*Федеральное государственное бюджетное образовательное учреждение* *высшего образования*

|  |  |
| --- | --- |
| **Gerb-BMSTU_01** | ***«Московский государственный технический университет  имени Н.Э. Баумана»***  ***(национальный исследовательский университет)*** |

ФАКУЛЬТЕТ ИНФОРМАТИКА И СИСТЕМЫ УПРАВЛЕНИЯ

КАФЕДРА Компьютерные Системы и сети (ИУ6)

**Отчет**

**по лабораторной работе №7**

**варианта №7**

**Название лабораторной работы:**

Подпрограммы. Средства отладки Delphi

**Дисциплина:**

Основы программирования

Студент гр. **ИУ6-12** **25/11/16\_\_\_\_\_\_\_\_\_\_ Векшин Роман**

(Подпись, дата) (И.О. Фамилия)

Преподаватель **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Черноусова Татьяна Геннадьевна**

(Подпись, дата) (И.О. Фамилия)

**Задание:**

Решить задачу, используя процедуру или функцию. Выбор обосновать. На примере полученной программы продемонстрировать умение:

1. назначать точку останова;
2. выполнить программу по шагам с заходом в процедуры и без захода;
3. определять значения переменных на конкретном шаге.

Задана матрица G(n,n), (n<=8). В каждой строке поменять местами максимальный и диагональный элементы. Напечатать исходную и преобразованную матрицы.

