**Семинар по применению динамических массивов**

1. **Одномерные массивы**

**Определение**

Динамическим одномерным массивом называется одномерный массив, размер которого заранее не известен, а память по такой массив выделяется из области линамической памяти на этапе **выполнения программы**.

**Выделение памяти под несколько значений в С**

**void \* сalloc(size\_t n, size\_t size);**

**Освобождение памяти**

**void free(void \*block);**

**Пример:**

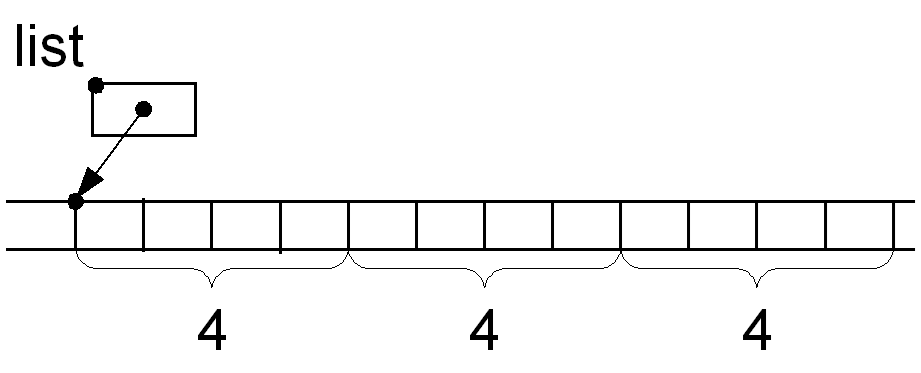
**int \*list;**

**list = (int \*) calloc(3,sizeof(int));**

**\*list=-244;**

**\*(list+1)=15;**

**\*(list+2)=-45;**

** …**

**free(list);**

**Операция выделения памяти для n значений С++:**

**<Указатель> =new <Имя типа>[<Количество>];**

**Операция освобождения памяти:**

**delete [ ] <Типизированный указатель>;**

**Пример:**

**int \*list;**

**list = new int [3];**

**\*list=-244; \*(list+1)=15; \*(list+2)=-45;**

**delete[ ] list;**

**Объявление массива:**

**int \*a; | int a[10];**

**a=calloc(10,4); |**

**a[2]=15; \*(a+2)=15;**

По правилам С++ имя массива является его адресом.

Поэтому, для нахождения положения элемета в одномерном массиве используется смещение этого элемента от начала массива, определяемого с помощью индекса. Формула расчета

**<Имя массива> + <номер индекса> \* <размер элемента>**

Именно по ней и выполняется расет. Однако, от пользователя этот расчет скрыт и ему достатоно правильно указать элемент.

Поэтому для адресации элементов массива независимо от способа описания можно использовать адресную арифметику:

**Адрес элемента (a+i) ⇔ &(a[i])**

**Элемент \*(a+i) ⇔ a[i]**

Для вывода адреса используется формат **%р**.

**Пример** вывода номера элемента, самого элемента и его адреса

**printf("%2d ----- %4d-----%p\n",i+1,\*(a+i),a+i);**

**Примеры обработки массивов**

**Задача 1.** **Создать одномерный дин. мас. В нем найти количество отрицательных элементов, кратных 3**

**поместить его на место элемента с ном.5**

**вывести адреса найд. отриц. элем. и массив до и после**

**// a={1,2,5,3,45,67,12}**

**// scanf(“%d”,a+i); -> &a[i]**

**// i=2**

**// \*(a+i)+4=9 к элементу с номером 2 добавили 4**

**// \*(a+i+4)=12 к адресу элемента добавили 4**

**(PrimDin1)**

**#include <stdio.h>**

**#include <stdlib.h>**

**#include <conio.h>**

**#include <time.h>**

**int \*a;**

**int i,j,n,kol3=0;**

**void main(int argc, char\* argv[])**

**{ srand((unsigned)time(NULL));**

**puts("Input n ");**

**scanf("%d",&n);**

**// создание массива**

**a=new int [n];**

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

**\*(a+i)=rand()%51-rand()%50;// a[i]**

**puts("Inputed Massiv");**

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

**printf("%4d",\*(a+i));**

**printf("\n");**

**// поиск элементов**

**puts(" Index Element Adress");**

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

**if((\*(a+i)<0)&&(\*(a+i)%3==0))**

**{ kol3++;**

**printf("%2d ------ %4d------%p\n",i+1,\*(a+i),a+i);**

**}**

**if(n>=5) // проверка, что 5 элемент есть**

**{**

**\*(a+4)=kol3;**

**printf("kol3= %4d \n",kol3);**

**puts("New Massiv after change element N 5");**

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

**printf("%4d",\*(a+i));**

**printf("\n");**

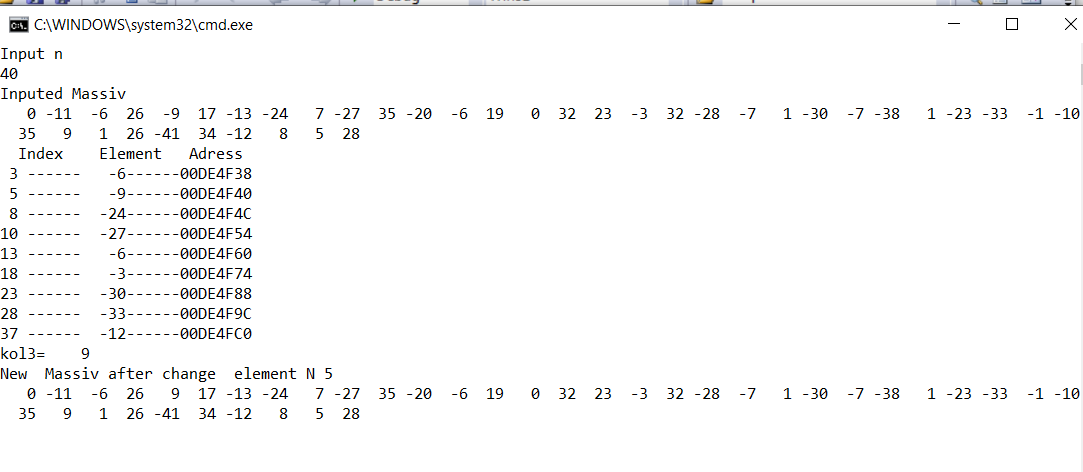
**}**

**else puts("Not Elem. N 5 ");**

**delete []a;**

**getch();**

**}**

****

**Расет адресов**

Возьмем для рачета числа

-6(инд. 3) с адресом 00DE4F38

-6 (инд. 13) с адресом 00DE4F60

От начала расположения элемента -6 с индексом 3 до элемента -6 с индексом 13 расположено 10 элементов целого типа (sizeof(int)=4).

**10\*4=40** байт.

Теперь посчитаем адреса

00DE4F60 - 00DE4F38=2816=2\*161+8\*160=**4010**.

Как показало сранениерезультаты расчета адресов и подсччет используемой памяти совпали. **Значит, в одномерном динамическом массиве элементы расположены подряд, как и в статическом массиве.**

**Задача 2. Создать одномерный дин. мас. В нем найти макс. сумму элементов и их индексы по схеме a[1]+a[n], a[2]+a[n-1],....**

