

Домашнее задание № 5

Study-Inf/1 курс/ПИ/ Информатика и программирование

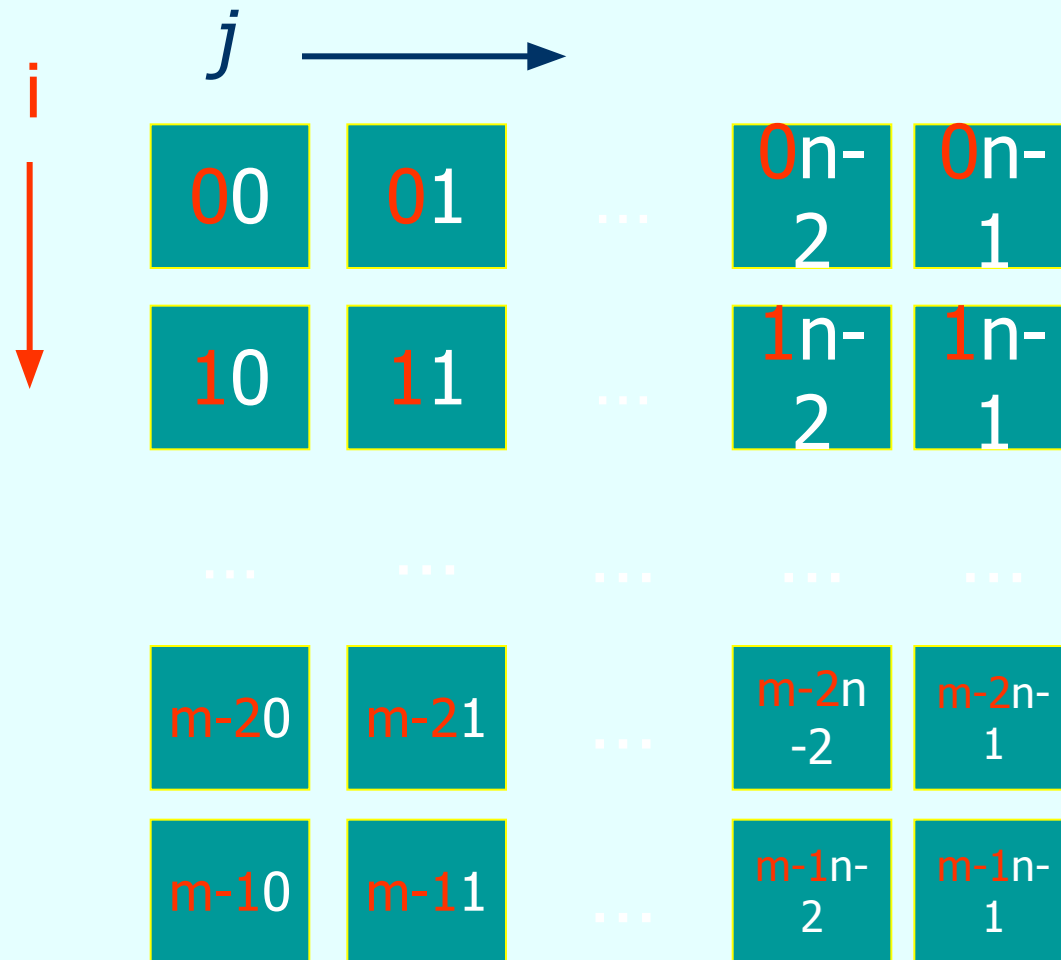
Study-Inf/1 курс/БИ/ Информатика

Домашние задания и самостоятельная работа

Срок сдачи задания - 9 ноября

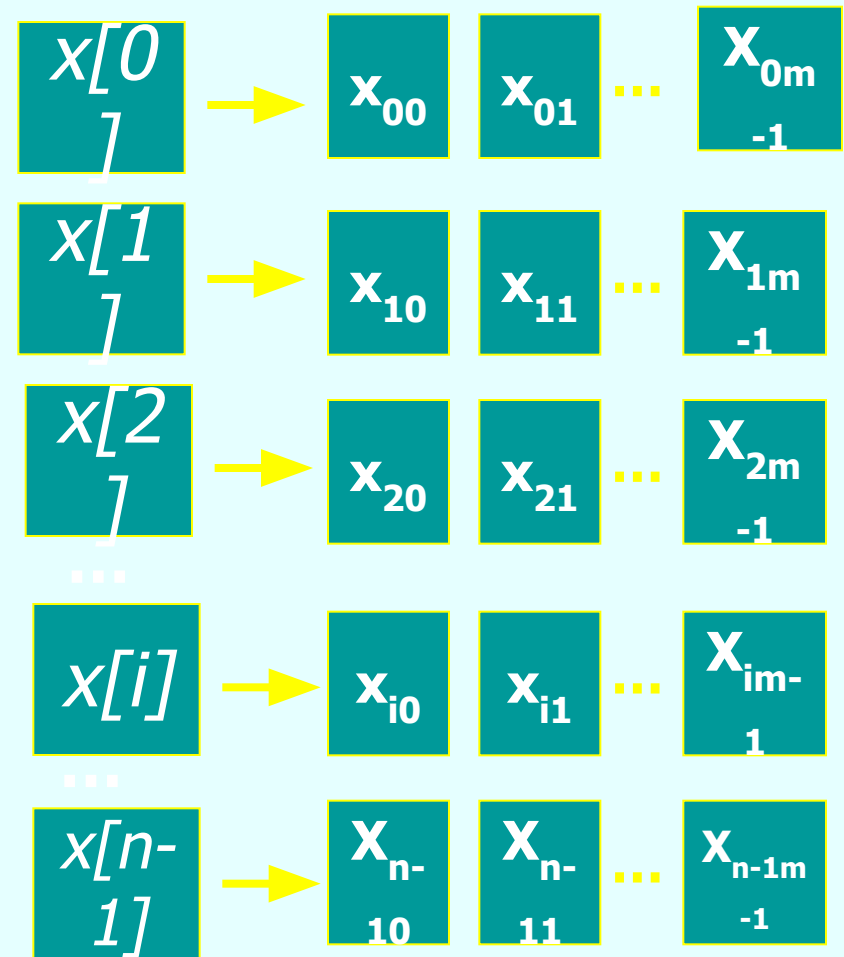