**Схема алгоритма:**

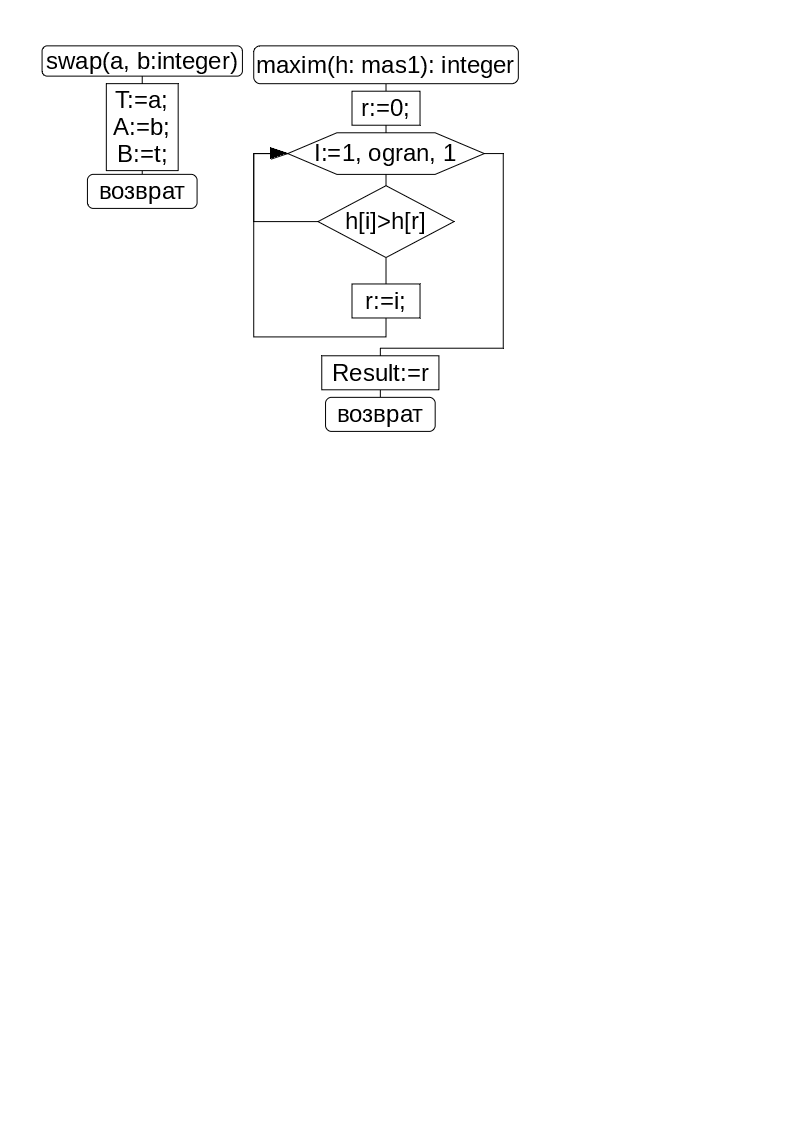
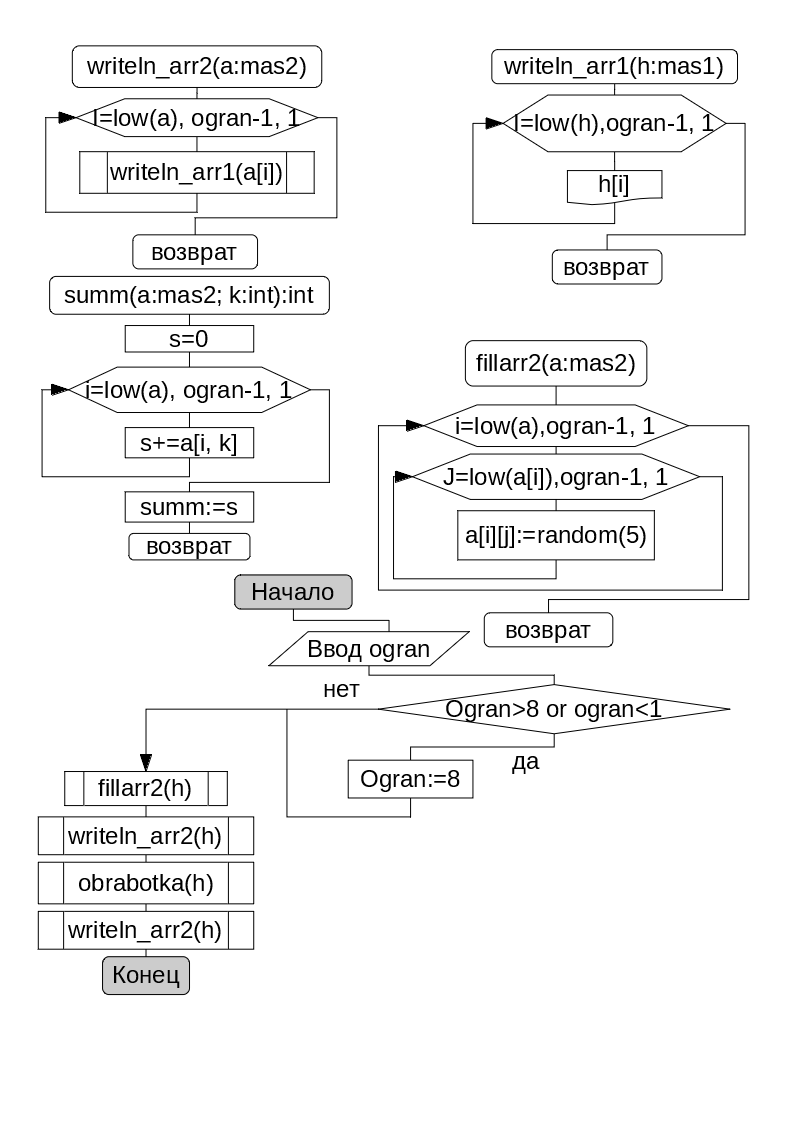


Рисунок 1-Схема алгоритма

**Код программы:**

program project1;

const

n = 8;

type

mas1 = array[0..n - 1] of integer;

mas2 = array[0..n - 1] of mas1;

var

ogran: byte;

**///Построчный вывод матрицы**

procedure writeln\_arr2(const a: mas2);

**///Процедура вывода одномерного массива**

procedure writeln\_arr1(const h: mas1);

var

i: byte;

begin

for i := low(h) to ogran - 1 do

Write(h[i]: 2);

writeln;

end;

var

i: byte;

begin

Write(' ');

for i := 1 to ogran do

Write(i: 2);

writeln();

for i := low(a) to ogran - 1 do

begin

Write(i + 1: 2, '|');

writeln\_arr1(a[i]);

end;

end;

**///Заполнение матрицы**

procedure fillarr2(var a: mas2);

var

i, j: byte;

begin

randomize;

for i := low(a) to ogran - 1 do

for j := low(a[i]) to ogran - 1 do

a[i][j] := random(10) - 5;

end;

**///функция возвращает индекс максимального элемента в строке**

function maxim(const h: mas1): integer;

var

i, r: integer;

begin

r := 0;

for i := 1 to ogran do

if h[i] > h[r] then

r := i;

Result := r;

end;

procedure swap(var a, b: integer);

var

t: integer;

begin

t := a;

a := b;

b := t;

end;

**///Процедура обработки массива**

procedure obrabotka(var h: mas2);

var

i, j: integer;

begin

for i := 0 to ogran - 1 do

swap(h[i][i], h[i][maxim(h[i])]);

end;

var

h: mas2;

begin

writeln(utf8toansi('Введите размер массива'));

readln(ogran);

if (ogran > 7) or (ogran < 1) then

ogran := 7;

randomize;

fillarr2(h);

writeln(utf8toansi('Сгенерированный массив:'));

writeln\_arr2(h);

obrabotka(h);

writeln(utf8toansi('Массив после обработки:'));

writeln\_arr2(h);

readln();

end.

