Многоступенчатые циклические вычислительные процессы. Двумерные массивы.

Цель работы: Научиться разрабатывать алгоритмы для решения задач на основе Многоступенчатых циклических вычислительных процессов для работы с двумерными массивами.

Используемое оборудование: ПК.

Задание №1

Постановка задачи: Найти сумму всех элементов массива 3x3. Массив задается явно внутри программы. Найти максимальный элемент.

Математическая модель:

S = a[1,1]+a[1,2]+a[1,3]+a[2,1]+a[2,2]+a[2,3]+a[3,1]+a[3,2]+a[3,3]

Блок схема:

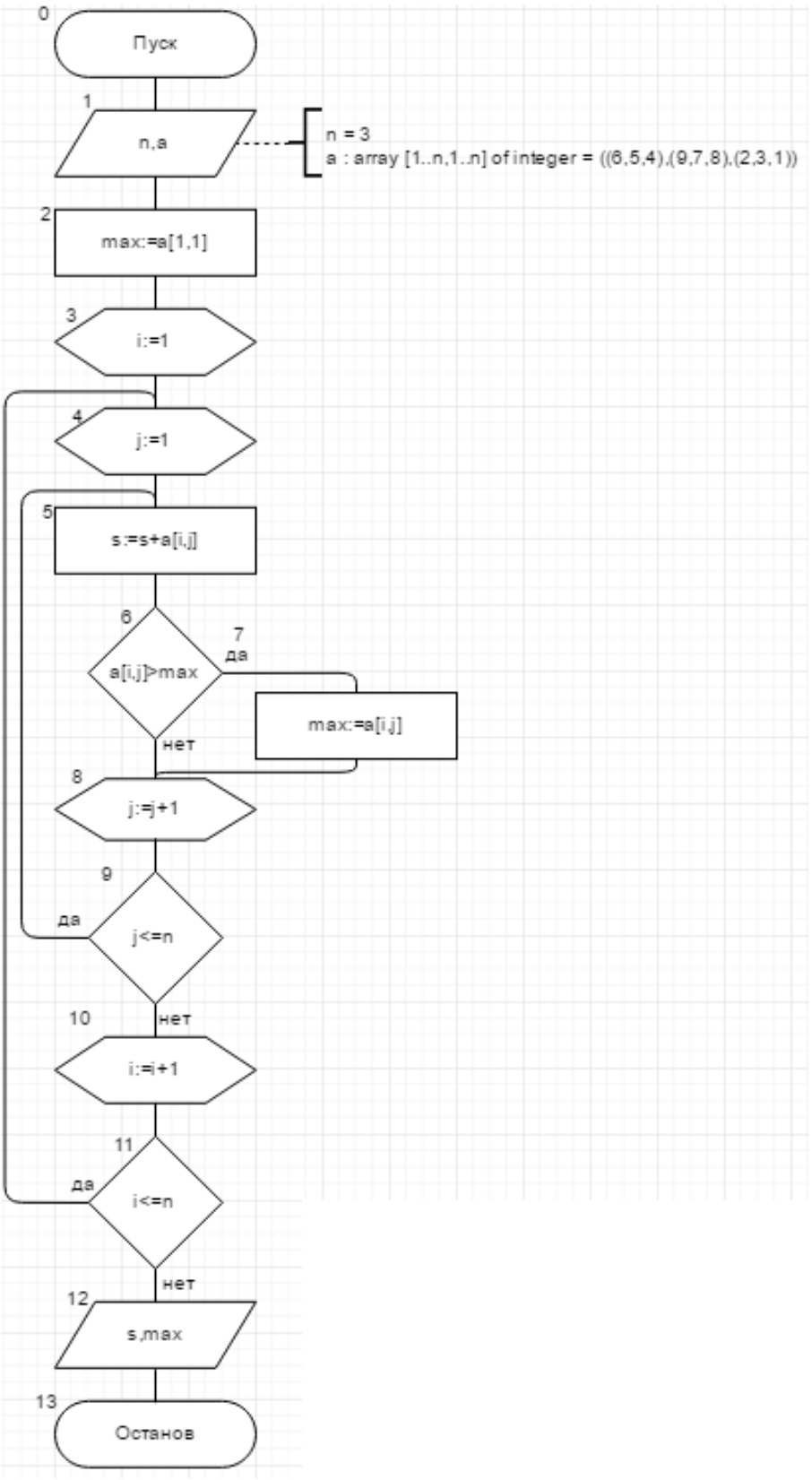

Рис 1 – блок схема к 1 задаче

Список идентификаторов:

Таблица 1 – список идентификаторов для 1 задачи

Название	Смысл	Тип
N	Ограничение массива	Integer
A	Массив, содержащий элементы для работы с ними	Array
I	Параметр цикла для работы с строковыми значениями массива	Integer
J	Параметр цикла для работы со столбцовыми значениями массива	Integer
Max	Переменная, содержащая в себе максимальный элемент	Integer
S	Сумма элементов массива	Integer

Код программы:

```
program mda;

const n=3;
a:array[1..n,1..n] of integer=((6,5,4),(9,7,8),(2,3,1));
var
i,j,max,s:integer;
begin
  max:=a[1,1];
  s:=0;
  for i:=1 to 3 do
  begin
    for j:=1 to 3 do
    begin
      s:=s+a[i,j];
      if max<a[i,j] then max:=a[i,j];
    end;
  end;
  Writeln(s);
  writeln(max);
end.
```

Результаты выполненной работы:

Рис 2 – результат выполнения 1 программы

Анализ результатов вычисления: Для явного задания массива массив был инициализирован в константах. Для работы с каждым элементом массива был создан цикл, проходящий по строкам массива и вложенный цикл для работы с элементами массива на строке. Для подсчета суммы элементов массива переменная s накапливает в себе сумму самой себя и очередного элемента массива. Для нахождения максимального элемента переменная max

сравнивается с данным элементом массива, и если он больше, то переменной max присваивается значение a[I,j].

Задание №2

Постановка задачи: Дан массив 3x3. Найти сумму элементов на главной диагонали и сумму элементов побочной диагонали.

Математическая модель:

Сумма главной диагонали $g = \sum_{i,j=1}^{n} a[i,j]$, шаг по $i,j = 1$

Сумма побочной диагонали $p = \sum_{i,j=n}^{1} a[i,j]$, шаг по $i,j = -1$

Блок схема:

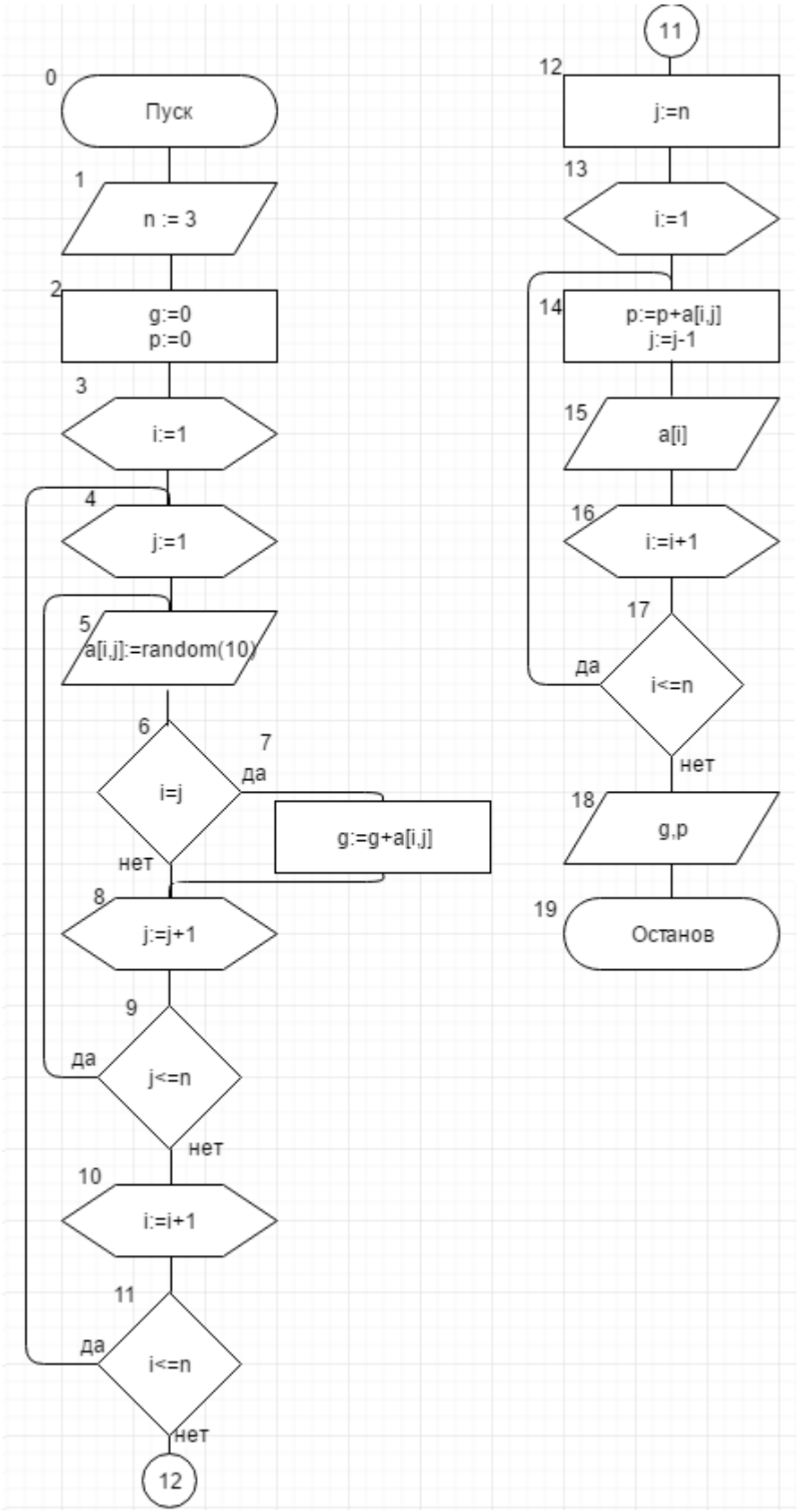

Рис 3 – блок схема к 2 задаче

Список идентификаторов:

Таблица 2 – список идентификаторов для 2 задачи

Название	Смысл	Тип
N	Ограничение массива	Integer
A	Массив для работы с элементами	Array
I	Параметр цикла для работы с строковыми значениями массива	Integer
J	Параметр цикла для работы с столбцовыми значениями массива	Integer
G	Переменная для хранения суммы элементов главной диагонали	Integer
P	Переменная для хранения суммы элементов побочной диагонали	Integer

Код программы:

```
program mda;
const n=3;
var a:array [1..n,1..n] of integer;
i,j,g,p:integer;
begin
  g:=0;
  p:=0;
  for i:=1 to n do
    for j:=1 to n do
    begin
      a[i,j]:=random(10);
      if i=j then g:=g+a[i,j]
    end;
  j:=n;
  for i:=1 to n do
  begin
    p:=p+a[i,j];
    j:=j-1;
    writeln(a[i]);
  end;
  writeln(g);
  writeln(p);
end.
```

Результаты выполненной работы:

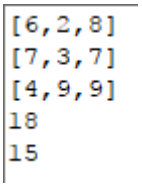

Рис 4 – результат работы 2 программы

Анализ результатов вычисления: Посчитать сумму элементов главной диагонали достаточно просто, достаточно в цикл инициализации массива добавить условие, при i=j g=g+a[i,j]. Но для подсчета суммы побочной

диагонали все немного сложнее, первоначально была идея создать условие, похожее на условие для главной диагонали, чтобы весь алгоритм прошел в одном цикле, однако, эта идея не увенчалась успехом. Мы остановились на варианте создания 1 цикла для работы с i и вручную j уменьшать от n do 1, тогда p=p+a[i,j], также в этом цикле был реализован построчный вывод массива.

