**Лабораторная работа №12**

**Многоступенчатые циклические вычислительные процессы. Двумерные массивы.**

2. Цель работы: научиться решать задачи посредством многоступенчатых вычислительных процессов.

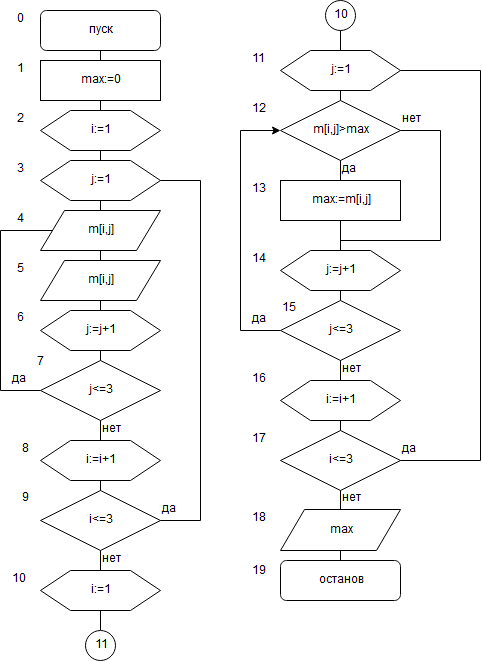
3. Используемое оборудование: ПК, Lazarus, PascalABC.

**Задание №1**

4. Найти сумму всех элементов массива 3x3. Массив задается явно внутри программы. Найти максимальный элемент.

5. if m[I,j]>max then max:=m[I,j]

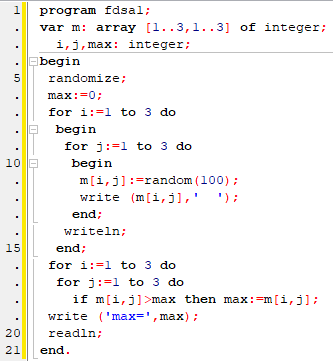
6.



7.

|  |  |  |
| --- | --- | --- |
| Имя | Смысл | Тип |
| m | массив | Array of integer |
| i | Номер строки/параметр цикла | integer |
| j | Номер столбца/параметр цикла | integer |
| max | Максимальный элемент массива | integer |

8.



9.



10. Для нахождения максимального элемента массива был организован многоступенчатый вычислительный процесс. Во внешнем цикле изменялась строка массива, во внутреннем – столбец. Если какой-то элемент при переборе был больше максимума – он становился максимумом.

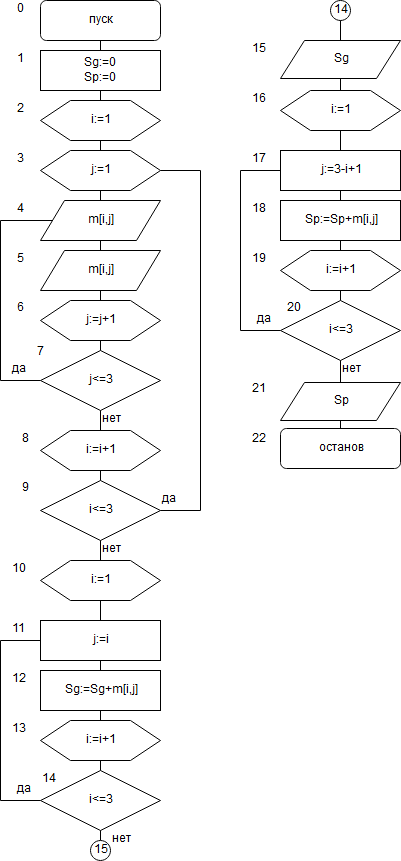
**Задание №2**

4. Дан массив 3x3. Найти сумму элементов на главной диагонали и сумму элементов побочной диагонали.

5. Главная диагональ: j:=i

Побочная диагональ: j:=n(размерность массива)-i+1

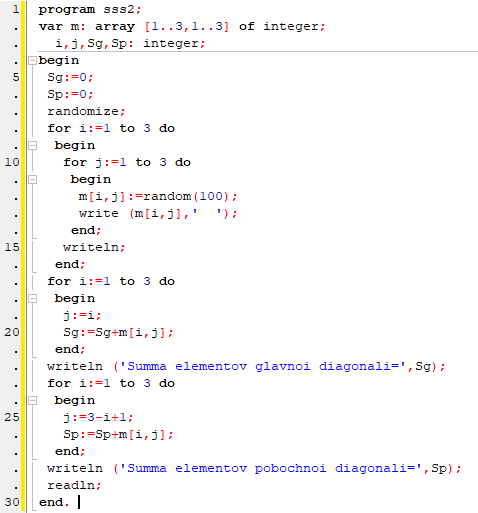
6.