**Отладка:**

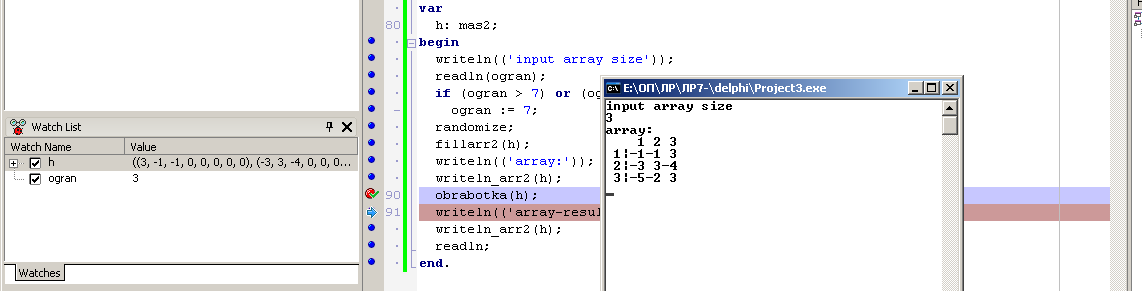


Рисунок -Скриншот отладки без входа в подпрограмму

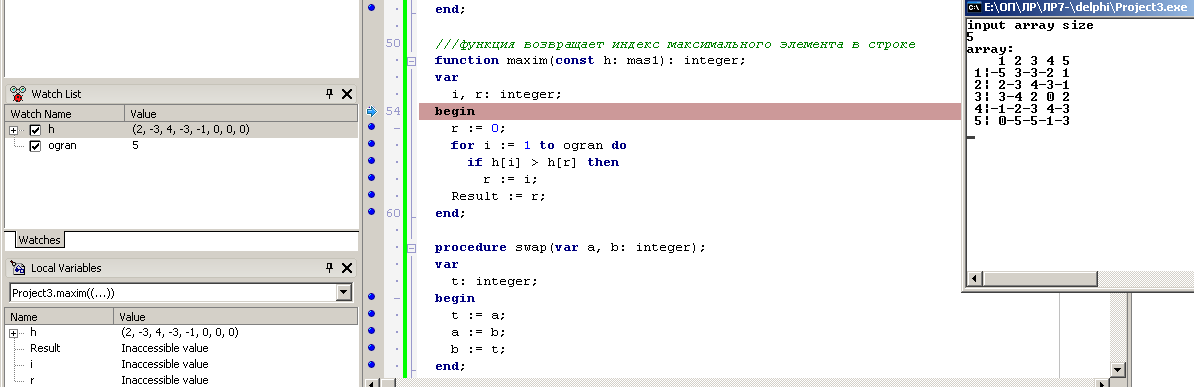


Рисунок -Скриншот отладки с заходом в подпрограмму

**Тестирование:**

Таблица -Результаты тестирования программы 1

|  |  |  |
| --- | --- | --- |
| Исходные данные | Ожидаемый результат | Полученный результат |
| 1 2 3  1| 0 3 4  2| 1 3-4  3| 1-4-5 | 1 2 3  1| 4 3 0  2| 1 3-4  3|-5-4 1 | 1 2 3  1| 4 3 0  2| 1 3-4  3|-5-4 1 |
| 1 2 3 4  1|-5 3 0-3  2| 2 0 3 3  3| 2 1 2 0  4|-5 3-4 4 | 1 2 3 4  1| 3-5 0-3  2| 2 3 0 3  3| 2 1 2 0  4|-5 3-4 4 | 1 2 3 4  1| 3-5 0-3  2| 2 3 0 3  3| 2 1 2 0  4|-5 3-4 4 |
| 1 2 3 4 5  1|-3 0 0 4 0  2| 2 2-5 0 2  3|-3 4-4 0-5  4|-4 1 2-4 2  5|-2 1-1 1 3 | 1 2 3 4 5  1| 4 0 0-3 0  2| 2 2-5 0 2  3|-3-4 4 0-5  4|-4 1-4 2 2  5|-2 1-1 1 3 | 1 2 3 4 5  1| 4 0 0-3 0  2| 2 2-5 0 2  3|-3-4 4 0-5  4|-4 1-4 2 2  5|-2 1-1 1 3 |
| 1 2 3 4 5 6 7  1| 0-3-1-3-2-4 1  2| 4 2 1-2 1-2 4  3|-4-4 2-4-4 4 2  4|-1-2-4 0-2 2 0  5| 0-2-4-5 3-2 1  6|-3 1-5-3 3 4 4  7|-1-2-1-1-1-5 2 | 1 2 3 4 5 6 7  1| 1-3-1-3-2-4 0  2| 2 4 1-2 1-2 4  3|-4-4 4-4-4 2 2  4|-1-2-4 2-2 0 0  5| 0-2-4-5 3-2 1  6|-3 1-5-3 3 4 4  7|-1-2-1-1-1-5 2 | 1 2 3 4 5 6 7  1| 1-3-1-3-2-4 0  2| 2 4 1-2 1-2 4  3|-4-4 4-4-4 2 2  4|-1-2-4 2-2 0 0  5| 0-2-4-5 3-2 1  6|-3 1-5-3 3 4 4  7|-1-2-1-1-1-5 2 |

**Пример работы программы:**

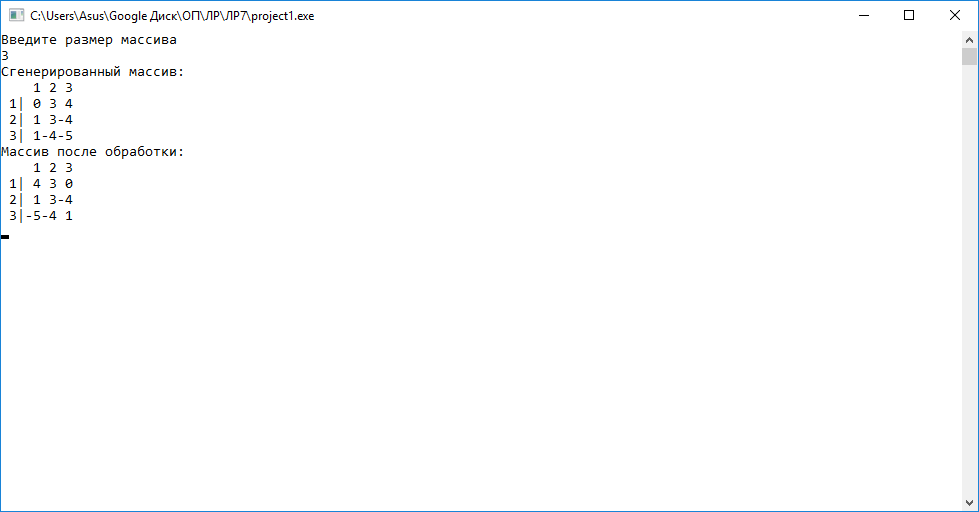


Рисунок 4-Пример работы программы

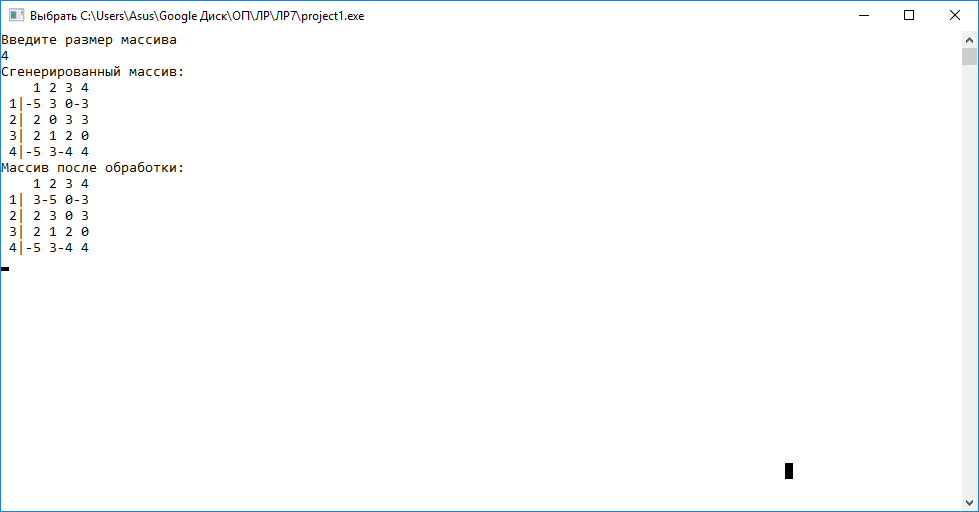


Рисунок 5-Пример работы программы

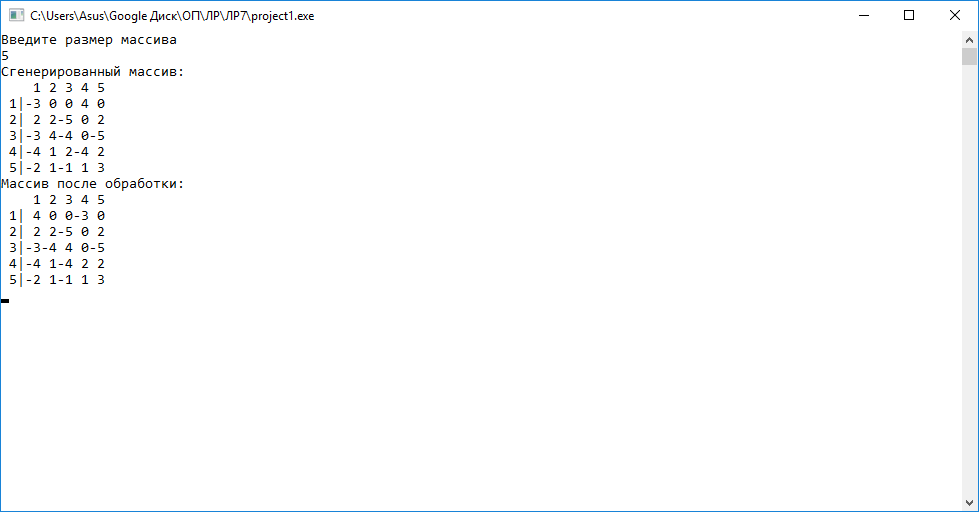


Рисунок 6-Пример работы программы

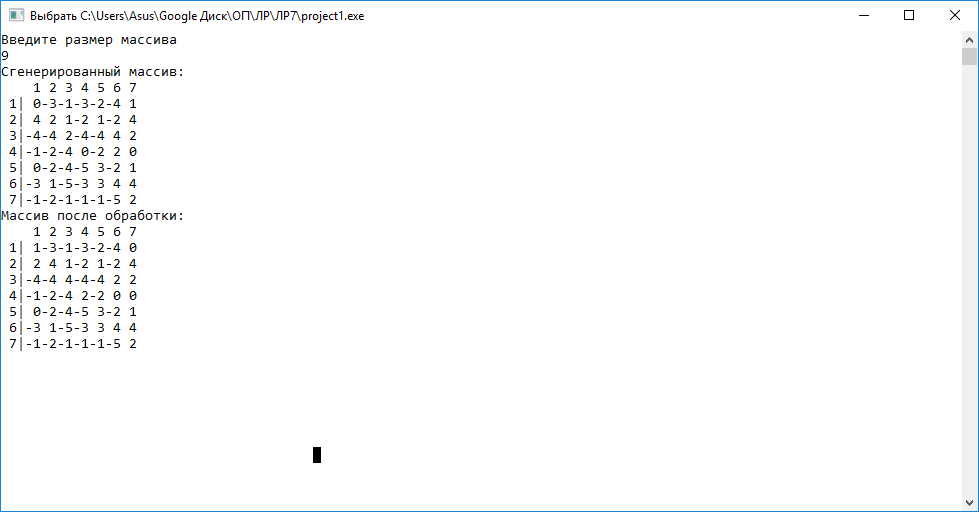


Рисунок 7-Пример работы программы

**Вывод:**

1. Разработан алгоритм для обработки матрицы в процедурном стиле и составлена его схема в среде LibreOffice Draw(см. рис. 1).
2. Создан код программы по схеме алгоритма в среде Lazarus
3. Проведена отладка программы по шагам с заходом в процедуры и без захода (см рис. 2-3)
4. Проведено тестирование (см. табл. 1).
5. Тестирование показало корректность работы программы (см. рис. 4-7)