**вывести адреса элем. составивщих махимальную сумму**

**(PrimDin2)**

**#include <stdio.h>**

**#include <stdlib.h>**

**#include <conio.h>**

**#include <time.h>**

**int \*a;**

**int i,j,n,msum,imax;**

**void main(int argc, char\* argv[])**

**{ srand((unsigned)time(NULL));**

**puts("Input n kratnoe 2");**

**scanf("%d",&n);**

**a=new int [n];// создание массива**

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

**\*(a+i)=rand()%50;**

**puts("Inputed Massiv");**

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

**printf("%4d",\*(a+i));**

**printf("\n");**

**// поиск нужных элементов и промежуточная печать**

**msum=-32000;imax=-1;**

**puts("-- Sum-- Ind\_1--Ind\_2");**

**for(i=0;i<(n/2);i++)**

**{ int s=\*(a+i)+\*(a+n-i-1);**

**printf("%6d %2d %2d\n",s,i+1,n-i);**

**if(s>msum)**

**{ imax=i;**

**msum=s;**

**}**

**}**

**printf("msum=%4d El1=%4d i1=%2d El2=%4d i2=%2d\n",msum,\*(a+imax),imax+1,\*(a+n-imax-1),n-imax);**

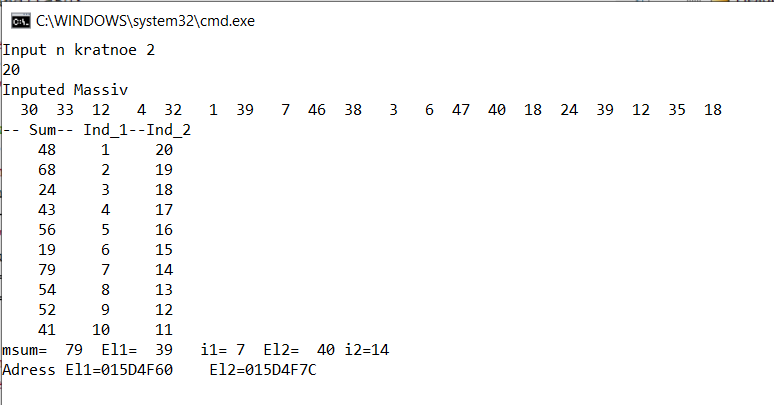
**printf("Adress El1=%p El2=%p\n",(a+imax),(a+n-imax-1));**

**printf("\n");**

**delete []a;**

**getch();**

**}**

****

**Задача 3.Создать одномерный дин. мас. В нем найти среднее ар. элем. стоящ. на четных местах. Сформировать новый массив и переписать в него все элементы, заменив все отриц.элем. средним ар.**

