МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

**«БЕЛГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ им. В. Г. ШУХОВА»**

**(БГТУ им. В.Г. Шухова)**

Кафедра программного обеспечения вычислительной техники и автоматизированных систем

**Лабораторная работа №11**

по дисциплине: Основы программирования тема: «Обработка типизированных файлов»

Выполнил: ст. группы ПВ-202 Буйвало Анастасия Андреевна

Проверил:

Притчин Иван Сергеевич Брусенцева Валентина Станиславовна

Белгород 2020 г.

# Лабораторная работа №11

**«Обработка типизированных файлов»**

**Цель работы**: получение навыков работы с файлами прямого доступа.

# Задания для подготовки к работе:

1. Изучите организацию работы с типизированными файлами и возможности Турбо Паскаля для организации прямого доступа к записям файла.
2. Разработайте алгоритм и составьте программу для решения задачи соответствующего варианта с использованием прямого доступа к

записям файлов. Во всех случаях, если возможно, не используйте вспомогательные файлы.

1. Разбейте задачу на подзадачи. Если задача небольшая, выделите самостоятельной̆ подзадачей обработку файла.
2. Опишите блок-схему алгоритма решения задачи в укрупненных блоках.
3. Для каждой подзадачи опишите используемые структуры данных, если необходимо, и спецификацию, и блок-схему алгоритма.
4. Опишите блок-схему алгоритма решения задачи с использованием блоков «предопределённый процесс».
5. Опишите процедуру для вывода содержимого файла на экран.
6. В программе обработки файла выведите содержимое файла на экран до и после обработки файла
7. Закодируйте алгоритм.
8. Файлы для обработки создайте с помощью отдельной программы.
9. Подберите наборы тестовых данных с обоснованием их выбора.

# Задания к работе:

* 1. Наберите программу, отладьте ее и протестируйте.
  2. Выполните анализ ошибок, выявленных при отладке программы.

# Задание варианта №7:

Из файла квадратных матриц порядка *N* (*N*−const) удалить матрицы, обратные последней

# Выполнение работы:

1. **Выделение подзадач**

Выделим следующие подзадачи:

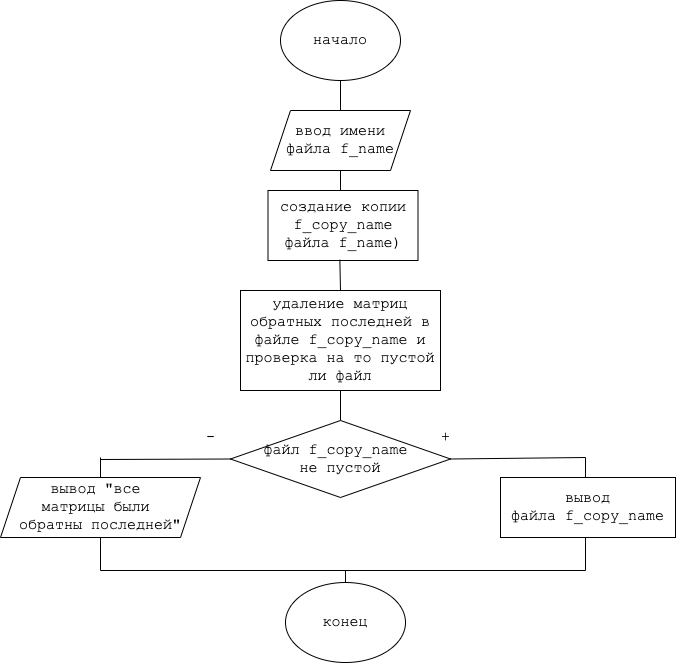
1. Создание файла с матрицами порядка N (N - const)
2. Создание копии исходного файла
3. Запись в массив b последней матрицы из файла
4. Проверка на то, является ли матрица обратной матрицей последней матрицы

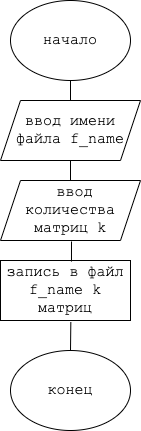
* Умножение матрицы на последнюю матрицу
* Проверка, является ли получившаяся матрица единичной

1. Удаление из копии файла матриц обратных последней матрице
2. Вывод на экран содержимого текстового документа

