Лекция 7
Многомерные массивы

1. Объявление массива
2. Доступ к отдельным элементам массива
3. Инициализация элементов массива
4. Объем памяти, занятой массивом
5. Операции с матрицами

тип имя_массива
[*Размер1*][*Размер2*]...[*РазмерN*];

int D[10][20];

Двумерные массивы

Одномерный массив
int a[6];

Двумерные массивы

Построим массив, состоящий из четырех таких массивов

int a[6];

Двумерные массивы

Построим массив, состоящий из четырех таких массивов

int a[6];

Перенумеруем элементы, используя два индекса и соответственно опишем этот массив:

int a[4][6]

a[2][1]

Двумерные массивы

Построим массив, состоящий из четырех таких массивов

int a[6];

Перенумеруем элементы, используя два индекса и соответственно опишем этот массив:

int a[4][6]

В памяти двумерный массив вытянут по строкам

Многомерные массивы

Трехмерный массив

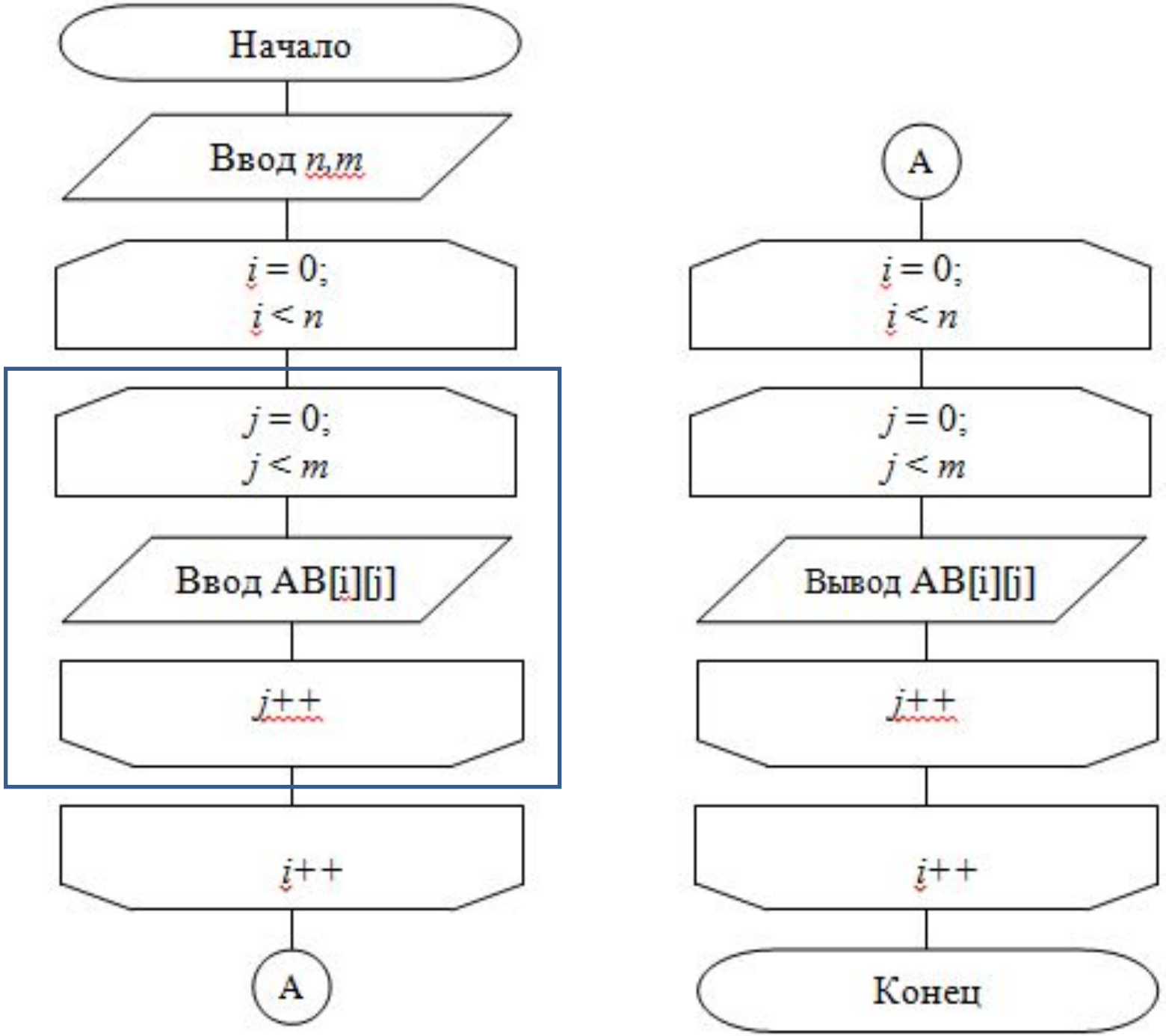
float x[2][3][4];

```
const int K=4, N=3, M=5;
int one[N];
int two[N][M];
int three[K][N][M];
for (int i=0;i<K;i++)
    one[i]=i*10;
for (int p=0;p<N;p++)
for (int q=0;q<M;q++)
    two[p][q]=p*q;
for (int i=0;i<K;i++)
for (int p=0;p<N;p++)
for (int q=0;q<M;q++)
    three[i][p][q]=one[i]+ two[p][q];
```