Задание №3

Постановка задачи: Дан массив 3x3. Заменить элементы, стоящие ниже главной диагонали нулями.

Блок схема:

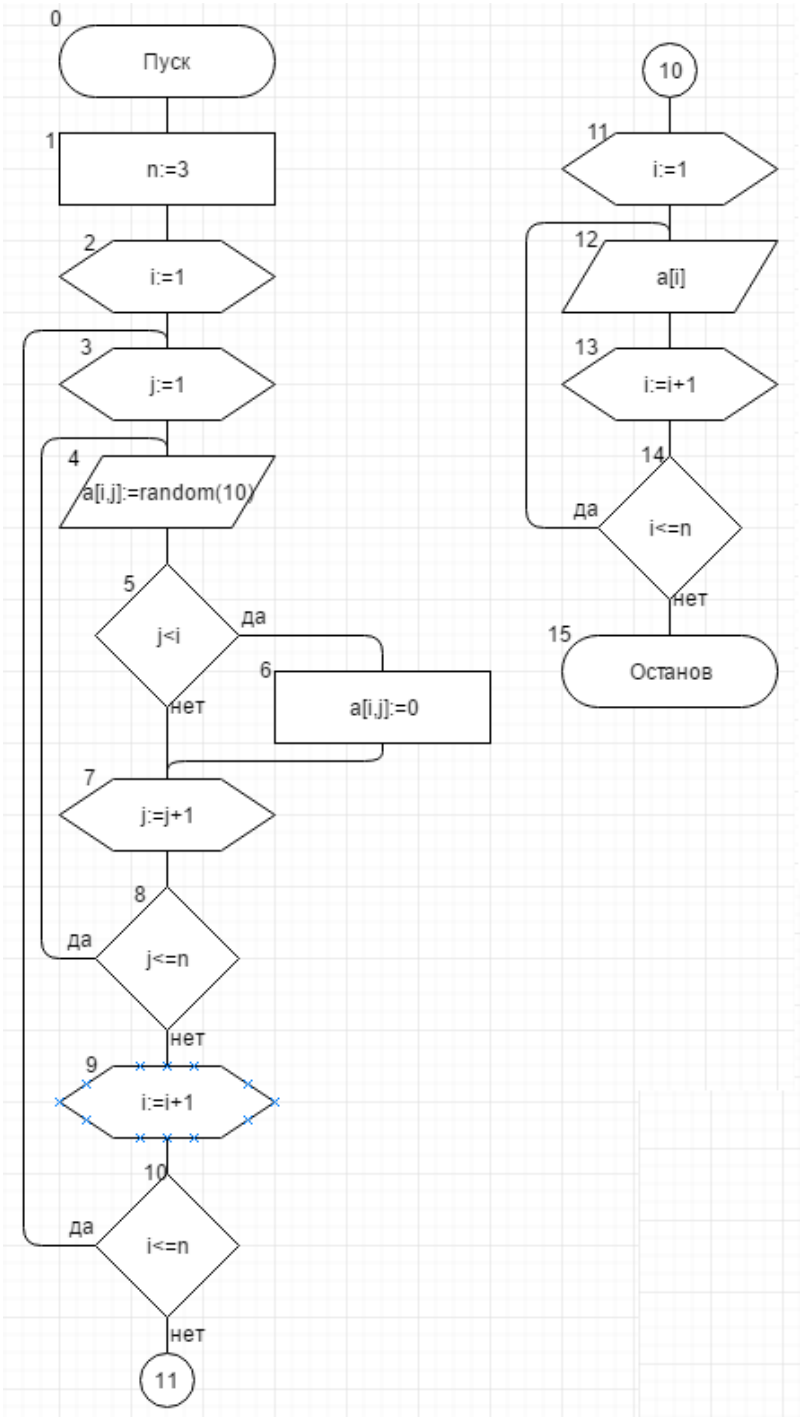

Рис 5 – блок схема к 3 задаче

Список идентификаторов:

Таблица 3 – список идентификаторов к 3 задаче

Наименование	Смысл	Тип
N	Ограничение массива	Integer
A	Массив с элементами для работы	Array
I	Параметр цикла для работы со строковыми значениями массива	Integer
J	Параметр цикла для работы с столбцовыми значениями массива	Integer

Код программы:

```
program mda;
const n=3;
var a:array [1..n,1..n] of integer;
i,j:integer;
begin
  for i:=1 to n do
    for j:=1 to n do
    begin
      a[i,j]:=random(10);
      if j<i then a[i,j]:=0;
    end;
  for i:=1 to n do
    writeln(a[i]);
end.
```

Результаты выполненной работы:

```
[2,6,5]
[0,3,1]
[0,0,0]
```

Рис 6 – результат работы 3 программы

Анализ результатов вычисления: Для нахождения элементов ниже главной диагонали достаточно ввести условие j<i для a[i,j] и при его выполнения присваивать ему значение 0.

Задание №4

Постановка задачи: Дана матрица 3х3. Найти суммы элементов каждой строки и упорядочить строки по возрастанию согласно их суммам.

Математическая модель:

Сумма элементов строки $c(i) = \sum_{j=1}^{n} a[i,j]$

Блок схема:

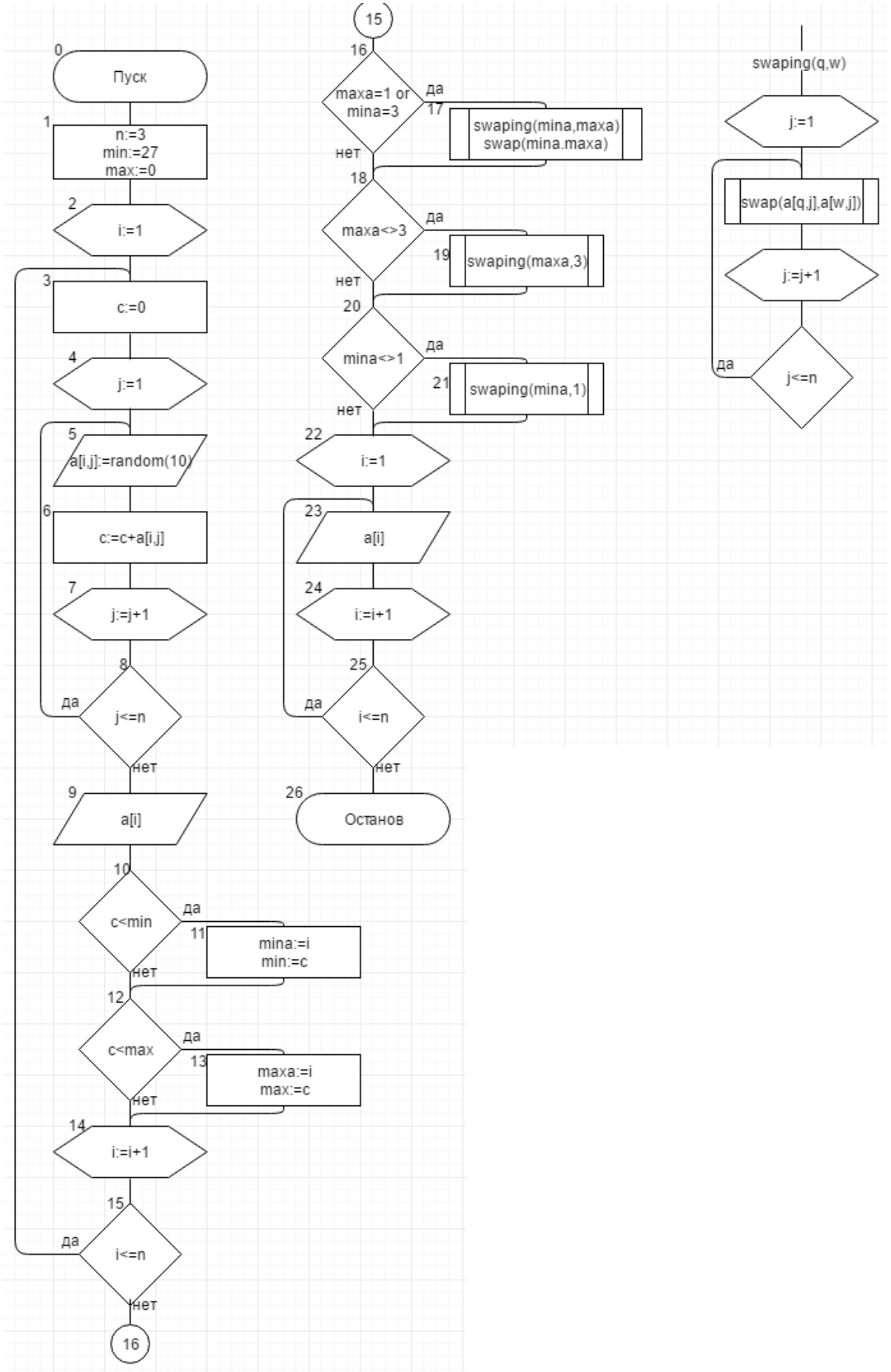

Рис 7 – блок схема к 4 задаче

Список идентификаторов:

Таблица 4 – список идентификаторов к 4 задаче

Наименование	Смысл	Тип
N	Ограничение массива	Integer
A	Массив с элементами для работы	Array
I	Параметр цикла для работы с строками массива	Integer
J	Параметр цикла для работы с столбцами массива	Integer
C	Переменная для хранения суммы элементов строки	Integer
Min	Переменная для хранения минимальной суммы	Integer
Max	Переменная для хранения максимальной суммы	Integer
Mina	Переменная для хранения номера строки с минимальной суммой	Integer
Maxa	Переменная для хранения номера строки с максимальной суммой	Integer

Код программы:

```
program mda;
const n=3;
var a:array [1..n,1..n] of integer;
i,j,c,min,max,mina,maxa:integer;