Опишем алгоритм в укрупненных блоках в терминах выделенных подзадач

# Блок-схемы алгоритмов в укрупненных блоках:





1. **Описание структур данных:**

const N = 2;

type t\_range = 1..N;

t\_matr = array [t\_range, t\_range] of real;

t\_file = file of t\_matr;

1. **Описание подпрограмм:**

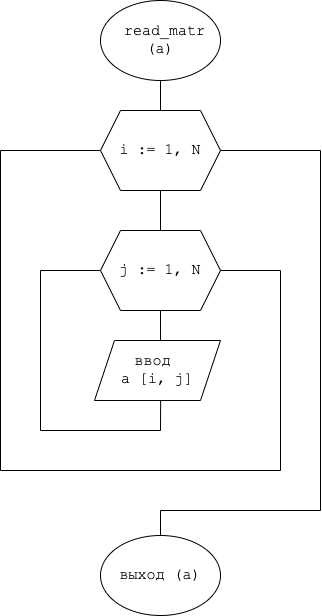
Создание файла

Процедура read\_matr

Спецификация.

1. Заголовок: procedure read\_matr (var a : t\_matr)
2. Назначение: ввод матрицы a.
3. Входные параметры: нет.
4. Выходные параметры: a.

Блок-схема:

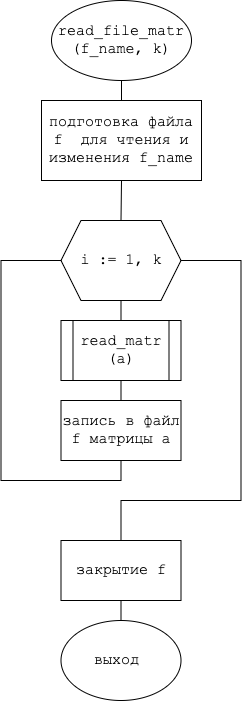


Процедура read\_file\_matr

Спецификация.

1. Заголовок: procedure read\_file\_matr (f\_name : string, k : byte);
2. Назначение: запись в файл с именем f\_name k матриц.
3. Входные параметры: f\_name, k.
4. Выходные параметры: нет.

Блок-схема:



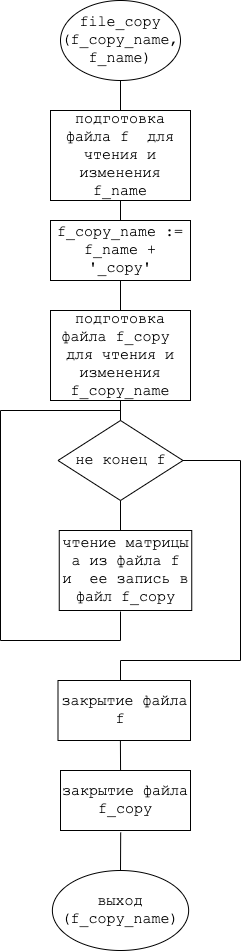
Основная программа

Процедура file\_copy

Спецификация.

1. Заголовок: procedure file\_copy (var f\_copy\_name : string, f\_name : string);
2. Назначение: Запись в файл с именем f\_copy\_name копии файла с именем f\_name
3. Входные параметры: f\_name.
4. Выходные параметры: f\_copy\_name.

Блок-схема:

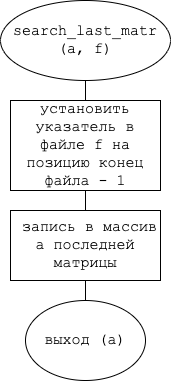


Процедура search\_last\_matr

Спецификация.

1. Заголовок: procedure search\_last\_matr (var a : t\_matr, f: t\_file);
2. Назначение: Запись в a последней матрицы файла f
3. Входные параметры: f.
4. Выходные параметры: a.

Блок-схема:

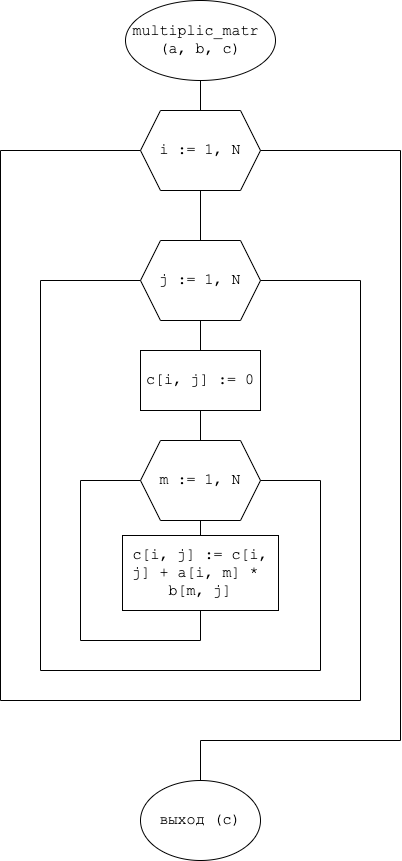


Процедура multiplic\_matr

Спецификация.

