**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ**

**Федеральное государственное бюджетное образовательное учреждение**

**высшего образования**

**РОССИЙСКИЙ ЭКОНОМИЧЕСКИЙ УНИВЕРСИТЕТ ИМЕНИ Г.В. ПЛЕХАНОВА**

**филиал в г. Пятигорске Ставропольского края**

Кафедра информационных технологий и правового регулирования управления

**ЛАБОРАТОРНАЯ РАБОТА №1**

**по дисциплине «Информатика»**

**Тема: «Изучение основ программирования разветвляющихся и циклических процессов»**

Вариант №4

**Выполнил:**

Студент I курса

группы 1а-ИСиТ(о)

Дубовой В. А.

**Проверила:**

к.э.н., доцент

Иноземцева С. А.

Пятигорск, 2022

СОДЕРЖАНИЕ

[**ЗАДАНИЕ №1** 3](#_Toc122335114)

[1.1.ПОСТАНОВКА И СПЕЦИФИКАЦИЯ ЗАДАЧИ №1 3](#_Toc122335115)

[1.2.ПРОЕКТИРОВАНИЕ И АЛГОРИТМИЗАЦИЯ 6](#_Toc122335116)

[1.3. КОДИРОВАНИЕ АЛГОРИТМА 7](#_Toc122335117)

[1.4. ОТЛАДКА И ТЕСТИРОВАНИЕ ПРОГРАММЫ 8](#_Toc122335118)

[1.5. РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ 10](#_Toc122335119)

[**ЗАДАНИЕ №2** 13](#_Toc122335120)

[2.1. ПОСТАНОВКА И СПЕЦИФИКАЦИЯ ЗАДАЧИ №2 13](#_Toc122335121)

[2.2. ПРОЕКТИРОВАНИЕ И АЛГОРИТМИЗАЦИЯ 15](#_Toc122335122)

[2.3. КОДИРОВАНИЕ АЛГОРИТМА 16](#_Toc122335123)

[2.4. ОТЛАДКА И ТЕСТИРОВАНИЕ ПРОГРАММЫ 17](#_Toc122335124)

[2.5. РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ 18](#_Toc122335125)

[СПИСОК ИСПОЛЬ ЗОВАННЫХ ИСТОЧНИКОВ 20](#_Toc122335126)

[ПРИЛОЖЕНИЕ 1. ЛИСТИНГ ПРОГРАММЫ К ЗАДАНИЮ №1 21](#_Toc122335127)

[ПРИЛОЖЕНИЕ 2. ЛИСТИНГ ПРОГРАММЫ К ЗАДАНИЮ №2 23](#_Toc122335128)

# **ЗАДАНИЕ №1**

# 1.1.ПОСТАНОВКА И СПЕЦИФИКАЦИЯ ЗАДАЧИ №1

* Название: принадлежность точки с координатами (х, у), введенными с клавиатуры пользователем, заштрихованной области координатной плоскости.
* Описание: для решения поставленной задачи будут использоваться стандартные арифметические операторы, методы и функции объектов проекта, условный оператор if, логические операторы. При разработке программы использовался разветвленный алгоритм.
* Управление режимами работы программы: запуск exe файла проекта, расположенного по адресу «Win32/Debug» в корневой папке проекта. После запуска программы необходимо ввести параметры точки (x, y), по желанию настроить параметр функции (по умолчанию параметр равен 5), нажать кнопку «Рассчитать» для того, чтобы понять принадлежит ли точка заштрихованной области функции. Завершение работы осуществляется с помощью крестика в правом верхнем углу приложения.
* Входные данные: координаты точки по оси X и оси Y и параметр функции, отвечающий за смещение графика вверх или вниз
* Выходные данные: Значение “Принадлежит”, если точка лежит в заштрихованной области функции, и “Не принадлежит” для противоположного результата работы программы.
* Ошибки времени выполнения: не обнаружены.
* Пример для отладки и тестирования: для отладки и тестирования программы предоставлена таблица различных значений. Составленная с помощью анализа прямых на графике.