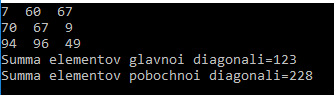
7.

|  |  |  |
| --- | --- | --- |
| Имя | Смысл | Тип |
| m |  | Array of integer |
| i | Номер строки/параметр цикла | integer |
| j | Номер столбца/параметр цикла | integer |
| Sg | Сумма элементов главной диагонали | integer |
| Sp | Сумма элементов побочной диагонали | integer |

8.



9.



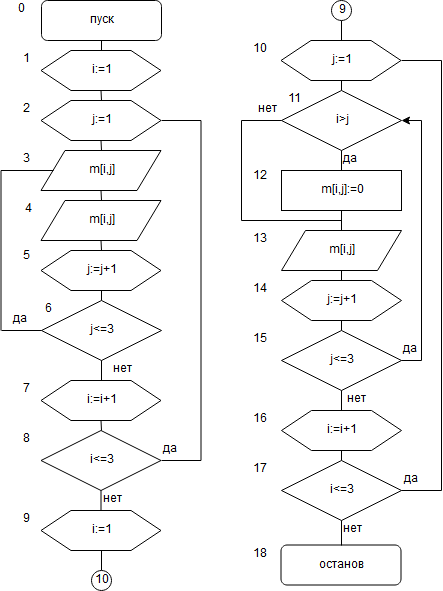
10. Для нахождения суммы главной диагонали в цикле складывались те элементы, у которых строка совпадала со столбцом. Для нахождения суммы побочной диагонали в цикле складывались те элементы, у которых столбцы равнялись 3-i+1.

**Задание№3**

4. Дан массив 3x3. Заменить элементы, стоящие ниже главной диагонали нулями.

5. if i>j then m[I,j]:=0

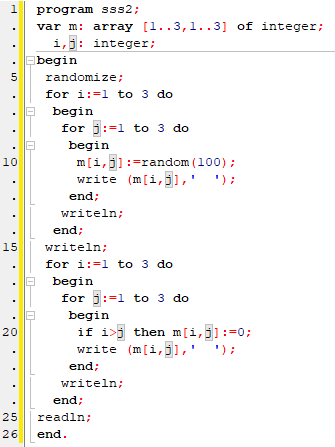
6.



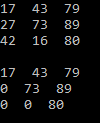
7.

|  |  |  |
| --- | --- | --- |
| Имя | Смысл | Тип |
| m | массив | Array of integer |
| i | Номер строки/параметр цикла | integer |
| j | Номер столбца/параметр цикла | integer |

8.



9.



10. На ноль заменялись те элементы, значение строки которых было больше, чем значение столбца (т.е. они находились под главной диагональю).

**Задание №4**

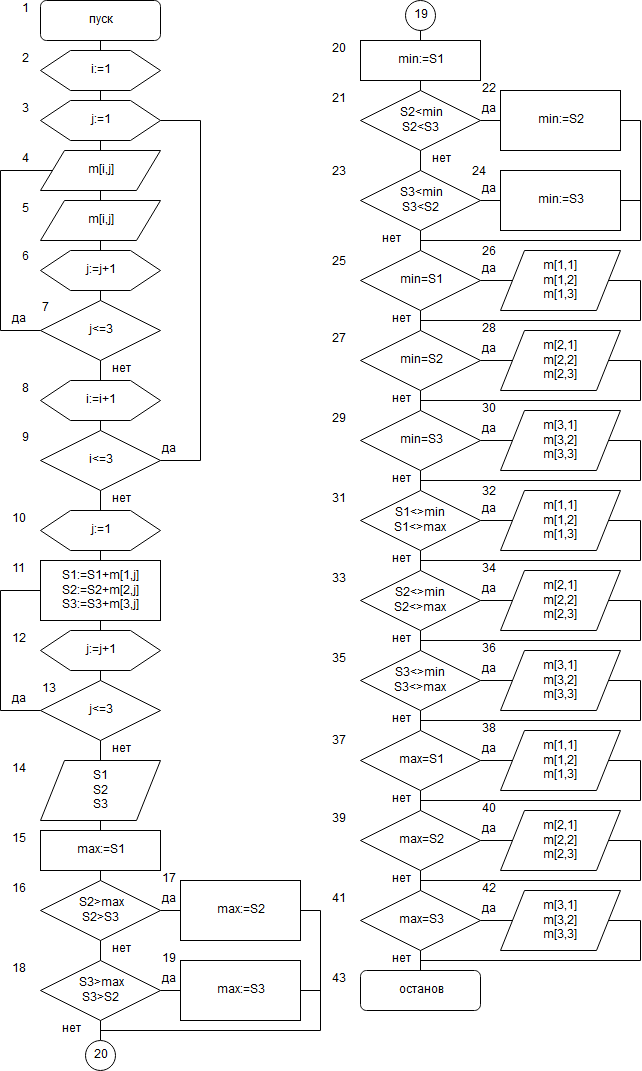
4. Дана матрица 3x3. Найти суммы элементов каждой строки и упорядочить строки по возрастанию согласно их суммам