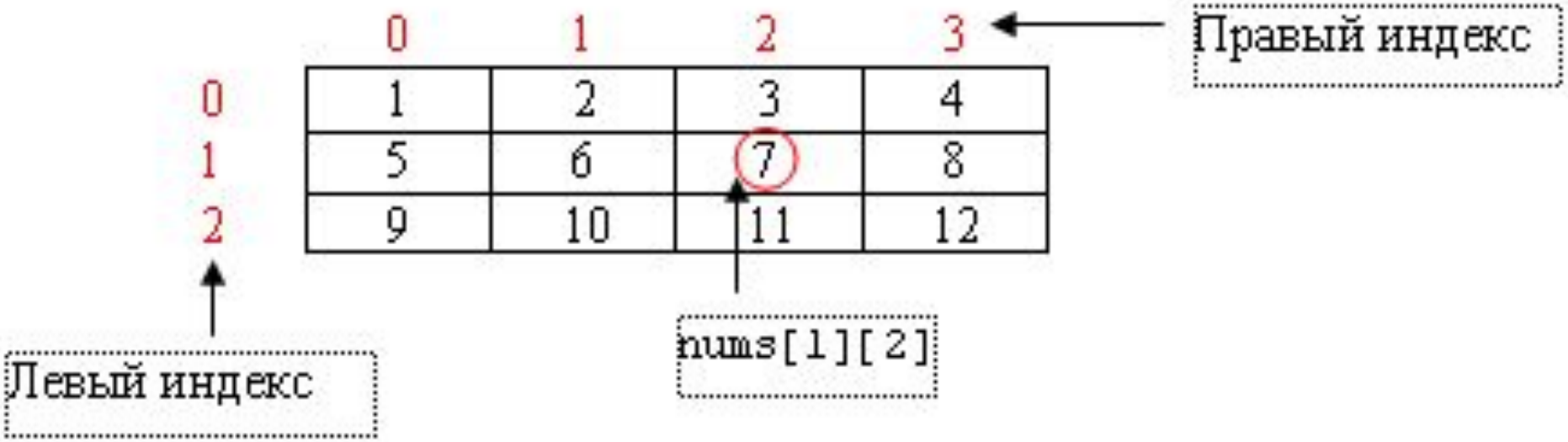
Ввод-вывод элементов двумерного массива

```
const int nm=20; //макс размер матрицы
int AB [nm][nm],m;
cin >> n >> m;
cout << "Элементы матрицы?\n";
for (int i=0;i<n;i++)
 for (int j=0;j<m;j++)
    cin >> AB[i][j];
```

```
//вывод матрицы
cout << "Введенная матрица" <<endl;
for (int i=0;i<n;i++)
{
 for (int j=0;j<m;j++)
   cout << AB[i][j] << " ";
   cout << endl;
}
```


Начало
Ввод n,m
i = 0;
i < n
j = 0;
j < m
Ввод AB[i][j]
j++
i++
A
A
i = 0;
i < n
j = 0;
j < m
Вывод AB[i][j]
j++
i++
Конец

```
int t, i, nums[3][4];
for(t=0; t < 3; ++t)
{
for(i=0; i < 4; ++i)
{
nums[t][i] = (t*4)+i+1;
cout<<nums[t][i]<<"\t";
}
cout<<"\n";
}
```


0
1
2
3
Правый индекс
0
1
2
1
2
3
4
5
6
7
8
9
10
11
12
Левый индекс
nums[1][2]

```
int sq[10][2] = {
1, 1,
2, 4,
3, 9,
4, 16,
5, 25,
6, 36,
7, 49,
8, 64,
9, 81,
10, 100
};
```

```
// How to initialize two-dimensional arrays

const int m=4, n =6;// maximum array dimensions

int a[m][n] = {{  0, 1,  2,   3,  4,  5},
                 {10,11,12,13,14,15},
                 {20,21,22,23,24,25},
                 {30,31,32,33,34,35}};

// third & fourth rows are set to zero
int b[m][n] = {0,1,2,3,4,5,10,11,12,13,14,15};

int d[m][n] = {0};   // the whole array is set to zero
```

```
// first dimension's size will be calculated by compiler
int c[ ][n]={
            { 0 , 1,  2,  3,  4,  5},
               {10,11,12,13,14,15},
               {20,21,22,23,24,25},
               {30,31,32,33,34,35}
            };
```