Таблица №1 – тестовые примеры для отладки программы

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ­№ | X | Y | Parametr | Output |
| 1. | 0 | 0 | 5 | Принадлежит |
| 2. | 0 | 5 | 5 | Принадлежит |
| 3. | 0 | -5 | 5 | Принадлежит |
| 4. | 5 | 0 | 5 | Принадлежит |
| 5. | -5 | 0 | 5 | Принадлежит |
| 6. | 2,5 | 2,5 | 5 | Принадлежит |
| 7. | -2,5 | -2,5 | 5 | Принадлежит |
| 8. | 2 | 3 | 5 | Принадлежит |
| 9. | -2 | -3 | 5 | Принадлежит |
| 10. | -2,6 | -2,5 | 5 | Не принадлежит |
| 11. | 2,5 | 2,6 | 5 | Не принадлежит |
| 12. | -2 | 3 | 5 | Не принадлежит |
| 13. | 2 | -3 | 5 | Не принадлежит |
| 14. | 0 | 5 | 6 | Принадлежит |
| 15. | 0 | -5 | 6 | Принадлежит |
| 16. | 5 | 0 | 6 | Принадлежит |
| 17. | -5 | 0 | 6 | Принадлежит |
| 18. | 2,5 | 2,5 | 6 | Принадлежит |
| 19. | -2,5 | -2,5 | 6 | Принадлежит |
| 20. | 2 | 3 | 6 | Принадлежит |
| 21. | -2 | -3 | 6 | Принадлежит |
| 22. | -2,6 | -2,5 | 6 | Принадлежит |
| 23. | 2,5 | 2,6 | 6 | Принадлежит |

# 1.2.ПРОЕКТИРОВАНИЕ И АЛГОРИТМИЗАЦИЯ

Алгоритм решения задачи представлен в виде блок-схем.