1. Заголовок: procedure multiplic\_matr(a, b : t\_matr, var c: t\_matr)
2. Назначение: Умножение матрицы a N-го порядка на матрицу b N-го порядка, с записью результата в матрицу c N-го порядка
3. Входные параметры: a, b.
4. Выходные параметры: c.

Блок-схема:

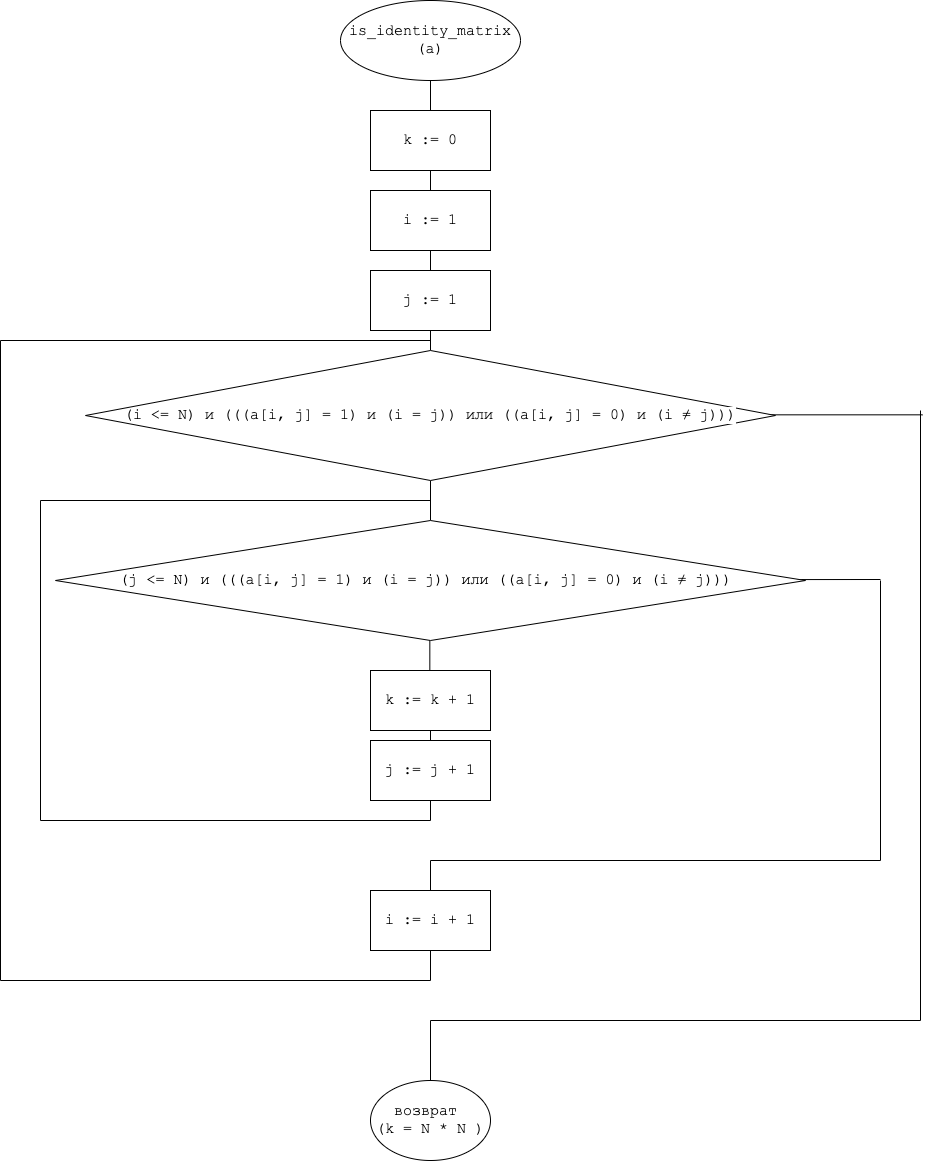


Функция is\_identity\_matrix

Спецификация.

