Министерство образования Республики Беларусь

Учреждение образования

БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Факультет компьютерных систем и сетей

Кафедра программного обеспечения информационных технологий

Дисциплина: Основы алгоритмизации и программирования (ОАиП)

ОТЧЕТ

по лабораторной работе №1

Тема работы: Действия над матрицами при

помощи процедур

Выполнил

студент: гр. 151004 Данилов Ф.А.

Проверил: Фадеева Е.П.

Минск 2022

Содержание

[1 Постановка задачи 4](#_Toc97795781)

[2 Описание алгоритмов решения задачи 5](#_Toc97795782)

[3 Структура данных 7](#_Toc97795783)

[3.1 Структура данных основной программы 7](#_Toc97795784)

[3.2 Структура данных алгоритма MultiplyMatrix 7](#_Toc97795785)

[3.3 Структура данных алгоритма MultiplyNum 8](#_Toc97795786)

[3.4 Структура данных алгоритма SumMatrix 8](#_Toc97795787)

[3.5 Структура данных алгоритма WriteMatrix 9](#_Toc97795788)

[4 Схема алгоритма решения задачи по ГОСТ 19.701-90 10](#_Toc97795789)

[4.1 Схема алгоритма подпрограммы MultiplyMatrix 11](#_Toc97795790)

[4.2 Схема алгоритма подпрограммы MultiplyNum 12](#_Toc97795791)

[4.3 Схема алгоритма подпрограммы SumMatrix 13](#_Toc97795792)

[4.4 Схема алгоритма подпрограммы WriteMatrix 14](#_Toc97795793)

[5 Результаты расчетов программы 15](#_Toc97795794)

[Приложение А 16](#_Toc97795795)

[Приложение Б 20](#_Toc97795796)

# Постановка задачи

## Постановка задачи 1

Разработать программу для расчёта матричных выражений. Программа должна иметь процедуры без использования глобальных параметров. Результат расчётов сравнить с MathCad 15.

Выполнить действия над матрицами:

где

## Постановка задачи 2 (реализованная)

Разработать программу для расчёта матричных выражений. Программа должна иметь процедуры без использования глобальных параметров. Вводится две матрицы [3, 3]. Результат расчётов сравнить с MathCad 15.

Выполнить действия над матрицами:

# Описание алгоритмов решения задачи

Таблица 1 – Описание алгоритмов решения задачи

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| №  п.п. | Наименование алгоритма | Назначение  алгоритма | Формальные параметры | Предпологаемый тип реализации |
|  | Основной алгоритм | Расчёт матричного выражения. Вызывает следующие подпрограммы:  multiplyNum(M, Num, Res),  multiplyMatrix(M1, M2, Res)  sumMatrix(M1, M2, Sign, Res),  writeMatrix(M) |  |  |
|  | sumMatrix  (M1, M2, Sign, Res) | Расчёт суммы(вычитания) матриц | M1 – получает от фактического параметра его копию;  M2 – получает от фактического параметра его копию;  Sign – знак суммы или разности (true = +, false = -)  Res – получает от фактического параметра адрес, возвращаемый параметр; | Процедура |

Продолжение таблицы 1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | multiplyNum(M, Num, Res) | Расчет произведения матрицы на число. | M – получает от фактического параметра его копию;  Num – получает от фактического параметра его копию;  Res – получает от фактического параметра адрес, возвращаемый параметр; | Процедура |
|  | multiplyMatrix(M1, M2, Res) | Расчет произведения матрицы на матрицу. | M1 – получает от фактического параметра его копию;  M2 – получает от фактического параметра его копию;  Res – получает от фактического параметра адрес, возвращаемый параметр; | Процедура |
|  | writeMatrix(M) | Вывод матрицы в консоль | M – получат от фактического параметра его копию | Процедура |

# Структура данных

## Структура данных основной программы

Таблица 2 – Структура данных основной программы

|  |  |  |
| --- | --- | --- |
| Элементы данных | Рекомендуемый тип | Назначение |
| Temp1, Temp2, Temp3 | array[1..3, 1..3] of Integer | Хранить значения операций над матрицами |
| A | array[1..3, 1..3] of Integer | Хранить значение матрицы А. |
| В | array[1..3, 1..3] of Integer | Хранить значение матрицы В. |