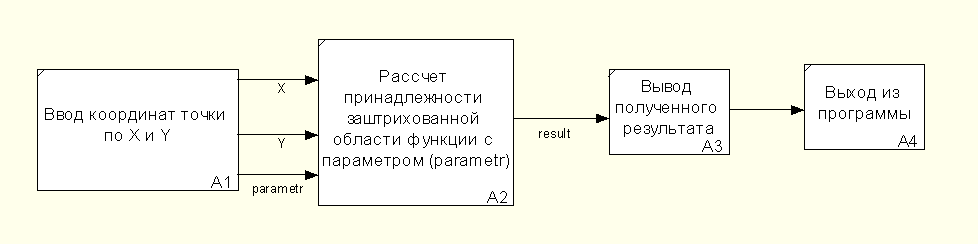


Рисунок 1 - Блочная структура программы

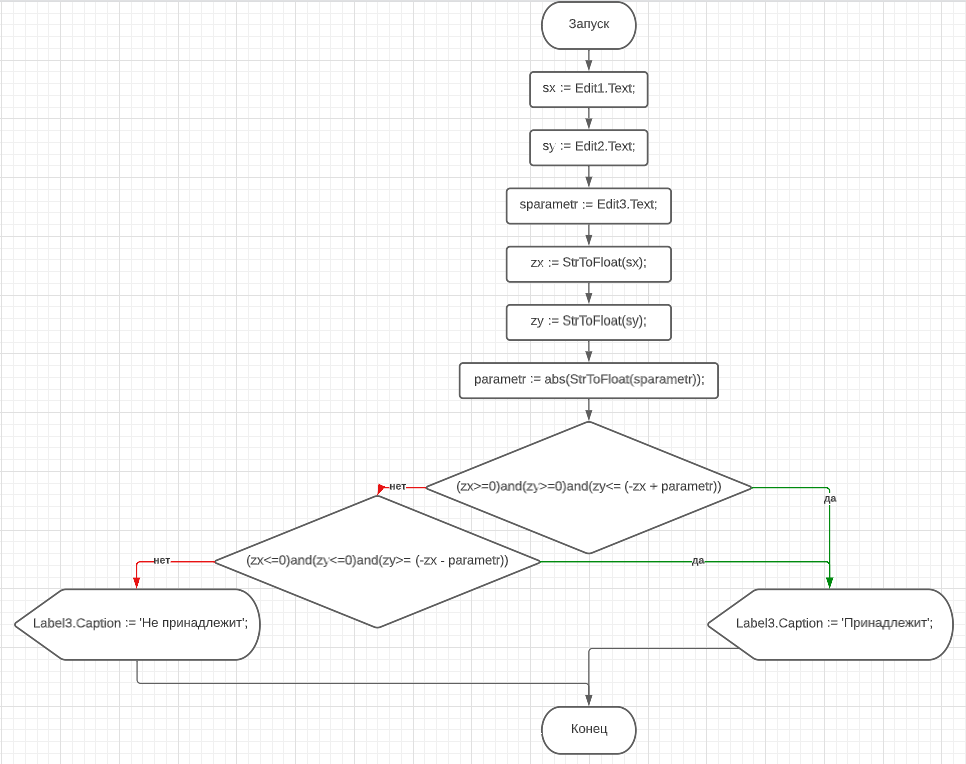


Рисунок 2 - Блок – схема программы, для определения принадлежности точки заштрихованной области функции по заданному параметру

# 1.3. КОДИРОВАНИЕ АЛГОРИТМА

Идентификаторы переменных:

* + zx, zy, parametr – Real – используется для хранения вводимых данных при вычислении принадлежности области.
  + sx, sy – String – используется для ввода входных данных в текстовом поле.

Идентификаторы констант: отсутствуют.

Ссылка на программу: <https://github.com/DeathOutsider/REU_Labs/tree/main/Lab2/Zadanie1>

# 1.4. ОТЛАДКА И ТЕСТИРОВАНИЕ ПРОГРАММЫ

Отладка и тестирование программы будет представлено в виде скриншотов некоторых вариантов из таблицы, приведенной в пункте 1.1

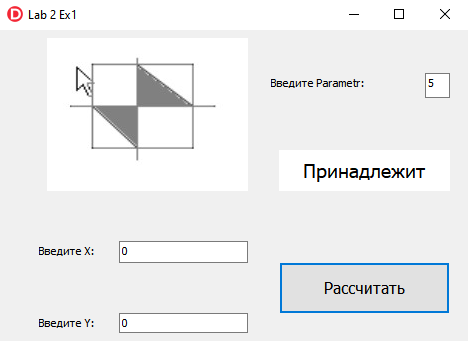
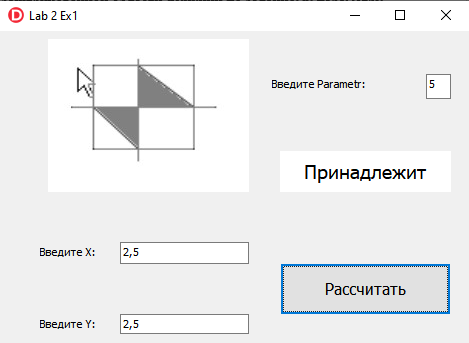


Рисунок 3 – вариант №1



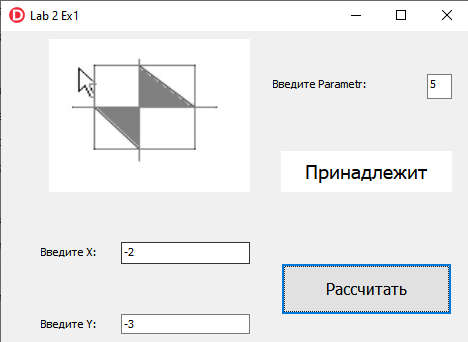
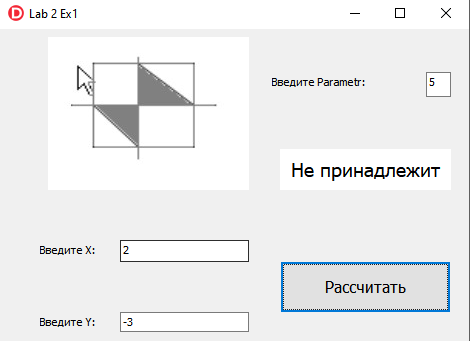
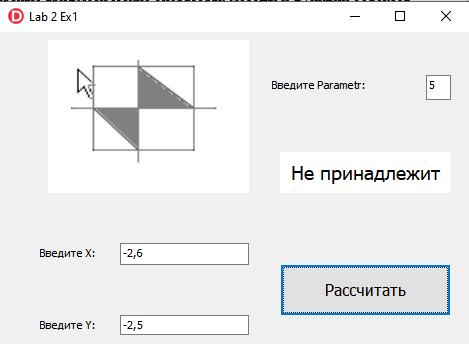
Рисунок 4 – вариант №6

Рисунок 5 – вариант 9

Рисунок 6 – вариант №13

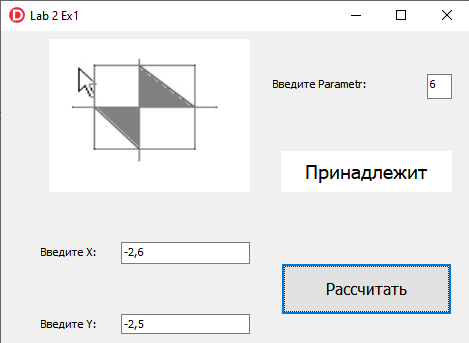
Рисунок 7 – вариант №10

Рисунок 8 – вариант №11

Таким образом, протестировав все примеры из таблицы и убедившись в корректной работе программы, решаем, что программа готова для практического использования.

# 1.5. РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ

Запустить файл Project1.exe, который находится в Win32/Debug/ корневой папки проекта

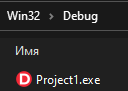


Рисунок 9 – Имя и Путь к файлу

После запуска программы открывается главное окно программы

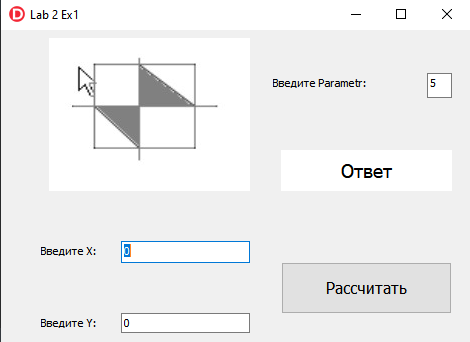


Рисунок 10 – Стартовый интерфейс программы

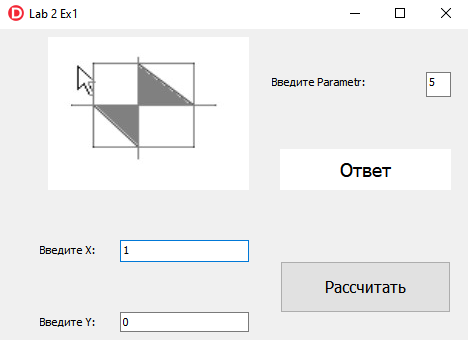
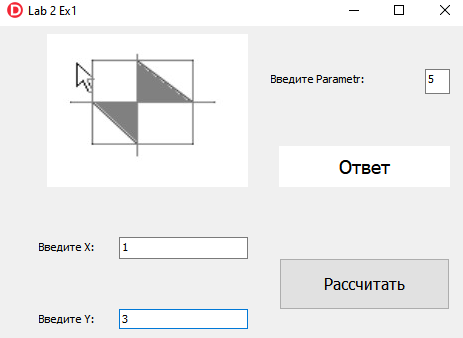
Далее необходимо ввести координаты точки (X и Y)

Рисунок 11, 12 – Ввод входных данных

При необходимости можно изменить дополнительный параметр функции

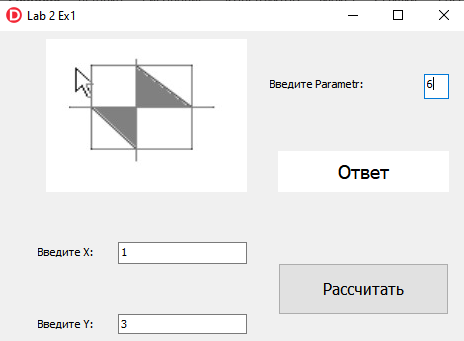


Рисунок 13 – Изменение дополнительного параметра

Далее следует нажать кнопку “Рассчитать”

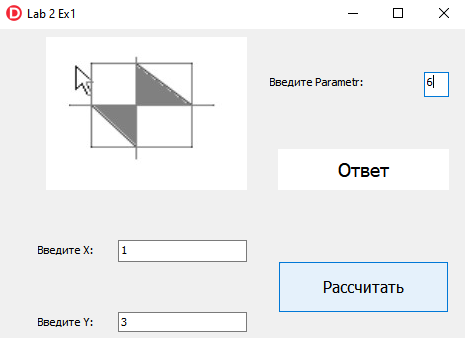


Рисунок 14 – Кнопка “Рассчитать”

И наслаждаться полученным результатом

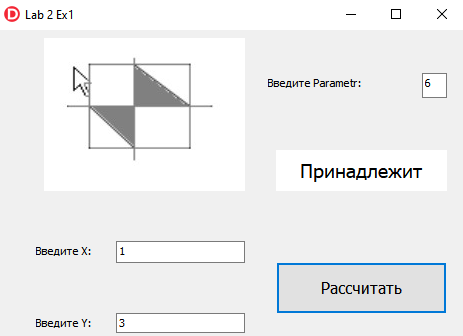


Рисунок 15 – Результат работы программы

Чтобы завершить работу с приложением следует закрыть его с помощью крестика в правом верхнем углу приложения

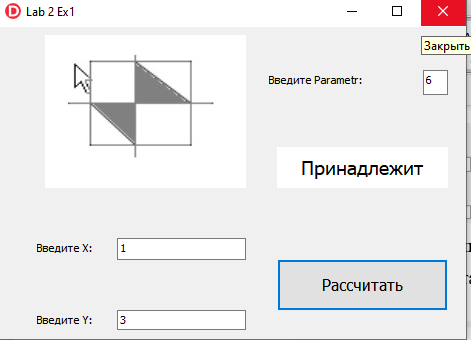


Рисунок 16 – Крестик в правом верхнем углу

# **ЗАДАНИЕ №2**

# 2.1. ПОСТАНОВКА И СПЕЦИФИКАЦИЯ ЗАДАЧИ №2

* Название: реализация циклического процесса с использованием подпрограммы.
* Описание: для решения поставленной задачи будут использоваться стандартные арифметические операторы, методы и функции объектов проекта, циклический оператор while do begin … end, логические операторы. При разработке программы использовался циклический алгоритм.
* Управление режимами работы программы: запуск exe файла проекта, расположенного по адресу «Win32/Debug» в корневой папке проекта. После запуска программы необходимо нажать кнопку «Рассчитать». После чего программа выполнит умножение заданного числа (123456789) на 9, 18, 27, …, 81 и выведет полученные значения на экран. Завершение работы осуществляется с помощью крестика в правом верхнем углу приложения.
* Входные данные: определяются программой
* Выходные данные: Результат умножения числа 123456789 на 9, 18,27, …, 81.
* Ошибки времени выполнения: не обнаружены.
* Пример для отладки и тестирования: для программы предоставлена таблица значений при произведении числа 123456789 на 9, 18, …, 81

Таблица №2 – результат работы программы после математических преобразований

|  |  |
| --- | --- |
| i | res |
| 9 | 1111111101 |
| 18 | 2222222202 |
| 27 | 3333333303 |
| 36 | 4444444404 |
| 45 | 5555555505 |
| 54 | 6666666606 |
| 63 | 7777777707 |
| 72 | 8888888808 |
| 81 | 9999999909 |

# 2.2. ПРОЕКТИРОВАНИЕ И АЛГОРИТМИЗАЦИЯ

Алгоритм решения задачи представлен в виде блок-схем.

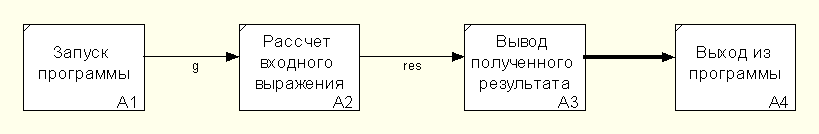


Рисунок 17 - Блочная структура программы №2

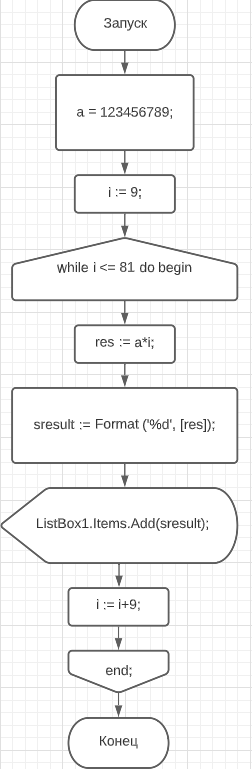


Рисунок 18 - Блок – схема программы №2, предназначенной для вычисления значений выражений

# 2.3. КОДИРОВАНИЕ АЛГОРИТМА

Идентификаторы переменных:

* a – Integer – Константа, используется для расчета конечного результата
  + i, res – Int64 – i используется, как один из элементов, участвующих в цикле. res используется для хранения произведения a и i
  + sresult – String – используется для вывода данных в текстовое поле.

Идентификаторы констант: a – Используется для расчета конечного результата

Ссылка на программу: <https://github.com/DeathOutsider/REU_Labs/tree/main/Lab2/Zadanie2>

# 2.4. ОТЛАДКА И ТЕСТИРОВАНИЕ ПРОГРАММЫ

Математическое решение выражение с помощью калькулятора, представлено в виде таблице в пункте 2.1 в разделе «Пример для отладки и тестирования»

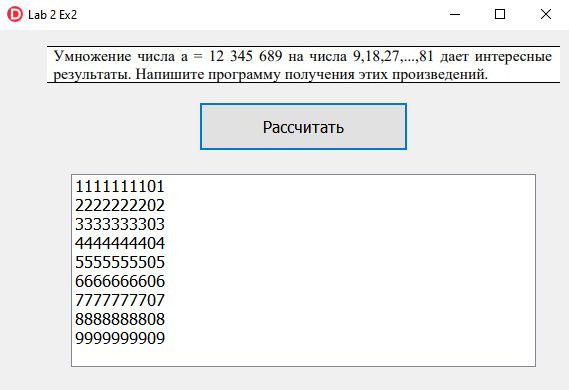


Рисунок 19 – Результат работы программы

Тестовый пример совпадает с полученным результатом, следовательно программа готова для практического использования.

# 2.5. РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ

Запустить файл Project1.exe, который находится в Win32/Debug/ корневой папки проекта

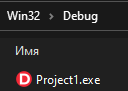


Рисунок 20 – Имя и Путь к файлу

После запуска программы открывается главное окно программы

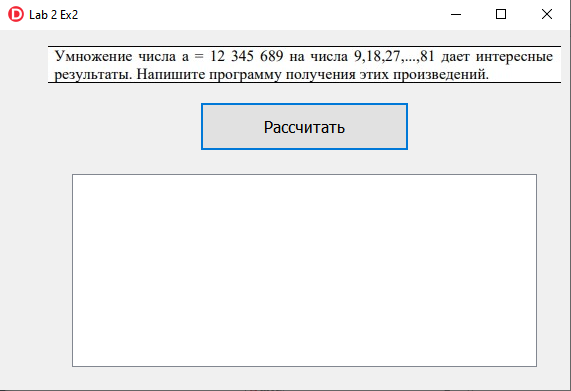


Рисунок 21 – Стартовый интерфейс программы

Далее следует нажать кнопку “Рассчитать”

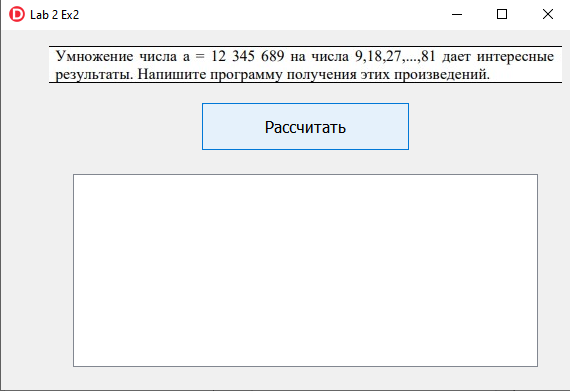


Рисунок 22 – Кнопка “Рассчитать”

И наслаждаться полученным результатом

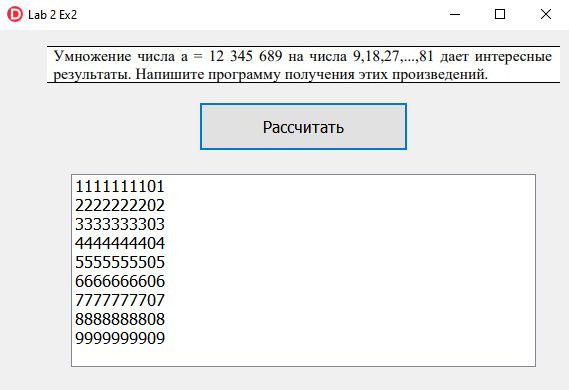


Рисунок 23 – Результат работы программы

Чтобы завершить работу с приложением следует закрыть его с помощью крестика в правом верхнем углу приложения

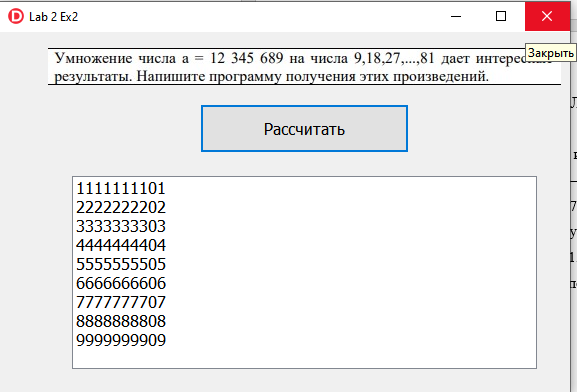


Рисунок 24 – Крестик в правом верхнем углу

# СПИСОК ИСПОЛЬ ЗОВАННЫХ ИСТОЧНИКОВ

1. Иноземцева С. А. Информатика и программирование: лабораторный практикум / С. А. Иноземцева. — Саратов: Вузовское образование, 2018.— 68 c. — ISBN 978-5-4487-0260-0. — Текст: электронный // Цифровой образовательный ресурс IPR SMART: [сайт]. — URL: https://www.iprbookshop.ru/75691.html (дата обращения: 15.12.2022). Режим доступа: для авторизир. пользователей. - DOI: <https://doi.org/10.23682/75691>

# ПРИЛОЖЕНИЕ 1. ЛИСТИНГ ПРОГРАММЫ К ЗАДАНИЮ №1

unit Unit1;

interface

uses

Winapi.Windows, Winapi.Messages, System.SysUtils, System.Variants, System.Classes, Vcl.Graphics,

Vcl.Controls, Vcl.Forms, Vcl.Dialogs, Vcl.StdCtrls, Vcl.Imaging.jpeg,

Vcl.ExtCtrls;

type

TForm1 = class(TForm)

Edit1: TEdit;

Edit2: TEdit;

Image1: TImage;

Label1: TLabel;

Label2: TLabel;

Label3: TLabel;

Button1: TButton;

Edit3: TEdit;

Label4: TLabel;

procedure Button1Click(Sender: TObject);

private

{ Private declarations }

public

{ Public declarations }

end;

var

Form1: TForm1;

implementation

{$R \*.dfm}

procedure TForm1.Button1Click(Sender: TObject);

var

zx, zy, parametr:real;

sx, sy, sparametr:string;

begin

sx := Edit1.Text;

sy := Edit2.Text;

sparametr := Edit3.Text;

zx := StrToFloat(sx);

zy := StrToFloat(sy);

parametr := abs(StrToFloat(sparametr));

if ( (zx>=0)and(zy>=0)and(zy<= (-zx + parametr)) )

then Label3.Caption := 'Принадлежит'

else if ( (zx<=0)and(zy<=0)and(zy>= (-zx - parametr)) )

then Label3.Caption := 'Принадлежит'

else Label3.Caption := 'Не принадлежит';

end;

end.

# ПРИЛОЖЕНИЕ 2. ЛИСТИНГ ПРОГРАММЫ К ЗАДАНИЮ №2

unit Unit1;

interface

uses

Winapi.Windows, Winapi.Messages, System.SysUtils, System.Variants, System.Classes, Vcl.Graphics,

Vcl.Controls, Vcl.Forms, Vcl.Dialogs, Vcl.StdCtrls, Vcl.Imaging.jpeg,

Vcl.ExtCtrls;

type

TForm1 = class(TForm)

Image1: TImage;

Button1: TButton;

ListBox1: TListBox;

procedure Button1Click(Sender: TObject);

private

{ Private declarations }

public

{ Public declarations }

end;

var

Form1: TForm1;

implementation

{$R \*.dfm}

procedure TForm1.Button1Click(Sender: TObject);

var

sresult:string;

i, res:int64;

const

a = 123456789;

begin

i := 9;

while i <= 81 do begin

res := a\*i;

sresult := Format ('%d', [res]);

//sresult := IntToStr(res);

ListBox1.Items.Add(sresult);

i := i+9;

end;

end;

end.