число_байтов = число_строк x
x число_столбцов x
x размер_типа_в_байтах

```
тип  имя[размер1] [размер2]... [размеры] ;
```

```
int  multidim[4] [10] [3] ;
```

```
// Магический квадрат
const int M=4; /* граница 2-го измерения массива*/
int mag[][M] ={
{16, 3,  2,   13},
{5, 10, 11,   8},
{9,   6,   7,  12},
{4,  15,14,    1}
};
// Вычисляем границу первого измерения:
int n =          /* граница 1-го измерения массива*/
sizeof(mag)       /* память под весь массив*/
/(sizeof(int)*M)     /* память, занятая строкой
массива*/;
for (int i=0;i<n;i++){
for (int j=0;j<M;j++) cout<<mag[i][j]<<"\t";
cout<<"\n"; // новая строка матрицы
```

16	3	2	13
5	10	11	8
9	6	7	12
4	15	14	1

http://www.pks.mpg.de/~flach/html/Magical_Square_Rus.HTM

Операции с матрицами

1. Сложение матриц - поэлементная операция

$$A_{mn}+B_{mn}=C_{mn}=\begin{vmatrix} a_{11}+b_{11} \; a_{12}+b_{12} \dots a_{1n}+b_{1n} \\ \dots\dots\dots\dots\dots\dots\dots\dots \\ a_{m1}+b_{m1} \; a_{m2}+b_{m2} \dots a_{mn}+b_{mn} \end{vmatrix}$$

```
for (int i=0;i<n;i++)
{
for (int j=0;j<m;j++)
{
c[i][j]= a[i][j]+ b[i][j];
cout<<c[i][j]<<"\t";
}
cout<< "\n";
}
```

2. Вычитание матриц - поэлементная операция

$$A_{mn} - B_{mn} = \begin{vmatrix} a_{11} - b_{11} & a_{12} - b_{12} \cdots & a_{1n} - b_{1n} \\ \cdots\cdots & \cdots\cdots & \cdots\cdots \\ a_{m1} - b_{m1} & a_{m2} - b_{m2} \cdots & a_{mn} - b_{mn} \end{vmatrix}$$

```
...
c[i][j]= a[i][j]- b[i][j];
...
```

3. Произведение матрицы на число – поэлементная операция

$$\lambda A = \begin{pmatrix} \lambda a_{11} & \lambda a_{12} & \dots & \lambda a_{1n} \\ \lambda a_{21} & \lambda a_{22} & \dots & \lambda a_{2n} \\ \dots & \dots & \dots & \dots \\ \lambda a_{m1} & \lambda a_{m2} & \dots & \lambda a_{mn} \end{pmatrix}$$

```
...
c[i][j]= n*a[i][j];
...
```

4. Умножение **A*B** матриц по правилу **строка на столбец** (число столбцов матрицы A должно быть равно числу строк матрицы B) $\mathbf{A_{mk}*B_{kn}=C_{mn}}$ причем каждый элемент $\mathbf{c_{ij}}$ матрицы $\mathbf{C_{mn}}$ равен сумме произведений элементов i-ой строки матрицы A на соответствующие элементы j-го столбца

I

$$C_{ij} = a_{i1}b_{1j} + a_{i2}b_{2j} + \ldots + a_{in}b_{nj} = \sum_{l=1}^{n} a_{il}b_{lj}$$

Например,

$$C_{11} = a_{11}b_{11} + a_{12}b_{21} + a_{13}b_{31} + \ldots + a_{1n}b_{n1}$$

$$C_{12} = a_{11}b_{12} + a_{12}b_{22} + a_{13}b_{32} + \ldots + a_{1n}b_{n2}$$

$$Пусть\ A_{23} = \begin{pmatrix} 1 & 0 & 2 \\ 3 & 1 & 0 \end{pmatrix}$$

$$B_{33} = \begin{pmatrix} -1 & 0 & 1 \\ 5 & 1 & 4 \\ -2 & 0 & 1 \end{pmatrix}$$

$$A_{23} * B_{33} = C_{23} = \begin{pmatrix} 1 & 0 & 2 \\ 3 & 1 & 0 \end{pmatrix} \begin{pmatrix} -1 & 0 & 1 \\ 5 & 1 & 4 \\ -2 & 0 & 1 \end{pmatrix} =$$

$$= \begin{pmatrix} 1*(-1)+0*5+2*(-2) & 1*0+0*1+2*0 & 1*1+0*4+2*1 \\ 3*(-1)+1*5+0*(-2) & 3*0+1*1+0*0 & 3*1+1*4+0*1 \end{pmatrix} =$$

$$= \begin{pmatrix} -5 & 0 & 3 \\ 2 & 1 & 7 \end{pmatrix}$$

```
int  sum;
// перемножение матриц
for (i = 0; i < n; ++i)
for (j = 0; j < n; ++j)
{
  sum = 0;
  for (k = 0; k < n; ++k)
  sum += a[i][k] * b[k][j];
  c[i][j] = sum;
}
```