## Структура данных алгоритма MultiplyMatrix

Таблица 3 – Структура данных алгоритма MultiplyMatrix(M1, M2, Res)

|  |  |  |  |
| --- | --- | --- | --- |
| Элементы данных | Рекомендуемый тип | Назначение | Тип параметра |
| i, j, k | Integer | Индексы | Локальные |
| M1 | array[1..3, 1..3] of Integer | Хранить значение матрицы левого операнда. | Формальный |
| M2 | array[1..3, 1..3] of Integer | Хранить значение матрицы правого операнда. | Формальный |
| Res | array[1..3, 1..3] of Integer | Хранить результат операции | Формальный |

## Структура данных алгоритма MultiplyNum

Таблица 4 – Структура данных алгоритма MultiplyNum(M, Num, Res)

|  |  |  |  |
| --- | --- | --- | --- |
| Элементы данных | Рекомендуемый тип | Назначение | Тип параметра |
| i, j, k | Integer | Индексы | Локальные |
| M | array[1..3, 1..3] of Integer | Хранить значение матрицы первого операнда. | Формальный |
| Num | array[1..3, 1..3] of Integer | Хранить значение числа второго операнда. | Формальный |
| Res | array[1..3, 1..3] of Integer | Хранить результат операции | Формальный |

## Структура данных алгоритма SumMatrix

Таблица 5 – Структура данных алгоритма SumMatrix(M1, M2, Sign, Res)

|  |  |  |  |
| --- | --- | --- | --- |
| Элементы данных | Рекомендуемый тип | Назначение | Тип параметра |
| i, j | Integer | Индексы | Локальные |
| M1 | array[1..3, 1..3] of Integer | Хранить значение матрицы левого операнда. | Формальный |
| M2 | array[1..3, 1..3] of Integer | Хранить значение матрицы правого операнда. | Формальный |
| Sign | Boolean | Если Sign = true, знак оператора ‘+’, иначе – ‘-‘ | Формальный |
| Res | array[1..3, 1..3] of Integer | Хранить результат операции | Формальный |

## Структура данных алгоритма WriteMatrix

Таблица 6 – Структура данных алгоритма WriteMatrix(M)

|  |  |  |  |
| --- | --- | --- | --- |
| Элементы данных | Рекомендуемый тип | Назначение | Тип параметра |
| i, j | Integer | Индексы | Локальные |
| M | array[1..3, 1..3] of Integer | Хранить выводимый массив | Формальный |

