

МОСКОВСКИЙ АВТОМОБИЛЬНО-

ДОРОЖНЫЙ ГОСУДАРСТВЕННЫЙ

ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ (МАДИ)

Кафедра

«Автоматизированные системы управления»

Курсовая работа по дисциплине

«Программирование»

Вариант №15

Тема:

«**Создание программного приложения для расчета тормозного пути грузового автомобиля при заданных условиях движения**»

Выполнил: студент группы 2бАСУ1

Никитин Н.С

Проверил: доцент каф. АСУ

Баринов К. А.

Москва, 2021

Оглавление

[**1 Задание** 4](#_Toc93520900)

[**2 Назначение программы** 5](#_Toc93520901)

[**3 Руководство пользователя** 5](#_Toc93520902)

[3.1 Окно приложения 5](#_Toc93520903)

[3.2 Заголовок программы 6](#_Toc93520904)

[3.3 Меню программы 6](#_Toc93520905)

[3.4 Подменю «Файл» 7](#_Toc93520906)

[3.5 Пункт «Создать» 7](#_Toc93520907)

[3.6 Пункт «Открыть…» 7](#_Toc93520908)

[3.7 Пункт «Сохранить» 8](#_Toc93520909)

[3.8 Пункт «Сохранить как…» 9](#_Toc93520910)

[3.9 Пункт «Выход» 9](#_Toc93520911)

[3.10 Пункт «Обработать данные» 10](#_Toc93520912)

[**4 Выбор среды разработки** 11](#_Toc93520913)

[**5 Форматы данных** 12](#_Toc93520914)

[5.1 Входные данные 12](#_Toc93520915)

[5.2 Выходные данные 13](#_Toc93520916)

[**6 Структура программы** 13](#_Toc93520917)

[6.1 Модуль <Project1.cpp> 13](#_Toc93520918)

[6.2 Константы 13](#_Toc93520919)

[6.3 Класс оконной формы <TFormMain> 13](#_Toc93520920)

[6.4 Компоненты формы 13](#_Toc93520921)

[6.5 Методы обработки событий 16](#_Toc93520922)

[6.6 Прочие методы класса 16](#_Toc93520923)

[6.7 Переменные 17](#_Toc93520924)

[**7 Описание алгоритмов** 19](#_Toc93520925)

[7.1 Блок-схема укрупненного алгоритма работы пользовательского интерфейса приложения 19](#_Toc93520926)

[7.2 Блок-схема алгоритма метода void \_\_fastcall TMainForm::ProcesDataClick(TObject \*Sender) 21](#_Toc93520927)

[7.3 Обозначение и смысл объектов из блок-схемы 21](#_Toc93520928)

[7.4 Блок-схема алгоритма метода bool \_\_fastcall TMainForm::SaveData () 27](#_Toc93520929)

[**8 Список литературы** 29](#_Toc93520930)

[**9 Текст программы с комментариями** 31](#_Toc93520931)

# **1 Задание**

Тормозной путь грузового автомобиля может быть подсчитан по формуле:

, где

 - время первой и второй фаз торможения (сек);

*=*9.81м/сек2 – ускорение свободного падения;

** - коэффициент инерции вращающихся масс;

 - коэффициент сцепления;

 - уклон;

 - скорость перед торможением;

 - коэффициент учета условий торможения.

, для груженого автомобиля.

=

, для порожнего автомобиля.

*i*=0.21; *t1*=0.17 сек; *t2*=0,77; =0.91.

Задания:

1. Определить тормозной путь для коэффициентов φ = 0.2, 0.39, 0.45, 0.49, 0.57, 0.63, 0.77, при значениях *V0*, изменяющихся от 20 до 50 км/час с шагом 5 км/час. Результат вывести в виде таблицы.

2. При заданном *V0* определить уклон *i*, при котором тормозной путь

лежит в пределах от 3 до 100 метров. Алгоритм “деления отрезка пополам” записать процедурным блоком.

# **2 Назначение программы**

Приложение вычисляет тормозной путь для различных коэффициентов сцепления, а также различные параметры тормозного пути, учитывая коэффицент учета условий торможения, коэффициент инерции вращающихся масс, время первой и второй фаз торможения и ускорение свободного падения. Определяет уклон при разных значениях тормозного пути и произвольного значения скорости автомобиля.

# **3 Руководство пользователя**

# 3.1 Окно приложения

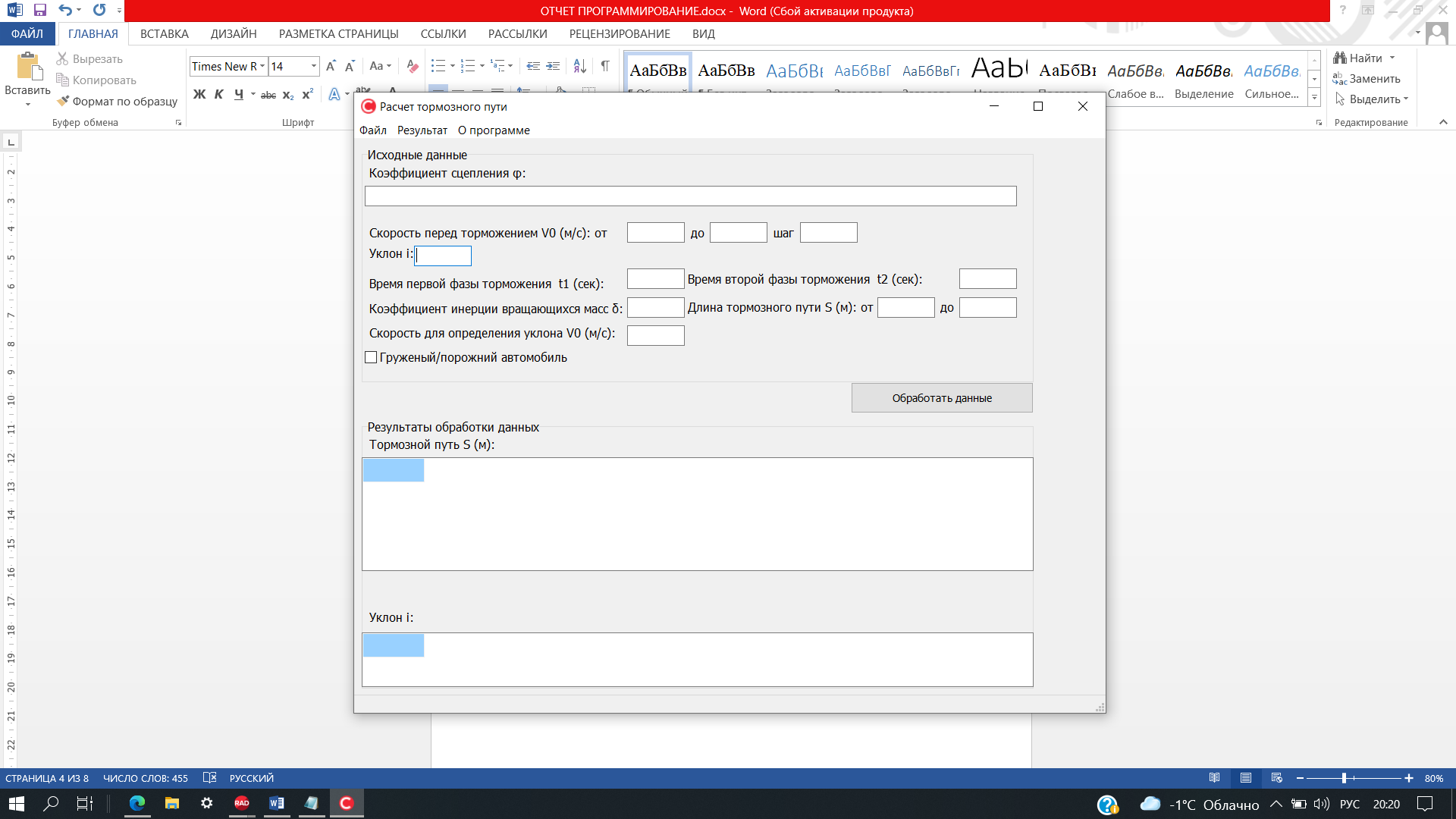


Рис 1. Окно приложения

Окно программы состоит из 5 частей: заголовок программы, главное меню программы, область ввода исходных данных, область вывода результатов обработки и строка вывода информации.

# 3.2 Заголовок программы

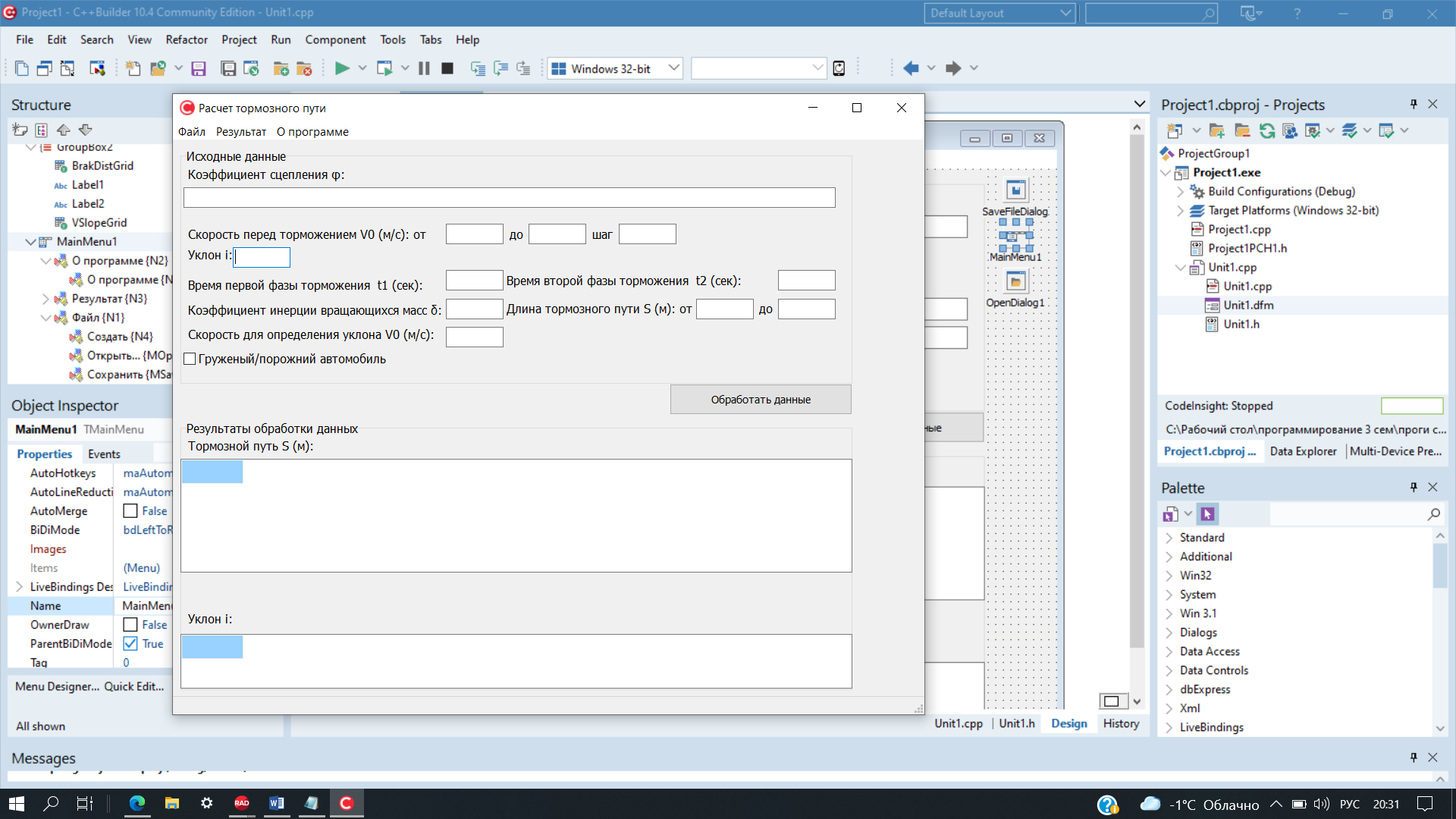


Рис 2. Заголовок программы

Заголовок программы (Рис. 2) состоит из: места вывода информации о программе и кнопок управления окном («Свернуть», «Развернуть на весь экран» («Восстановить») и «Закрыть»).

Место вывода информации о программе

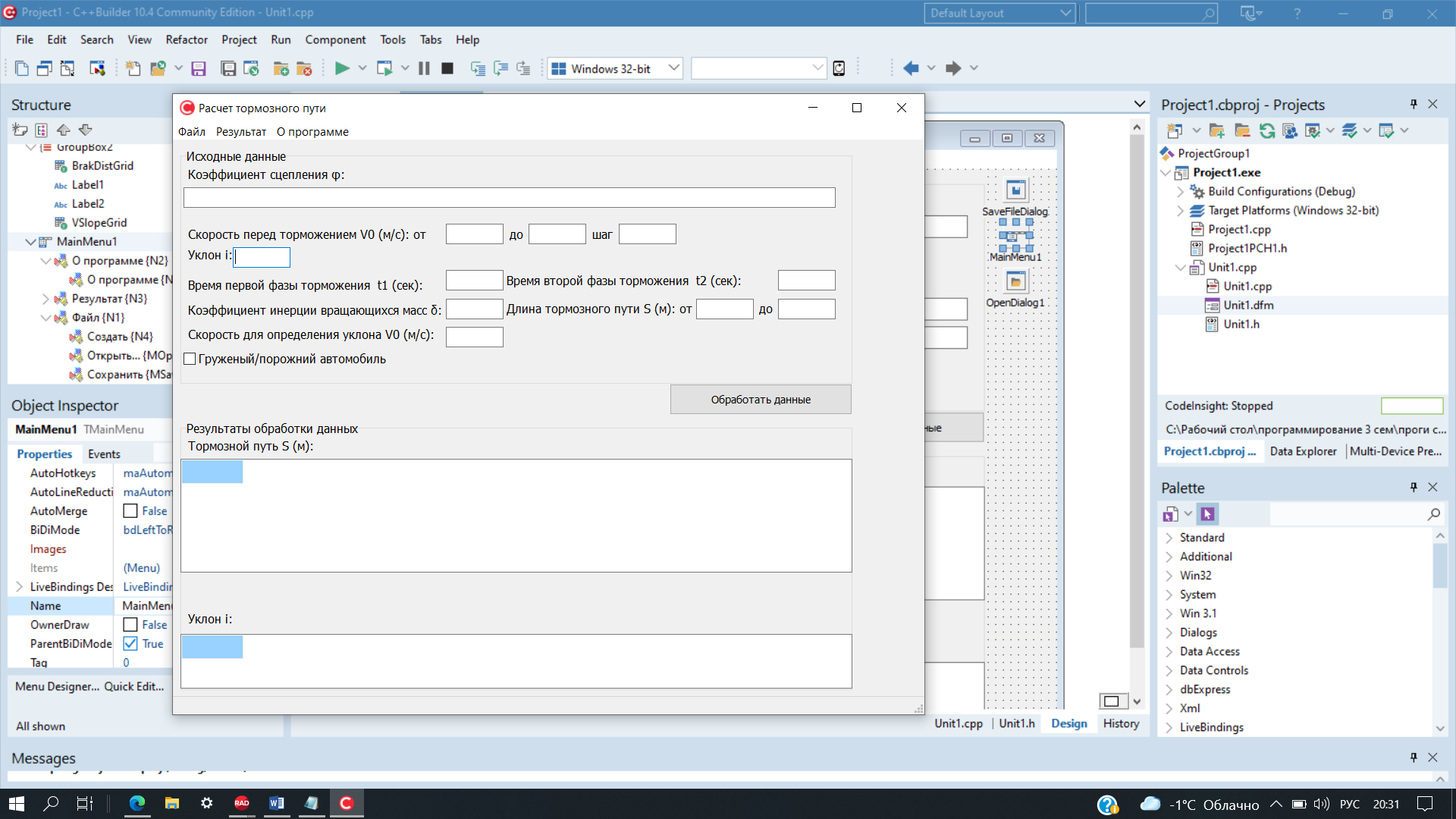


Рис 3. Место вывода информации о программ

В левой части заголовка программы находится информация о самой программе.

Кнопки управления окном

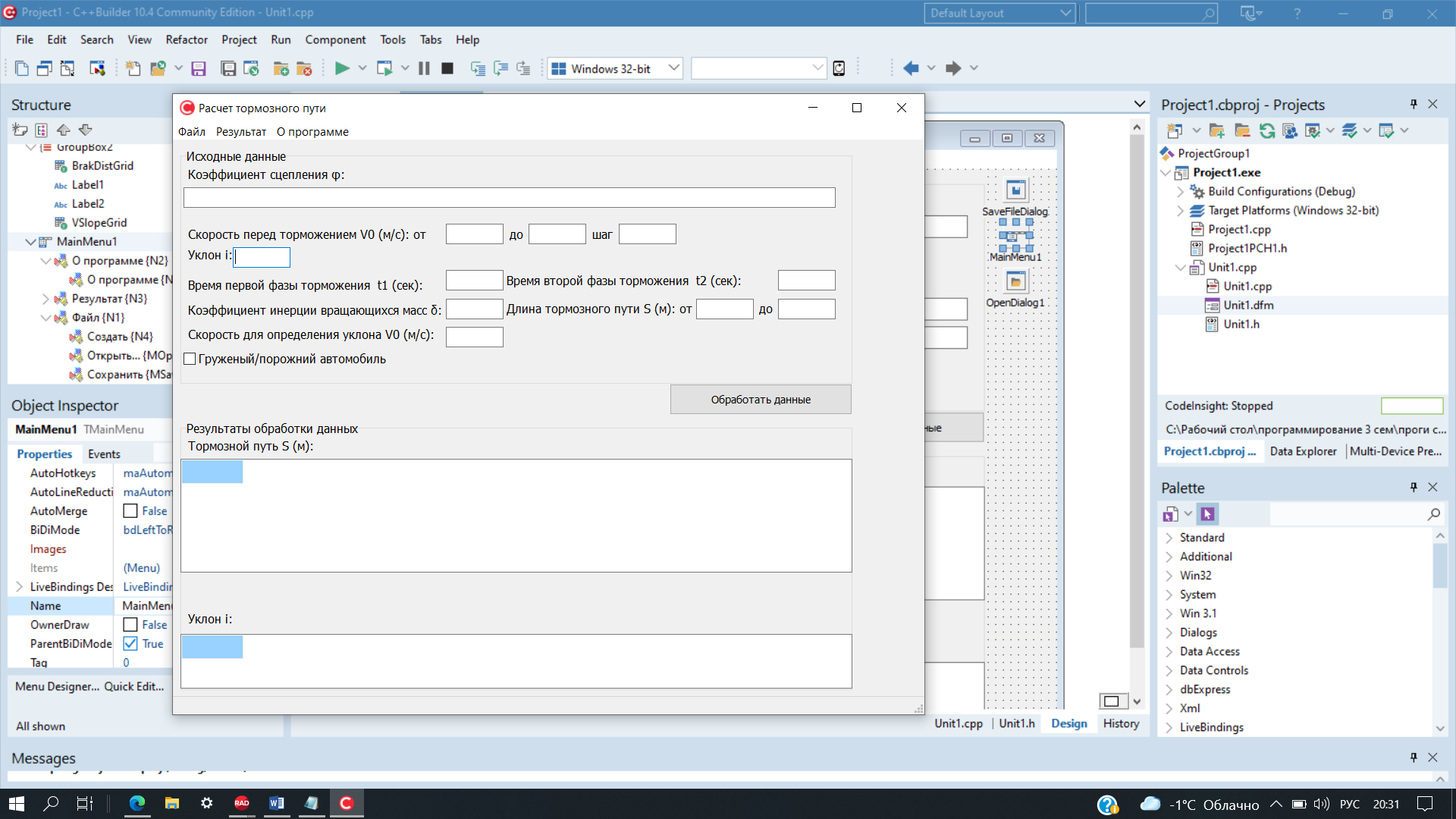


Рис 4. Кнопки управления окном

В заголовке программы в правом углу находятся три кнопки управления окном. При нажатии на кнопку «свернуть» (самая левая) программа сворачивается в панель задач. Чтобы обратно развернуть окно в предыдущее состояние, нужно нажать на иконку программы в панели задач. При нажатии на кнопку «Развернуть на весь экран» (посередине) окно программы разворачивается на весь экран. При этом кнопка меняется на кнопку «восстановить», при нажатии на которую окно восстанавливает размер, который был до нажатия на кнопку «Развернуть на весь экран». При нажатии на кнопку «Закрыть» (самая правая) программа будет закрыта.

# 3.3 Меню программы

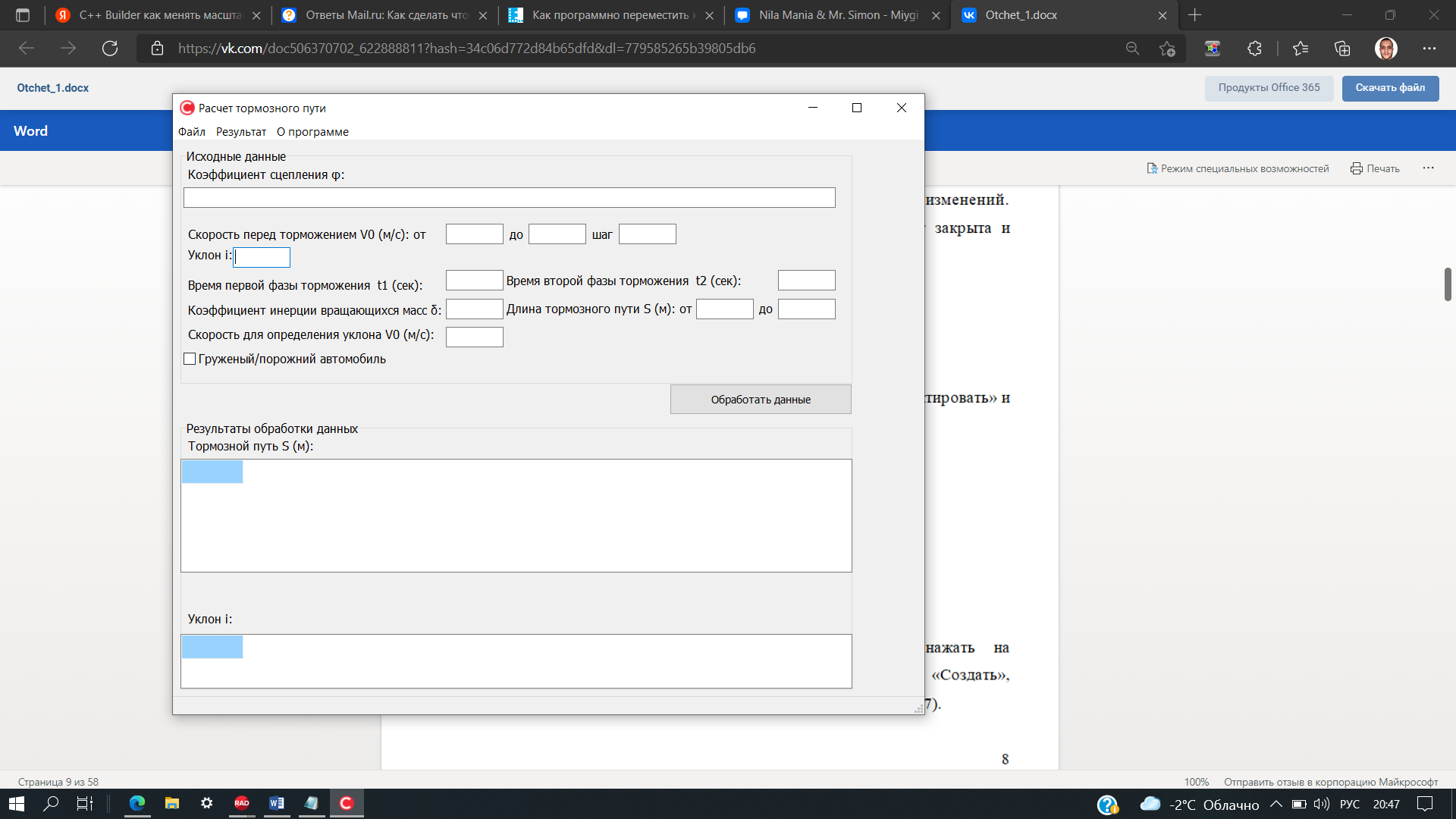


Рис 5. Меню программы

Меню программы состоит из 3 подменю: «Файл», «О программе» и «Результат»

# 3.4 Подменю «Файл»

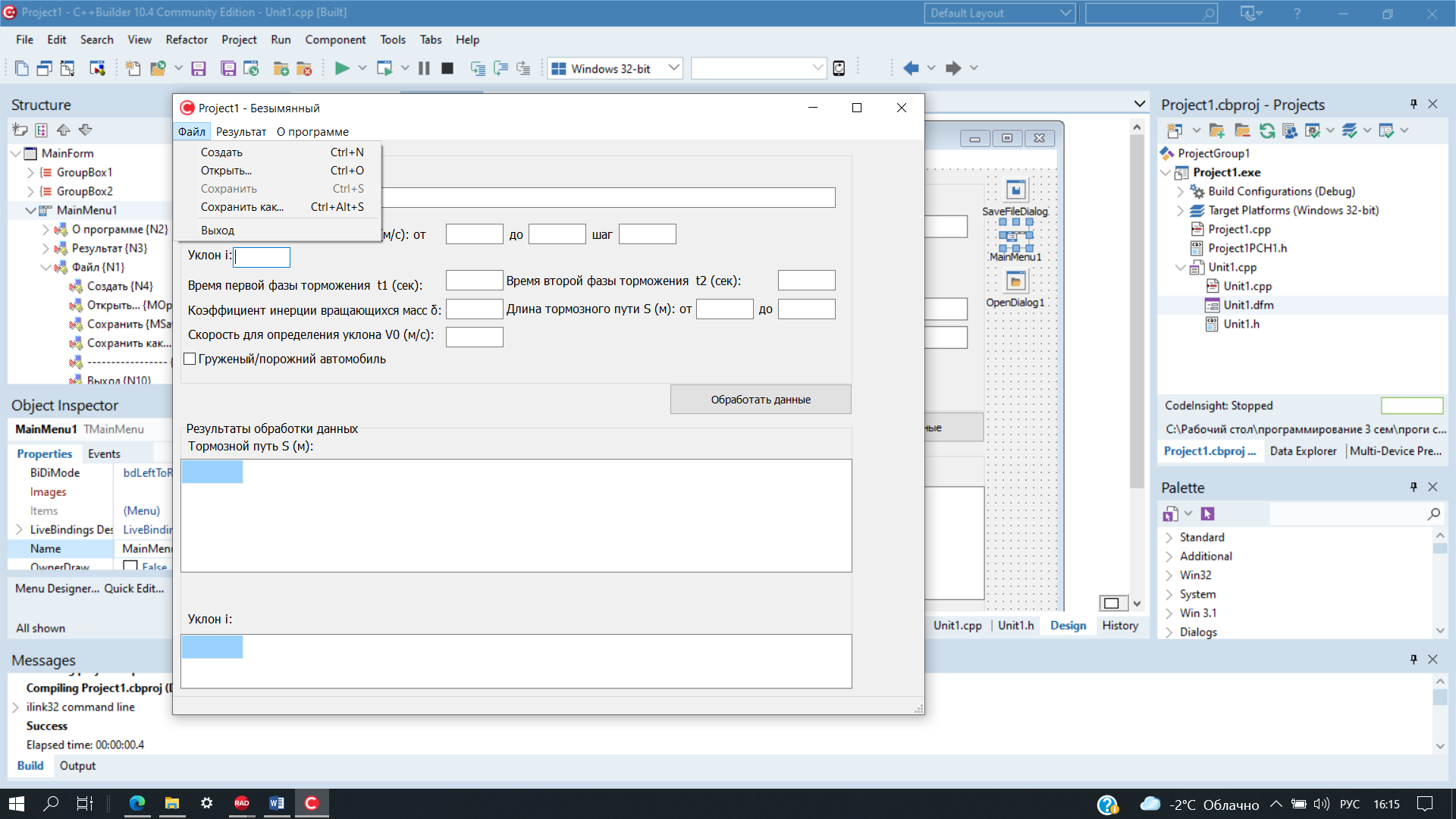


Рис 6. Подменю «Файл»

Чтобы открыть раздел меню «Файл», нужно нажать на соответствующую надпись. В этом разделе есть 5 пунктов: «Создать», «Открыть…», «Сохранить», «Сохранить как…» и «Выход»

# 3.5 Пункт «Создать»

При нажатии на пункт «Создать» (комбинация клавиш Ctrl+N), программа создает новый чистый набор исходных данных (очищаются поля ввода исходных данных и вывода результата). Если был открыть файл, то он закрывается.

# 3.6 Пункт «Открыть…»

При нажатии на пункт «Открыть…» (комбинация клавиш Ctrl+O), появляется окно выбора файла.

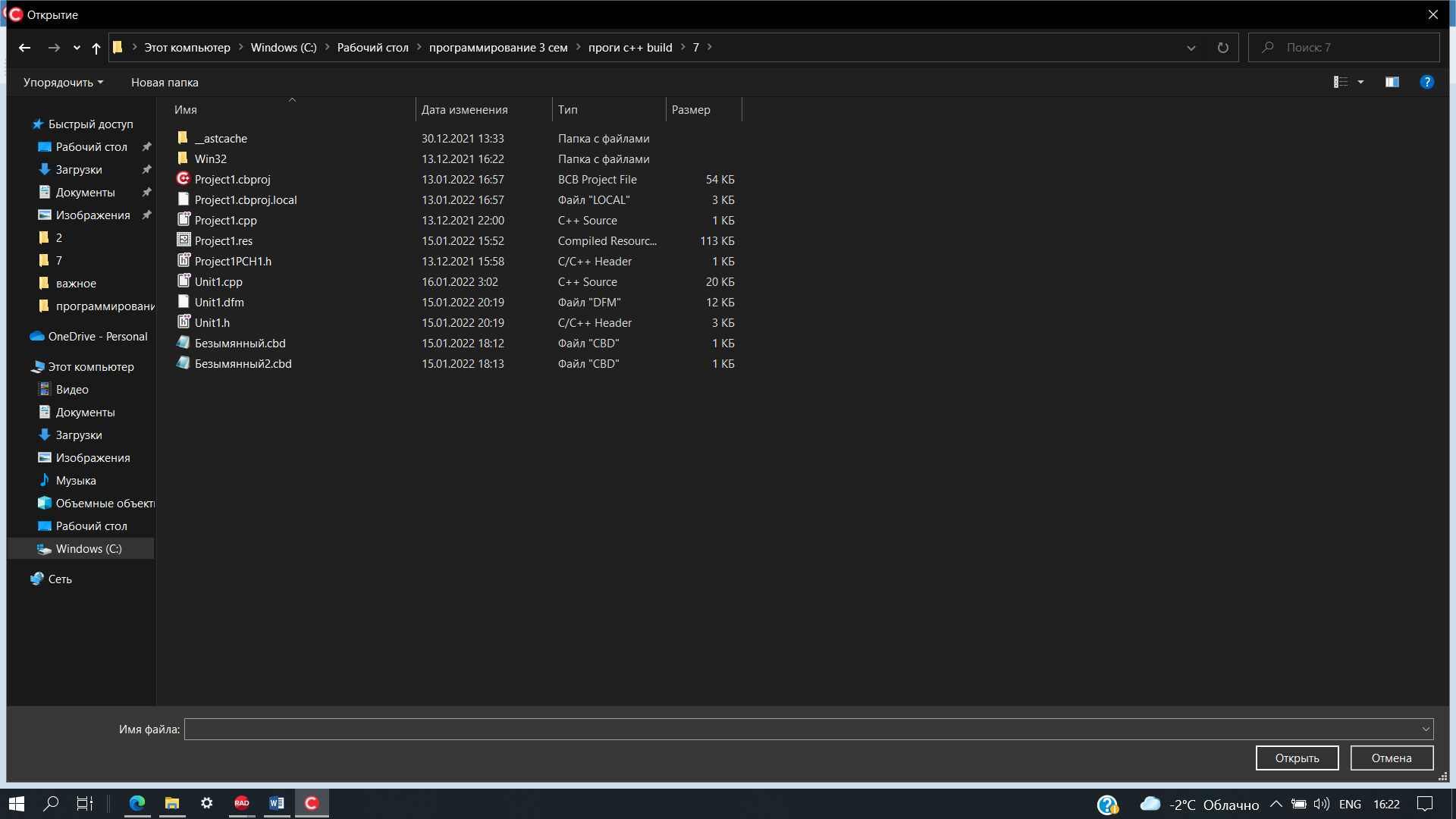


Рис 7. Окно выбора файла

Для выбора файла нужно выбрать его из пункта и нажать на кнопку «Открыть», либо написать имя файла в поле «Имя файла» (полный путь к нему), либо выбрать его двойным нажатием. При нажатии на кнопку «Отмена» или закрытии окна, открытие файла будет прервано.

Если не получится открыть файл, высветится окно с сообщением, при этом открытие будет прервано.

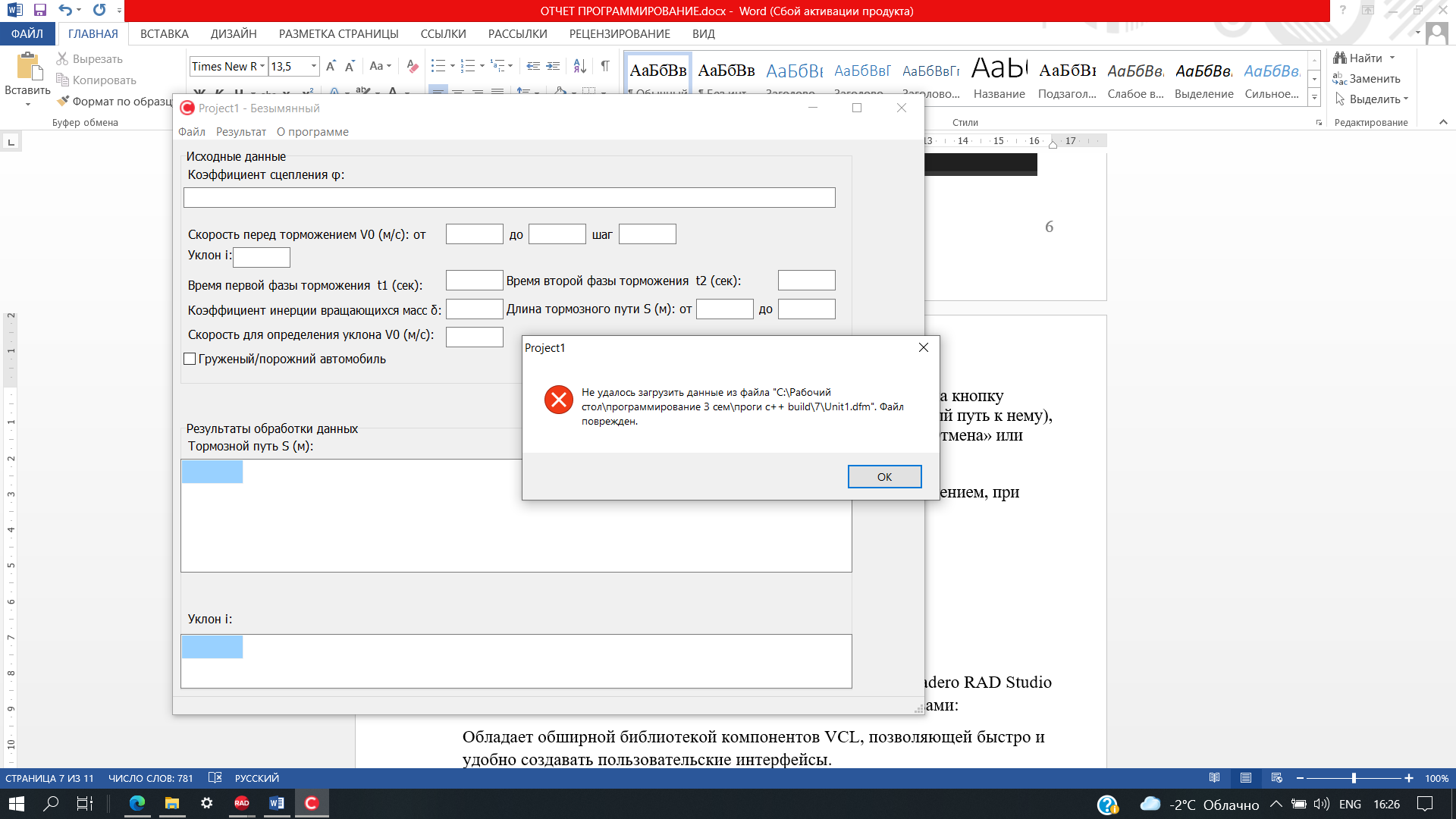


Рис 8. Окно с сообщением о невозможности открыть файл

# 3.7 Пункт «Сохранить»

При нажатии на пункт «Сохранить» (комбинация клавиш Ctrl+S), программа сохранит изменения исходных данных в открытый файл. При отсутствии открытого файла высветится окно выбора имени файла для сохранения.

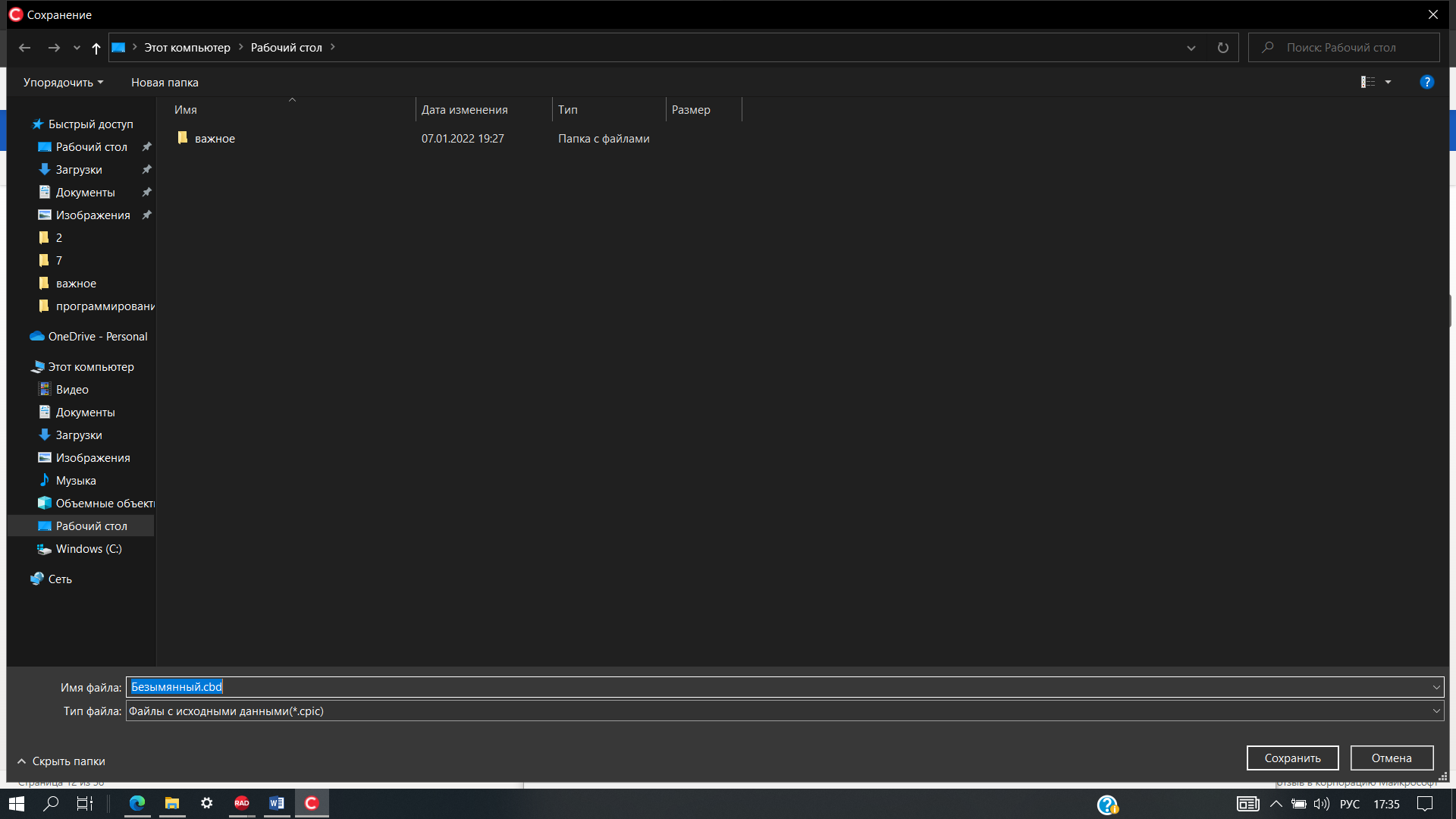


Рис 9. Окно выбора имени файла для сохранения

В окне выбора имени файла для сохранения надо либо выбрать существующий файл и нажать на кнопку «Сохранить», либо выбрать с двойным нажатием, либо ввести имя или путь к файлу, при этом, если файл с таким именем не будет найден, он будет создан. При нажатии на кнопку «Отмена» или закрытии окна, сохранение будет прервано.При отсутствии несохраненных изменений пункт не доступен.

# 3.8 Пункт «Сохранить как…»

При нажатии на пункт «Сохранить как…» (комбинация клавиш Ctrl+Alt+S), программа выведет окно выбора файла для сохранения, в котором для выбора файла надо либо выбрать существующий файл и нажать на кнопку «Сохранить», либо выбрать двойным нажатием, либо ввести имя или путь к файлу, при этом, если файл с таким именем не будет найден, он будет создан. При нажатии на кнопку «Отмена» или закрытии окна, сохранение будет прервано.

# 3.9 Пункт «Выход»

При нажатии на пункт «Выход» программа закроется.

Подменю «Результаты»

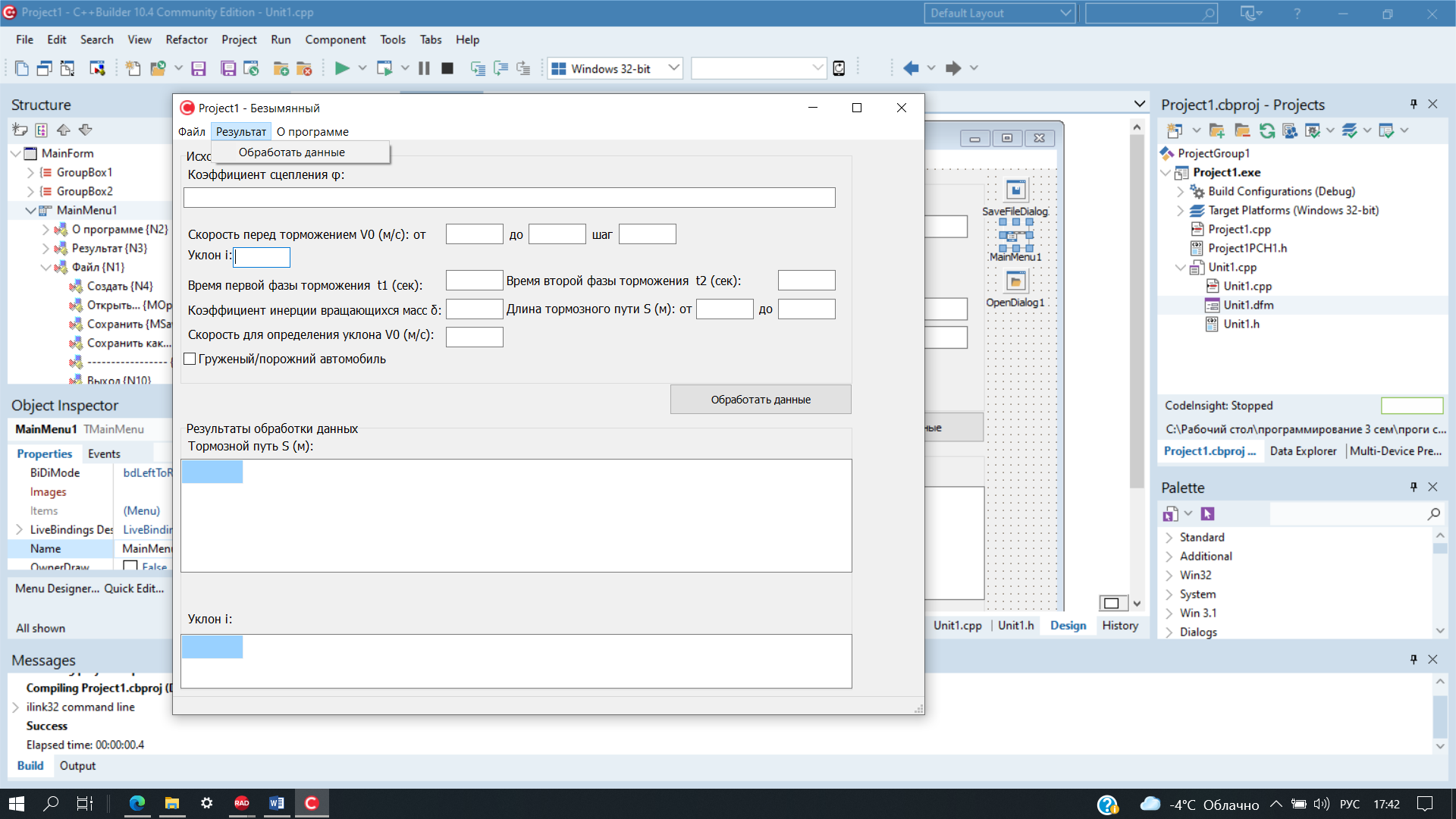


Рис 10. Подменю «Результаты»

Чтобы открыть раздел меню «Результаты», нужно нажать на соответствующую надпись. В этом разделе есть пункт «Обработать данные».

# 3.10 Пункт «Обработать данные»

При нажатии на пункт «Обработать данные», после чего в области вывода результатов обработки появятся результаты обработки исходных данных.

Если значения данных будут недопустимы, некорректны или отсутствовать, то появится окно с соответствующим сообщением.

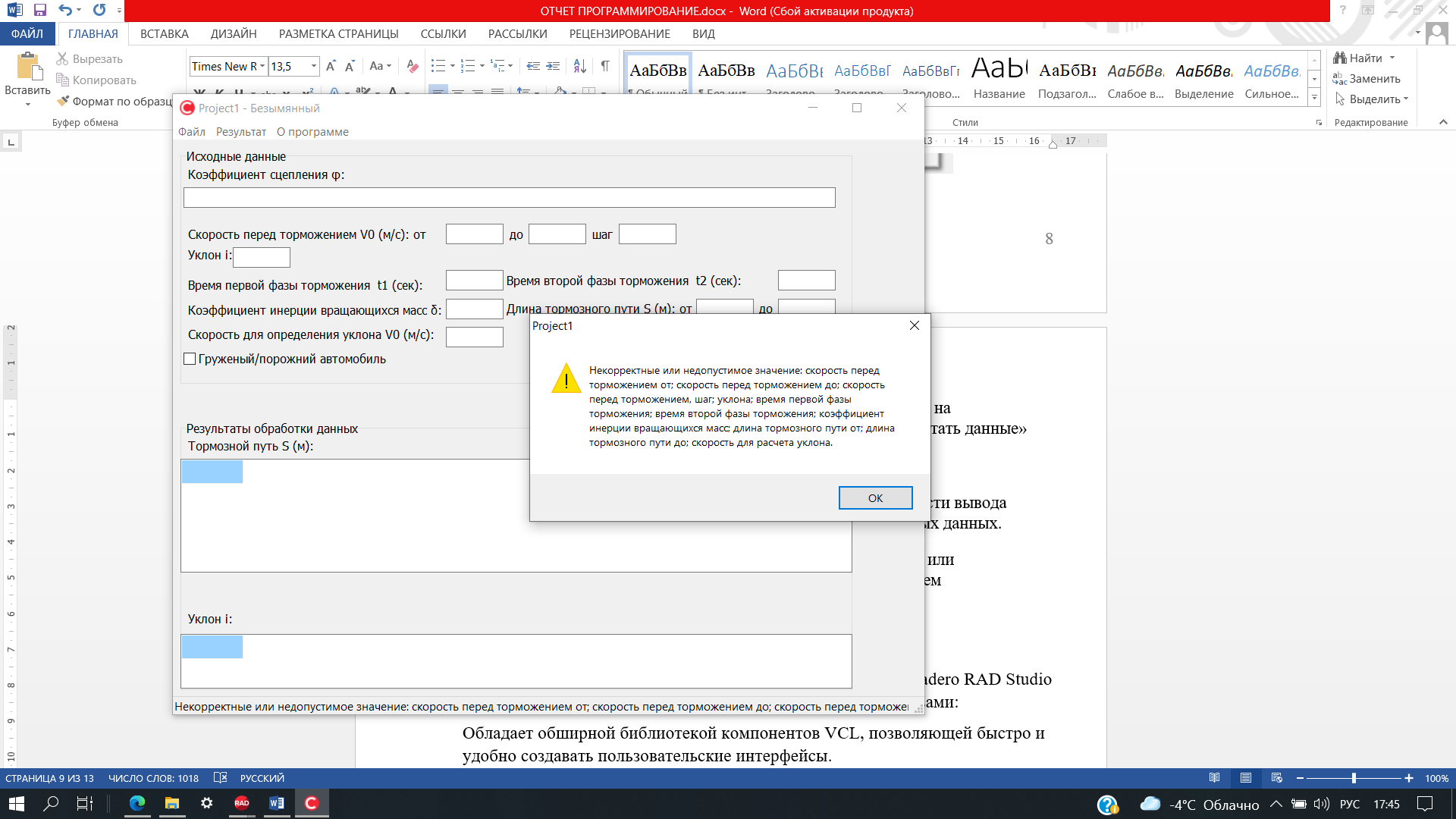


Рис 11. Окно с сообщением об ошибке

Области ввода исходных данных

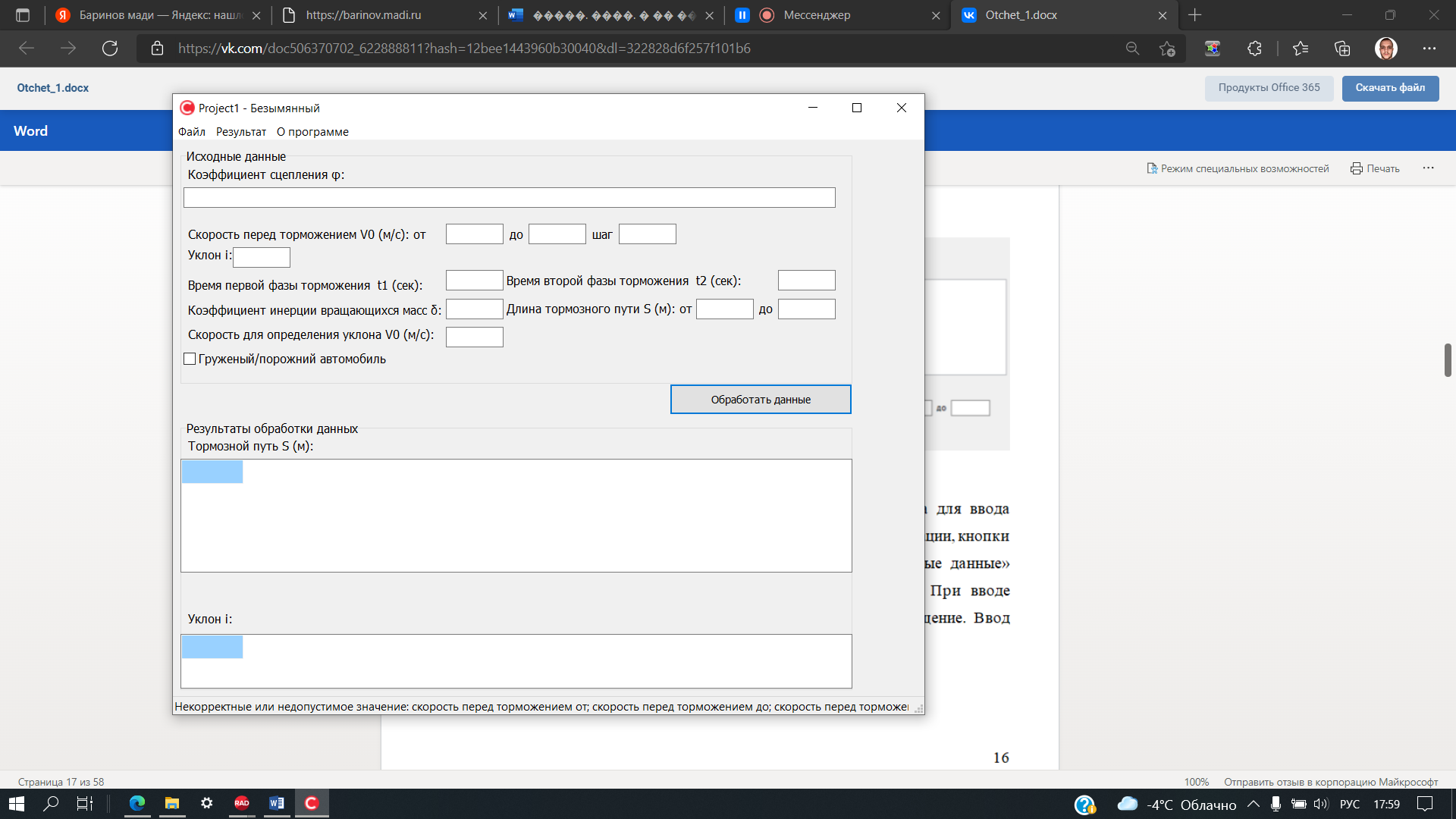


Рис 12. Области ввода исходных данных

В области ввода исходных данных находятся поля для ввода данных.