5. Транспонирование матрицы А. Транспонированную матрицу обозначают A^T или A'

$$Если\ A = \begin{pmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{22} & a_{22} & \dots & a_{2n} \\ \dots & \dots & \dots & \dots \\ a_{m1} & a_{m2} & \dots & a_{mn} \end{pmatrix},\ то$$

$$A^T = A' = \begin{pmatrix} a_{11} & a_{21} & \dots & a_{m1} \\ a_{12} & a_{22} & \dots & a_{m2} \\ \dots & \dots & \dots & \dots \\ a_{1n} & a_{2n} & \dots & a_{mn} \end{pmatrix}$$

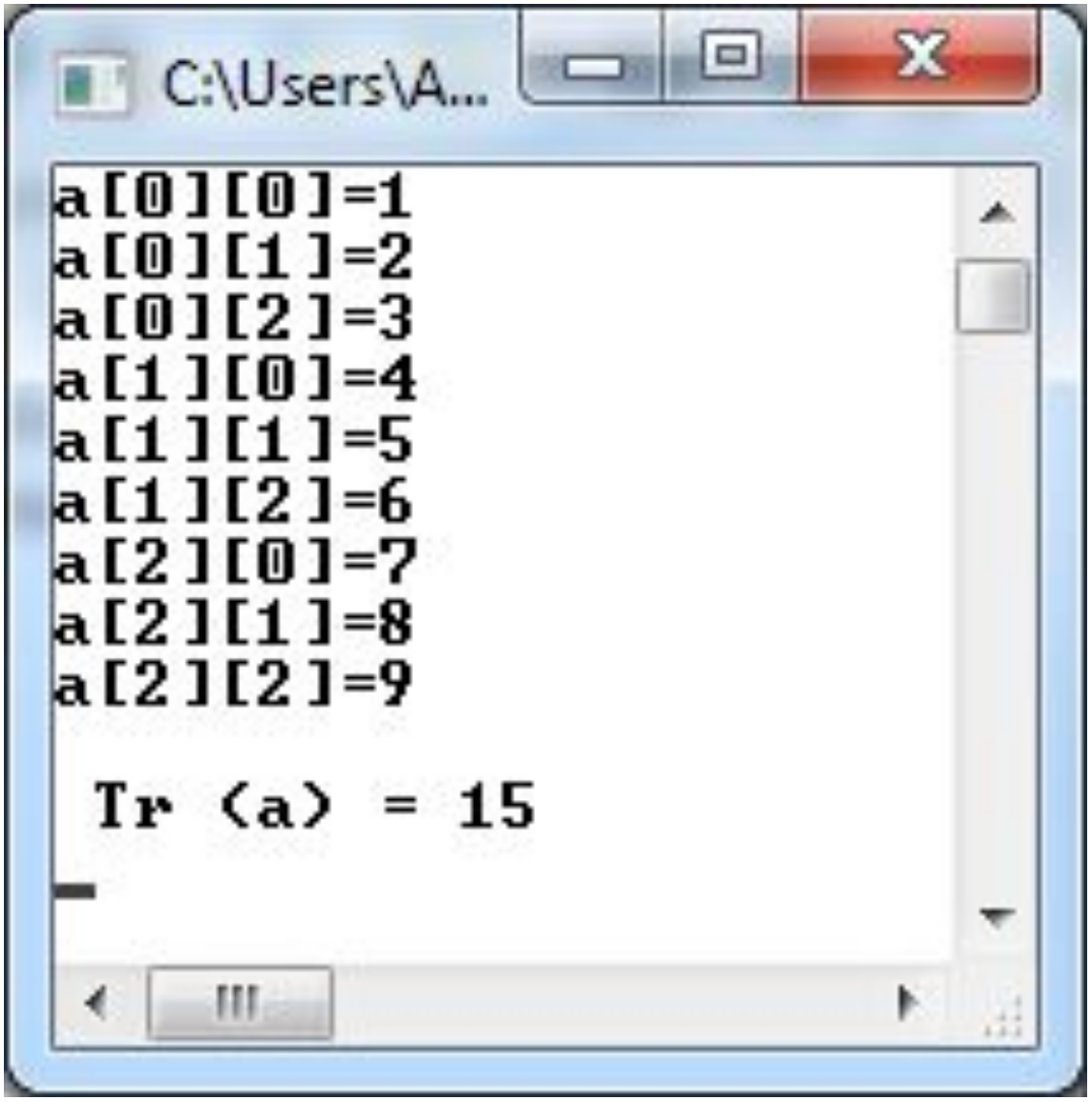
$$A = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{pmatrix},$$