1. Заголовок: function is\_identity\_matrix(a : t\_matr): boolean
2. Назначение: возвращает значение «истина» если матрица а является единичной, иначе «ложь»
3. Входные параметры: a.
4. Выходные параметры: нет.

Блок-схема:

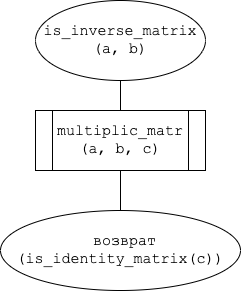


Функция is\_inverse\_matrix

Спецификация.

1. Заголовок: function is\_inverse\_matrix(a, b : t\_matr): boolean
2. Назначение: возвращает значение «истина», если матрица a является обратной матрицей матрицы b, иначе «ложь»
3. Входные параметры: a, b.
4. Выходные параметры: нет.

Блок-схема:



Процедура file\_matr\_del

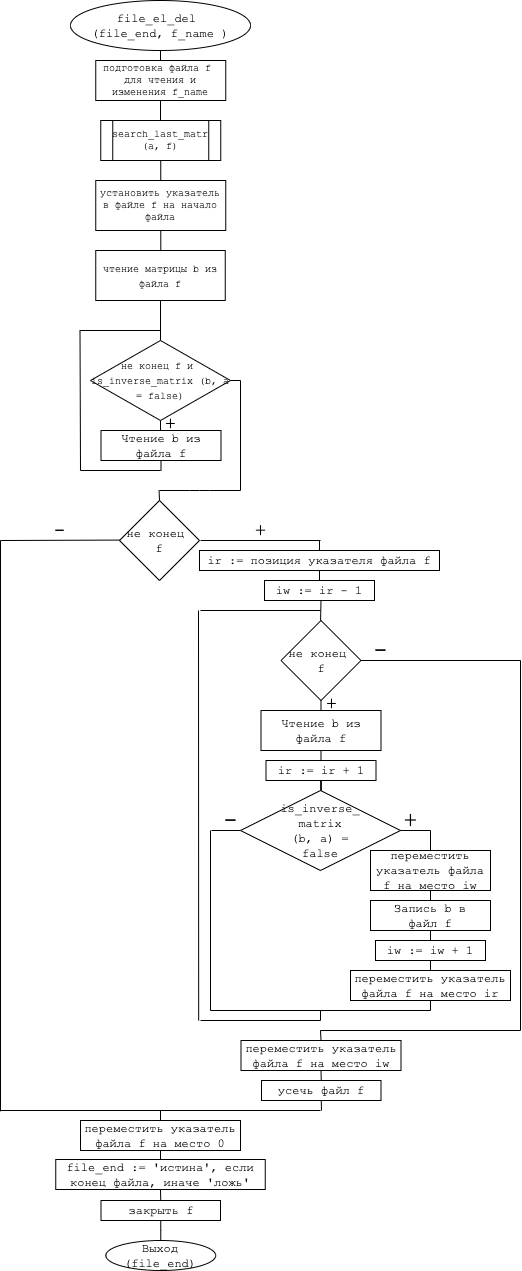
Спецификация.

1. Заголовок: procedure file\_matr\_del(var file\_end:

boolean; f\_name: string)

1. Назначение: Удаление матриц обратных последней из файла с именем f\_name, file\_end := «истина», если файл с именем f\_name пустой, иначе «ложь»
2. Входные параметры: f\_name.
3. Выходные параметры: file\_end.

Блок-схема:

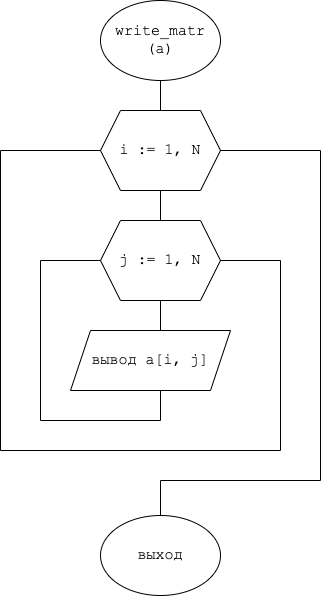


Процедура write\_matr

Спецификация.