Область вывода результатов обработки

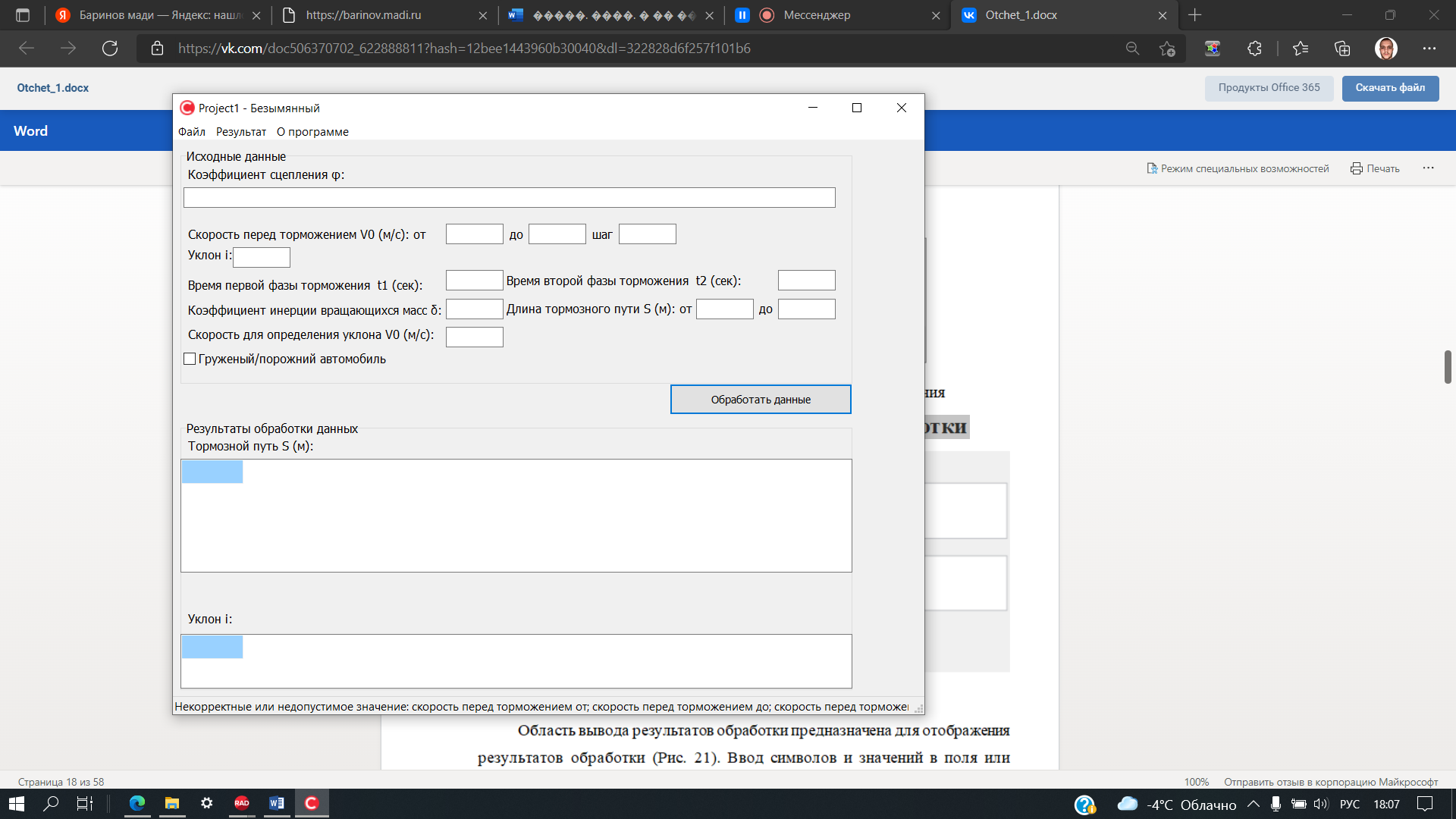


Рис 13. Области вывода результатов обработки данных

Область вывода результатов обработки предназначена для отображения результатов обработки. Ввод символов и значений в поля или таблицы заблокирован.

Строка вывода информации

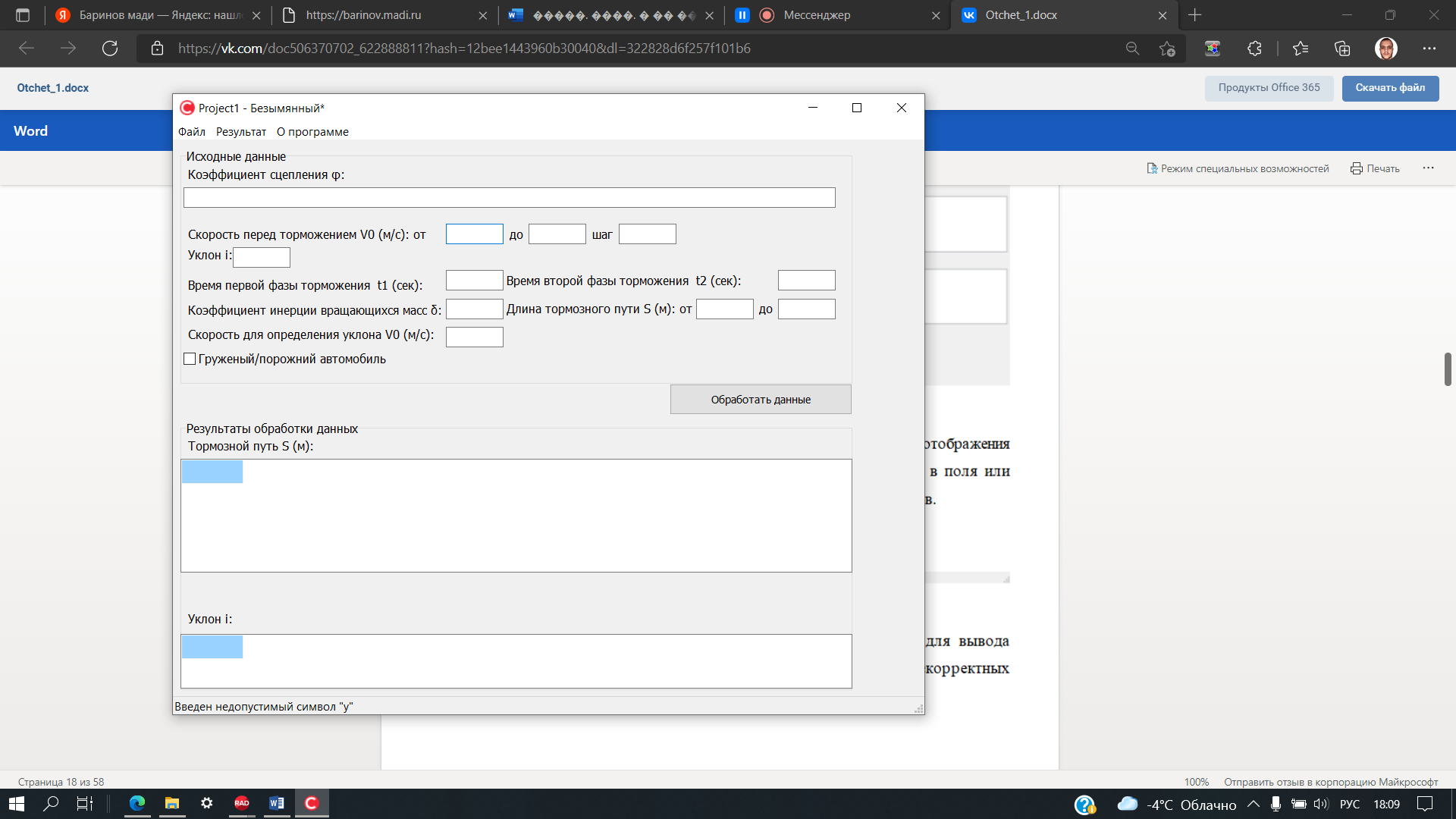


Рис 14. Строка вывода информации

Строка вывода информации предназначена для вывода различных сообщений об ошибках, недопустимых и некорректных значениях.

# **4 Выбор среды разработки**

Для данного приложения я выбрал среду разработки Embarcadero RAD Studio C++ Builder, так как она обладает следующими преимуществами:

Обладает обширной библиотекой компонентов VCL, позволяющей быстро и удобно создавать пользовательские интерфейсы. Обладает инструментами для создания графиков и таблиц. Обеспечивает высокую скорость разработки приложений. Возможность создания пользовательских функци.

C++ Builder объединяет в себе комплекс объектных библиотек, например: STL, VCL, CLX, MFC и др., а также, компилятор, отладчик, редактор кода и многие другие компоненты.Данная среда разработки содержит инструменты, которые делают разработку пользовательского интерфейса визуальной, упрощает программирование благодаря встроенному WYSIWYG (What You See Is What You Get, «что видишь, то и получишь») — редактору интерфейса и пр.

# **5 Форматы данных**

# 5.1 Входные данные

В качестве файла для хранения исходных данных был выбран двоичный файл, так как:

* в программе присутствуют средства для ввода и редактирования данных, а также сохранения и чтения из файлов;
* при записи в двоичный файл между значениями нет необходимости ставить пробельные символы разделители, которые приведут к увеличению размера файла;
* в двоичный файл можно записывать (читать) данные блоками в отличие от текстового файла, например, записать весь массив в двоичный файл можно с помощью одной операции без использования цикла

Структура файла

Структура файла с исходными данными:

В файл записывается набор элементов, состоящий из:

* массив коэффициентов сцепления
* нижнего значения расчетной скорости движения, м/с
* верхнего значения расчетной скорости движения, м/с
* шага изменения скорости, м/с
* уклона
* время первой фазы торможения, сек
* время второй фазы торможения, сек
* коэффициента инерции вращающихся масс
* нижнего значения тормозного пути для вычисления уклона, м
* верхнего значения тормозного пути для вычисления уклона, м

Для ввода исходных данных в файл с помощью клавиатуры в программе предусмотрены поля для ввода значений.

# 5.2 Выходные данные

Для вывода обработанных данных в программе предусмотрено окно вывода, в котором присутствуют поля и таблицы.

# **6 Структура программы**

# 6.1 Модуль <Project1.cpp>

Модуль содержит: Unit1.cpp, Unit1.dfm и Unit1.h. Модуль содержит главную форму программы и занимается обработкой всех действий на ней, сохранение и загрузкой, а также вычислением результата.

# 6.2 Константы

|  |  |
| --- | --- |
| Определение | Назначение |
| #define DefFileName L"Безымянный" | Название файла по умолчанию |

# 6.3 Класс оконной формы <TFormMain>

Оконная форма предназначена для ввода исходных данных и вывод результатов обработки.Форма создается автоматически при запуске программы.

# 6.4 Компоненты формы

|  |  |  |
| --- | --- | --- |
| Имя | Класс | Назначение |
| MainMenu | TMainMenu | Предназначен для запуска основных функций программы |
| Label14 | TLabel | Предназначен для подписи поля ввода Edit11 |
| Label15 | TLabel | Предназначен для подписи поля ввода Edit2 |
| Label16 | TLabel | Предназначен для подписи поля ввода Edit3 |
| Label17 | TLabel | Предназначен для подписи поля ввода Edit4 |
| Label18 | TLabel | Предназначен для подписи поля ввода Slope |
| Label19 | TLabel | Предназначен для подписи поля ввода TimeFirstBrakPhase |
| Label20 | TLabel | Предназначен для подписи поля ввода TimeSecBrakPhase |
| Label21 | TLabel | Предназначен для подписи поля ввода BrakDistFrom |
| Label22 | TLabel | Предназначен для подписи поля ввода BrakDistTo |
| LinkLabel1 | TLinkLabel | Предназначен для подписи поля ввода InertCoefRotatMas |
| LinkLabel2 | TLinkLabel | Предназначен для подписи поля ввода Vslope |
| Label1 | TLabel | Предназначен для подписи таблицы BrakDistGrid |
| Label2 | TLabel | Предназначен для подписи таблицы VSlopeGrid |
| ProcesData | TButton | Предназначен для запуска обработки исходных данных |
| StatusBar | TStatusBar | Предназначен для вывода информации о возникшей ошибке. |
| SaveFileDialog | TSaveDialog | Предназначен для получения имени файла, в который будут сохранятся исходные данные |
| OpenDialog1 | TOpenDialog | Предназначен для получения имени файла, который будет открыт |
| GroupBox1 | TGroupBox | Предназначен для визуального деления исходных данных от результата обработки данных |
| GroupBox2 | TGroupBox | Предназначен для визуального деления результата обработки данных от исходных данных |
| Edit11 | TEdit | Предназначен для ввода коэфициентов φ |
| Edit2 | TEdit | Предназначен для ввода нижней границы скорости |
| Edit3 | TEdit | Предназначен для ввода верхней границы скорости |
| Edit4 | TEdit | Предназначен для ввода шага скорости |
| TimeFirstBrakPhase | TEdit | Предназначен для ввода первой фазы торможения |
| TimeSecBrakPhase | TEdit | Предназначен для ввода второй фазы торможения |
| BrakDistFrom | TEdit | Предназначен для ввода нижней границы тормозного пути для определения уклона |
| BrakDistTo | TEdit | Предназначен для ввода вехней границы тормозного пути для определения уклона |
| InertCoefRotatMas | TEdit | Предназначен для ввода коэфициента инерции вращающихся масс |
| Vslope | TEdit | Предназначен для ввода скорости для определения уклона |

# 6.5 Методы обработки событий

|  |  |
| --- | --- |
| Имя | Условие вызова |
| MCreateFileClick | Нажатие на пункт меню «Файл» – «Создать» или комбинация клавиш Ctrl+N |
| MOpenFileClick | Нажатие на пункт меню «Файл» – «Открыть…» или комбинация клавиш Ctrl+O |
| MSaveFileClick | Нажатие на пункт меню «Файл» – «Сохранить» или комбинация клавиш Ctrl+S |
| MSaveClick | Нажатие на пункт меню «Файл» – «Сохранить как…» или комбинация клавиш Ctrl+Alt+S |
| Exit | Нажатие на пункт меню «Файл» – «Выход» или комбинация клавиш Ctrl+F4 |
| ProcesDataClick | Нажатие на пункт меню «Результат» – «Обработать данные» или нажатие на кнопку «Обработать данные» |
| KeyPress | Нажатие клавиш в полях ввода класса TEdit |

# 6.6 Прочие методы класса

|  |  |  |
| --- | --- | --- |
| Заголовок метода | Назначение | Возвращаемое значение |
| bool \_\_fastcall TMainForm::SaveData () | Сохранение исходных данных в файл | Возвращаемое значение типа bool. True означает, что удалось сохранить исходные данные, false, что при сохранении произошла ошибка и не удалось сохранить |
| oid \_\_fastcall TMainForm::ClearOldResults () | Очистка всех полей формы |  |
| bool \_\_fastcall TMainForm::CheckData (bool ToSave) | Проверка входных данных на корректность и допустимость | Возвращаемое значение типа bool. True означает, что все исходные данные корректны или допустимы, false, что есть некорректные или недопустимые |

# 6.7 Переменные

Глобальные

Глобальных переменных нет.

Локальные

|  |  |  |  |
| --- | --- | --- | --- |
| Название функции | Имя переменой | Тип | Назначение |
| CheckData | ErrMess | String | Хранит сообщение об ошибках, что были найдены в исходных данных |
|  | ErrorF | bool | Хранит ошибку при проверке корректности исходных данных |
| SaveData | FCount | int | Хранит количество значений в поле Edit11 |
|  | str | String | Хранит поле Edit11 |
|  | TypeCar | bool | Хранит значение поля СheckTypeCar |
|  | DataEdit | double | Хранит значение DBL\_MIN при условии, что одно из полей ввода исходных данных пустое |
|  | CountByte | unsigned int | Хранит количество байт в файле |
| ProcesDataClick | FCount | int | Хранит количество значений в поле Edit11 |
|  | stream | stringstream | Хранит строку str |
|  | str | String | Хранит поле Edit11 |
|  | StartMean | double | Хранит значение поля Edit2 |
|  | FinishMean | double | Хранит значение поля Edit3 |
|  | StMean | double | Хранит значение поля Edit4 |
|  | CountSpeed | int | Хранит значение количества шагов для скорости |
|  | TypeCar | bool | Хранит значение поля CheckTypeCar |
|  | ir | double | Хранит значение поля Slope |
|  | t1 | double | Хранит значение поля TimeFirstBrakPhase |
|  | t2 | double | Хранит значение поля TimeSecBrakPhase |
|  | b | double | Хранит значение поля InertCoefRotatMas |
|  | v | double | Хранит значение поля VSlope |
|  | Vf | double | Хранит значение поля BrakDistFrom |
|  | Vt | double | Хранит значение поля BrakDistTO |
| functionS | Ky | double | Хранит значение коэффициента учета условий торможения |
| algВivSegmentHalf | res | double | Хранит значение Ку, возвращаемое процедурой functionS |
|  | i | double | Хранит значение деления отрезка пополам |
| MOpenFileClick | FileName | String | Хранит имя файла |
|  | ByteCount | double | Хранит значение размера файла |
|  | TypeCar | bool | Хранит значение типа автомобиля |

# **7 Описание алгоритмов**

# 7.1 Блок-схема укрупненного алгоритма работы пользовательского интерфейса приложения

Отсутствует

Нажатие на поле ввода CheckTypeCar

Инициализация приложения (создание формы, отображение главной формы на экране)

Действие пользователя

Выбор пункта главного меню “Файл” – “Создать”

TMainForm::MCreateFileClick(TObject \*Sender)

Выбор пункта главного меню “Файл” – “Сохранить”

TMainForm::MSaveFileClick(TObject \*Sender)

Выбор пункта главного меню “Файл” – “Сохранить как…”

TMainForm::MSaveAsClick(TObject \*Sender)

Выбор пункта главного меню “Редактировать” – “Очистить форму”

TMainForm::ClearClick(TObject \*Sender)

Выбор пункта главного меню “Результат” – “Обработать данные” или кнопку Обработать данные

TMainForm::ProcesDataClick(TObject \*Sender)

Выбор пункта главного меню “Справка” – “О программе”

TMainForm::MOpenFileClick (TObject \*Sender)

oid \_\_fastcall TMainForm::informationClick(TObject \*Sender)

Выбор пункта главного меню “Файл” – “Открыть”

TMainForm::Edit2KeyPress(TObject \*Sender, System::WideChar &Key)

void \_\_fastcall TMainForm::CheckTypeCarClick(TObject \*Sender)

Нажатие клавиши в полях ввода

0

1

Выбор пункта главное меню “Файл” – “Выход”

void \_\_fastcall TMainForm::FormResize(TObject \*Sender)

void \_\_fastcall TMainForm::Exit(TObject \*Sender)

CanClose?

Деиницилизация приложения

void \_\_fastcall TMainForm::FormCloseQuery(TObject \*Sender, bool &CanClose)

## 7.2 Блок-схема алгоритма метода void \_\_fastcall TMainForm::ProcesDataClick(TObject \*Sender)

# 7.3 Обозначение и смысл объектов из блок-схемы

FCount – переменная целочисленного типа, которая будет считать количество значение коэфициента φ.

NUM – переменная типа данных с плавающей точкой, используется для передачи данных из строкового потока.

str – переменная типа данных String, используется для передачи данных из текстового поля.

stream – строковый поток.

CheckData() – процедура, отвечающая за проверку на различные ошибки.

FI – димаческий массив, отвечающий за хранение коэфициентов φ.

ColCount– значение столбцов в таблице.

BrakDistGrid– таблица для вычисления значений тормозного пути.

StartMean–начальная скорость.

FinishMean – конечная скорость.

StMean– шаг скорости.

CountSpeed– вычисление количества строк для таблицы BrakDistGrid.

TypeCar – переменная булевского типа, отвечает за тип автомобиля груженый/порожний.

ir –переменная отвечает за уклон.

t1 – переменная отвечает за первую фазу торможения.

t2 – переменная отвечает за вторую фазу торможения.

b – переменная отвечает за коэфициент вращающихся масс.

vF – тормозной путь для определения уклона от.

vT – тормозной путь для определения уклона до.

f – переменная принимает значения массива FI.

Ky – переменная для вычисления коэфициента учета торможения.

algВivSegmentHalf (bool TypeCar, double &t1, double &t2, double &b, double &f, double &v, double &From, double &To, double &e) – процедура, отвечающая за вычисление уклона

CheckData()=0

true

Создание потока строкового типа и локальных переменных FCount=0, NUM, str, stream

1

0

Преобразование str в unicode строку и операция вставки данных в строковый поток unicodeunicode

Создание динамического массива FI

Пока stream не достиг конца

Операция вставки данных в NUM из потока stream и Fcount = Fcount + 1

1

0

Длинна FI = Fcount, сбрасываем все флаги состояния stream, перемещение указателя потока stream в позицию 0

i = 0 ; i меньше длины массива FI ; i = i + 1

Загрузка данных в массив FI из потока stream

0

1

Вычисление значения ColCount в BrakDistGrid ColCount = Fcount

Создание локальных переменных и загрузка данных из однострочного редактора StartMean, FinishMean, StMean

Создание CountSpeed = (FinishMean – StartMean)/ StMean и вычисление количества строк в BrakDistGrid, RowCount = CountSpeed +1

i = 1 ; i меньше CountSpeed+1 ; i = i + 1

Заполнение значений скорости для таблицы BrakDistGrid BrakDistGrid разадресация Сells[0][i] = StartMean

StartMean>FinishMean

StartMean снова заполняем из однострочного редактора, получая начальную скорость

BrakDistGrid разадресация Сells[0][0] = V0\\f

Присваиваем свойствам таблицы VslopeGrid требуемые значения RowCount =2, ColCount = Fcount + 1, Cells[0][0] = “f”, Cells[0][1]= “Уклон”

Создание локальных переменных TypeCar, ir, t1, t2, b, v, vF, vT и заполнение их из формы

i = 1 ; i <= Fcount ; i = i + 1

BrakDistGrid разадресация Cells[0][i] = FI[i-1]

1

0w

Создание локальных переменных f = FI[i-1], Ky

TypeCar = 1

Ky =1,07\*b-1,26(f-0,3)+(0,1-0,21\* (f-0,3))\*ir

Ky =1,3\*b-1,2(f-0,3)+(0,08-0,19\* (f-0,3))\*ir

j=1; j<CountSpeed; j= j+1

BreakDistGrid разадресация Cells [i][j] = EdMean\*(t1+t2)-gr/(1+b)\*(f\*pow(t2,2)/2- ir\*pow((t1+t2),2)/2)+((EdMean-gr/(1+b))\*((0.5\*f-ir)\*t2-t1)\*(1+b)\*Ky)/(2\*gr\*(f-ir))

StartMean = StartMean + StMean

Создание локальной переменной e = 0,01, Заполнение первой строки VslopeGrid коэфициентами φ

0

1

algВivSegmentHalf (TypeCar, t1, t2, b, f, v, vF, vT, e)

VS=Минимальномальному числу вещественного типа

Создание локальной переменной VS =algВivSegmentHalf (TypeCar, t1, t2, b, f, v, vF, vT, e)

VslopeGrid заполнение 2 строки, уклона значением переменной VS

VslopeGrid заполнение 2 строки, уклона значением “-”

# 7.4 Блок-схема алгоритма метода bool \_\_fastcall TMainForm::SaveData ()

Создание потока строкового типа и локальных переменных FCount=0, NUM, str, stream

Преобразование str в unicode строку и операция вставки данных в строковый потокunicodeunicode

Создание динамического массива FI

Пока stream не достиг конца

Операция вставки данных в NUM из потока stream и Fcount = Fcount + 1

1

0

сбрасываем все флаги состояния stream, перемещение указателя потока stream в позицию 0

Длина FI = FCount

i = 0 ; i меньше длины массива FI ; i = i + 1

Загрузка данных в массив FI из потока stream

Подсчет количества байт в файле

Создание переменной вещественного типа и присваивание ей DBL\_MIN при условии пустого поля

Открытие файла

1

0

Если файл не открыт

Вывод сообщения “Не удалось создать файл”

Запись всех данных в файл

Закрываем диологовое окно

# **8 Список литературы**

* 1. Бьерн Страуструп. Язык программирования С++. Специальное издание. — М.: Бином, 2011. — 1136 с.
  2. Б. Керниган, Д. Ритчи. Язык программирования Си. 2-е издание — М.: Вильямс, 2009. — 304 с.
  3. Шилдт Г. С++: базовый курс, 3-е издание.: Пер. с англ. — М.: Издательский дом "Вильямс", 2010. — 624 с.
  4. Х.М. Дейтел, П.Дж. Дейтел. Как программировать на С++: Пятое издание. Пер. с англ. — М: ООО «Бином-Пресс», 2008. — 1456 с.
  5. Прата С. Язык программирования С++. Лекции и упражнения, 6-е изд.: Пер. с англ. — М.: ООО "И.Д. Вильямс", 2012. — 1248 с.
  6. Липпман С, Лажойе Ж., My Б. Язык программирования С++. Базовый курс, 5-е изд.: Пер. с англ. — М.: ООО "И.Д. Вильямс", 2014. — 1120 с.
  7. А.Я. Архангельский. Программирование в С++ Builder. 7-е издание. — М.: «Бином-Пресс», 2010. — 896 с. (1230 с.)
  8. Бобровский СИ. Технологии С++ Builder. Разработка приложений для бизнеса. Учебный курс. — СПб.: Питер, 2007. — 560 с.
  9. Пахомов Б.И. Самоучитель C/C++ и С++ Builder 2007. — СПб.: БХВ-Петербург, 2008. — 672 с.
  10. Липпман С. Весь С++ от азов до совершенства. — СПб.: ДМК Пресс, 2007. — 1104 с.
  11. Г. Буч. Объектно-ориентированный анализ и проектирование с примерами приложений на С++. — М.: Вильямс, 2008. — 721 с.
  12. Культин Н.Б. С++ Builder. — 2-е изд., перераб. и доп. — СПб.: БХВ-Петербург, 2008. — 464 с.
  13. Онлайн справочная система библиотеки MSDN по языку С++ и стандартным библиотекам:

https://msdn.microsoft.com/ru-ru/library/hh875057.aspx

* 1. Онлайн документация по последней и предыдущим версиям среды разработки Embarcadero С++ Builder:

http://docwiki.embarcadero.com/RADStudio/Seattle/en/Main\_Page

# **9 Текст программы с комментариями**

#ifndef Unit1H

#define Unit1H

//---------------------------------------------------------------------------

#include <System.Classes.hpp>

#include <Vcl.Controls.hpp>

#include <Vcl.StdCtrls.hpp>

#include <Vcl.Forms.hpp>

#include <Vcl.Grids.hpp>

#include <Vcl.Menus.hpp>

#include <Vcl.Dialogs.hpp>

#include <Vcl.ExtCtrls.hpp>

#include <Vcl.ComCtrls.hpp>

//---------------------------------------------------------------------------

#define DefFileName L"Безымянный" //значение по умолчанию

#define gr 9.81

class TMainForm : public TForm

{

\_\_published: // IDE-managed Components

TMainMenu \*MainMenu1;

TMenuItem \*N1;

TMenuItem \*N2;

TMenuItem \*N3;

TSaveDialog \*SaveFileDialog;

TOpenDialog \*OpenDialog1;

TMenuItem \*N4;

TMenuItem \*MOpenFile;

TMenuItem \*MSaveFile;

TMenuItem \*N8;

TMenuItem \*N9;

TMenuItem \*N10;

TMenuItem \*N7;

TGroupBox \*GroupBox1;

TLabel \*Label14;

TLabel \*Label15;

TEdit \*Slope;

TLabel \*Label16;

TEdit \*Edit2;

TLabel \*Label17;

TEdit \*Edit3;

TLabel \*Label18;

TEdit \*Edit4;

TLabel \*Label19;

TEdit \*TimeFirstBrakPhase;

TLabel \*Label20;

TEdit \*TimeSecBrakPhase;

TLinkLabel \*LinkLabel1;

TEdit \*InertCoefRotatMas;

TLabel \*Label21;

TEdit \*BrakDistFrom;

TLabel \*Label22;

TEdit \*BrakDistTo;

TLinkLabel \*LinkLabel2;

TEdit \*VSlope;

TCheckBox \*CheckTypeCar;

TButton \*ProcesData;

TEdit \*Edit11;

TStatusBar \*StatBarInfo;

TGroupBox \*GroupBox2;

TStringGrid \*BrakDistGrid;

TLabel \*Label1;

TLabel \*Label2;

TStringGrid \*VSlopeGrid;

TMenuItem \*N5;

void \_\_fastcall FormCreate(TObject \*Sender);

void \_\_fastcall MCreateFileClick(TObject \*Sender);

void \_\_fastcall MOpenFileClick(TObject \*Sender);

void \_\_fastcall ProcesDataClick(TObject \*Sender);

void \_\_fastcall Exit(TObject \*Sender);

void \_\_fastcall informationClick(TObject \*Sender);

void \_\_fastcall MSaveFileClick(TObject \*Sender);

void \_\_fastcall Edit2KeyPress(TObject \*Sender, System::WideChar &Key);

void \_\_fastcall CheckTypeCarClick(TObject \*Sender);

void \_\_fastcall SaveASClick(TObject \*Sender);

void \_\_fastcall FormResize(TObject \*Sender);

// void \_\_fastcall MAddColDataGridClick(TObject \*Sender);

// void \_\_fastcall MDelColDataGridClick(TObject \*Sender);

//void \_\_fastcall TimeFirstBrakPhaseChange(TObject \*Sender);

private: // User declarations

bool \_\_fastcall CheckData (bool ToSave = false);

bool \_\_fastcall SaveData ();

void \_\_fastcall ClearOldResults();

public: // User declarations

\_\_fastcall TMainForm(TComponent\* Owner);

};

//---------------------------------------------------------------------------

extern PACKAGE TMainForm \*MainForm;

//---------------------------------------------------------------------------

#endif

#include <vcl.h>

#pragma hdrstop

#include "Unit1.h"

//---------------------------------------------------------------------------

#pragma package(smart\_init)

#pragma resource "\*.dfm"

TMainForm \*MainForm;

//---------------------------------------------------------------------------

#include <sstream>

#include <string.h>

#include <cstring>

#include <fstream>

using namespace std;

\_\_fastcall TMainForm::TMainForm(TComponent\* Owner)

: TForm(Owner)

{

}

//---------------------------------------------------------------------------

void \_\_fastcall TMainForm::FormCreate(TObject \*Sender)

{

MainForm->Caption=String(Application->Title)+L" - "+DefFileName;

}

//---------------------------------------------------------------------------

void \_\_fastcall TMainForm::MCreateFileClick(TObject \*Sender)

{

if (MSaveFile->Enabled)

{

String FileName=SaveFileDialog->FileName;

if (FileName==L"")

{

FileName=DefFileName;

}

switch (Application->MessageBox((String(L"Присутствуют не сохраненые изменения. Сохранить изменения в файле \"")

+FileName+"\"?").w\_str(),Application->Title.w\_str(), MB\_YESNOCANCEL|MB\_ICONQUESTION))

{

case mrCancel:

return;

case mrNo:

break;

case mrYes:

if (!CheckData(true))

return;

if (SaveFileDialog->FileName==L"") {

SaveFileDialog->FileName=DefFileName;

if (!SaveFileDialog->Execute()) {

if (SaveFileDialog->FileName==DefFileName) {

SaveFileDialog->FileName=L"";

}

return;

}

}

if (!SaveData())

return;

break;

}

}

//выполнить зачистку старых данных

ClearOldResults();

//for (int i=1; i < DataGrid->ColCount; i++)

//DataGrid->Cells[i][1]=L"";

//DataGrid->ColCount=2;

Edit11->Text=L"";

Edit2->Text=L"";

Edit3->Text=L"";

Edit4->Text=L"";

Slope->Text=L"";

TimeFirstBrakPhase->Text=L"";

TimeSecBrakPhase->Text=L"";

InertCoefRotatMas->Text=L"";

BrakDistFrom->Text=L"";

BrakDistTo->Text=L"";

VSlope->Text=L"";

//MDelColDataGrid->Enabled = false;

Caption=String(Application->Title)+L" - "+ DefFileName;

MSaveFile->Enabled = false;

CheckTypeCar->Checked=false;

}

//---------------------------------------------------------------------------

//---------------------------------------------------------------------------

bool \_\_fastcall TMainForm::CheckData (bool ToSave) // проверка файла

{

bool ErrorF = false;

String ErrMess;

double TempValue;

bool ErrorFEdit = false;

if (!(Edit2->Text==L""&&ToSave) && (!TryStrToFloat(Edit2->Text, TempValue)||TempValue<0))

{

if (ErrorF)

ErrMess+=L". Некорректные или недопустимое значение: cкорость перед торможением от";

else

{

ErrorF=true;

ErrMess=L"Некорректные или недопустимое значение: cкорость перед торможением от";

}

ErrorFEdit = true;

}

if (!(Edit3->Text==L""&&ToSave) && (!TryStrToFloat(Edit3->Text, TempValue)||TempValue<0)) {

if (ErrorFEdit)

ErrMess+=L"; cкорость перед торможением до";

else {

if (ErrorF)

ErrMess+=L". Некорректные или недопустимое значение: cкорость перед торможением до";

else {

ErrorF=true;

ErrMess=L"Некорректные или недопустимое значение: cкорость перед торможением до";

}

ErrorFEdit = true;

}

}

if (!(Edit4->Text==L""&&ToSave) && (!TryStrToFloat(Edit4->Text, TempValue)||TempValue<0)) {

if (ErrorFEdit)

ErrMess+=L"; cкорость перед торможением, шаг";

else {

if (ErrorF)

ErrMess+=L". Некорректные или недопустимое значение: cкорость перед торможением, шаг";

else {

ErrorF=true;

ErrMess=L"Некорректные или недопустимое значение: cкорость перед торможением, шаг";

}

ErrorFEdit = true;

}

}

if (!(Slope->Text==L""&&ToSave) && (!TryStrToFloat(Slope->Text, TempValue)||TempValue<0)) {

if (ErrorFEdit)

ErrMess+=L"; уклона";

else {

if (ErrorF)

ErrMess+=L". Некорректные или недопустимое значение: уклона";

else {

ErrorF=true;

ErrMess=L"Некорректные или недопустимое значение: уклона";

}

ErrorFEdit = true;

}

}

if (!(TimeFirstBrakPhase->Text==L""&&ToSave) && (!TryStrToFloat(TimeFirstBrakPhase->Text, TempValue)||TempValue<0)) {

if (ErrorFEdit)

ErrMess+=L"; время первой фазы торможения";

else {

if (ErrorF)

ErrMess+=L". Некорректные или недопустимое значение: время первой фазы торможения";

else {

ErrorF=true;

ErrMess=L"Некорректные или недопустимое значение: время первой фазы торможения";

}

ErrorFEdit = true;

}

}

if (!(TimeSecBrakPhase->Text==L""&&ToSave) && (!TryStrToFloat(TimeSecBrakPhase->Text, TempValue)||TempValue<0)) {

if (ErrorFEdit)

ErrMess+=L"; время второй фазы торможения";

else {

if (ErrorF)

ErrMess+=L". Некорректные или недопустимое значение: время второй фазы торможения";

else {

ErrorF=true;

ErrMess=L"Некорректные или недопустимое значение: время второй фазы торможения";

}

ErrorFEdit = true;

}

}

if (!(InertCoefRotatMas->Text==L""&&ToSave) && (!TryStrToFloat(InertCoefRotatMas->Text, TempValue)||TempValue<0)) {

if (ErrorFEdit)

ErrMess+=L"; коэффициент инерции вращающихся масс";

else {

if (ErrorF)

ErrMess+=L". Некорректные или недопустимое значение: коэффициент инерции вращающихся масс";

else {

ErrorF=true;

ErrMess=L"Некорректные или недопустимое значение: коэффициент инерции вращающихся масс";

}

ErrorFEdit = true;

}

}

if (!(BrakDistFrom->Text==L""&&ToSave) && (!TryStrToFloat(BrakDistFrom->Text, TempValue)||TempValue<0)) {

if (ErrorFEdit)

ErrMess+=L"; длина тормозного пути от";

else {

if (ErrorF)

ErrMess+=L". Некорректные или недопустимое значение: длина тормозного пути от";

else {

ErrorF=true;

ErrMess=L"Некорректные или недопустимое значение: длина тормозного пути от";

}

ErrorFEdit = true;

}

}

if (!(BrakDistTo->Text==L""&&ToSave) && (!TryStrToFloat(BrakDistTo->Text, TempValue)||TempValue<0)) {

if (ErrorFEdit)

ErrMess+=L"; длина тормозного пути до";

else {

if (ErrorF)

ErrMess+=L". Некорректные или недопустимое значение: длина тормозного пути до";

else {

ErrorF=true;

ErrMess=L"Некорректные или недопустимое значение: длина тормозного пути до";

}

ErrorFEdit = true;

}

}

if (!(VSlope->Text==L""&&ToSave) && (!TryStrToFloat(VSlope->Text, TempValue)||TempValue<0)) {

if (ErrorFEdit)

ErrMess+=L"; скорость для расчета уклона";

else {

if (ErrorF)

ErrMess+=L". Некорректные или недопустимое значение: скорость для расчета уклона";

else {

ErrorF=true;

ErrMess=L"Некорректные или недопустимое значение: скорость для расчета уклона";

}

ErrorFEdit = true;

}

}

if (ErrorF) {

ErrMess+=L".";

StatBarInfo->SimpleText=ErrMess;

Application->MessageBox(ErrMess.w\_str(),Application->Title.w\_str(), MB\_OK|MB\_ICONEXCLAMATION);

return false;

}

return true;

}

void \_\_fastcall TMainForm::Exit(TObject \*Sender)

{

Close();

}

void \_\_fastcall TMainForm::MOpenFileClick(TObject \*Sender)

{

if (MSaveFile->Enabled)

{

String FileName=SaveFileDialog->FileName;

if (FileName==L"")

{

FileName=DefFileName;

}

switch (Application->MessageBox((String(L"Присутствуют не сохраненые изменения. Сохранить изменения в файле \"")+

FileName+"\"?").w\_str(),Application->Title.w\_str(), MB\_YESNOCANCEL|MB\_ICONQUESTION))

{

case mrCancel:

return;

case mrNo:

break;

case mrYes:

if (!CheckData(true))

return;

if (SaveFileDialog->FileName==L"") {

SaveFileDialog->FileName=DefFileName;

if (!SaveFileDialog->Execute()) {

if (SaveFileDialog->FileName==DefFileName) {

SaveFileDialog->FileName=L"";

}

return;

}

}

if (!SaveData())

return;

break;

}

}

if (!OpenDialog1->Execute())

return;

unsigned int ByteCount;

fstream file;

file.open(AnsiString(OpenDialog1->FileName).c\_str(), ios::in | ios::binary);

if (!file) {

Application->MessageBox((String(L"Не удалось открыть файл \"")+OpenDialog1->FileName+"\"!").w\_str(),Application->Title.w\_str(), MB\_OK|MB\_ICONERROR);

return;

}

file.read((char\*)&ByteCount, sizeof(ByteCount));

file.seekg(0, ios::end);

file.clear();

if (ByteCount != file.tellg()){

file.close();

Application->MessageBox((String(L"Не удалось загрузить данные из файла \"")+OpenDialog1->FileName+"\". Файл поврежден.").w\_str(), Application->Title.w\_str(), MB\_OK|MB\_ICONERROR);

return;

}

file.clear();

file.seekg(sizeof(ByteCount));

double DataEdit;

file.read((char\*)&DataEdit, sizeof(DataEdit));

if (DataEdit==DBL\_MIN)

Edit2->Text=L"";

else

Edit2->Text=DataEdit;

file.read((char\*)&DataEdit, sizeof(DataEdit));

if (DataEdit==DBL\_MIN)

Edit3->Text=L"";

else

Edit3->Text=DataEdit;

file.read((char\*)&DataEdit, sizeof(DataEdit));

if (DataEdit==DBL\_MIN)

Edit4->Text=L"";

else

Edit4->Text=DataEdit;

file.read((char\*)&DataEdit, sizeof(DataEdit));

if (DataEdit==DBL\_MIN)

Slope->Text=L"";

else

Slope->Text=DataEdit;

file.read((char\*)&DataEdit, sizeof(DataEdit));

if (DataEdit==DBL\_MIN)

TimeFirstBrakPhase->Text=L"";

else

TimeFirstBrakPhase->Text=DataEdit;

file.read((char\*)&DataEdit, sizeof(DataEdit));

if (DataEdit==DBL\_MIN)

TimeSecBrakPhase->Text=L"";

else

TimeSecBrakPhase->Text=DataEdit;

file.read((char\*)&DataEdit, sizeof(DataEdit));

if (DataEdit==DBL\_MIN)

InertCoefRotatMas->Text=L"";

else

InertCoefRotatMas->Text=DataEdit;

file.read((char\*)&DataEdit, sizeof(DataEdit));

if (DataEdit==DBL\_MIN)

BrakDistFrom->Text=L"";

else

BrakDistFrom->Text=DataEdit;

file.read((char\*)&DataEdit, sizeof(DataEdit));

if (DataEdit==DBL\_MIN)

BrakDistTo->Text=L"";

else

BrakDistTo->Text=DataEdit;

file.read((char\*)&DataEdit, sizeof(DataEdit));

if (DataEdit==DBL\_MIN)

VSlope->Text=L"";

else

VSlope->Text=DataEdit;

bool TypeCar;

file.read((char\*)&TypeCar, sizeof(TypeCar));

CheckTypeCar->Checked=TypeCar;

int ColCount;

file.read((char\*)&ColCount, sizeof(ColCount));

DynamicArray<Double> DGrid;

try

{

DGrid.Length=ColCount;

} catch (...)