# Схема алгоритма решения задачи по ГОСТ 19.701-90

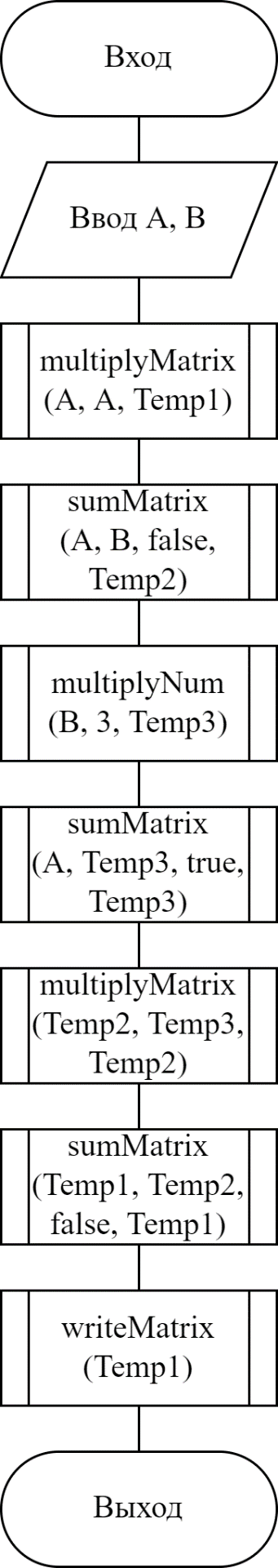


Рисунок 1 – Схема алгоритма решения задачи

## Схема алгоритма подпрограммы MultiplyMatrix

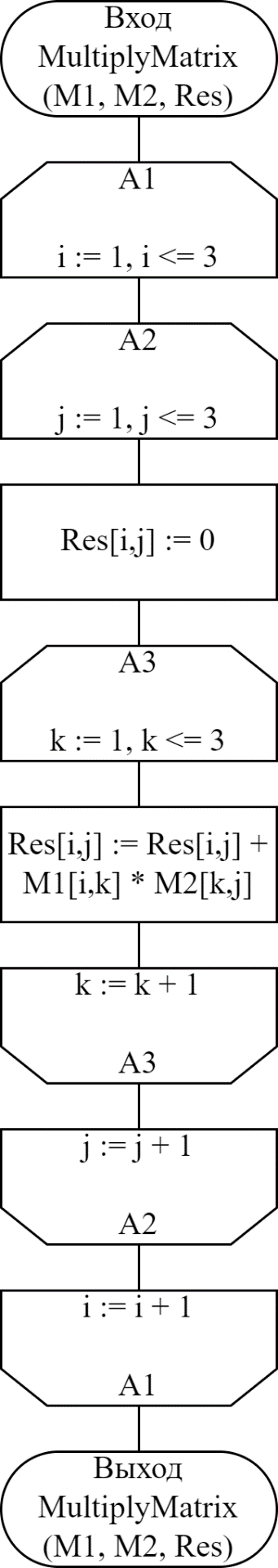


Рисунок 2 – Схема алгоритма подпрограммы MultiplyMatrix(M1, M2, Res)

## Схема алгоритма подпрограммы MultiplyNum

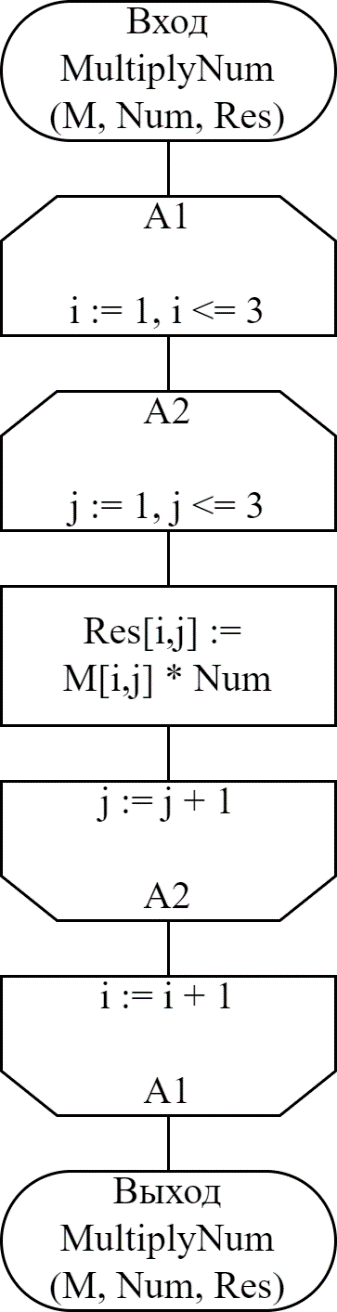


Рисунок 3 – Схема алгоритма подпрограммы MultiplyNum(M, Num, Res)