1. Заголовок: procedure write\_matr (a : t\_matr)
2. Назначение: вывод матрицы a.
3. Входные параметры: a.
4. Выходные параметры: нет.

Блок-схема:

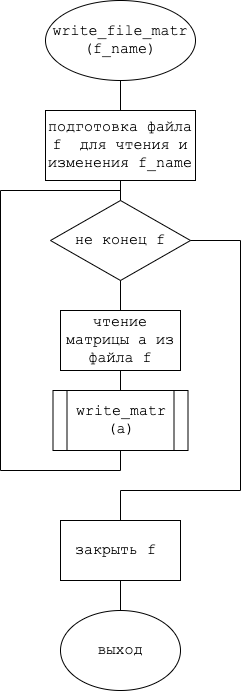


Процедура write\_file\_matr

Спецификация.

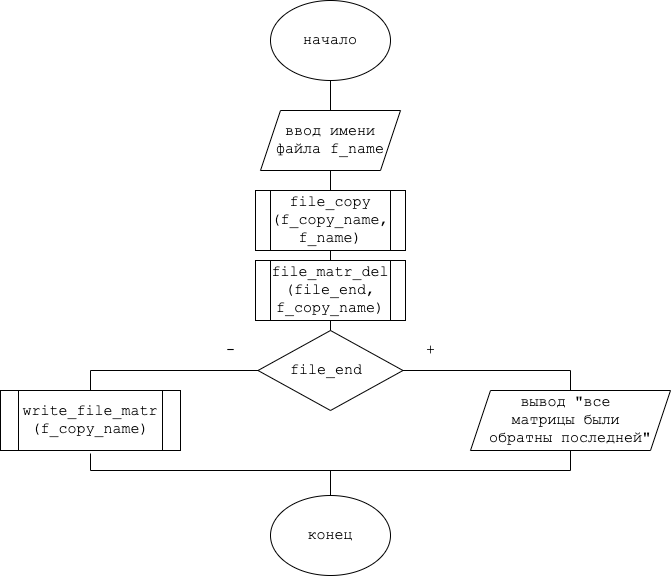
1. Заголовок: procedure write\_file\_matr (f\_name : string)
2. Назначение: вывод матриц из файла с именем f\_name.
3. Входные параметры: f\_name.
4. Выходные параметры: нет.

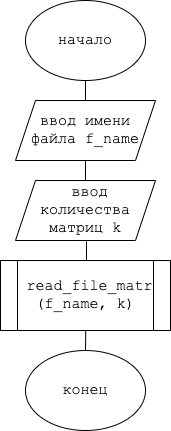
Блок-схема:



# Блок-схемы алгоритмов решения задачи c блоками

**«предопределенный процесс»**

****

****

1. **Тестовые данные:**

|  |  |  |
| --- | --- | --- |
| № | Исходные данные | Результаты |
| 1 | 1 1  2 2  2 2  3 3 | 1 1  2 2  2 2  3 3 |
| 2 | 0.5 0  0 0.5  2 0  0 2 | 2 0  0 2 |
| 3 | -1 0  0 -1  -1 0  0 -1 | «все матрицы были обратны последней» |
| 4 | -1 0  0 -1  2 0  0 2  -1 0  0 -1 | 2 0  0 2 |

1. **Текст программы:**

const N = 2;

type t\_range = 1..N;

t\_matr = array [t\_range, t\_range] of real;

t\_file = file of t\_matr;

{Ввод матрицы a}

procedure read\_matr (var a : t\_matr);

var i, j : t\_range;

begin

for i := 1 to N do

for j := 1 to N do

read (a[i, j]);

end;

{Запись в файл с именем f\_name k матриц}

procedure read\_file\_matr (f\_name : string; k : byte);

var f : t\_file;

a : t\_matr;

i : byte;

begin

assign(f, f\_name);

rewrite(f);

for i:= 1 to k do

begin

read\_matr (a);

write(f, a);

writeln('---------');

end;

close(f);

end;

var k : byte;

f\_name : string;

begin

write('введите имя файла: ');

read(f\_name);

write('Введите кол-во матриц которые вы хотите ввести: ');

read(k);

writeln('Введите ',k,' матриц:');

read\_file\_matr(f\_name, k);

end.

const N = 2;

type t\_range = 1..N;

t\_matr = array [t\_range, t\_range] of real;

t\_file = file of t\_matr;

