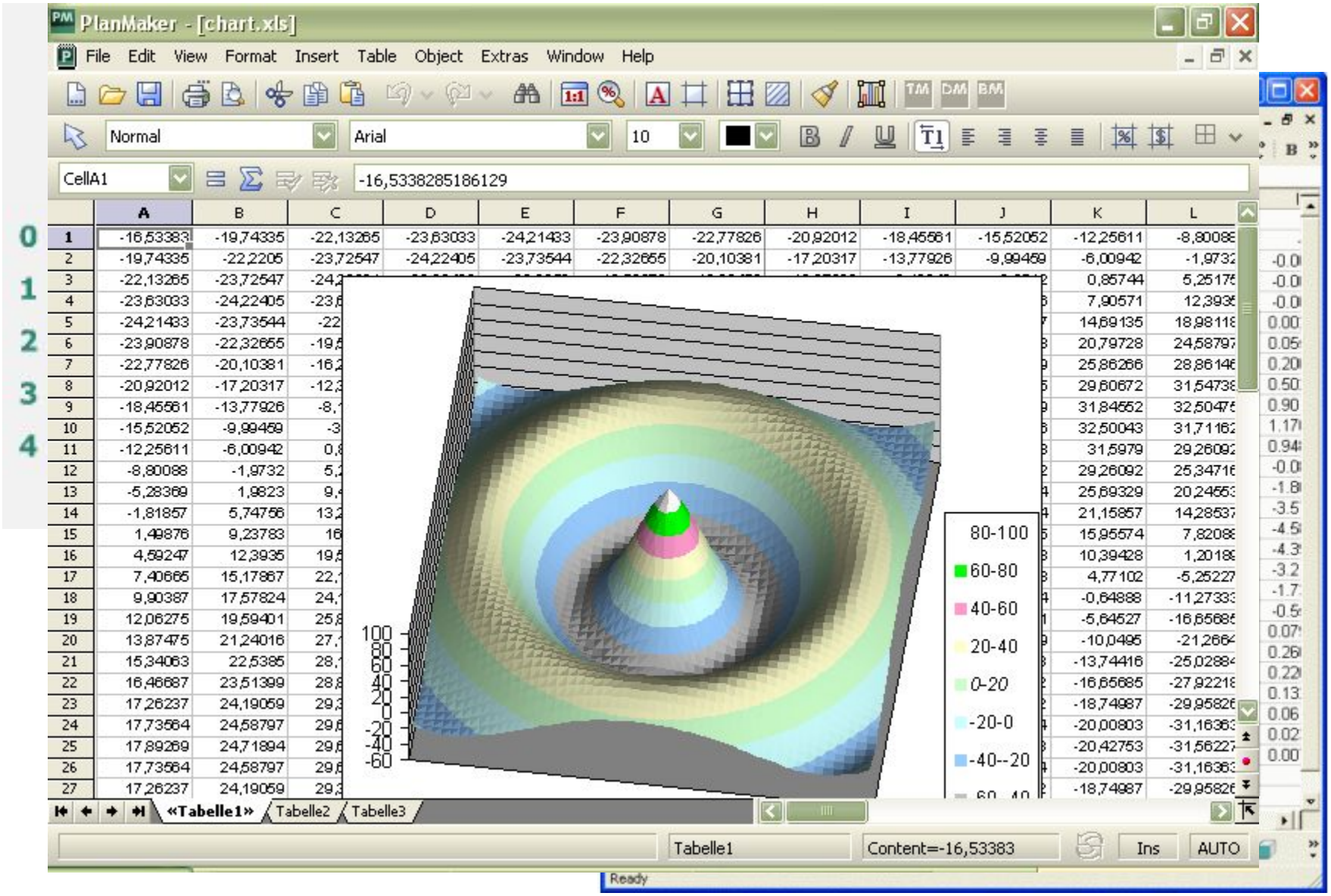
Двумерные массивы

В математике **таблицы чисел, состоящие из строк и столбцов называются *матрицами* и записываются в круглых скобках.**

$$A = \begin{pmatrix} 3 & 21 & 17 & 36 \\ 45 & 67 & 89 & 22 \\ 91 & 34 & 78 & 57 \\ 11 & 18 & 65 & 20 \\ 56 & 81 & 54 & 16 \end{pmatrix}$$

Использование двумерных массивов для построения поверхностей.



Массив **— это пронумерованная последовательность величин одинакового типа, обозначаемая одним именем. Элементы массива располагаются в последовательных ячейках памяти, обозначаются именем массива и индексом. Каждое из значений, составляющих массив, называется его *компонентой* (или *элементом* массива).**

Способ организации данных, при котором каждый элемент определяется номером строки и номером столбца, на пересечении которых он расположен, называется *двумерным массивом*

	1	2	3	4	5	6
1	7	4	2	7	5	4
2	4	1	3	8	9	9
3	1	5	0	6	0	0

В математике:

$A_{i;j}$

В Pascal:

```
A[i,j]
```

```
A[2,4]
```

```
A[1,2]
```

```
A[3,5]
```

Самый простой способ описания массива

```
a : array [1..10, 1..20] of real;
```

Имя массива

Количество строк

Тип данных в массиве

Описание как массив массивов:

```
a: array [1..10] of array [1.. 20] of real;
```

Количество столбцов

Одномерный массив

Каждый элемент которого в свою очередь является одномерным массивом

Количество строк и столбцов через константу.

```
Const
  m=10; n =20;
Var
  a : array[1..m,1..n] of integer;
```

В разделе констант указываем число строк и столбцов

Определяем пользовательский тип , двумерный массив

Определ... данных.

```
type t=array[1..m,1..n] of integer;
var   a : t;
```

Массив констант.

```
const
a: array[1..3,1..5] of  integer =
((3,-2,1,4,3),
 (-5,-9,0,3,7),
 (-1,2,1,-4,0));
```

Непосредственно в программе указываем значения элементов массива.