**вывести адреса четных элем. и все массивы**

**(PrimDin3)**

**#include <stdio.h>**

**#include <stdlib.h>**

**#include <conio.h>**

**#include <time.h>**

**float \*a,\*b,srch=0;**

**int i,j,kolch=0,n;**

**void main(int argc, char\* argv[])**

**{ srand((unsigned)time(NULL));**

**puts("Input n ");**

**scanf("%d",&n);**

**a=new float [n];**

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

**\*(a+i)=float(1.0\*rand()/1000-1.0\*rand()/1000);**

**puts("Inputed Massiv");**

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

**printf("%7.2f",\*(a+i));**

**printf("\n");**

**// поиск суммы элем.и их кол-ва На четных местах**

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

**{ srch+=(\*(a+i));**

**kolch++;**

**printf("%2d -- %6.1f--%p\n",i+1,\*(a+i),a+i);**

**}**

**srch=srch/kolch;**

**printf("\nsrarifm=%7.2f -- kol=%4d \n",srch,kolch);**

**b=new float [n];**

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

**if(\*(a+i)<0)**

**\*(b+i)=srch;**

**else**

**\*(b+i)=\*(a+i);**

**puts("NEW Massiv");**

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

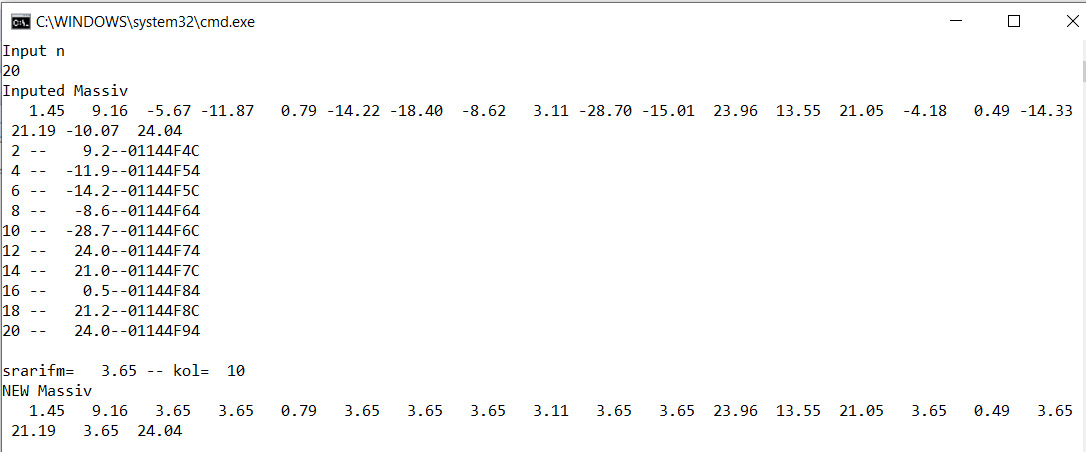
**printf("%7.2f",\*(b+i));**

**printf("\n");**

**delete []a;**

**getch();**

**}**

****

**Задача 4. Создать динамический массив указанного размера.**

**Удалить из массива все отрицательные элементы.**

**Вывести на печать исходный и полученный массивы.**

**masiv1b\_fun.cpp**

**#include <stdio.h>**

**#include <conio.h>**

**// Подпрограмма удаления**

**void Udal(int \*massiv,int &n)**

**{ int j,i;**

**j=-1;**

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

**if(\*(massiv+i)>=0)**

**{j++;**

**\*(massiv+j)=\*(massiv+i);**

**}**

**for(i=j+1;i<n;i++)\*(massiv+i)=0;**

**n=j+1;**

**}**

**// Подпрограмма печати**

**void PrintMas(int \*massiv,int n,char \*s)**

**{int \*p,i;**

**puts(s);**

**p=massiv;**

**for(i=0;i<n;i++)printf("%4d",\*p++);**

**printf("\n");**

**}**

**// Основная программа**

**void main()**

**{int \*mas,\*ptr,i,j,n;**

**puts("Input n");**

**scanf("%d",&n);**

**mas= new int [n];**

**printf("Input %3d Elementov\n",n);**

**ptr=mas;**

**for(i=0;i<n;i++)scanf("%d",ptr++);**

**PrintMas(mas,n," Inputed Massiv");**

**Udal(mas,n);**

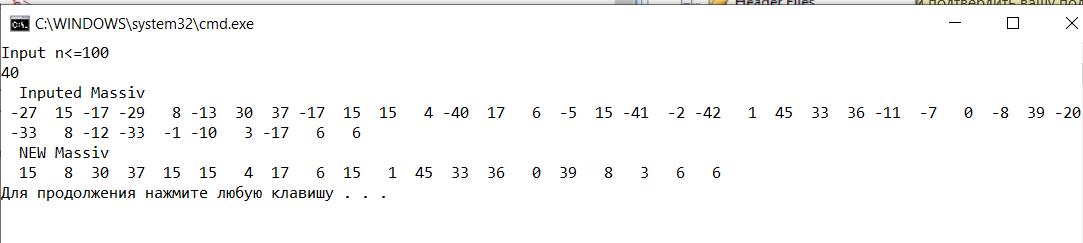
**if(n==0)puts(" New massiv empty");**

**else PrintMas(mas,n," NEW Massiv");**

**delete []mas;**

**getch();**

**}**



1. **Двумерные массивы**

**Определение**

Динамическим двумерным массивом называется двумерный массив, размер которого заранее не известен, а память по такой массив выделяется из области линамической памяти на этапе **выполнения программы**.

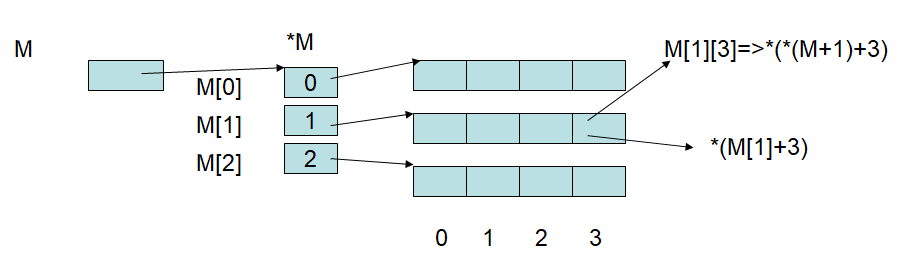
Двумерные динамиеские массивы описываются с помоцью указателя на указатель.

**<Тип элемента> \*\*<Имя матрицы>;**

Например

указатель

**int \*\*M**

указывает на массив указателей **int \*M**, каждый из элементов которого, в свою очередь, адресует одномерный массив элементов целого типа.