$$A^T = \begin{pmatrix} 1 & 4 \\ 2 & 5 \\ 3 & 6 \end{pmatrix}$$

```
...
c[i][j]= a[j][i];
...
```

```
// След матрицы
const int N=3;
double a[N][N];
int i,j;
// Блок ввода значений
for (i=0;i<N;i++)
for (j=0;j<N;j++)
{
cout<<"a["<<i<<"]["<<j<<"]=";
cin>>a[i][j];
}
```

```
// След матрицы

double tr_a=0.0;

for (i=0;i<N;i++)
tr_a+=a[i][i];

cout<<"\n Tr (a) = "<<tr_a<<"\n";
```


C:\Users\A...
a[0][0]=1
a[0][1]=2
a[0][2]=3
a[1][0]=4
a[1][1]=5
a[1][2]=6
a[2][0]=7
a[2][1]=8
a[2][2]=9
Tr (a) = 15

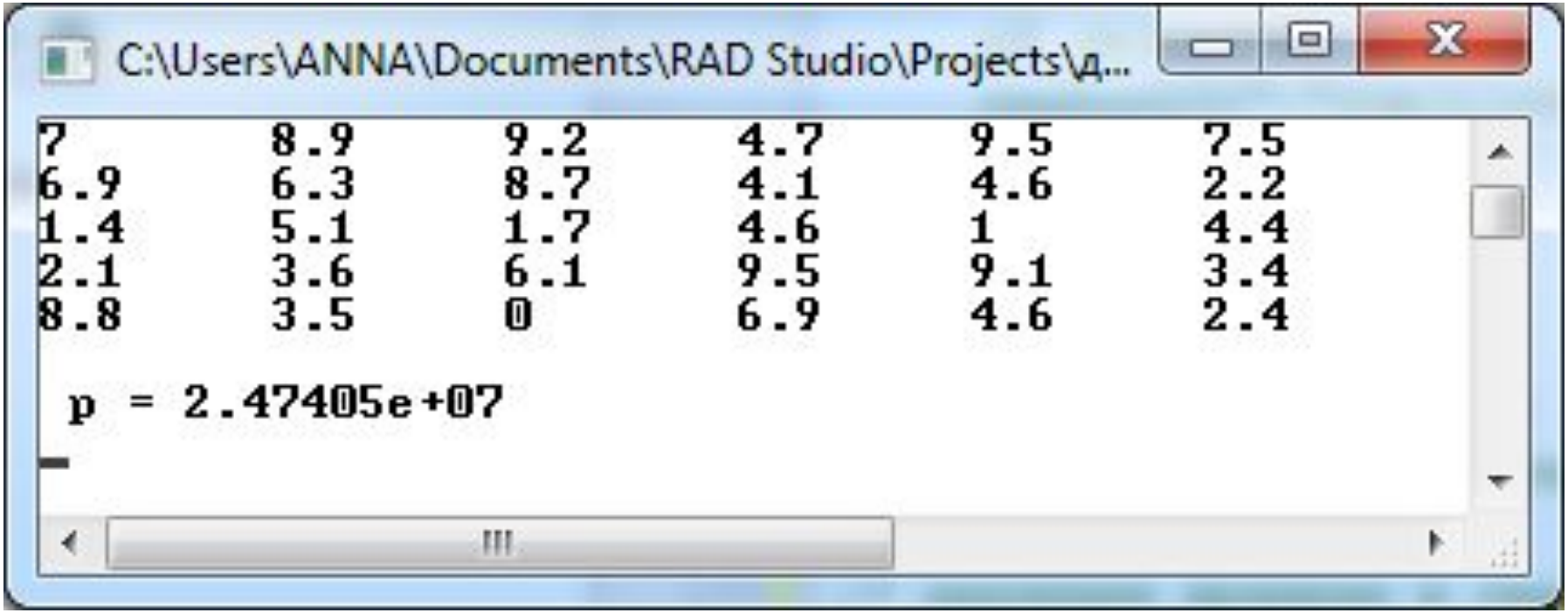
Вычисление произведения сумм элементов строк матрицы:

$$P = \prod_{i=1}^{N} \sum_{j=1}^{M} a_{ij}$$

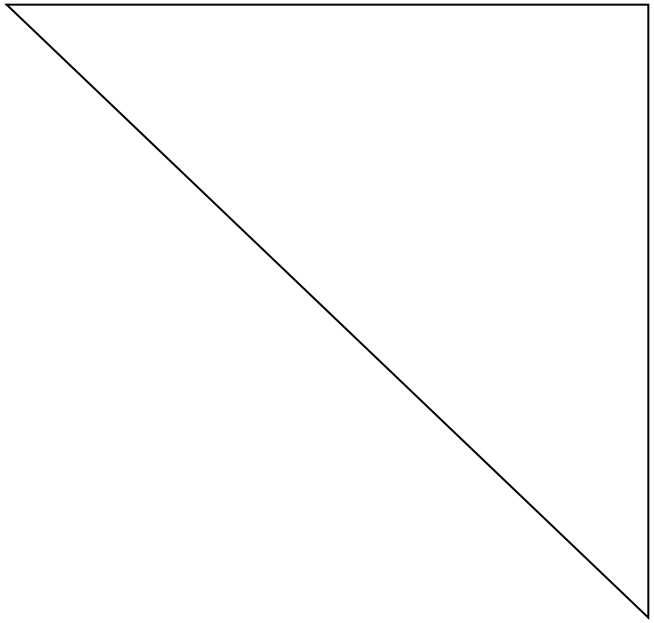
```
double a[5][6];
int i,j;
randomize();
for (i=0;i<5;i++)
for (j=0;j<6;j++)
  a[i][j] = (rand()%100) * 0.1;
```

```
double p,s;
// произведение
p=1.0;
for (i=0;i<5;i++)
{
// s - сумма элементов строки
   s = 0;
for (j=0; j<6;j++)
   s += a[i][j];
   p *= s;
}
```

```
// Вывод результатов
for (i=0;i<5;i++)
{
for (j=0;j<6;j++)
  cout<<a[i][j]<<"\t";
  cout<<"\n";
}
  cout<<"\n p = "<<p<<"\n";
```


C:\Users\ANNA\Documents\RAD Studio\Projects\д...
7 8.9 9.2 4.7 9.5 7.5
6.9 6.3 8.7 4.1 4.6 2.2
1.4 5.1 1.7 4.6 1 4.4
2.1 3.6 6.1 9.5 9.1 3.4
8.8 3.5 0 6.9 4.6 2.4
p = 2.47405e+07

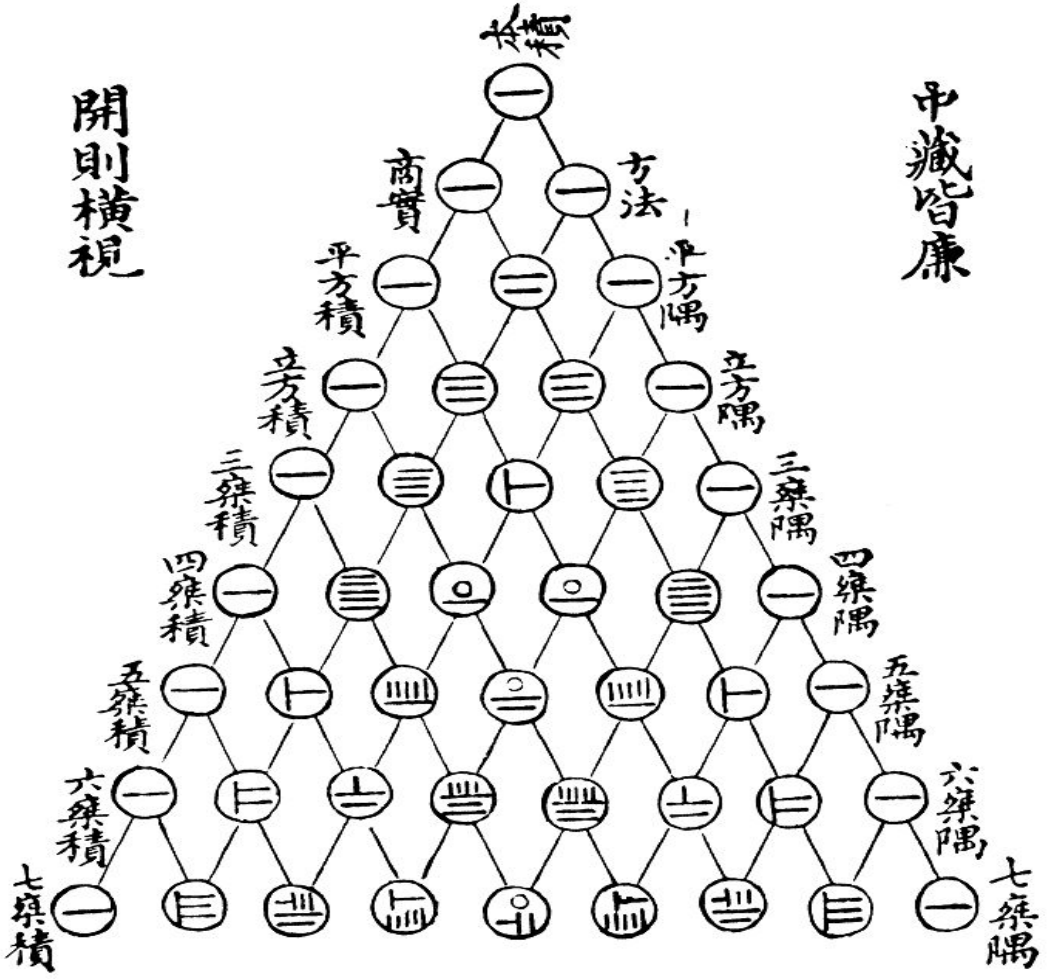
```
/*сумма элементов верхнего
правого треугольника матрицы*/
```



```
/*сумма элементов верхнего правого
  треугольника матрицы*/
/*максимальный размеры матрицы*/

 const  nmax = 10;
 float  a[nmax][nmax];
 int n,i,j;

 cout << "введите размерность<10\n";
 cin >> n;
 cout << "введите матрицу по
              строкам\n";
```

```
for (i=0; i<n; i++)
 for (j=0; j<n; j++)
      cin >> a[i][j];

float sum=0;
for (i=0; i<n; i++)
 for (j=i; j<n; j++)
       sum = sum + a[i][j];

cout << "sum= " << sum;
```