5. S1:=S1+m[1,j]

S2:=S2+m[2,j]

S3:=S3+m[3,j]

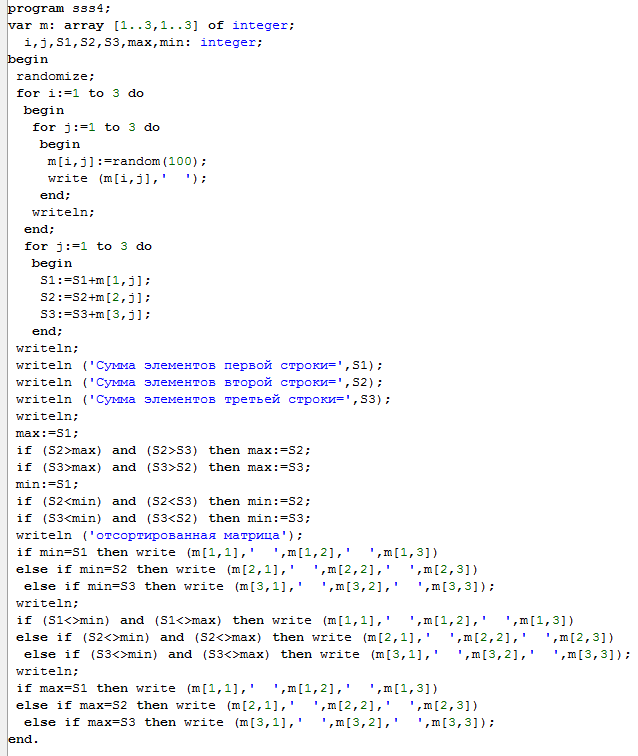
6.



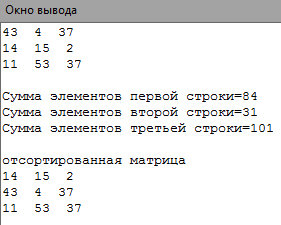
7.

|  |  |  |
| --- | --- | --- |
| Имя | Смысл | Тип |
| m | массив | Array of integer |
| i | Номер строки/параметр цикла | integer |
| j | Номер столбца/параметр цикла | integer |
| S1 | Сумма элементов первой строки | integer |
| S2 | Сумма элементов второй строки | integer |
| S3 | Сумма элементов третьей строки | integer |
| max | Максимальная сумма элементов строки | integer |
| min | Минимальная сумма элементов строки | integer |

8.



9.



10. Для сортировки массива сначала были найдены суммы элементов первой, второй и третьей строк. Далее переменным min и max были присвоены значения самой маленькой и самой большой суммы соответственно. Далее если сумма какой-либо строки совпадала с min, то она выводилась на экран. Потом если сумма какой-либо строки не совпадала ни с min ни с max, то она выводилась на экран. И если сумма какой-либо строки совпадала с max, то она выводилась на экран.

11. Вывод: Многоступенчатые вычислительные процессы отлично подходят для различных действий с двумерными массивами, потому что удобно параметром внешнего цикла назначать переменную, отвечающую за строки массива, а параметром внутреннего цикла – переменную отвечающую за столбцы массива.