Заполне

Цикл отвечающий за перебор строк. Берем первую, вторую и так далее строки

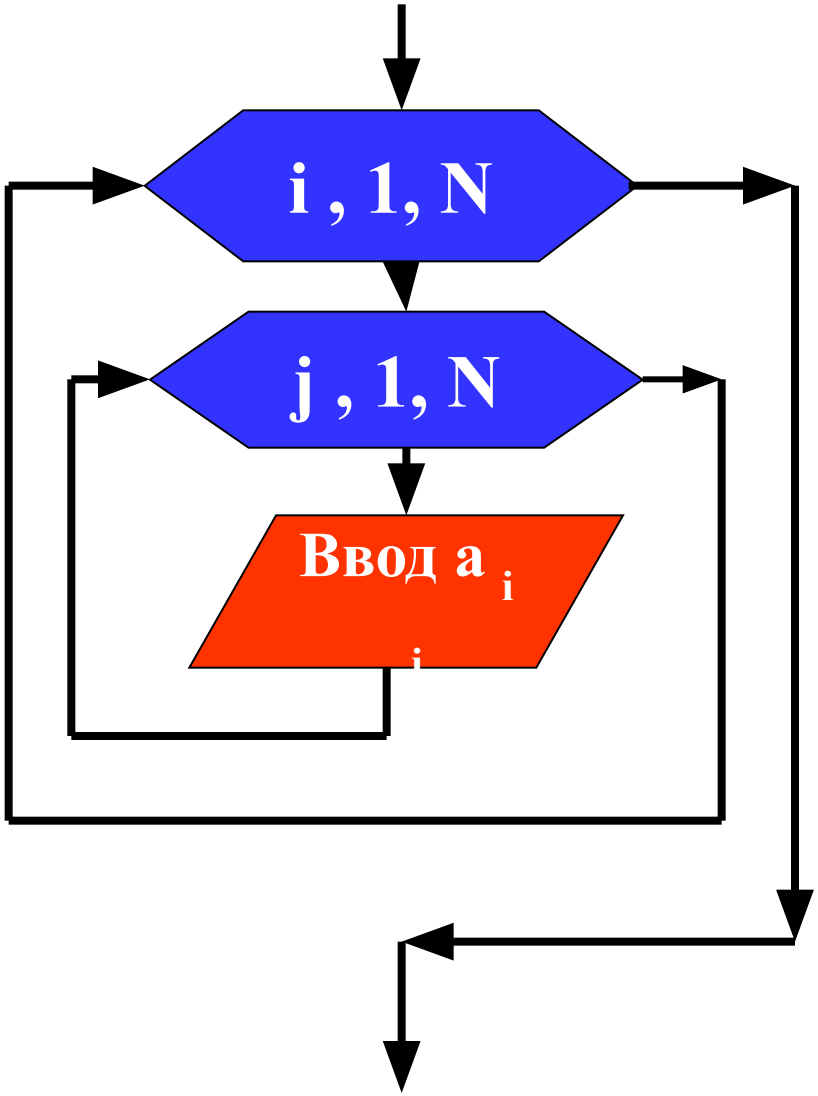
```
For i := 1 to 3 do begin
  For i := 1 to 4 do
```

```
('A[ ', i, ', ',j, ']=  ');
n(a[i, j])
```

```
CRT - программа завершена
A[ 1, 1]=  1
A[ 1, 2]=  2
A[ 1, 3]=  5
A[ 1, 4]=  3
A[ 2, 1]=  2
A[ 2, 2]=  1
A[ 2, 3]=  2
A[ 2, 4]=  4
A[ 3, 1]=  3
A[ 3, 2]=  2
A[ 3, 3]=  3
A[ 3, 4]=  4
```

Цикл отвечающий за перебор ячеек в каждой строке.

Блок-схема заполнения с клавиатуры:



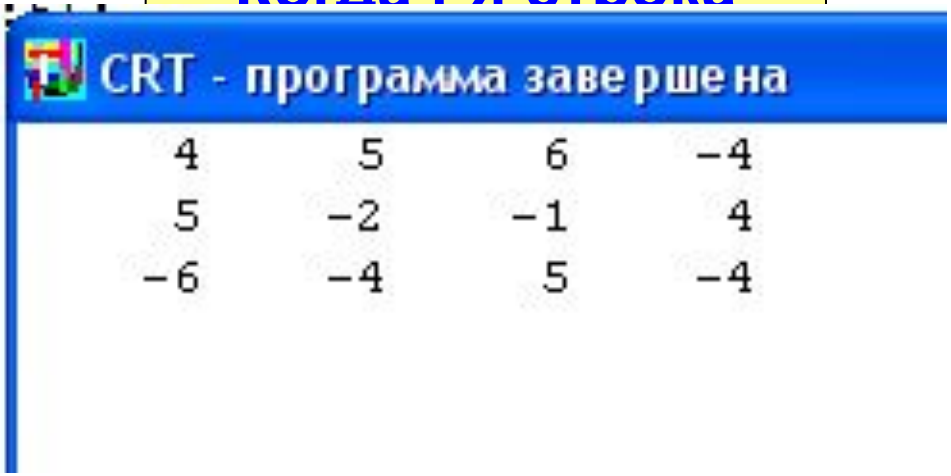
Цикл отвечающий за перебор строк. (Внешний цикл)

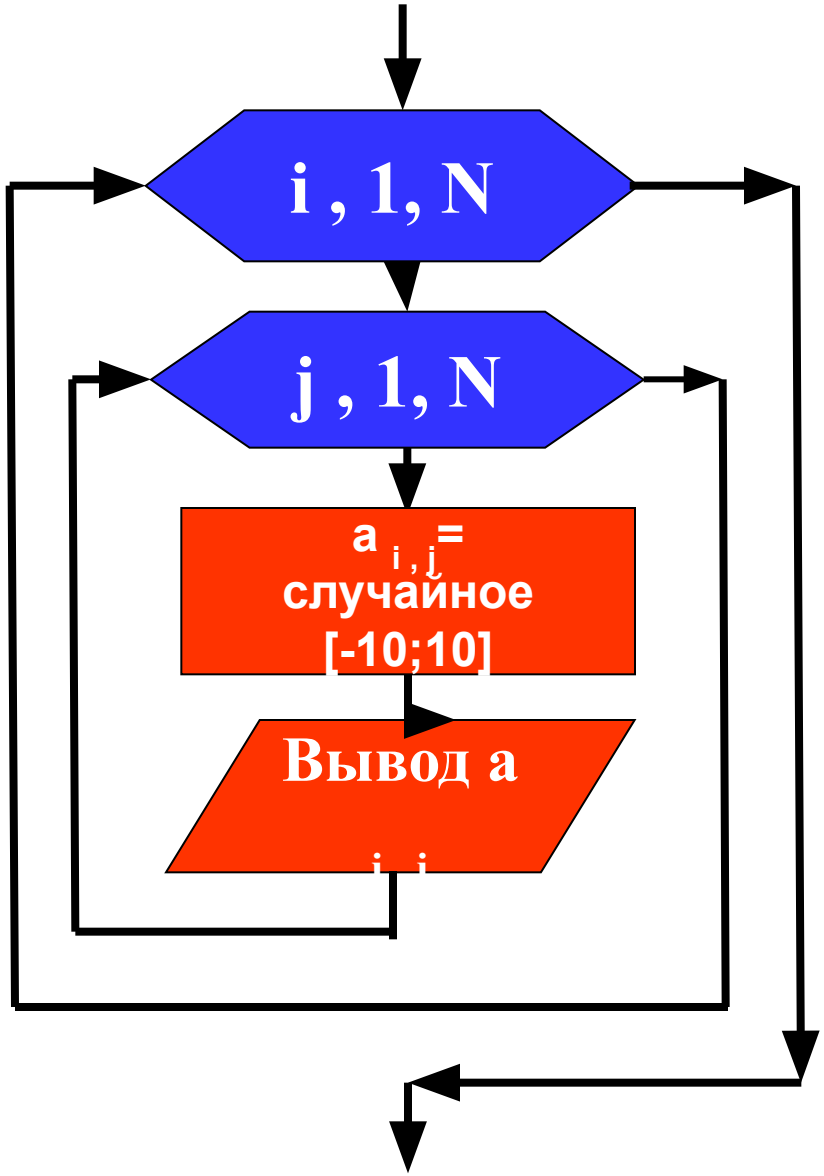
Цикл отвечающий за перебор ячеек в каждой строке. (Внутренний цикл)

Заполнение массива случайными числами:

```
For i := 1 to 3 do begin
        For j := 1 to 4 do begin
                a[i, j] := random(21) - 10;
                write(a[i, j]:6);
        end;
        writeln;
end;
```

Когда i-я строка





Цикл отвечающий за перебор строк. (Внешний цикл)

Цикл отвечающий за перебор ячеек в каждой строке. (Внутренний цикл)

Заполнение массива по правилу:

Заполнить п… размером N x N … правилу:

```
11111
22222
33333
44444
55555
```

```
ClrScr;
Write('ВВеди
For i:=1 to n d
For j:=1 to n
a[i,
write(a[i,j]:4);
end;
writeln;
end;
```