Многомерные массивы

Инициализация матриц



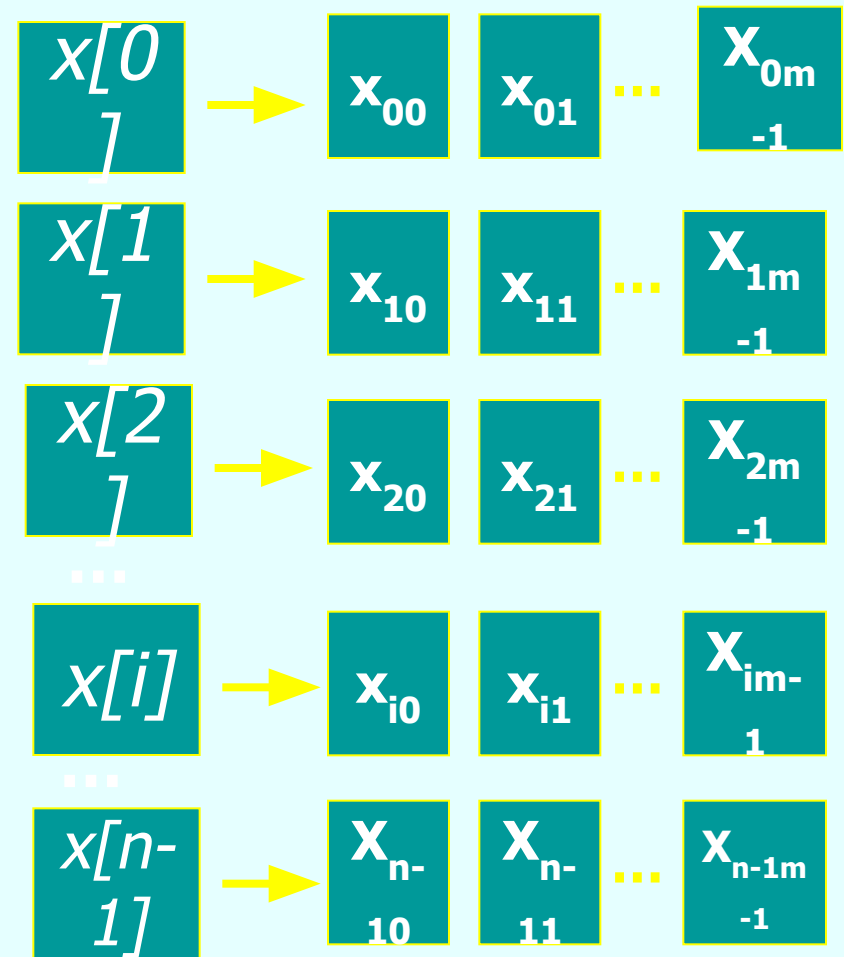
Механизм выделения памяти

```
int **x, i;  
x=(int**)malloc(sizeof( int*)*n);  
for(i=0;i<n;i++)  
x[i] = (int*)malloc(sizeof(int)*n);
```



Механизм освобождения памяти

```
for(int i=n-1;i>=0;i--)  
    free (x[i]);  
free(x);
```



Обращение к элементу матрицы $x[i,j]$

```
for(i=0;i<n;i++)  
  for(j=0;j<m;j++)  
    ...
```

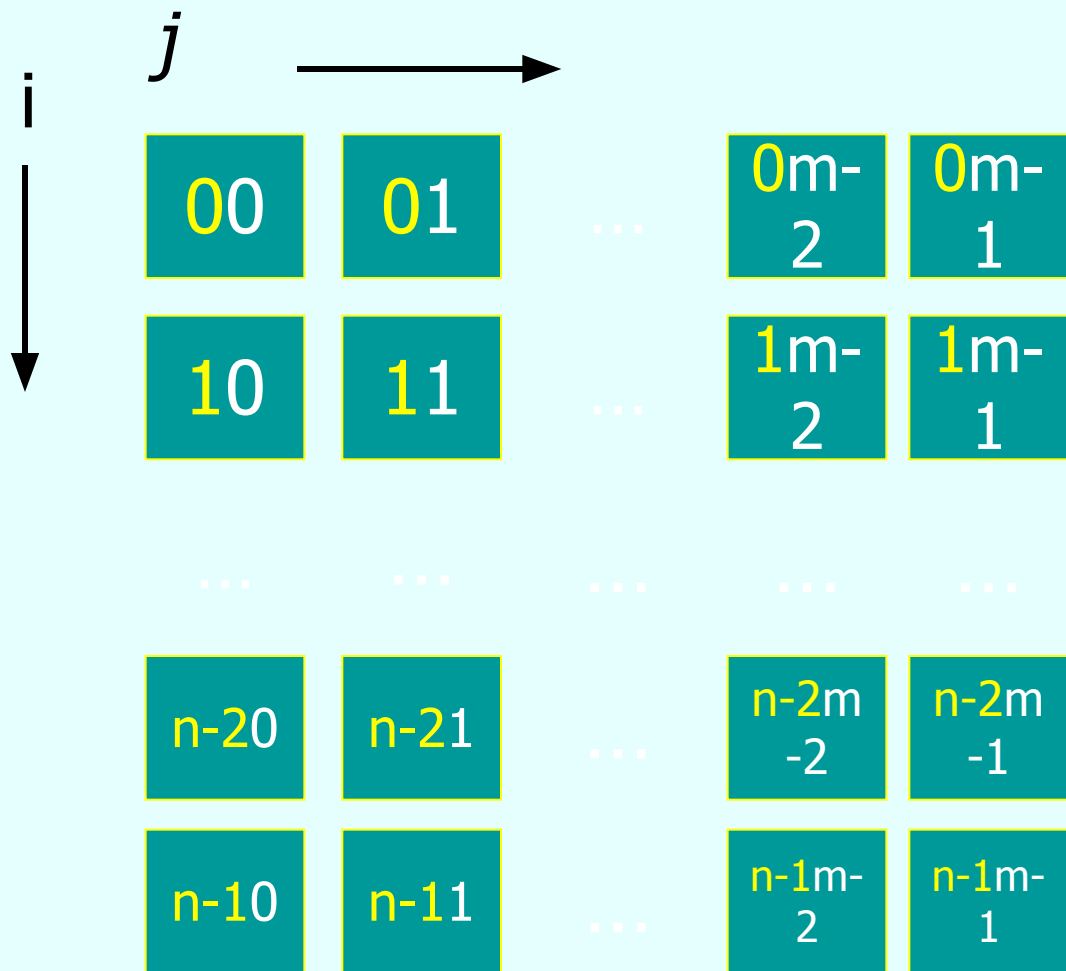
фиксируется строка

фиксируется столбец

индекс строки

индекс столбца

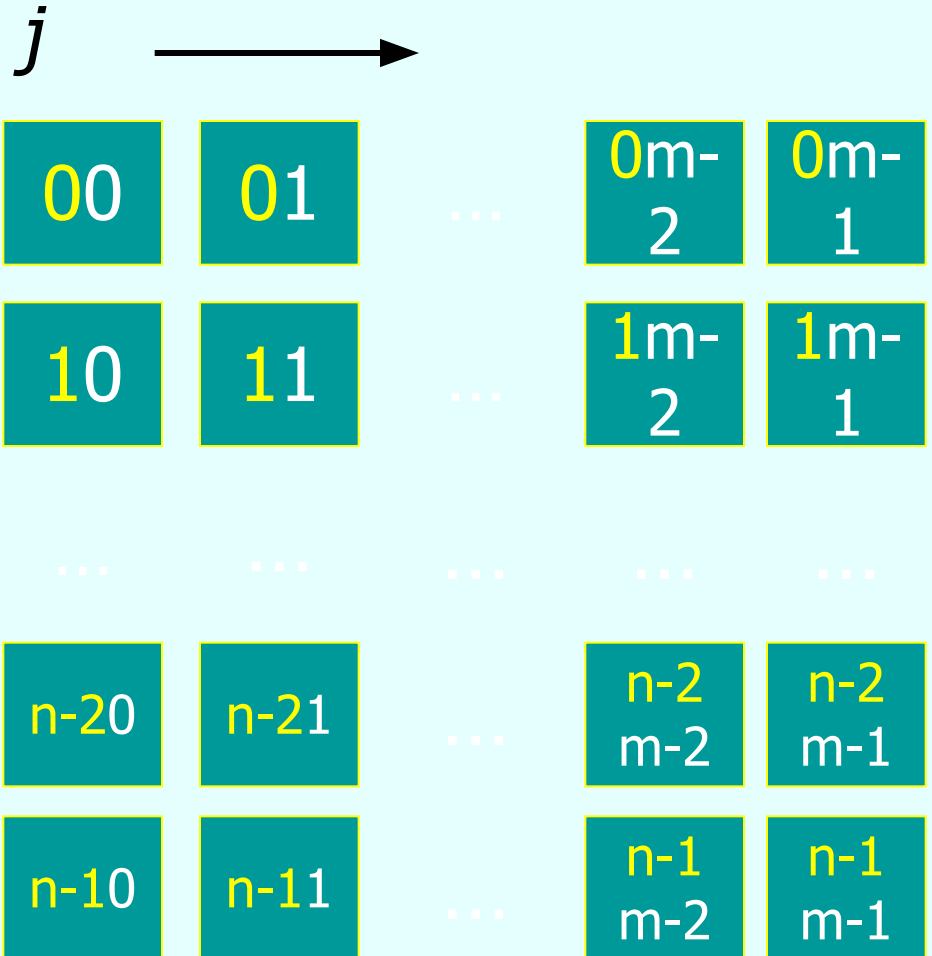
$x[i][j]$



```
for(i=0;i<m;i++)  
  for(j=0;j<n;j++)  
    ... x[j][i]
```

фиксируется строка

фиксируется столбец



Инициализация элементов матрицы. Ввод данных с клавиатуры.

...

```
float **A;
```

```
int n,m,i,j;
```

```
printf ("Введите количество строк матрицы: ");
```

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

```
printf ("Введите количество столбцов матрицы: ");
```

```
scanf("%d",&m);
```

```
A = new float*[n];
```



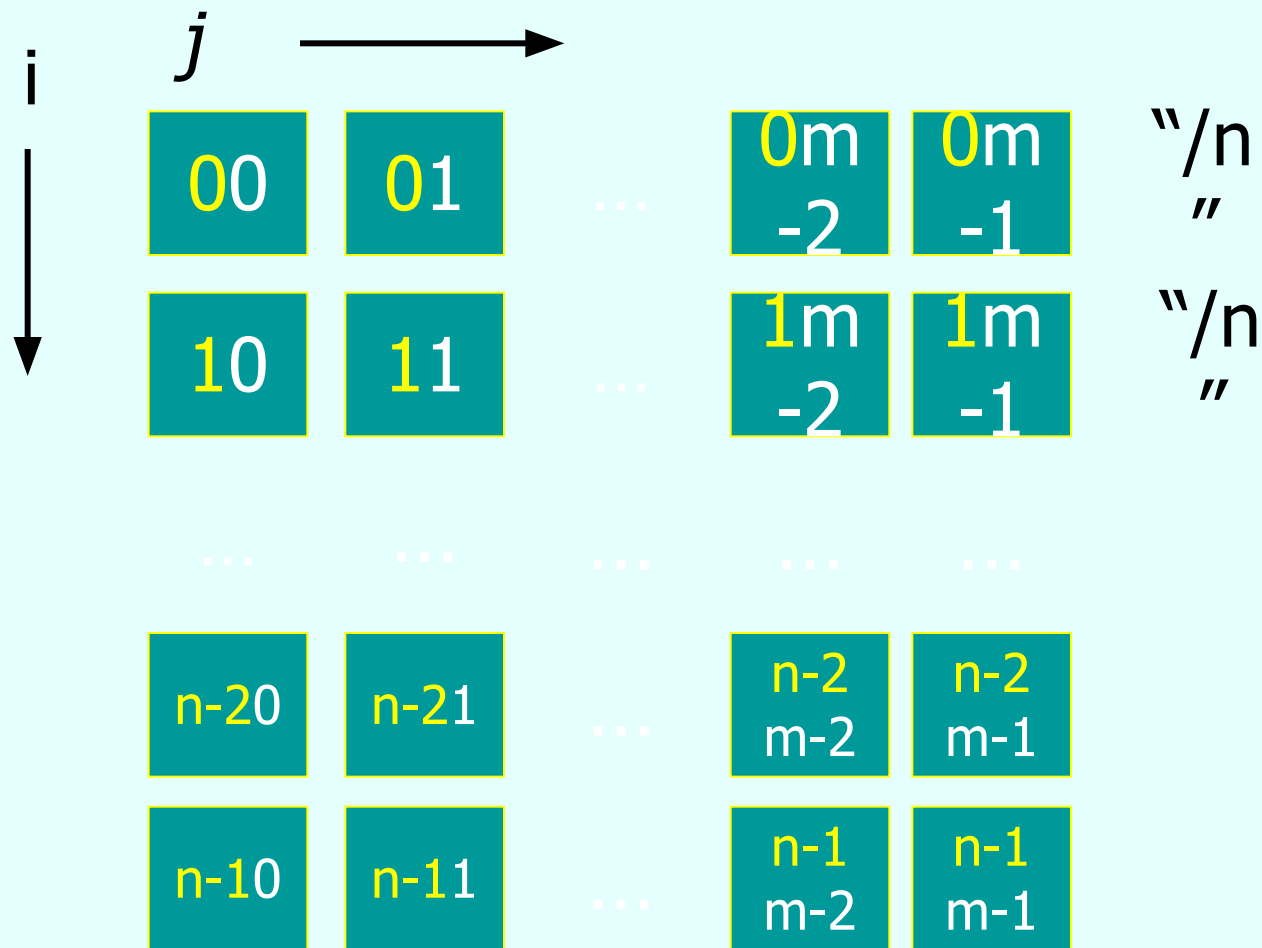
```
for(i=0;i<n;i++)  
  A[i] = new float[m];  
for(i=0;i<n;i++)  
  for(j=0;j<m;j++)  
    { printf("A[%d][%d]= ",i,j);  
      scanf("%f",&(A[i][j]));  
    }
```

...

Получение значений случайным образом

```
...  
float **x;  
int n,m,i,j;  
...  
x = (float**) malloc(sizeof(float*)*n);  
for( i=0;i<n;i++)  
    x[i] = (float*)malloc(sizeof(float)*m);  
for(i=0;i<n;i++)  
    for(j=0;j<m;j++)  
        x[i][j] = rand()%200/(rand()%100+1.);  
...
```

Печать элементов матрицы



...

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

```
{
```

```
for( j=0;j<m;j++)
```

```
    printf("%8.3f ",x[i][j]);
```

```
// переход на новую строку экрана
```

```
printf("\n");
```

```
}
```

...

Выделение областей матриц

Выделение строки с номером k :



...

```
for (int i=0; i< $m$ ; i++)
```

Обращение к элементу $x[k][i]$;

...

Выделение столбца с номером f

0f

1f

...

n-2f

n-1f

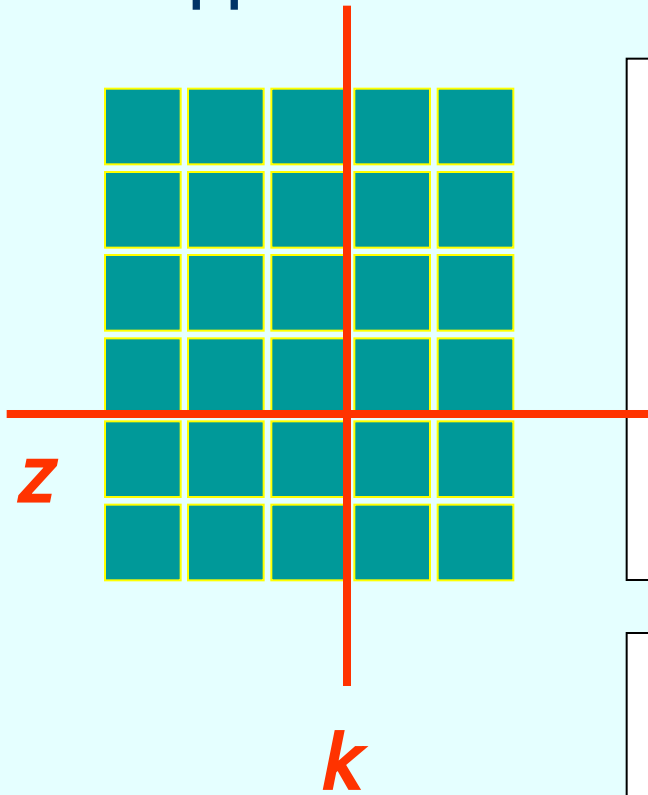
...

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

Обращение к элементу $x[i][f]$;

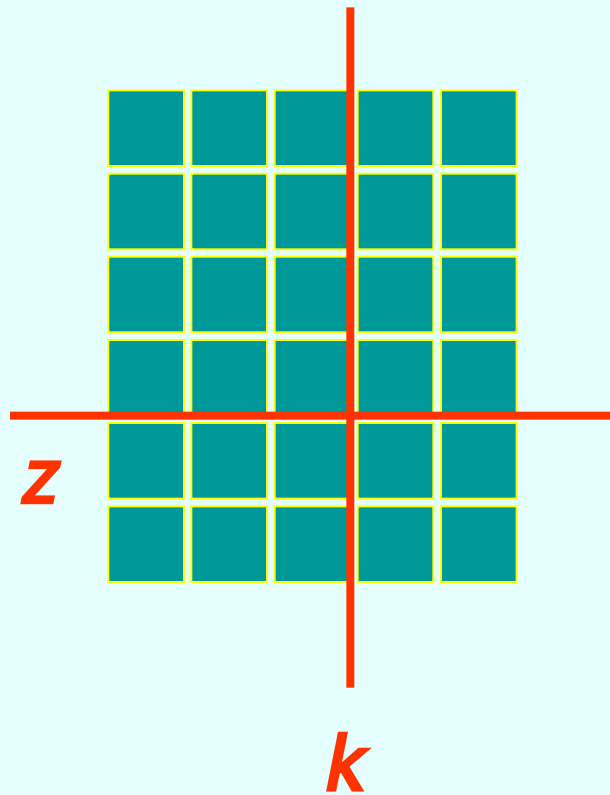
...

Выделение блоков матриц



```
...  
for (int i=0;i<n;i++)  
    for(int j=0;j<=k;j++)  
        Обращение к элементу  $x[i][j]$ ;  
...
```

```
...  
for (int i=0;i<=z;i++)  
    for(int j=0;j<m;j++)  
        Обращение к элементу  $x[i][j]$ ;  
...
```



```
...  
for (int i=z+1;i<n;i++)  
  for(int j=k+1;j<m;j++)  
    Обращение к элементу  $x[i][j]$ ;  
...
```


Квадратные матрицы

0,0				0,n-1
	1,1		1,n-2	
		2,2		
	i,n-1-i		i,i	

$$m==n$$

...

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

Обращение к элементу $x[i][i]$;

...

...

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

Обращение к элементу

$x[i][n-i-1]$;

...

0,0				0,n-1
	1,1		1,n-2	
		2,2		
	i,n-1-i		i,i	

...

```
for (int i=0;i<n-1;i++)
  for (int j=i+1;j<n;j++)
```

Обращение к элементу
 $x[i][j]$;

...

...

```
for (int i=1;i<n;i++)
  for (int j=0;j<i;j++)
```

Обращение к элементу
 $x[i][j]$;

...

0,0				0,n-1
	1,1		1,n-2	
		2,2		
	i,n-1-i		i,i	

...

```
for (int i=0;i<n-1;i++)
  for (int j=0;j<n-i-1;j++)
```

Обращение к элементу $x[i][j]$;

...

...

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

Обращение к элементу $x[i][j]$;

...

В вещественной матрице размерности ***$n \times m$*** элементов найти минимальный элемент и его местоположение в матрице.

```
#include <conio.h>
#include <stdio.h>
#include <stdlib.h>
#include <values.h>
```

```
...
```

```
{
```

```
printf("Введите количество строк: ");
```

```
int n,m,i,j;  
srand(time(NULL));  
scanf("%d",&n);  
printf("Введите количество столбцов: ");  
scanf("%d",&m);  
float **x = (float**) malloc(sizeof(float*)*n);  
for(i=0;i<n;i++)  
    x[i]=(float*)malloc(sizeof(float)*m);  
for(i=0;i<n;i++)  
    {  
        for(j=0;j<m;j++)  
            { x[i][j]=rand()%100/(rand()%100)+1.);
```

```
printf("%8.2f",x[i][j]);}  
printf("\n");  
}  
int imin,jmin;  
float min = MAXFLOAT;  
for (i=0;i<n;i++)  
    for( j=0;j<m;j++)  
        if (min>x[i][j]) {  
            min = x[i][j];  
            imin=i;  
            jmin=j;}  
printf("Минимальный элемент x[%d,%d] =  
%8.2f \n",imin,jmin,min);
```

```
for(i=n-1;i>=0;i--)
```

```
    free( x[i]);
```

```
    free( x);
```

```
    ...
```

```
}
```

Отсортировать строки целочисленной матрицы $A[n \times m]$ по возрастанию минимальных элементов строк.

```
#include <conio.h>  
#include <stdio.h>  
#include <stdlib.h>  
#include <values.h>  
  
...  
{ ...  
    printf("Введите количество строк: ");
```



```
int n,m,i ,j;  
srand(time(NULL));  
scanf("%d",&n);  
printf("Введите количество столбцов: ");  
scanf("%d",&m);  
int **a = (int**)malloc(sizeof(int*)*n);  
for(i=0;i<n;i++)  
    a[i]=(int*)malloc(sizeof(int)*m) ;  
int* min = new int [n];  
for (i=0;i<n;i++)  
    min[i] = MAXINT;  
for(i=0;i<n;i++)  
{
```

```
for(j=0;j<m;j++)
{a[i][j]=rand()%101;
 if (min[i]>a[i][j]) min[i] = a[i][j];
 printf("%4d",a[i][j]); }
printf(" min = %4d",min[i]);
printf("\n");
}
int k;
for (i=0;i<n-1;i++)
for (j=0;j<n-1-i;j++)
{ int temp;
 if(min[j]>min[j+1]) {
 for( k=0;k<m;k++)
 { temp = a[j][k];
```

```
    a[j][k] = a[j+1][k];  
    a[j+1][k] = temp;  
}
```

```
    temp = min[j];  
    min[j] = min[j+1];  
    min[j+1] = temp; }  
}
```

```
printf("\n");
```

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

```
{
```

```
    for(int j=0;j<m;j++)
```

```
{
```

```
    printf("%4d",a[i][j]);}  
    printf("  min = %4d",min[i]);  
    printf("\n");  
}  
for(i=n-1;i>=0;i--)  
    free( a[i]);  
free( a);  
system("pause");  
}
```