## Схема алгоритма подпрограммы SumMatrix

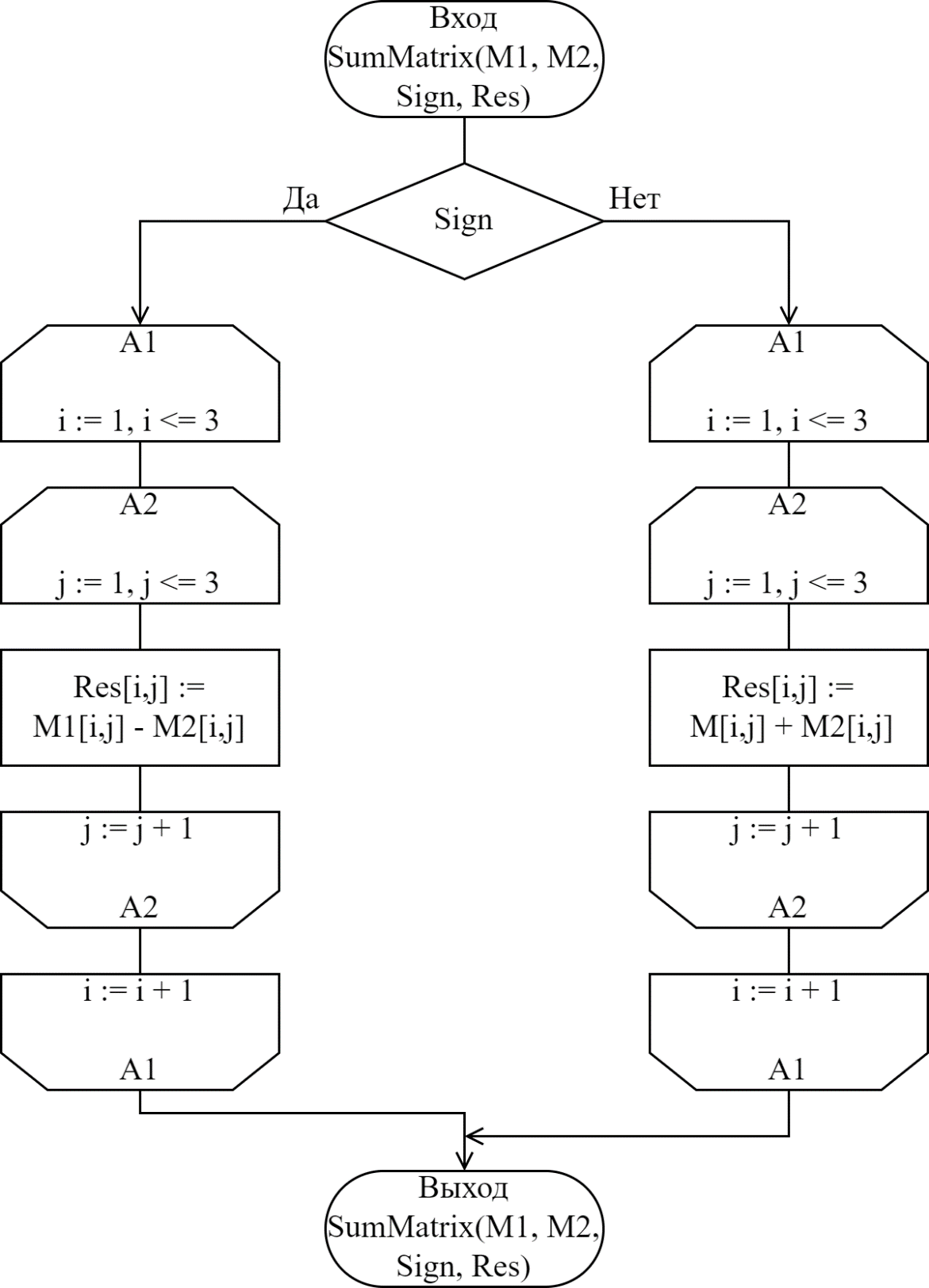


Рисунок 4 – Схема алгоритма подпрограммы SumMatrix(M1, M2, Sign, Res)

## Схема алгоритма подпрограммы WriteMatrix

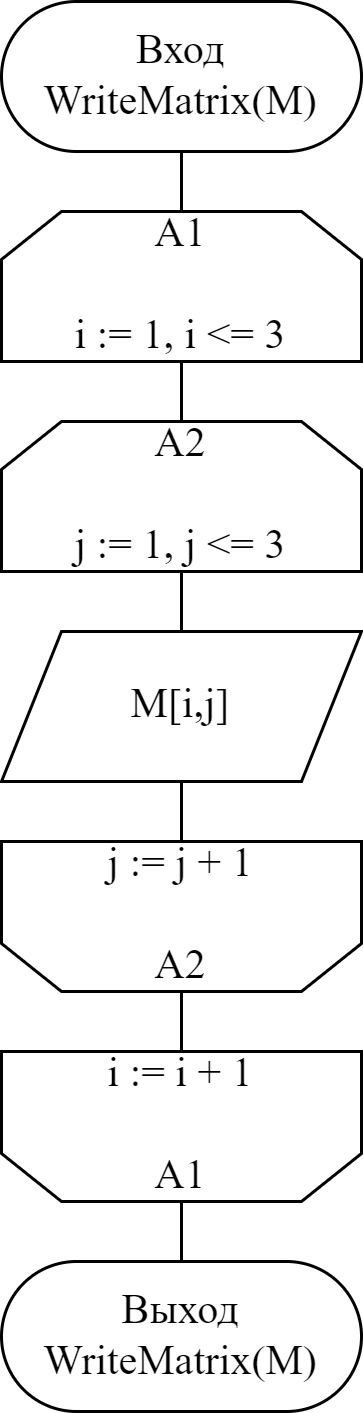


Рисунок 5 – Схема алгоритма подпрограммы WriteMatrix(M)

# Результаты расчетов программы

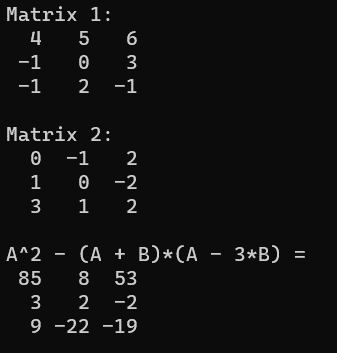


Рисунок 6 – Результаты расчётов программы

Приложение А

(обязательное)

Исходный код программы

program Lab1\_20V1;

{

Calculate A^2 - (A + B)\*(A - 3\*B), where

(( 4, 5, 6), (( 0,-1, 2),

A = (-1, 0, 3), B = ( 1, 0,-2),

(-1, 2,-1)) ( 3, 1, 2))

using procedures.