CRT - программа завершена

```
ВВеди N = 9
   1   1   1   1   1   1   1   1   1
   2   2   2   2   2   2   2   2   2
   3   3   3   3   3   3   3   3   3
   4   4   4   4   4   4   4   4   4
   5   5   5   5   5   5   5   5   5
   6   6   6   6   6   6   6   6   6
   7   7   7   7   7   7   7   7   7
   8   8   8   8   8   8   8   8   8
   9   9   9   9   9   9   9   9   9
```

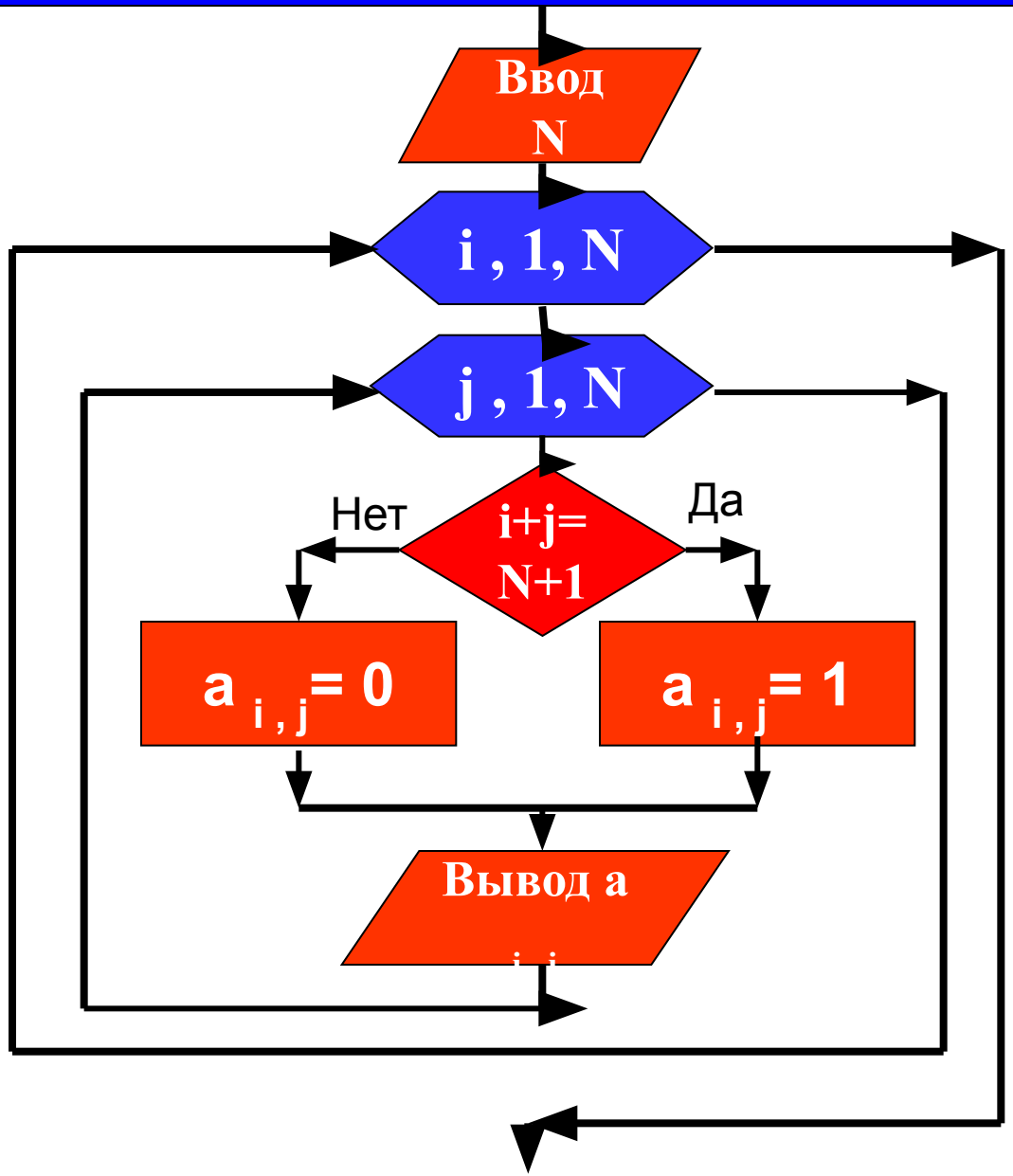
каждой ячейке строки равно номеру строки.


```
);Readln(n);
gin
gin
n+1) then a[i,j]:=1
        else a[i,j]:=0;
,j]);
end;
writeln;
end;
```

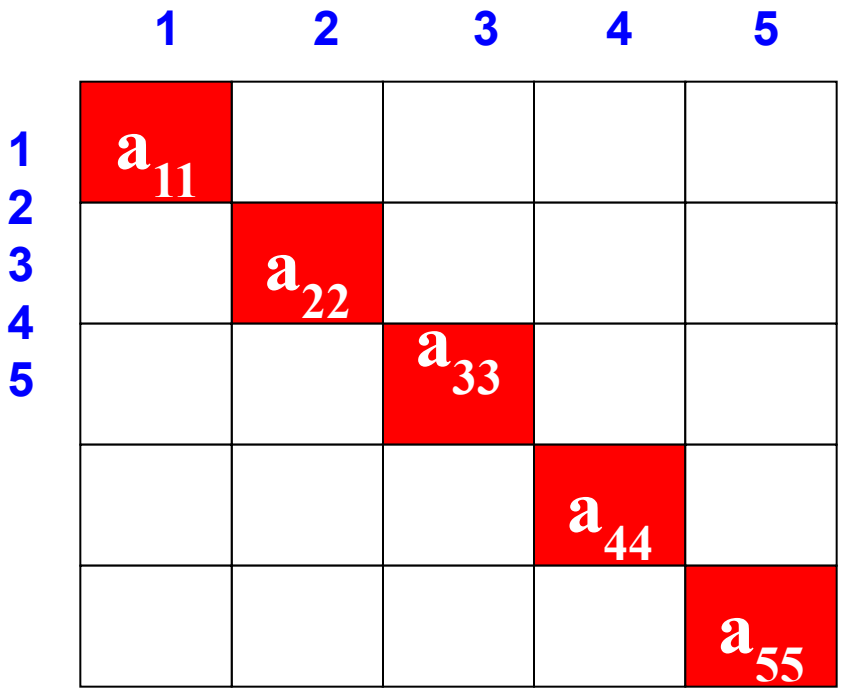
Если побочная диагональ то заполнить ячейку 1 во всех остальных случаях 0

01000

10000



Фрагмент блок-схемы задачи заполнения побочной диагонали единицами



Удовлетворяет неравенству

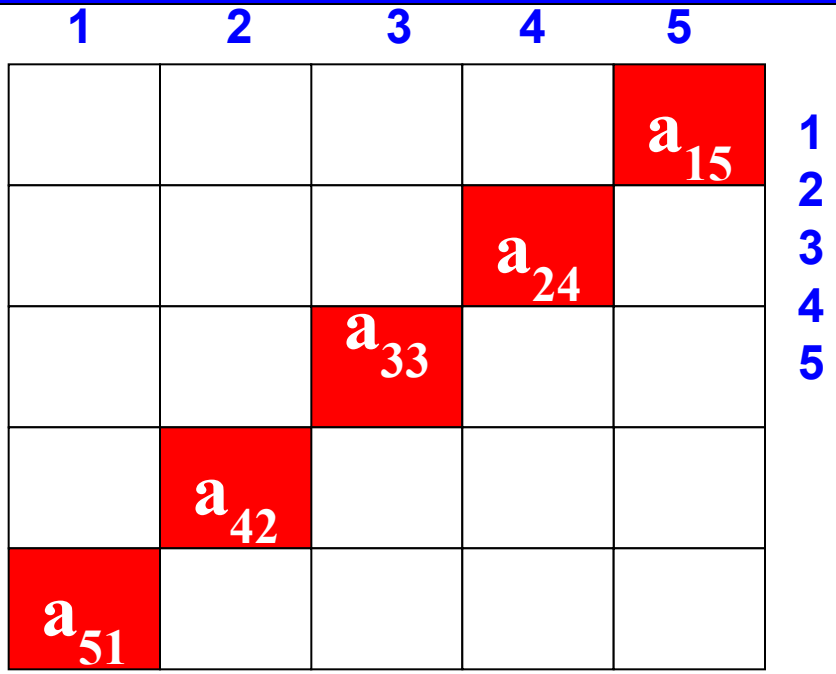
i < j

Удовлетворяет неравенству

i > j

Удовлетворяет неравенству

$i+j<n+1$



Удовлетворяет неравенству

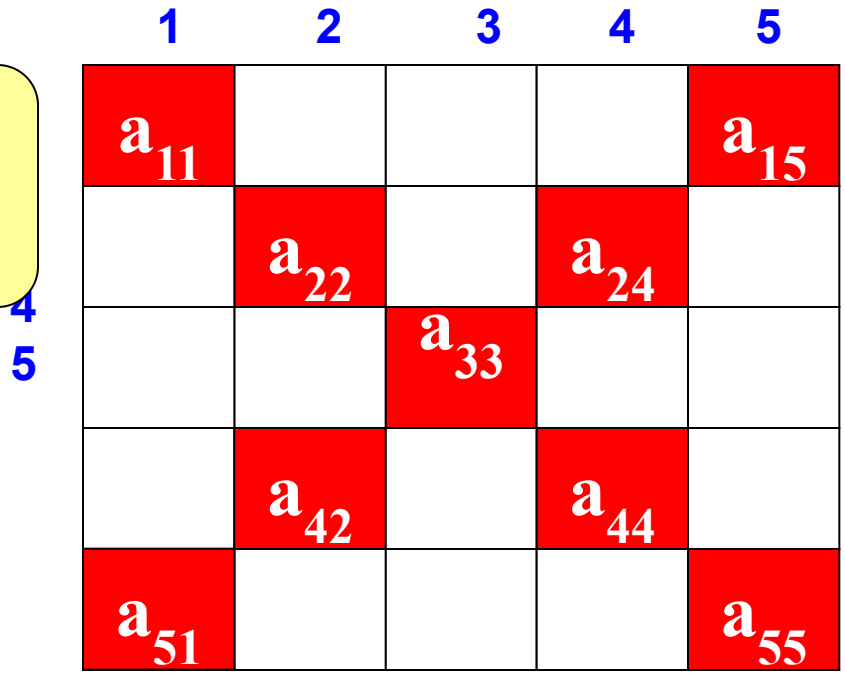
$i+j>n+1$

Системы неравенств

```
(i+j<n+1) And (i<j)
```

```
(i+j<n+1)
And (i>j)
```

```
(i+j>n+1)
And (i<j)
```




```
(i+j>n+1) And (i>j)
```

```
Write('ВВеди N = '); Readln(n);
 For i:=1 to n do begin
  For j:=1 to n do begin
     if (i+j>=n+1) and (i<=j) then a[i,j]:=1
                              else a[i,j]:=0;
     write(a[i,j]:2);
      end;
 writeln;
 end;
```

```
CRT - программа завершена
ВВеди N = 9
 0 0 0 0 0 0 0 0 1
 0 0 0 0 0 0 0 1 1
 0 0 0 0 0 0 1 1 1
 0 0 0 0 0 1 1 1 1
 0 0 0 0 1 1 1 1 1
 0 0 0 0 0 1 1 1 1
 0 0 0 0 0 0 1 1 1
 0 0 0 0 0 0 0 1 1
 0 0 0 0 0 0 0 0 1
```

Второй способ. Два прохода по массиву

Замечание.

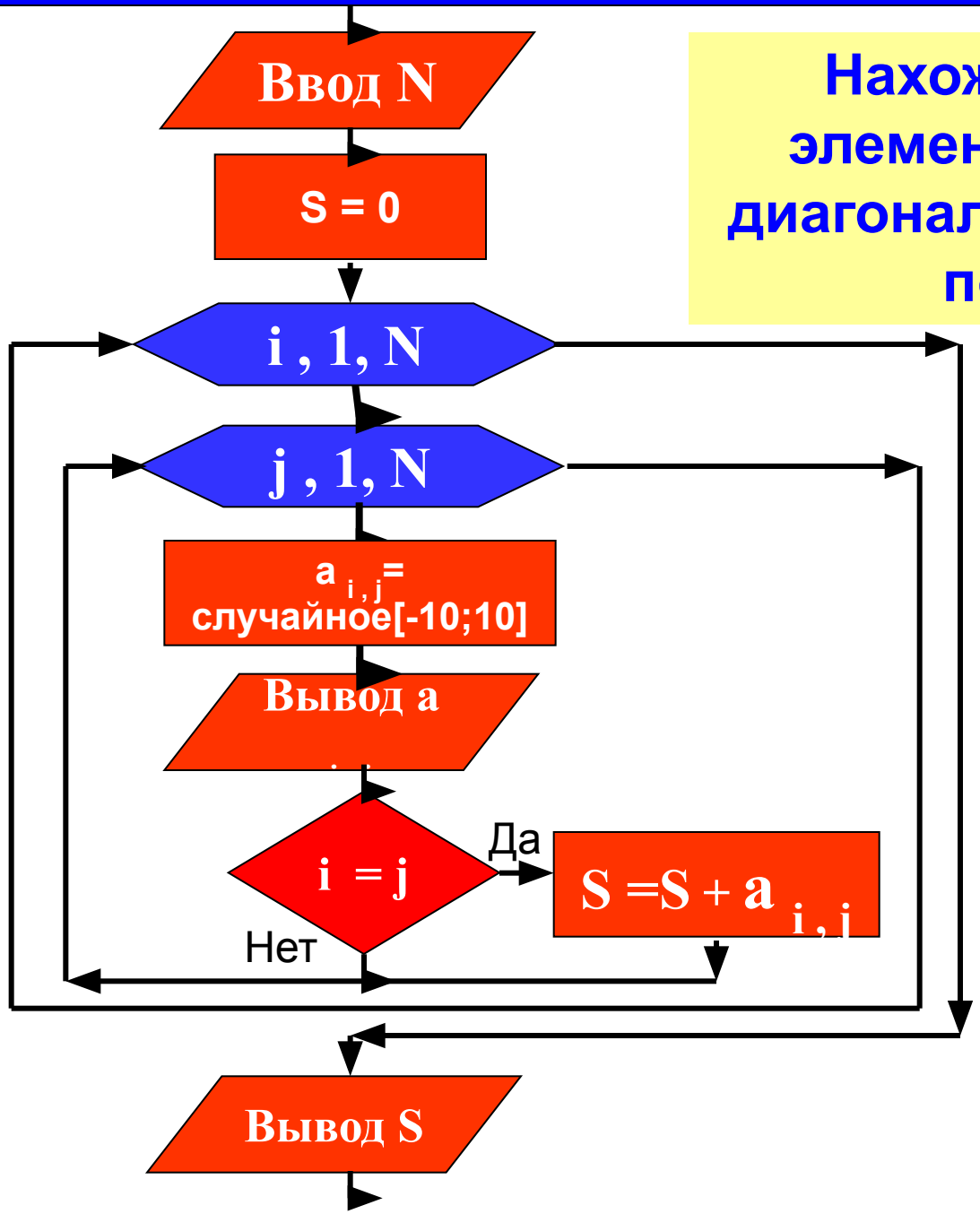
Если нужно что то сделать только с главной диагональю, то можно обойтись без вложенных циклов

```
For i:=1 to n do s:=s+ a[i,i];
```

```
end;
```