procedure swaping(q,w: integer);
var j:integer;
begin
  for j:=1 to n do
    swap(a[q,j],a[w,j]);
end;
begin
  min:=27;
  max:=0;
  for i:=1 to n do
  begin
    c:=0;
    for j:=1 to n do
    begin
      a[i,j]:=random(10);
      c:=c+a[i,j];
    end;
    writeln(a[i]);
    if c<min then
    begin
      mina:=i;
      min:=c;
    end;
    if c>max then
    begin
      maxa:=i;
      max:=c;
    end;
  end;
  if (maxa=1) or (mina=3) then
  begin
    swaping(mina,maxa);
    swap(mina,maxa);
  end;
  if maxa<>3 then
    swaping(maxa,3);
  if mina<>1 then
   swaping(mina,1);
  writeln('Упорядоченая матрица:');
```

```
  for i:=1 to n do
    writeln(a[i]);
end.
```

Результаты выполненной работы:

```
[9,3,3]
[9,0,1]
[7,7,1]
Упорядоченая матрица:
[9,0,1]
[7,7,1]
[9,3,3]
```

Рис 8 – результат работы 4 программы

Анализ результатов вычисления: Создание данного алгоритма оказалось совсем не легким. Для упорядочивания строк по возрастанию была создана процедура перемены строк местами, она работает следующим образом: в процедуру передается номера строк, которые нужно поменять местами, процедура циклом проходит по элементам строк и поочередно меняет их местами. Но не только это вызывало трудность. При такой ситуации, в которой строка с максимальной суммой элементов на 1 позиции, а строка с минимальной суммой на 3 следуя нашему коду эти 2 строки меняются местами 2 раза и получается, что все остается по-прежнему и массив остается не отсортированным. Для исправления данной ситуации было добавлено условие – если максимальная строка на 1 месте, или минимальная строка на 3 месте, тогда поменять их местами и поменять местами значения переменных maxa и mina. Данное введение помогло решить проблемы и кода и работать алгоритму безотказно.

Вывод: Нами были решены все задачи согласно их постановке, все алгоритмы решают задачи эффективно.