**Способы выделения памяти под динамический двумерный массив:**

**Для выделения памяти под массив в С** испльзуется функция **calloc**

**int \*\*M**

**int p,l;** // количество столбцов p и строк l;

**M=(int \*\*)calloc(l,sizeof(int\*));**

**for (i=0;i<l;i++)**

**\*(M+i)=(int \*)calloc(p,sizeof(int));**

**Для выделения памяти под массив в С++** испльзуется операция **new**

**M=new int\* [l];**

**for (i=0;i<l;i++)**

**(\*(M+i)=new int[p];**

Для нахождения положения элемета в двумерном массиве массиве тоже используется смещение этого элемента от начала массива, определяемого с помощью индексов. Ожнако, так как двумерный массив – это массив одномерных массивов, это смещение определяется с помощью двух индексов номером строки и номером элемкета в строке.

***Формула расчета***

**<Имя массива> + <номер индекса строки> \* <количество элементов в строке> \* <размер элемента> + <номер столбца> \* <размер элемента>.**

Именно по ней и выполняется расчет. Однако, от пользователя этот расчет скрыт и ему достатоно правильно указать элемент.

Поэтому для адресации элементов массива независимо от способа описания можно использовать адресную арифметику:

**Адрес элемента (\*(a+i)+j) ⇔ &(a[i][j])**

**Элемент \*(\*(a+i)+j) ⇔ a[i][j]**

Пример печати самого элемента и его адреса

**printf("%5d-----%p\n",\*(\*(a+i)+j),(\*(a+i)+j));**

**Пример обработки массивов**

**Задача 5.Создать динамический двумерный массив. Сформировать его вводом с клавиатуры или случайными числами заменить в матрице все отрицательные элементы их модулем. Пользуясь указателями вывести на экран их адреса**