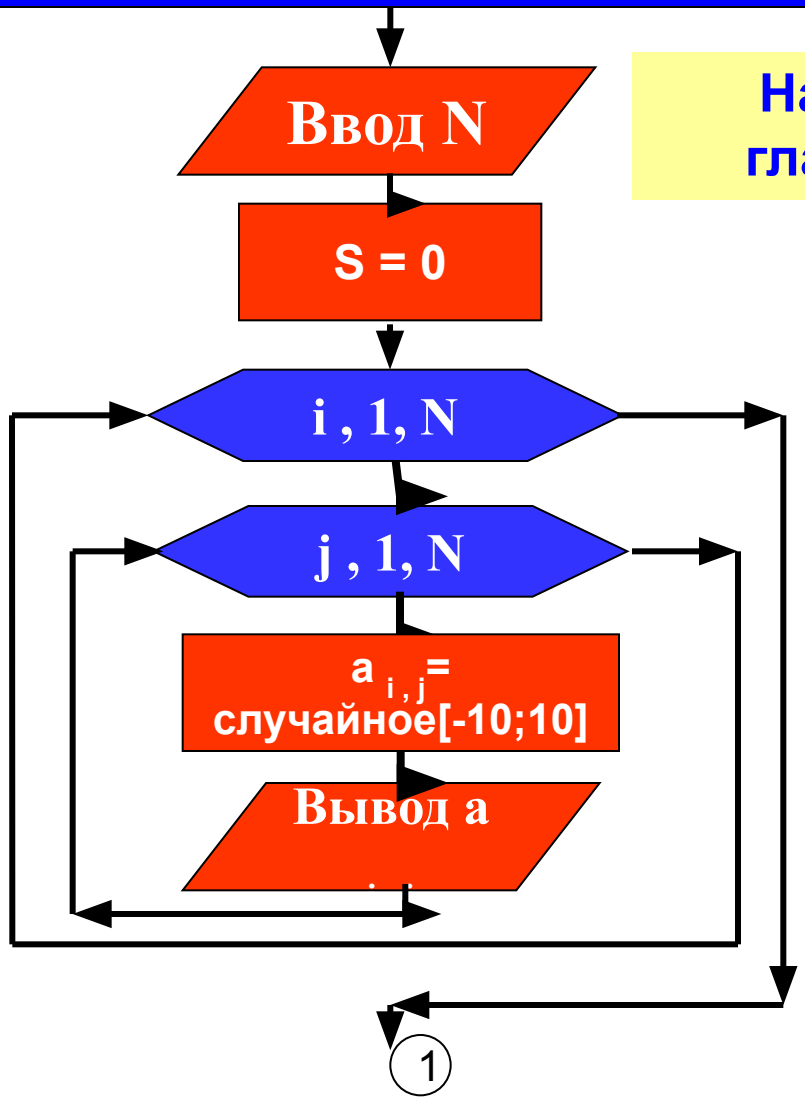
Находим сумму.

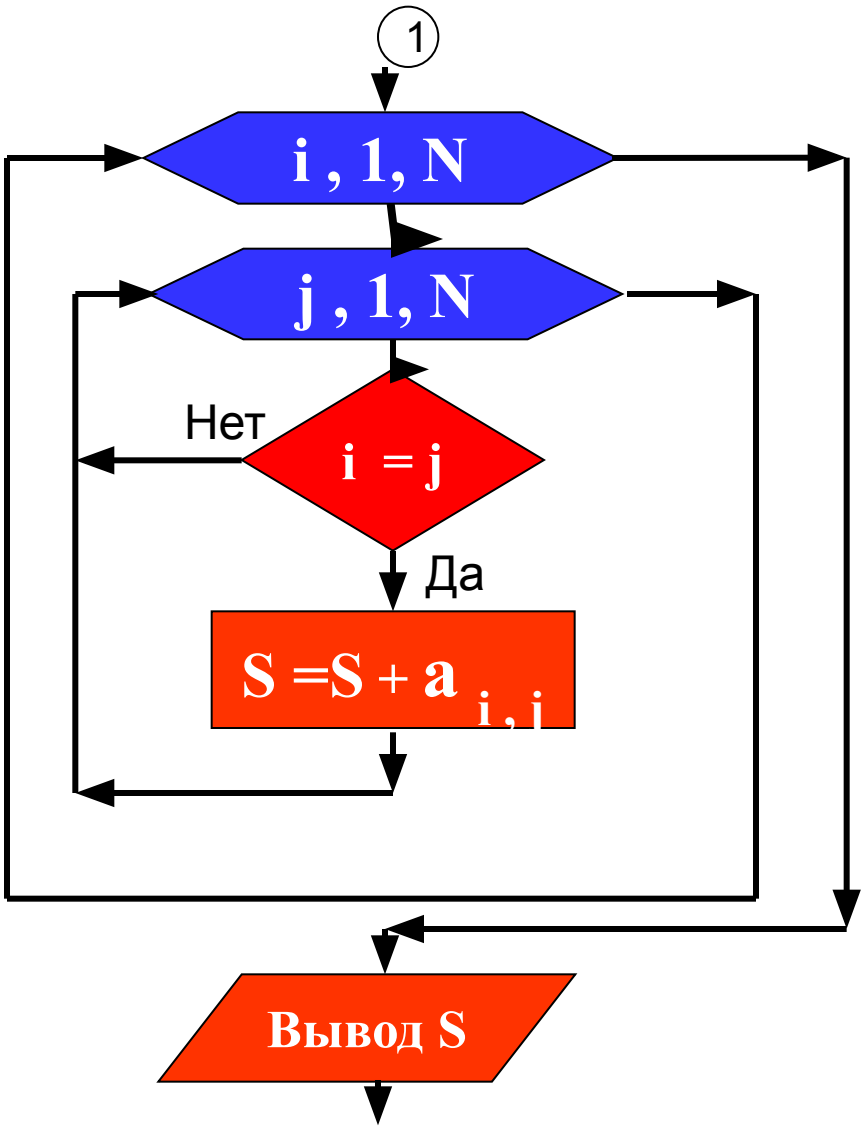
```
For i:=1 to n do
  For j:=1 to n do
    if (i=j) then s:=s+ a[i,j];
writeln('Сумма элементов =',s:5);
```