古法七乘方圖



本積	方法	上廉	一廉	二廉	三廉	四廉	五廉	六廉	七廉

Треугольник Паскаля

$$(a+b)^n = \sum_{k=0}^{n} \binom{n}{k} a^k b^{n-k}$$

0:						1								
1:					1		1							
2:				1		2		1						
3:			1		3		3		1					
4:		1		4		6		4		1				
5:	1		5		10		10		5		1			
6:		1		6		15		20		15		6		1
7:	1		7		21		35		35		21		7		1
8:		1		8		28		56		70		56		28		8		1
9:	1		9		36		84		126		126		84		36		9		1
10:		1		10		45		120		210		252		210		120		45		10		1
11:	1		11		55		165		330		462		462		330		165		55		11		1
12:		1		12		66		220		495		792		924		792		495		220		66		12		1
13:	1		13		78		286		715		1287		1716		1716		1287		715		286		78		13		1
14:		1		14		91		364		1001		2002		3003		3432		3003		2002		1001		364		91		14		1

$(a+b)^0$	$=$	1	0								1											
$(a+b)^1$	$=$	$a+b$	1							1		1										
$(a+b)^2$	$=$	$a^2+2ab+b^2$	2						1		2		1									
$(a+b)^3$	$=$	$a^3+3a^2b+3ab^2+b^3$	3					1		3		3		1								
$(a+b)^4$	$=$	$a^4+4a^3b+6a^2b^2+4ab^3+b^4$	4				1		4		6		4		1							
$(a+b)^5$	$=$	$\dots$	5			1		5		10		10		5		1						
$(a+b)^6$	$=$	$\dots$	6	1		6		15		20		15		6		1						

Построение треугольника Паскаля (вид 1)

1

1 1

1 2 1

1 3 3 1

1 4 6 4 1

1 5 10 10 5 1

1 6 15 20 15 6 1

1 7 21 35 35 21 7 1

1 8 28 56 70 56 28 8 1

Построение треугольника Паскаля (вид 2)