{

DGrid.Length=0;

file.close();

Application->MessageBox(L"Не хватает оперативной памяти для загрузки данных.",Application->Title.w\_str(), MB\_OK|MB\_ICONEXCLAMATION);

return;

}

file.read((char\*)&DGrid[0], sizeof(DGrid[0])\*ColCount);

file.close();

stringstream stream;

String str = "";

for (int i=0; i < ColCount; i++)

{

str=str + FloatToStr(DGrid[i])+" " ;

}

Edit11->Text = str ;

ClearOldResults();

SaveFileDialog->FileName=OpenDialog1->FileName;

MainForm->Caption=String(Application->Title)+L" - "+ExtractFileName(SaveFileDialog->FileName);

}

//---------------------------------------------------------------------------

double functionS (bool TypeCar, double &ir, double &t1, double &t2, double &b, double &f, double &v,double &Sp) {

double Ky;

if (TypeCar)

Ky=1.07\*b-1.26\*(f-0.3)+(0.1-0.21\*(f-0.3))\*ir;

else

Ky=1.3\*b-1.2\*(f-0.3)+(0.08+0.19\*(f-0.3))\*ir;

return (v\*(t1+t2)-gr/(1+b)\*(f\*pow(t2,2)/2-ir\*pow((t1+t2),2)/2)+((v-gr/(1+b))\*((0.5\*f-ir)\*t2-t1)\*(1+b)\*Ky)/(2\*gr\*(f-ir)))-Sp;

}

//---------------------------------------------------------------------------

double algВivSegmentHalf (bool TypeCar, double &t1, double &t2, double &b, double &f, double &v, double &From, double &To, double &e) {

double il=-90, ir=90;

double Sp=(To-From)/2.0;

do {

double f1=functionS(TypeCar, il, t1, t2, b, f, v,Sp);

double HalfI=(il+ir)/2.0;

double f2= functionS(TypeCar, HalfI, t1, t2, b, f, v,Sp);

if (f1\*f2<0)

ir=HalfI;

else

il=HalfI;

}while (fabs(ir-il)>e);

double i=(ir+il)/2.0;

Sp=0;

double res=functionS(TypeCar, i, t1, t2, b, f, v,Sp);

if (res>From&&res<To)

return i;

else

return DBL\_MIN;

}

void \_\_fastcall TMainForm::ProcesDataClick(TObject \*Sender)

{

if (!CheckData())

return;

//подсчет количества значений по коэффицентам fi из поля edit

int FCount=0;

float NUM;

//int n;

stringstream stream; // Edit11->Text;

AnsiString str= Edit11->Text;

char \*str2 = new char[ str.Length() + 1 ] ;

strcpy (str2,str.c\_str());

stream<<str2;

DynamicArray<double> FI;

while(!stream.eof())

{

stream>>NUM;

FCount++;

}

//

FI.Length = FCount;

stream.clear();

stream.seekg(0);

for(int i = 0; i < FI.Length; i++)

stream>>FI[i];

BrakDistGrid-> ColCount= FCount+1;

//-------------------------------------------------------------------------

double StartMean=StrToFloat(Edit2->Text); //присвоение скоростей от, до, шаг

double FinishMean=StrToFloat(Edit3->Text);

double StMean=StrToFloat(Edit4->Text);

int CountSpeed=(FinishMean-StartMean)/StMean+1;//высчитываем кол-во строк

BrakDistGrid->RowCount=CountSpeed+1;

double EdMean=StartMean;

//-------------------------------------------------------------------------

for (int i=1; i < CountSpeed+1; i++) //заполняем 1 столбец скорости

{

BrakDistGrid->Cells[0][i]=EdMean;

EdMean+=StMean;

if (EdMean>FinishMean)

EdMean=FinishMean;

}

//------------------------------------------------------------------------

BrakDistGrid->Cells[0][0]=L"V0\\f";//продолжаем заполнение таблицы

VSlopeGrid->RowCount=2;

VSlopeGrid->ColCount=FCount+1;

VSlopeGrid->Cells[0][0]=L"f";

VSlopeGrid->Cells[0][1]=L"Уклон";

bool TypeCar=CheckTypeCar->Checked;

double ir =StrToFloat(Slope->Text);

double t1 = StrToFloat(TimeFirstBrakPhase->Text);

double t2 = StrToFloat(TimeSecBrakPhase->Text);

double b = StrToFloat(InertCoefRotatMas->Text);

double v = StrToFloat(VSlope->Text);

double vF=StrToFloat(BrakDistFrom->Text);

double vT=StrToFloat(BrakDistTo->Text);

for (int i=1; i <= FCount; i++)

{

BrakDistGrid->Cells[i][0] = FI[i-1];

EdMean=StartMean;

double f = FI[i-1];

double Ky;

if (TypeCar)

Ky=1.07\*b-1.26\*(f-0.3)+(0.1-0.21\*(f-0.3))\*ir;

else

Ky=1.3\*b-1.2\*(f-0.3)+(0.08+0.19\*(f-0.3))\*ir;

for (int j=1; j <= CountSpeed; j++) {

BrakDistGrid->Cells[i][j]=EdMean\*(t1+t2)-gr/(1+b)\*(f\*pow(t2,2)/2-

ir\*pow((t1+t2),2)/2)+((EdMean-gr/(1+b))\*

((0.5\*f-ir)\*t2-t1)\*(1+b)\*Ky)/(2\*gr\*(f-ir));

EdMean+=StMean;

if (EdMean>FinishMean)

EdMean=FinishMean;

}

double e=0.01;

VSlopeGrid->Cells[i][0]=f;

double VS=algВivSegmentHalf (TypeCar, t1, t2, b, f, v, vF, vT, e);

if (VS==DBL\_MIN)

VSlopeGrid->Cells[i][1]=L"-";

else

VSlopeGrid->Cells[i][1]=VS;

}

//SaveProcData->Enabled=true;

}

void \_\_fastcall TMainForm::ClearOldResults ()

{

BrakDistGrid->ColCount=1;

BrakDistGrid->RowCount=1;

BrakDistGrid->Cells[0][0]=L"";

VSlopeGrid->ColCount=1;

VSlopeGrid->RowCount=1;

VSlopeGrid->Cells[0][0]=L"";

// SaveProcData->Enabled=false;

}

void \_\_fastcall TMainForm::informationClick(TObject \*Sender)

{

ShowMessage("Создание программного приложения для расчета тормозного пути грузового автомобиля при заданных условиях движения. Проект разработан студентом группы 2бАСУ1 Московского автомобильно-дорожного государственного технического университета Никитиным Н.С. под руководством Баринова К.А.");

// доб. удаление результатов

}

//---------------------------------------------------------------------------

bool \_\_fastcall TMainForm::SaveData ()

{

DynamicArray<Double> DGrid;

// int ColCount = DataGrid->ColCount-1;

int ColCount = 0;

int NUM;

stringstream stream; // Edit11->Text;

AnsiString str= Edit11->Text;

char \*str2 = new char[ str.Length() + 1 ] ;

strcpy (str2,str.c\_str());

stream<<str2;

while(!stream.eof())

{

stream>>NUM;

ColCount++;

}

try

{

DGrid.Length=ColCount;

}

catch (...)

{

DGrid.Length=0;

Application->MessageBox(L"Не хватает оперативной памяти, для сохранения данных!"

,Application->Title.w\_str(),MB\_OK|MB\_ICONEXCLAMATION);

return false;

}

stream.clear();

stream.seekg(0);

for (int i=0; i < ColCount; i++)

{

stream>>DGrid[i];

}

double DataEdit;

bool TypeCar = CheckTypeCar->Checked;

unsigned int CountByte = sizeof(ColCount)+ColCount\*sizeof(DGrid[0])+sizeof(CountByte)+sizeof(DataEdit)\*10+sizeof(TypeCar);

fstream file;

file.open(AnsiString(SaveFileDialog->FileName).c\_str(), ios::out | ios::binary);

if (!file)

{

Application->MessageBox((String(L"Не удалось создать файл \"")

+SaveFileDialog->FileName+"\"!").w\_str(),Application->Title.w\_str(),MB\_OK|MB\_ICONERROR);

return false;

}

file.write((char\*)&CountByte, sizeof(CountByte));

if (Edit2->Text==L"")

DataEdit=DBL\_MIN;

else

TryStrToFloat(Edit2->Text, DataEdit);

file.write((char\*)&DataEdit, sizeof(DataEdit));

if (Edit3->Text==L"")

DataEdit=DBL\_MIN;

else

TryStrToFloat(Edit3->Text, DataEdit);

file.write((char\*)&DataEdit, sizeof(DataEdit));

if (Edit4->Text==L"")

DataEdit=DBL\_MIN;

else

TryStrToFloat(Edit4->Text, DataEdit);

file.write((char\*)&DataEdit, sizeof(DataEdit));

if (Slope->Text==L"")

DataEdit=DBL\_MIN;

else

TryStrToFloat(Slope->Text, DataEdit);

file.write((char\*)&DataEdit, sizeof(DataEdit));

if (TimeFirstBrakPhase->Text==L"")

DataEdit=DBL\_MIN;

else

TryStrToFloat(TimeFirstBrakPhase->Text, DataEdit);

file.write((char\*)&DataEdit, sizeof(DataEdit));

if (TimeFirstBrakPhase->Text==L"")

DataEdit=DBL\_MIN;

else

TryStrToFloat(TimeFirstBrakPhase->Text, DataEdit);

file.write((char\*)&DataEdit, sizeof(DataEdit));

if (InertCoefRotatMas->Text==L"")

DataEdit=DBL\_MIN;

else

TryStrToFloat(InertCoefRotatMas->Text, DataEdit);

file.write((char\*)&DataEdit, sizeof(DataEdit));

if (BrakDistFrom->Text==L"")

DataEdit=DBL\_MIN;

else

TryStrToFloat(BrakDistFrom->Text, DataEdit);

file.write((char\*)&DataEdit, sizeof(DataEdit));

if (BrakDistTo->Text==L"")

DataEdit=DBL\_MIN;

else

TryStrToFloat(BrakDistTo->Text, DataEdit);

file.write((char\*)&DataEdit, sizeof(DataEdit));

if (VSlope->Text==L"")

DataEdit=DBL\_MIN;

else

TryStrToFloat(VSlope->Text, DataEdit);

file.write((char\*)&DataEdit, sizeof(DataEdit));

file.write((char\*)&TypeCar, sizeof(TypeCar));

file.write((char\*)&ColCount, sizeof(ColCount));

file.write((char\*)&DGrid[0], sizeof(DGrid[0])\*ColCount);

DGrid.Length=0;

file.close();

MSaveFile->Enabled = false;

return true;

}

void \_\_fastcall TMainForm::MSaveFileClick(TObject \*Sender)

{

if (!CheckData(true))

return;

if (SaveFileDialog -> FileName==L"")

{

SaveFileDialog->FileName=DefFileName;

if (!SaveFileDialog->Execute()) {

if (SaveFileDialog->FileName==DefFileName) {

SaveFileDialog->FileName=L"";

}

return;

}

}

if (!SaveData())

return;

MainForm->Caption=String(Application->Title)+L" - "+ExtractFileName(SaveFileDialog->FileName);

}

//---------------------------------------------------------------------------

void \_\_fastcall TMainForm::Edit2KeyPress(TObject \*Sender, System::WideChar &Key)

{

if (!MSaveFile->Enabled) {

MSaveFile->Enabled= true;

MainForm->Caption+=L"\*";

}

if (!((Key >= '0' && Key <= '9') || (int)Key == 22 || (int)Key == 13 || (int)Key == 24 || (int)Key == 26||(int)Key == 8|| (int)Key == 3 || (int)Key == 44)) {//проверка коррекности введеного символа

StatBarInfo -> SimpleText=String(L"Введен недопустимый символ \"") + Key + L"\"";

Key = 0;

return;

}

StatBarInfo -> SimpleText = L"";

}

//---------------------------------------------------------------------------

void \_\_fastcall TMainForm::CheckTypeCarClick(TObject \*Sender)

{

if (!MSaveFile->Enabled) {

MSaveFile->Enabled= true;

MainForm->Caption+=L"\*";

}

}

//---------------------------------------------------------------------------

void \_\_fastcall TMainForm::SaveASClick(TObject \*Sender)

{

if (!CheckData(true))

return;

String OldFileName = SaveFileDialog->FileName;

SaveFileDialog->FileName=DefFileName;

if (!SaveFileDialog->Execute()) {

SaveFileDialog->FileName = OldFileName;

return;

}

if (!SaveData()) {

SaveFileDialog->FileName=OldFileName;

return;

}

MainForm->Caption=String(Application->Title)+L" - "+ExtractFileName(SaveFileDialog->FileName);

}

//---------------------------------------------------------------------------