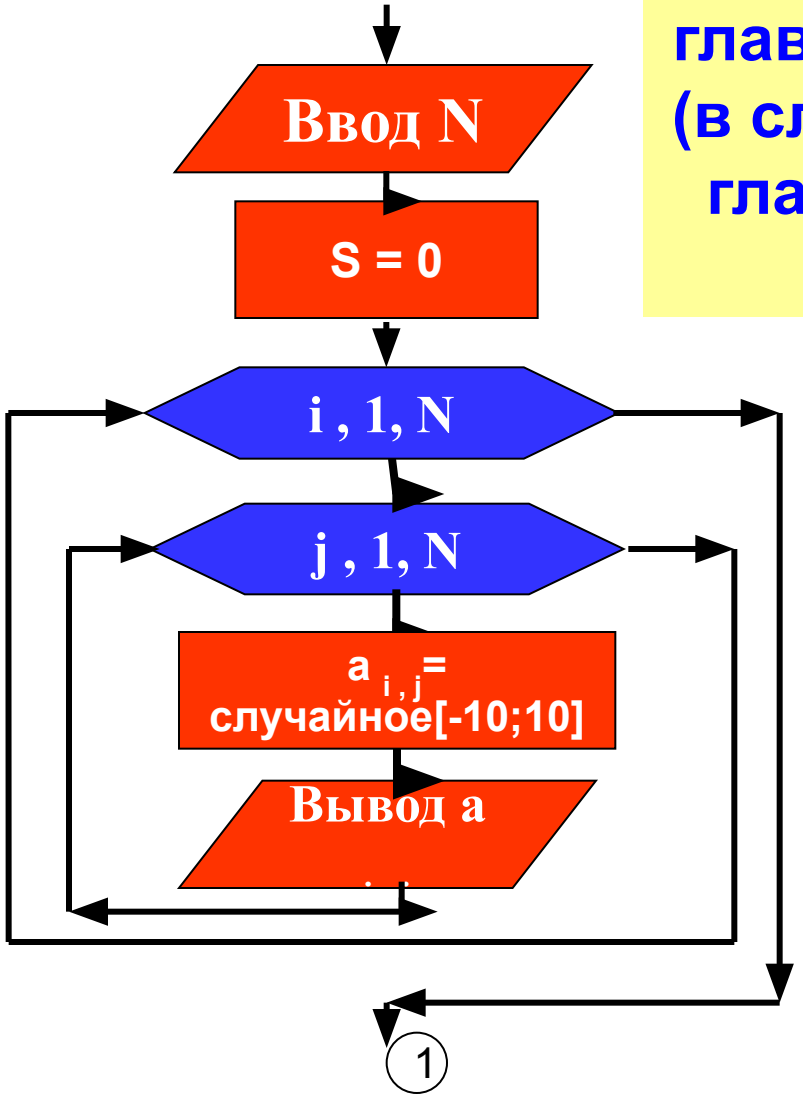
Нахождение суммы элементов на главной диагонали за один проход по массиву

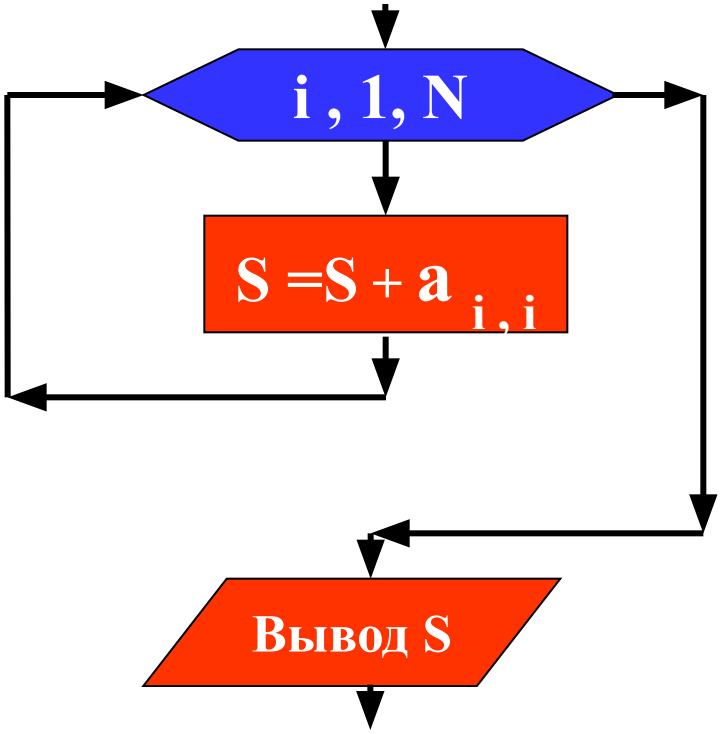
Нахождение суммы элементов на главной диагонали за два прохода





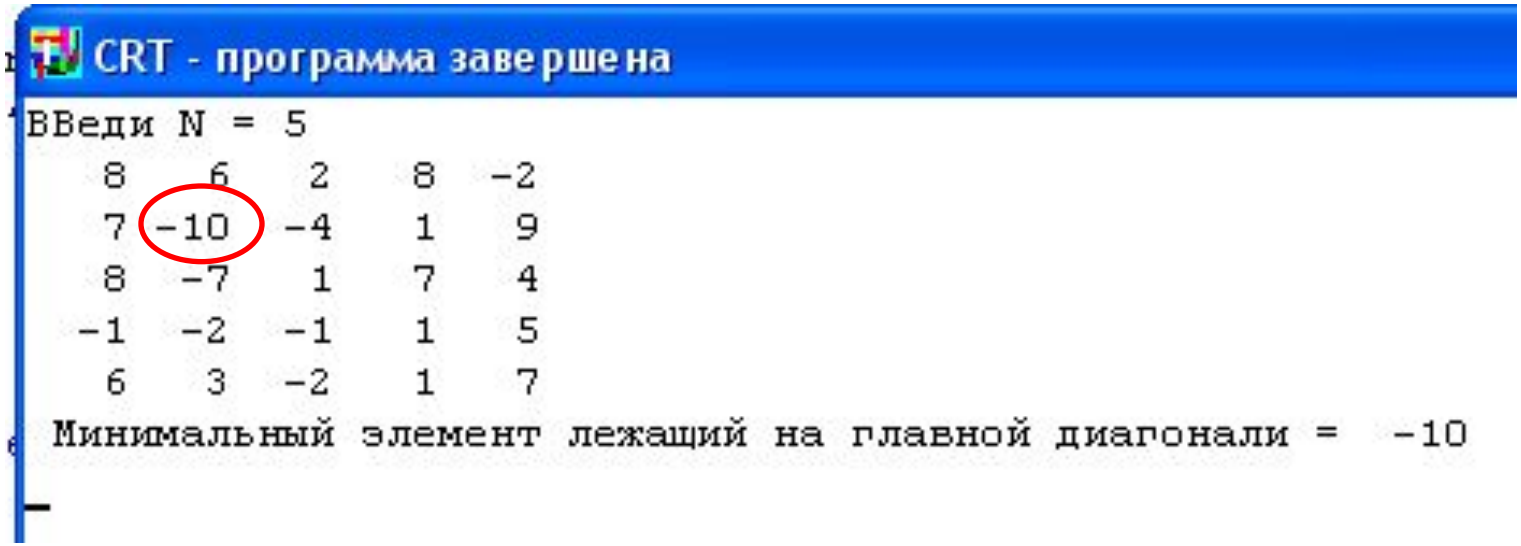
Нахождение суммы элементов на главной диагонали за два прохода (в случае когда речь идет только о главной диагонали и остальной массив не нужен)





Один или два прохода по массиву? Рассуждение второе.

Заполнить двумерный массив N x N случайными числами из интервала [-10 ; 10] и найти минимальный элемент лежащий на главной диагонали.



Стандартный способ без анализа задачи

Write('ВВеди N = ');Readln(n);

Замечание.