{Запись в файл с именем f\_copy\_name копии файла с именем f\_name}

procedure file\_copy(var f\_copy\_name : string; f\_name : string);

var f, f\_copy : t\_file;

a : t\_matr;

begin

assign(f, f\_name);

reset(f);

f\_copy\_name := f\_name + '\_copy';

assign(f\_copy, f\_copy\_name);

rewrite(f\_copy);

while not EoF(f) do

begin

read(f, a);

write(f\_copy, a);

end;

close(f);

close(f\_copy);

writeln('Была создана копия файла ',f\_name,' с именем: ',f\_copy\_name);

end;

{Запись в a последней матрицы файла f}

procedure search\_last\_matr (var a : t\_matr ; f : t\_file);

begin

seek (f, FileSize(f) - 1);

read (f, a);

end;

{Умножение матрицы a N-го порядка на матрицу b N-го порядка,с записью результата в матрицу c N-го порядка}

procedure multiplic\_matr (a, b : t\_matr; var c : t\_matr);

var i, j, m : t\_range;

begin

for i := 1 to N do

for j := 1 to N do

begin

c[i,j] := 0;

for m := 1 to N do

c[i,j] := c[i,j] + a[i,m] \* b[m,j];

end;

end;

{возвращает значение "истина" если матрица а является единичной, иначе "ложь"}

function is\_identity\_matrix(a : t\_matr) : boolean;

var k : integer;

i, j : t\_range;

begin

k := 0;

i := 1;

j := 1;

while (i <= N) and (((a[i, j] = 1) and (i = j)) or ((a[i, j] = 0) and

(i <> j))) do

begin

while (j <= N) and (((a[i, j] = 1) and (i = j)) or ((a[i, j] = 0) and

(i <>j)))do

begin

k := k + 1;

j := j + 1;

end;

i := i + 1;

end;

is\_identity\_matrix := k = N \* N;

end;

{возвращает значение «истина», если матрица a является обратной матрицей матрицы b, иначе «ложь»}

function is\_inverse\_matrix(a, b : t\_matr) : boolean;

var c : t\_matr;

begin

multiplic\_matr(a, b, c);

is\_inverse\_matrix := is\_identity\_matrix(c);

end;

{Удаление матриц обратных последней из файла с именем f\_name}

procedure file\_matr\_del (var file\_end : boolean; f\_name : string);

var f : t\_file;

ir, iw : integer;

a, b : t\_matr;

begin

assign(f, f\_name);

reset(f);

search\_last\_matr(a, f);

seek (f, 0);

read (f, b);

while (not Eof(f)) and (is\_inverse\_matrix (b, a) = false) do

read (f, b);

if not EoF (f) then

begin

ir := FilePos(f);

iw := ir - 1;

while not Eof (f) do

begin

read (f, b);

ir := ir + 1;

if is\_inverse\_matrix (b, a) = false then

begin

seek (f, iw);

write (f, b);

iw := iw + 1;

seek (f, ir);

end;

end;

seek (f, iw);

truncate (f);

end;

seek (f, 0);

file\_end := EoF (f);

close(f);

end;

{Вывод матрицы a}

procedure write\_matr(a : t\_matr);

var i, j : t\_range;

begin

for i := 1 to N do

begin

for j := 1 to N do

write(a[i, j], ' ');

writeln;

end;

end;

{Вывод матриц из файла с именем f\_name}

procedure write\_file\_matr(f\_name : string);

var f : t\_file;

a : t\_matr;

i : byte;

begin

assign(f, f\_name);

reset(f);

while not EoF(f) do

begin

read(f, a);

write\_matr(a);

writeln('---------');

end;

close(f);

end;

var f\_name, f\_copy\_name : string;

file\_end : boolean;

begin

write('введите имя файла: ');

read(f\_name);

file\_copy (f\_copy\_name, f\_name);

file\_matr\_del (file\_end, f\_copy\_name);

if file\_end = false then

begin

writeln('матрицы файла не обратные последней:');

write\_file\_matr(f\_copy\_name);

end

else

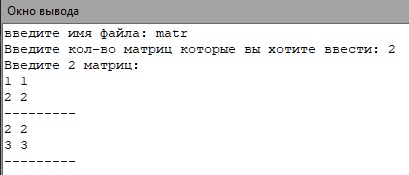
write('Все матрицы были обратны последней');

end.

# Анализ допущенных ошибок:

* При наборе программы ошибок допущено не было;

# Результаты работы программы:

****