Структуры



Книга

Название

Автор

Цена

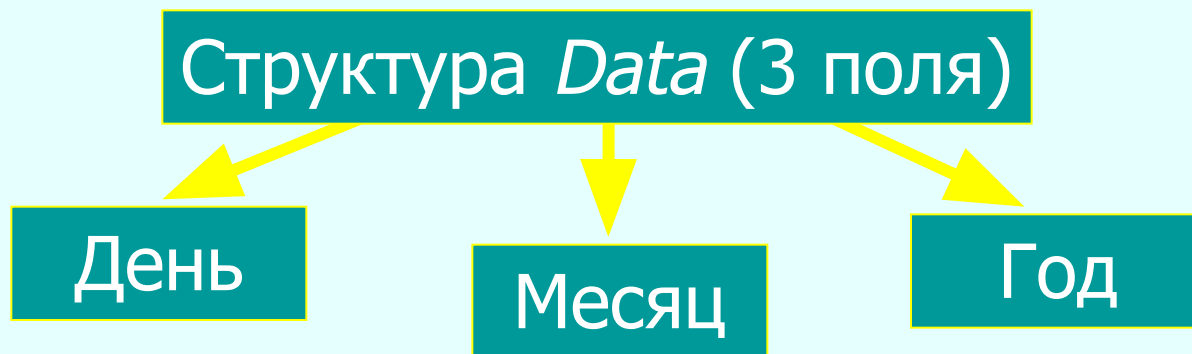
Издательство

Тираж

Синтаксис:

```
struct [имя]  
{ тип поле1;  
  тип поле2;  
...  
}
```

```
struct Data{  
  int day;  
  int mounth;  
  int year;  
}  
...  
struct Data k;
```



Обращение к полям структуры

- <имя переменной>.<имя поля>

```
k.day
```

```
k.mounth
```

```
k.year
```

- <имя указателя на структуру>-><имя поля>

```
struct Data* f;
```

```
f->day; f->mounth;
```

```
f->year;
```

Вложенность структур

```
struct Student{  
    struct Name{  
        char surname[30];  
        char name[20];  
        char patronymic[30];  
    };  
    int ball;  
    char sex;  
};  
...  
struct Student ss;  
ss.Name.name = ;  
...
```


Оператор определения собственного (пользовательского) типа

Синтаксис:

typedef <стандартный тип> <задаваемое имя>;

```
typedef struct Student student;  
student s1,s2;  
typedef int MYTYPE;  
typedef double precision;
```

Дан массив записей, содержащих информацию о сдаче студентами одной группы экзаменов по математике, физике и программированию. Расположить записи в массиве по убыванию оценки по математике. Вывести отсортированный массив на экран.

```
#include <conio.h>  
#include <stdio.h>  
#include <stdlib.h>  
#include <values.h>  
  
...  
{
```

```
typedef struct {  
    char Name[80];  
    int m;  
    int f;  
    int p;  
} Student;  
int n = 5;  
Student *student;  
student = new Student[n];  
for (int i=0;i<n;i++)  
{printf("Имя: ");  
    scanf("%s",student[i].Name);
```

```
printf("Математика: ");
scanf("%d",&student[i].m);
printf(" Физика: ");
scanf("%d",&student[i].f);
printf(" Программирование: ");
scanf("%d",&student[i].p);
}
printf(" Исходные данные: ");
for(i=0;i<n;i++) {
printf("%30s%3d%3d%3d\n",student[i].Name,
student[i].m,student[i].f,student[i].p);
}
for (i=1;i<n;i++)
```

```
{ Student S_vs = student[i];  
  int vs = student[i].m;  
  int j = i-1;  
  while(vs>student[j].m&& j>=0)  
    {student[j+1]=student[j];  
      j--;}  
  student[j+1] = S_vs;  
}  
printf("\n После сортировки:\n");  
for(i=0;i<n;i++) {  
  printf("%30s%3d%3d%3d\n",student[i].Name,  
    student[i].m,student[i].f,student[i].p);  
}  
getch();  
}
```

Объединения

Хранение разнотипных данных в одной области памяти.

Синтаксис:

```
union [имя] {  
    тип поле1;  
    тип поле2;  
    ...  
}
```

Размер объединения - это размер его максимального элемента.

В каждый момент времени может быть сохранен только один из элементов объединения.

```
union MyUnion{  
char k[2];  
int m;}
```