В данном случае можно не бегать по всему массиву а пройти только по главной диагонали, обойдясь без вложенных циклов.

```
m:=a[1,1];
For i:=2 to n do
    if (a[i,i]<m)  then m:=a[i,i];
```

for j:=1 to n do

if (a[i,j]<m) and (i=j) then m:=a[i,j];

writeln(' Минимальный элемент =',m:5);

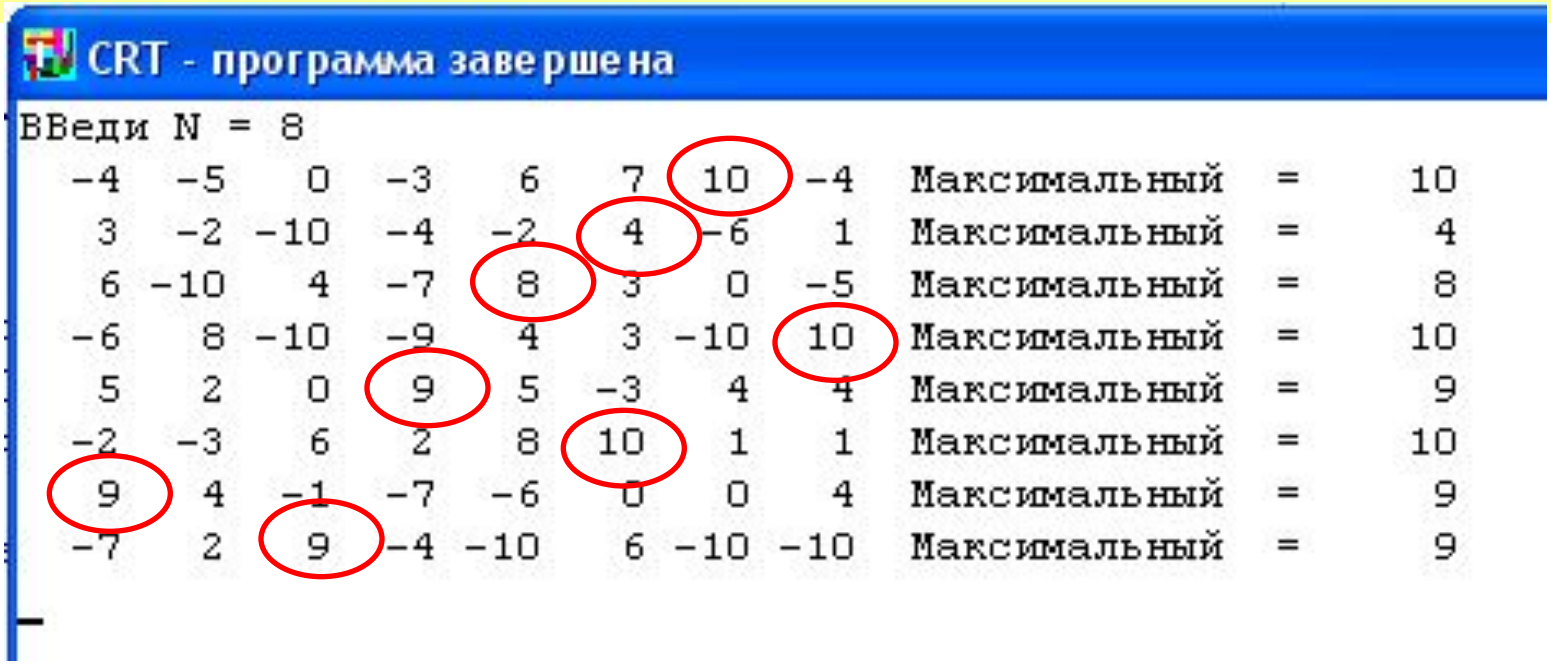
С анализом исходных данных задачи

```
Write('ВВеди N = ');Readln(n);
 m:=10;
  For i:=1 to n do begin
   For j:=1 to n do begin
                  a[i,j]:=random(21)-10;
                  write(a[i,j]:4);
                  if (a[i,j]<m) and (i=j) then m:=a[i,j];
                  end;
   Writeln;
   end;
  writeln(' Минимальный эле                    а
главной диагонали =',m:5);
```

Предполагаем, что самое маленькое число 10, правая граница исходного интервала.

Заполняем, выводим на экран и сразу проверяем на минимальность в главной диагонали.

Заполнить двумерный массив N x N случайными числами из интервала [-10 ; 10] и найти максимальный элемент в каждой строке.



Решение в один п
с анализом зад

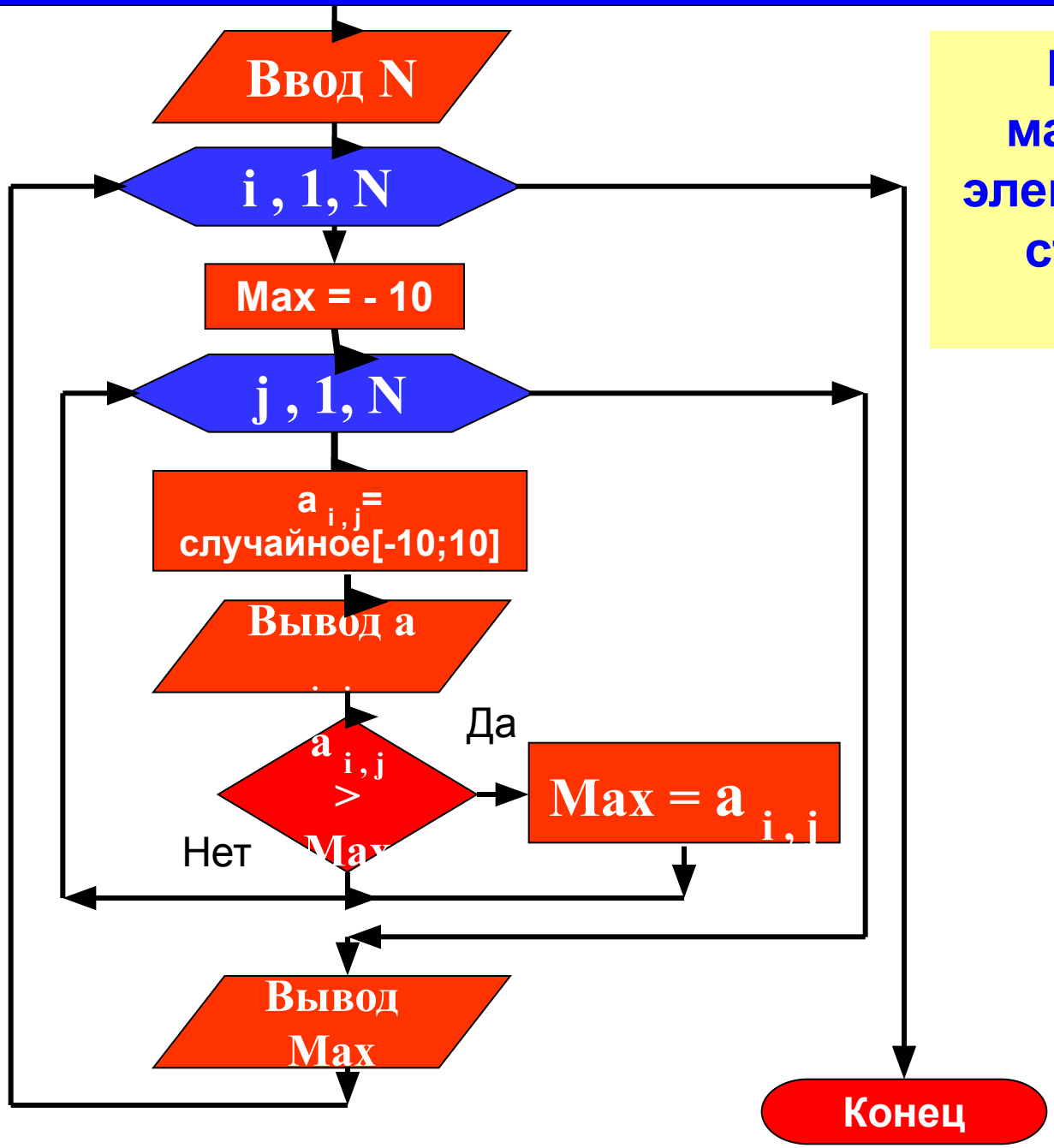
```
Wr
Fo
  m
  Fo
        a[i,j]:=random(21)-10;
        write(a[i,j]:4);
        if (a[i,j]>max) then max:= a[i,j];
      end;
  write('  MAx  = ',max:5);
  writeln;
end;
```

Перебираем строки

Заполняем элемент массива и выводим его на экран

Если в строке встречается элемент
го, то он
тся
ьным

Выводим наибольший элемент в строке



Нахождение максимального элемента в каждой строке в один проход

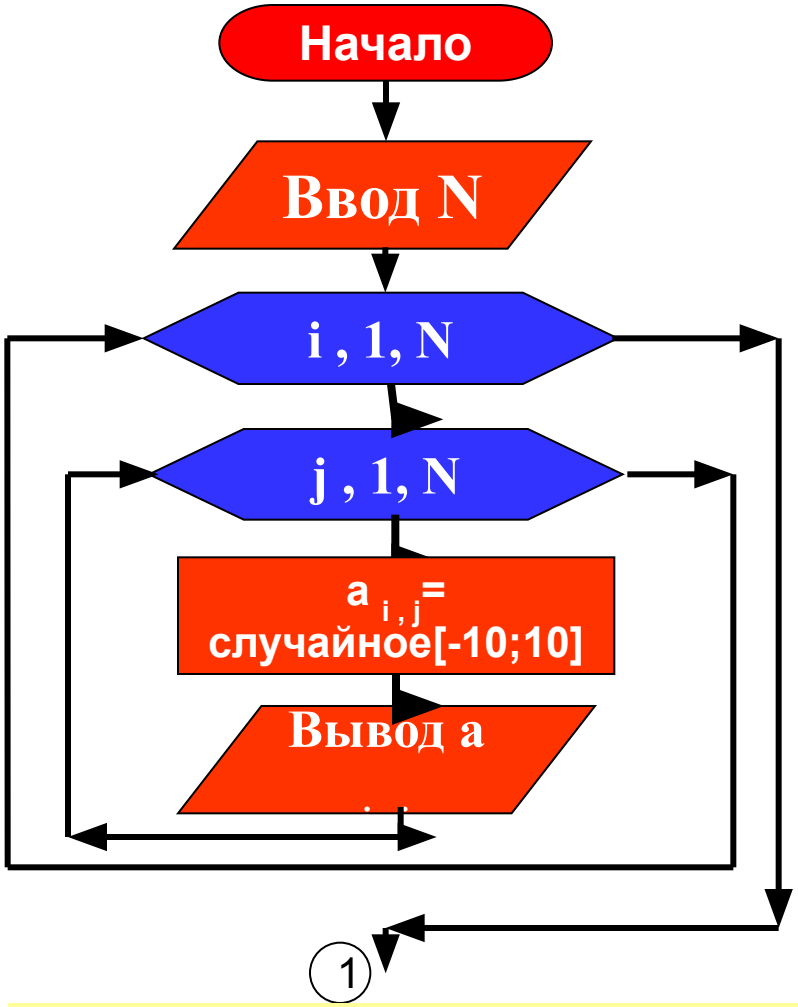
```
For i:=
  max:
  For j
    [i,j];
    writ
      5);
end;
```

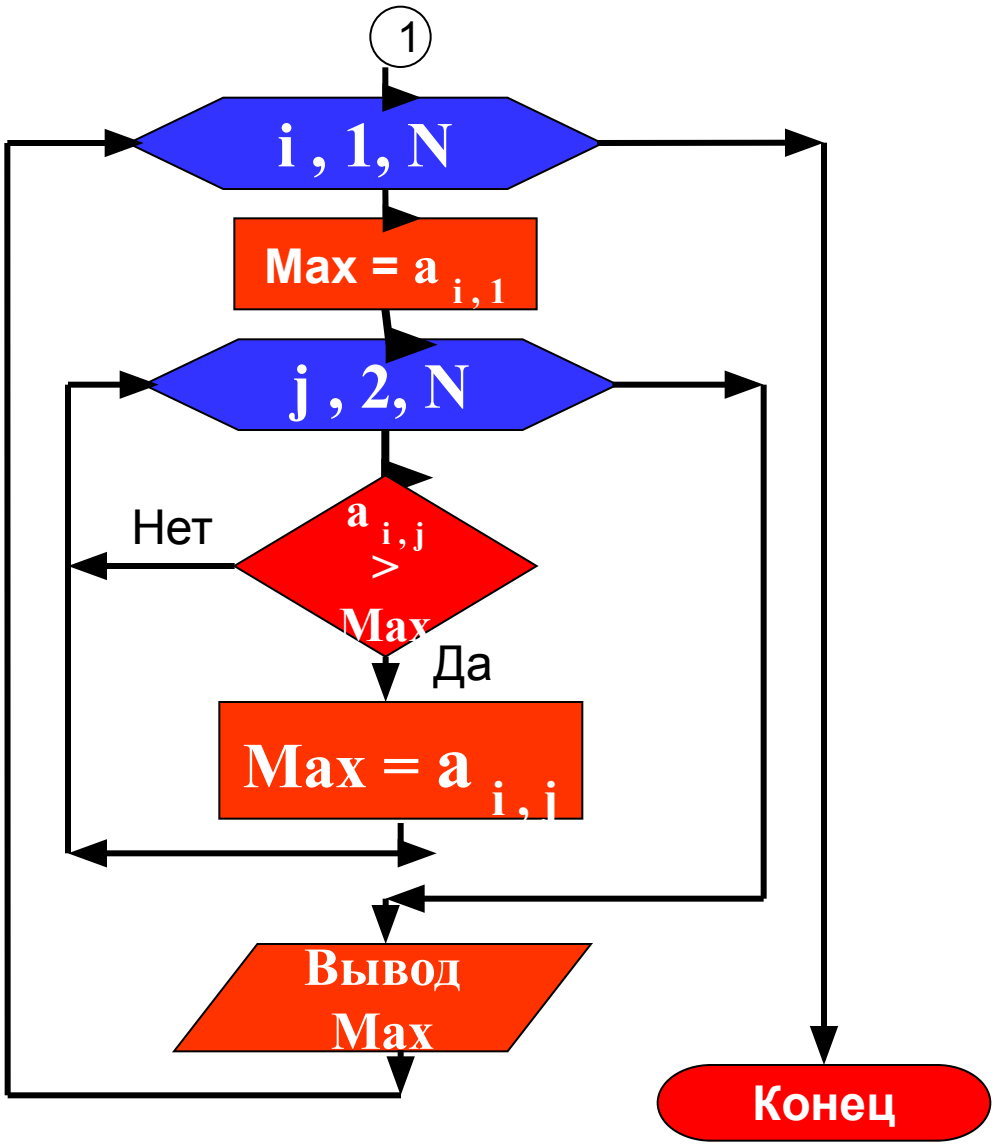
Предполагаем, что мент в оит на те

Стандартным способом, первый проход – заполнение, второй проход – поиск максимального в строках

Вывод наибольшего элемента в строке

Идем по строке и если находим элемент больший чем максимальный, то он становится максимальным





Нахождение максимального элемента в каждой строке в два прохода