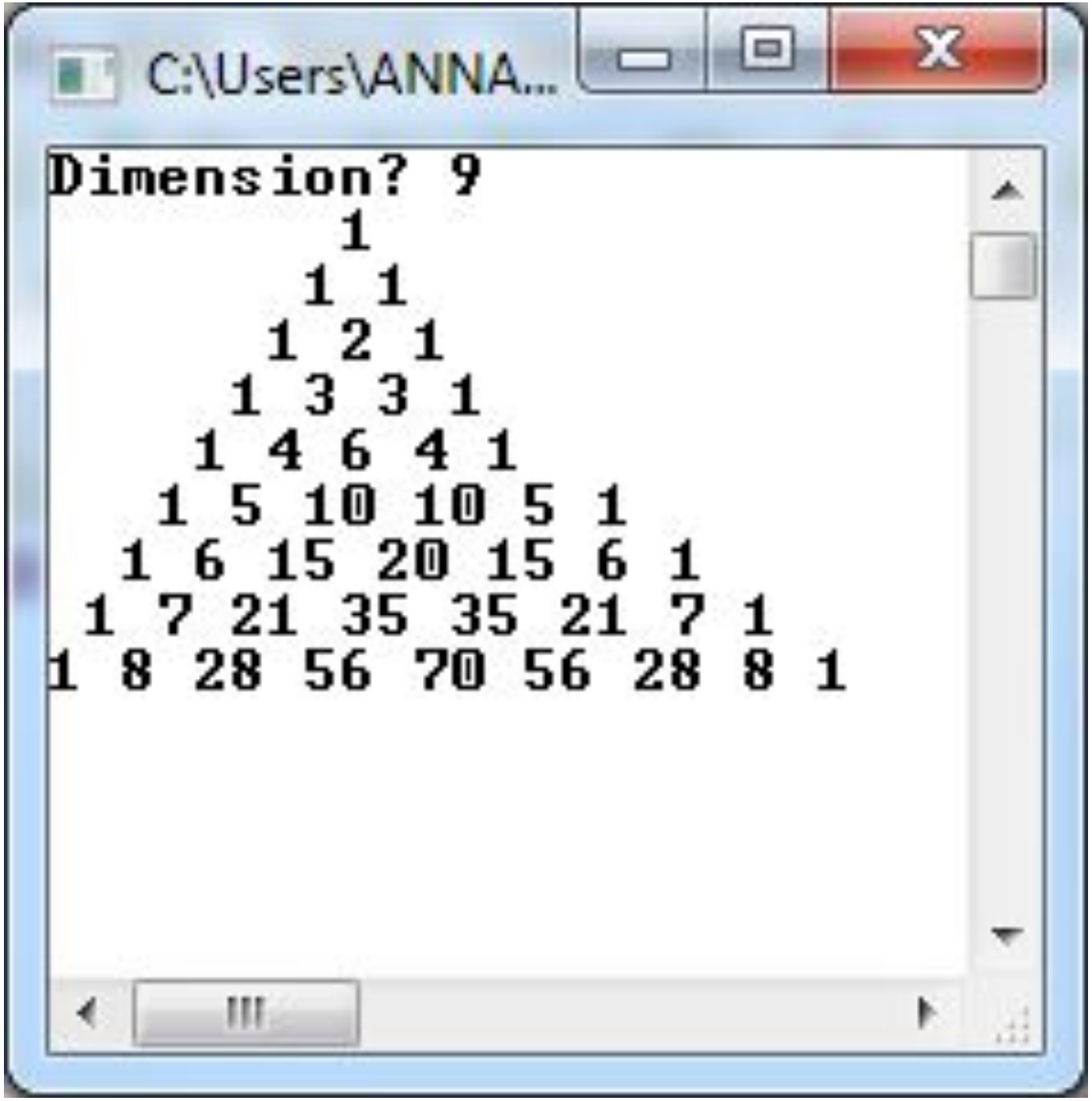
```
                1
              1   1
            1   2   1
          1   3   3   1
        1   4   6   4   1
      1   5  10  10   5   1
    1   6   15  20  15   6   1
  1   7  21  35  35  21   7   1
1   8   28  56  70  56  28  8   1
```

```
//треугольник Паскаля

#include <iomanip.h>

// Установить ширину поля, где n —
//количество позиций, символов

cout<<setw(n);
```

```
//треугольник Паскаля


 const int nmax=10;
 int n,i,j;
 int Ma[nmax][nmax];
 cout<<"Dimension? ";
 cin>>n;
```

```
//расчёт
Ma[0][0]=1;
for (i=1; i<n; i++)
   {
    Ma[i][0]=1;
    Ma[i][i]=1;

for (j=1; j<i; j++)
    Ma[i][j]=Ma[i-1][j-1]+Ma[i-1][j];
       }
```

```
//вывод на экран
 cout<<"\t"<<setw(n);
 for (i=0; i<n; i++)
  {
  cout<<setw(n-j);
 for (j=0; j<i+1; j++)
  cout<<Ma[i][j]<<" ";
  cout<<"\n";
  }
```


C:\Users\ANNA...
Dimension? 9
1
1 1
1 2 1
1 3 3 1
1 4 6 4 1
1 5 10 10 5 1
1 6 15 20 15 6 1
1 7 21 35 35 21 7 1
1 8 28 56 70 56 28 8 1

Представить целочисленную квадратную матрицу 4x4 в виде массива. Присвоить элементам на главной диагонали значение 1, выше главной диагонали - 2, ниже – 0.

1 a[0][0]	**2** a[0][1]	**2** a[0][2]	**2** a[0][3]
0 a[1][0]	**1** a[1][1]	**2** a[1][2]	**2** a[1][3]
0 a[2][0]	**0** a[2][1]	**1** a[2][2]	**2** a[2][3]
0 a[3][0]	**0** a[3][1]	**0** a[3][2]	**1** a[3][3]

```
const int n=4;
int a[n][n] = {0};
for (int i=0; i<=n-1; i++)
for (int j=0; j<=n-1; j++)
    {
      if (i == j) a[i][j]=1;
      else if (i < j) a[i][j]=2;
    }
for (int i=0; i<=n-1; i++)
    {
      cout << endl;
for (int j=0; j<=n-1; j++)
cout << a[i][j];
    }
```

1 2 2 2

0 1 2 2

0 0 1 2

0 0 0 1