**(PrinDinMatr)**

**#include <stdio.h>**

**#include <stdlib.h>**

**#include <time.h>**

**// ПОДПРОГРАММА ФОРМИРОВАНИЯ МАТРИЦЫ**

**int \*\*matrvv(int &l,int &p)**

**{int \*\*m;int i,j;**

**printf(" input size of massiv  \n");**

**scanf("%d %d",&l,&p);**

**printf(" input %4d strok iz %4d elementov\n",l,p);**

**m=new int\* [l];**

**for (i=0;i<l;i++)**

**{ m[i]=new int[p];**

**for (j=0;j<p;j++)**

**scanf("%3d",\*(m+i)+j);}**

**return m;**

**}**

**// ПОДПРОГРАММА ФОРМИРОВАНИЯ МАТРИЦЫ RAND**

**int \*\*matrrand(int &l,int &p,int xn,int xk)**

**{int \*\*m;int i,j;**

**printf(" input size of massiv  \n");**

**scanf("%d %d",&l,&p);**

**m=new int\* [l];**

**for (i=0;i<l;i++)**

**{ m[i]=new int[p];**

**for (j=0;j<p;j++)**

**\*(\*(m+i)+j)=rand()%xk+xn;}**

**return m;**

**}**

**// ПОДПРОГРАММА ПЕРЕФОРМИРОВАНИЯ МАТРИЦЫ**

**void sortmas(int \*\*m,int n,int l)**

**{int i,j;**

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

**{ printf("%3d stroka:\n",i+1);**

**for(j=0;j<l;j++)**

**if (\*(\*(m+i)+j)<0)**

**{**

**printf("%5d-----%p\n",\*(\*(m+i)+j),(\*(m+i)+j));**

**\*(\*(m+i)+j)=abs(\*(\*(m+i)+j));**

**}**

**}**

**}**

**int main(int argc, char\* argv[])**

**{int n,l,\*\*mat,i,j;**

**srand((unsigned)time(NULL));**

**mat=matrrand(n,l,-50,100);**

**printf("\n ===== inputed massiv ==== \n");**

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

**for(j=0;j<l;j++)**

**printf("%4d%c",\*(\*(mat+i)+j),(j==l-1)?'\n':' ');**

**sortmas(mat,n,l);**

**printf("\nperest massiv\n");**

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

**for(j=0;j<l;j++)**

**printf("%4d%c",\*(\*(mat+i)+j),(j==l-1)?'\n':' ');**

**// УДАЛЕНИЕ МАТРИЦЫ**

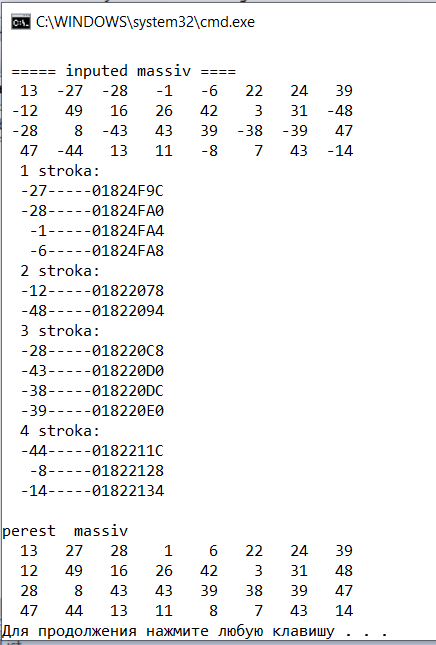
**for (i=0;i<n;i++)**

**delete [] mat[i];**

**delete [] mat;**

**return 0;**

**}**

****

**Проверка адресной арифметики**

**Проверка адресов в одной строке**

Для проверки вычисления адресов возьмем данные последнего теста

Выберем элементы первой строки

элемент -27 (01824F9С) и

элемент -6 (01824FA8)

Они находятся в одной строке. Между ними расположено 3 элемента целого типа (sizeof (int)=4). 3\* 4= 12 байт.

01824FA816 – 01824F9С16 = 0С16 = 0 \* 161 + 12 \* 160 = 1210

Как показал расчет адресов, мы тоже получили 12 байт. Это соответствует расположению элементов в одномерном динамическом массиве.

**Расчет адресов элементов одного столбца.**

Для проверки вычисления адресов возьмем данные последнего теста

элемент 28 (01824FA0) первый элемент третьего столбца и

элемент -43 (018220D0) третий элемент третьего столбца.

Между этими элементами расположено ровно 2 строки по 8 элементов (16 элементов) целого типа,

то есть **2\*8\*** s**izeof(int)** = 2\*8\*4 = 64 байта.

***Рассмотрим адреса***

Как видим, адрес третьего элемента столбца меньше, чем адрес первого. Это говорит об отрицательной разнице адресов, чего в статической матрицу быть не может.

018220D0-01824FA0= FFFF D13016=-**1198410**

Можно вычесть из большего адреса меньший, тогда разница будет положительной

01824FA0-018220D0=2ED016=

2\*163+14\* 162+13\*161 +0\*160=8192+3584+208=**11984.**

Но и этот результат тоже не равен 64.

Таким образом, смещение адресов и размещение элементов не совпадают, и разница может быть даже отрицательной. Это происходит потому, что каждый раз, когда система выделяет память под новую строку в цикле, блок память выбирается не последовательно предыдущему, поэтому элементы всей матрицы не располагаются в одном блоке памяти последовательно, как в случае статического двумерного массива.

**Задание для самостоятельной работы**

**Размеры массивово вводятся с клавиатуры.**

1. В одномерном массиве, состоящем из **N** вещественных чисел, вычислить сумму положительных элементов с нечетными номерами. Вывести на печать исходный массив и всю найденную информацию с комментариями.
2. В одномерном массиве, состоящем из **N** вещественных чисел, вычислить максимальное произведение положительных подряд идущих элементов. Вывести на печать исходный массив и всю найденную информацию с комментариями.
3. В одномерном массиве, состоящем из **N** целых чисел удалить из массива все кратные двум исходного массива. Вывести на печать исходный и сформированный массивы и всю найденную информацию с комментариями
4. В данной квадратной матрице порядка N , содержащей вещественные числа разных знаков, переместить наибольший элемент каждой строки на побочную диагональ. Вывести на печать матрицу на всех этапах преобразования.
5. В данной квадратной матрице порядка N, содержащей целые числа разных знаков, вычеркнуть строку, содержащую максимальный элемент на главной диагонали. Вывести на печать матрицу на всех этапах преобразования.