *динамические массивы*.
2. Под объявлением одномерного *динамического массива* понимают объявление указателя на переменную заданного типа для того, чтобы данную переменную можно было использовать как *динамический массив*.
3. Работа с одномерными динамическими массивами начинается с выделения участка памяти, а завершается освобождением ранее выделенного участка.
4. Выделение и *освобождение памяти* под одномерный *динамический массив* выполняется с помощью операций или функций для работы с динамической памятью.
5. Адресация элементов *динамического массива* осуществляется с помощью индексированного имени.
6. Указатель на одномерный *динамический массив* не обязательно должен показывать на начальный элемент некоторого массива.