Output: result of expression

}

{$APPTYPE CONSOLE}

{$R \*.res}

uses

System.SysUtils;

// Matrix of size (3,3)

type

TMatrix = Array[1..3, 1..3] of Integer;

// Multiplication of matrix and Integer

procedure multiplyMatrix(M: TMatrix; Num: Integer;

var Res: TMatrix); overload;

var i, j: Integer;

begin

for i:= 1 to 3 do

for j:= 1 to 3 do

Res[i,j] := M[i,j] \* Num;

end;

// Multiplication of two matrices

procedure multiplyMatrix(M1, M2: TMatrix;

var Res: TMatrix); overload;

var

i,j,k: Integer;

begin

for i := 1 to 3 do

for j := 1 to 3 do

begin

Res[i,j] := 0;

for k := 1 to 3 do

Res[i,j] := Res[i,j] + M1[i,k]\*M2[k,j];

end;

end;

// Sum\* of two matrices

// Take sign value.

// If Sign = true, result is sum

// If Sign = false, result is difference

procedure sumMatrix(M1, M2: TMatrix; Sign: Boolean;

var Res: TMatrix); overload;

var

i,j: Integer;

begin

if(Sign) then

begin

for i := 1 to 3 do

for j := 1 to 3 do

Res[i,j] := M1[i,j] + M2[i,j];

end

else

begin

for i := 1 to 3 do

for j := 1 to 3 do

Res[i,j] := M1[i,j] - M2[i,j];

end;

end;

// Output matrix on screen

procedure writeMatrix(M: TMatrix);

var

i,j: Integer;

begin

for i := 1 to 3 do

begin

for j := 1 to 3 do

write(M[i,j]:3, ' ');

writeln;

end;

end;

procedure inputMatrix(var M: TMatrix);

var

InpStr: String;

i,j, Error: Integer;

begin

for i := 1 to 3 do

begin

for j := 1 to 3 do

begin

repeat

writeln('Input [', i, ', ', j, ']:');

readln(InpStr);

val(InpStr, M[i,j], Error);

if (Error <> 0) then

writeln('Error! Enter a valid Integer.');

until Error = 0;

end;

end;

writeln;

end;

// Temp1, Temp2, Temp3 - temporary matrices

// to store results of operations

var

A, B, Temp1, Temp2, Temp3: TMatrix;

begin

inputMatrix(A);

inputMatrix(B);

// Output A

writeln('A:');

writeMatrix(A);

writeln;

// Output B

writeln('B:');

writeMatrix(B);

// Output expression

writeln;

writeln('A^2 - (A + B)\*(A - 3\*B) =');

// Perform calculations

// (A^2)

multiplyMatrix(A, A, Temp1);

// (A + B)

sumMatrix(A, B, true, Temp2);

// (3\*B)

multiplyMatrix(B, 3, Temp3);

// (A - 3\*B)

sumMatrix(A, Temp3, false, Temp3);

// (A + B)\*(A - 3\*B)

multiplyMatrix(Temp2, Temp3, Temp2);

// A^2 - (A + B)\*(A - 3\*B)

sumMatrix(Temp1, Temp2, false, Temp1);

// Output result

writeMatrix(Temp1);

// Stop console from closing

readln;

end.

Приложение Б

(обязательное)

Тестовые наборы

Тестовые наборы для проверки ввода данных

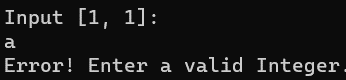
Тест 1

Тестовая ситуация: Проверка работоспособности.

Исходные данные: a

Ожидаемый результат: Error! Enter a valid Integer.

Полученный результат программы:



Тест 2

Тестовая ситуация: Проверка работоспособности.

Исходные данные: 1

Ожидаемый результат:

Полученный результат программы:



Тестовые наборы для проверки вычислений

Тест 1

Тестовая ситуация: Проверка работоспособности.

Исходные данные:

(( 4, 5, 6), (( 0,-1, 2),

А = (-1, 0, 3), B = ( 1, 0, -2),

(-1, 2,-1)) ( 3, 1, 2))

Ожидаемый результат: ((85, 8, 53),

(3, 2, -2),

(9, -22, -19)

Полученный результат программы:



Полученный результат в программе MathCad 15:

