МОСКОВСКИЙ АВТОМОБИЛЬНО-ДОРОЖНЫЙ

ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ (МАДИ)

Кафедра

«Автоматизированные системы управления»

Курсовая работа по дисциплине

«Программирование»

Вариант № 24

Тема:

«Разработка программного приложения для определения пропускной способности проезжей части на перегоне»

Выполнил: студент группы 2бАСУ1

Халилов Р.Э.

Проверил: доцент каф. АСУ

Баринов К.А.

Москва 2023

**Оглавление**

[**1. Задание** 4](#_Toc156203258)

[**2. Назначение программы** 5](#_Toc156203259)

[**3. Руководство пользователя** 6](#_Toc156203260)

[**4. Выбор среды разработки** 16](#_Toc156203261)

[**5. Форматы данных** 17](#_Toc156203262)

[**5.1.Входные данные** 17](#_Toc156203263)

[**5.2 Выходные данные** 18](#_Toc156203264)

[**6. Структура программы** 19](#_Toc156203265)

[**6.1. Модуль Main** 19](#_Toc156203266)

[**6.2. Вспомогательные типы данных** 19](#_Toc156203267)

[**6.3. Константы** 19](#_Toc156203268)

[**6.4. Класс оконной формы Main** 19](#_Toc156203269)

[**6.5. Функции** 23](#_Toc156203270)

[**6.6. Переменные** 27](#_Toc156203271)

[**7. Описание алгоритмов** 28](#_Toc156203272)

[**7.1. Блок-схема укрупненного алгоритма работы пользовательского интерфейса приложения** 28](#_Toc156203273)

[**7.2. Блок-схема алгоритма метода выбора пункта меню “Открыть”** 29](#_Toc156203274)

[**[void \_\_fastcall TTForm1::dowloadClick(TObject \*Sender)]** 29](#_Toc156203275)

[**7.3. Блок-схема алгоритма метода выбора пункта меню “Обработать”** 30](#_Toc156203276)

[**[void \_\_fastcall TTForm1::bProcClick(TObject \*Sender)]** 30](#_Toc156203277)

[**7.4. Блок-схема алгоритма метода выбора пункта меню “Сохранить таблицу как …”** 31](#_Toc156203278)

[**[void \_\_fastcall TTForm1::savegridClick(TObject \*Sender)]** 31](#_Toc156203279)

[**7.5. Блок-схема алгоритма метода выбора пункта меню “Сохранить график как…”** 32](#_Toc156203280)

[**[void \_\_fastcall TTForm1::SaveChartClick(TObject \*Sender)]** 32](#_Toc156203281)

[**7.6. Блок-схема алгоритма метода выбора пункта меню “Вставить столбец”** 33](#_Toc156203282)

[**[void \_\_fastcall TTForm1::bInsColClick(TObject \*Sender)]** 33](#_Toc156203283)

[**8. Текст программы с комментариями** 34](#_Toc156203284)

# **1. Задание**

Вариант №24

Пропускная способность проезжей части улицы на перегоне определяется по формуле:

, где

*k* - коэффициент плотности;

*N* - пропускная способность одной полосы (авт/час);

*n* - число полос движения.

Величина *N* определятся из условий безопасности движения по формуле:

 , где

v - скорость (м/с);

а - расстояние между остановившимися транспортными средствами в метрах; t - промежуток времени между началом торможений двух последовательно движущихся автомашин в сек;

С - коэффициент торможения (сек2/м);

l - эмпирический коэффициент;

t=4 сек.

**Задания:**

1. Определить N для следующих наборов коэффициентов а и l

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| a | 4.2 | 5.0 | 5.4 | 6.0 |
| l | 4 | 5.0 | 5.0 | 5.5 |

при С=0.062; 0.078; 0.083 и изменении v от 30 до 60 с шагом 5. Результаты вывести в виде 3-х таблиц с соответствующими надписями.

2. Произвести расчет изменения Q=f(v) при фиксированном значении С, a=4.2; l=4; n=2; k=0.8.

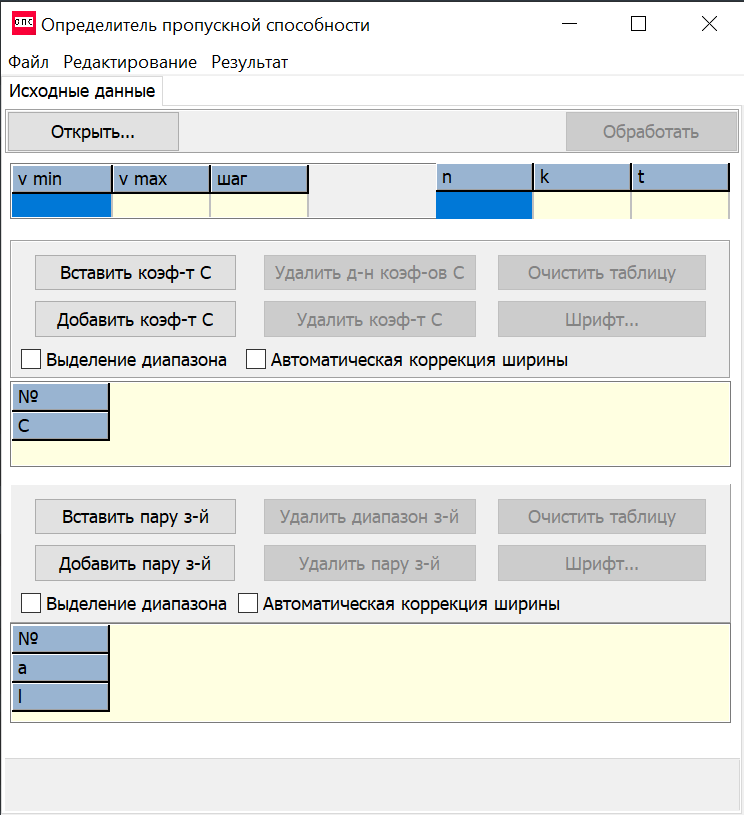
3. Вывести результаты в виде графиков.

# **2. Назначение программы**

Разрабатываемое приложение предназначено для определения пропускной способности проезжей части на перегоне. Оно учитывает различные параметры и условия движения.

# **3. Руководство пользователя**

При запуске программы, интерфейс приложения представляет собой следующее (рис.1):



*Рисунок 1 (Вид приложения при запуске)*

Для обработки данных пользователь может внести данные двумя способами:

1. Загрузить данные из файла нажав на кнопку “Открыть…”, при нажатии откроется диалоговое окно открытия (рис.2):

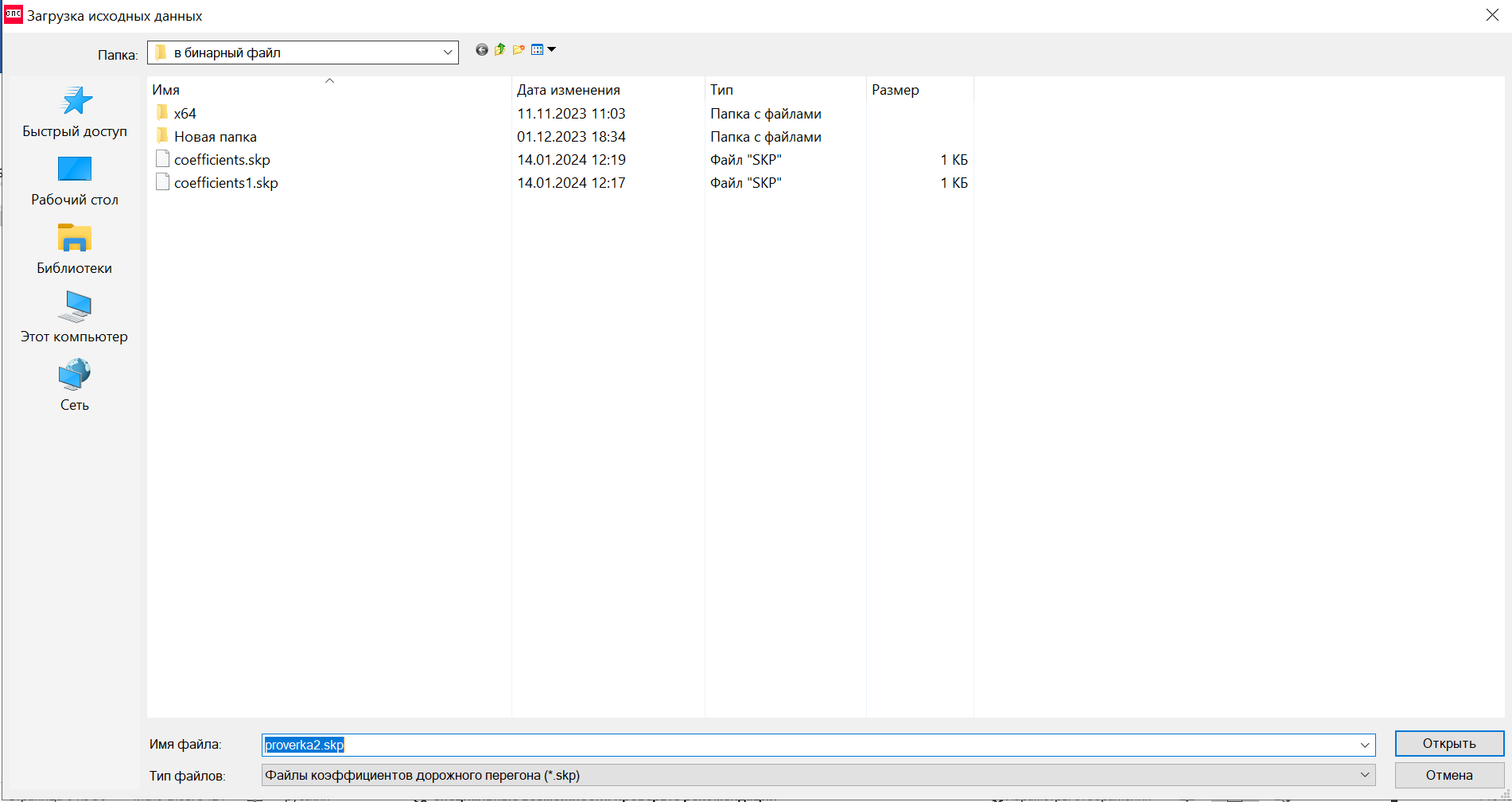


Рисунок 2 (Диалоговое окно открытия)

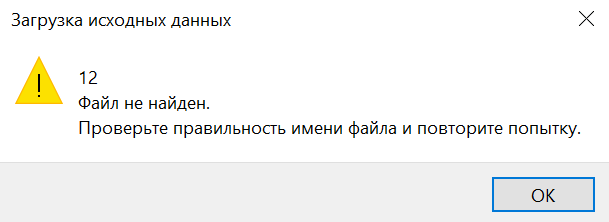
Пользователь может выбрать файл, либо ввести его имя. Затем необходимо нажать “Открыть”. Если файла не существует (рис.3):  
 

Рисунок 3 (Окно если загружаемого файла не существует)

1. Ввести данные вручную, ввод будет контролироваться(рис.4):



Рисунок 4 (Сообщение о недопустимом символе)

Пользователь может настраивать таблицу, нажав на соответствующие кнопки, а именно:

* Добавить столбец;
* Удалить диапазон;
* Настроить шрифт;
* Включить режим “Выделение диапазона”;
* Включить режим “Автоматическое коррекция ширины”;
* Самостоятельно настроить ширину, двигая за границы столбцов (при выключенном “Автоматическое коррекция ширины”);

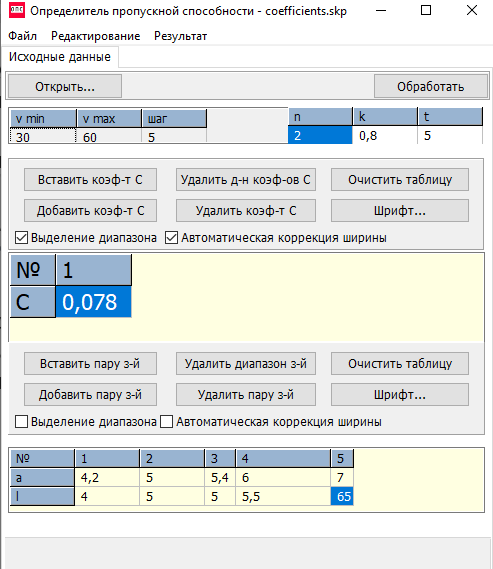


Рисунок 5 (Пример настроенного ввода данных)

Будет выведено сообщение при удалении диапазона столбцов (рис.6) или очистки таблицы (рис.7):

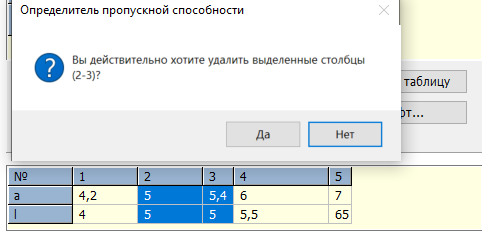


Рисунок 6 (Предупреждение об удалении диапазона)

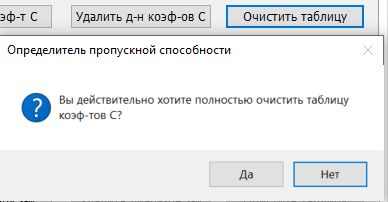


Рисунок 7 (Предупреждение о полной очистки таблицы)

Если же пользователь вставить неправильное значение в ячейку, то она выделиться красным (рис.8):

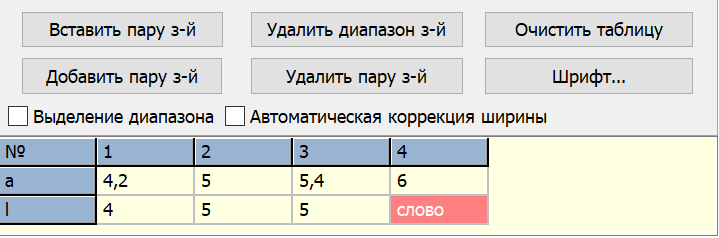


Рисунок 8(Ввод некорректного значения)

Кнопка “Обработать” не будет доступна пока данные для неё не будут корректны и допустимы. При нажатии на кнопку пользователь перемещается на вкладку “Результаты”, где в табличном редакторе будут выведены обработанные значения(рис.9):

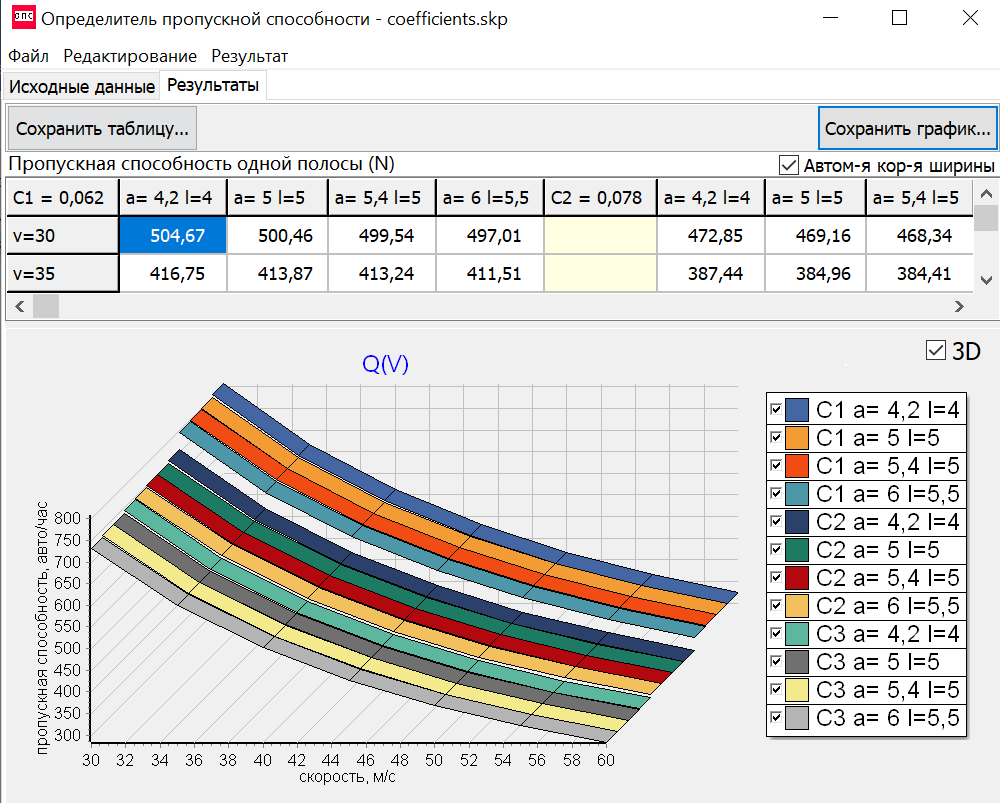


Рисунок 9(Результаты)

На этой вкладке пользователь может:

* Выключить “Автоматическое коррекция ширины” и настроить самостоятельно;
* Выключить тип 3D для линий графика;
* Настраивать необходимый размер графика и таблицы

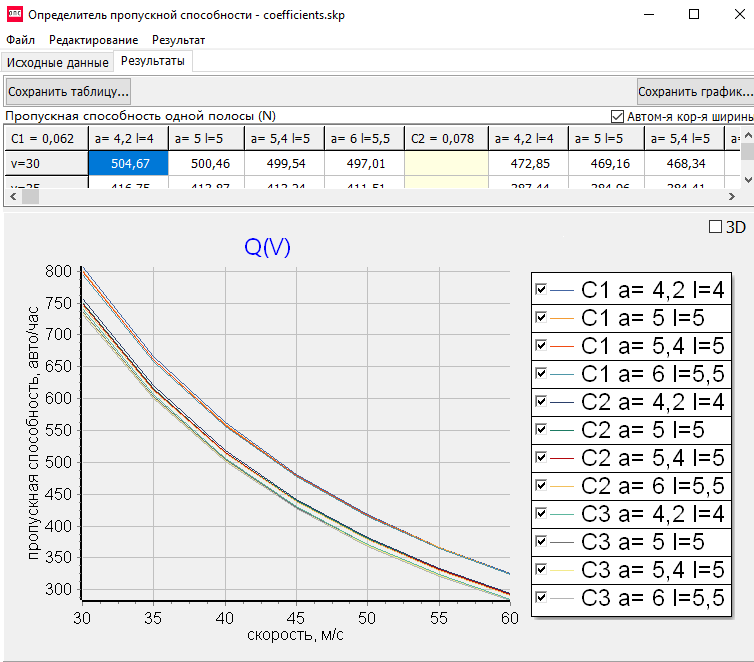


Рисунок 10 (Пример настроенной вкладки “Результаты”)

Для сохранения исходных данных пользователь может нажать во вкладке меню “Сохранить” или “Сохранить как…”.

Кнопка “Сохранить” будет доступна, только когда таблицы будут заполнены корректными и допустимыми значениями.

Если пользователь загрузил исходные данные из файла, то программа сохранит исходные данные в этот файл.

Если же пользователь не загружал данные из файла, а ввёл их вручную, откроется диалог, в котором пользователь может выбрать файл, в который нужно сохранить данные (рис.11) или он может вписать имя файла сам, если файла с таким именем не существует, будет создан файл с именем, которое вписал пользователь, иначе выведется сообщение о перезаписи(рис.12):

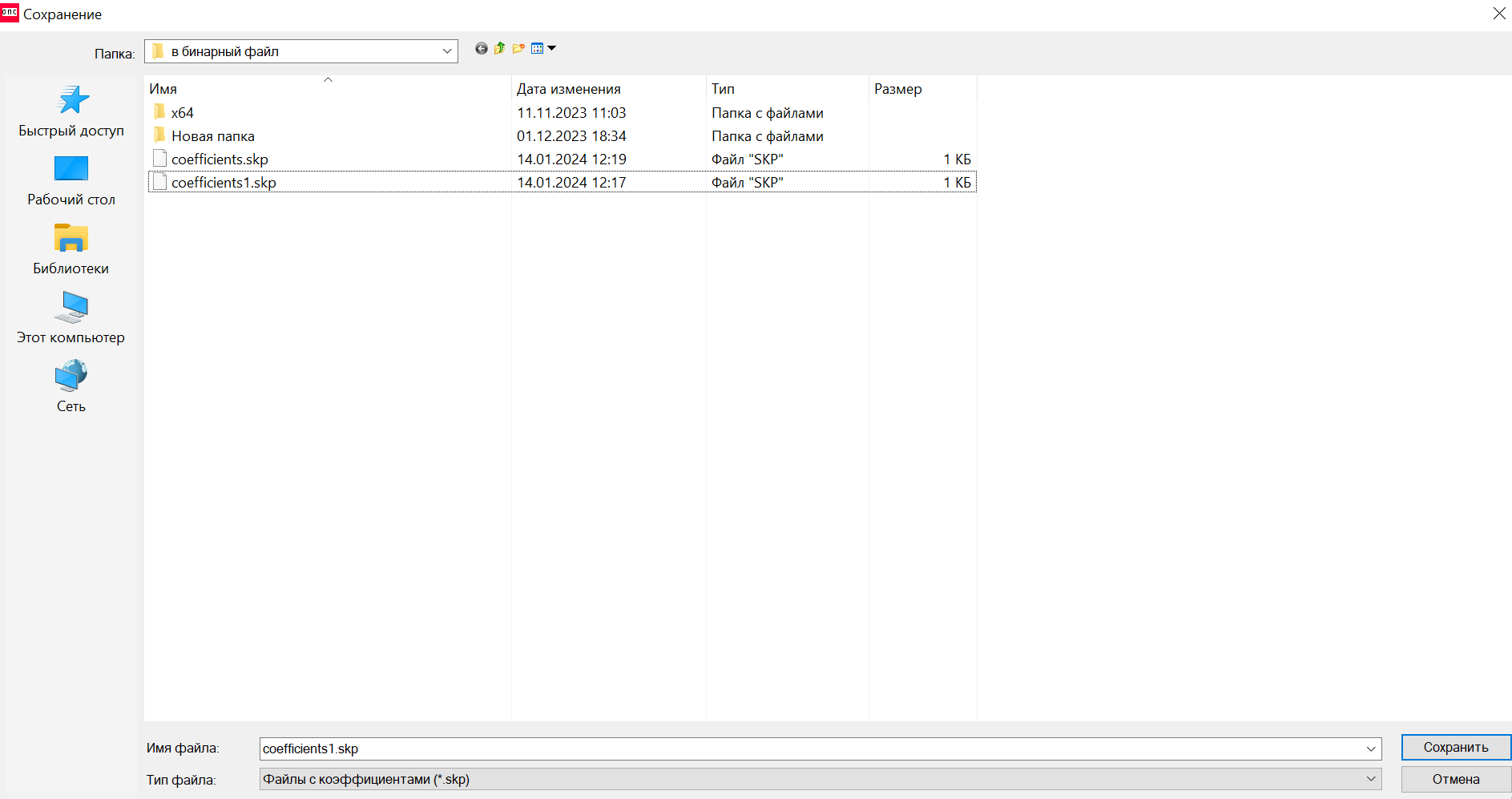


Рисунок 11 (Диалог сохранения)

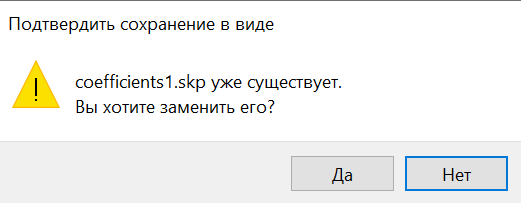


Рисунок 12 (Сообщение о перезаписи)

Главное меню с функционалом приложения расположено в его верхней части:



Рисунок 13 (Пункты главного меню)

При нажатии на пункт “Файл” откроется вкладка с функционалом программы, предназначенным для работы с данными:

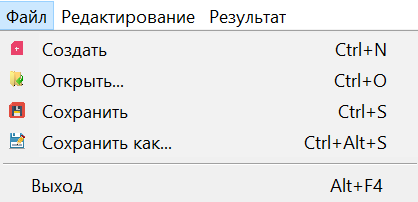


Рисунок 14 (Пункт "Файл")

* “Создать” проверяет доступна есть ли не сохранённые данные в таблицах, если есть то выводиться сообщение(рис.15), если нет, то все таблицы очищаются

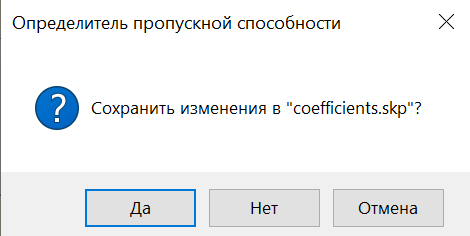


Рисунок 15 (Проверка на не сохранённые данные)

* “Открыть…” дублирует кнопку “Загрузить…” из вкладки “Исходные данные”.
* “Выход” осуществляет проверку на не сохранённые данные, если они есть то выводит запрос на сохранение(рис.16), если их нет то закрывает программу.

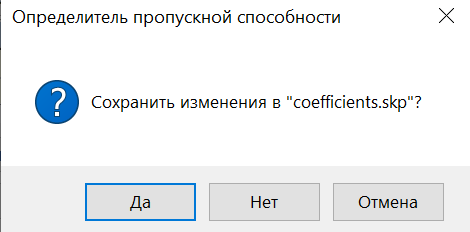


Рисунок 16 (Проверка на не сохранённые данные)

При нажатии на пункт “Редактировать” откроется вкладка с функционалом программы, предназначенным для редактирования таблицы:

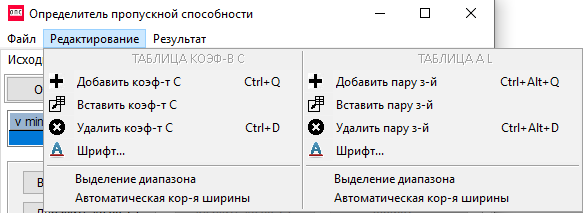


Рисунок 17 (Пункт "Редактировать")

При нажатии на пункт “Результаты” откроется вкладка с функционалом программы, предназначенным для работы с результатами:

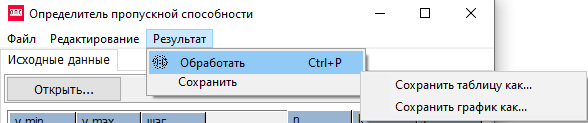


Рисунок 18 (Пункт "Результаты")

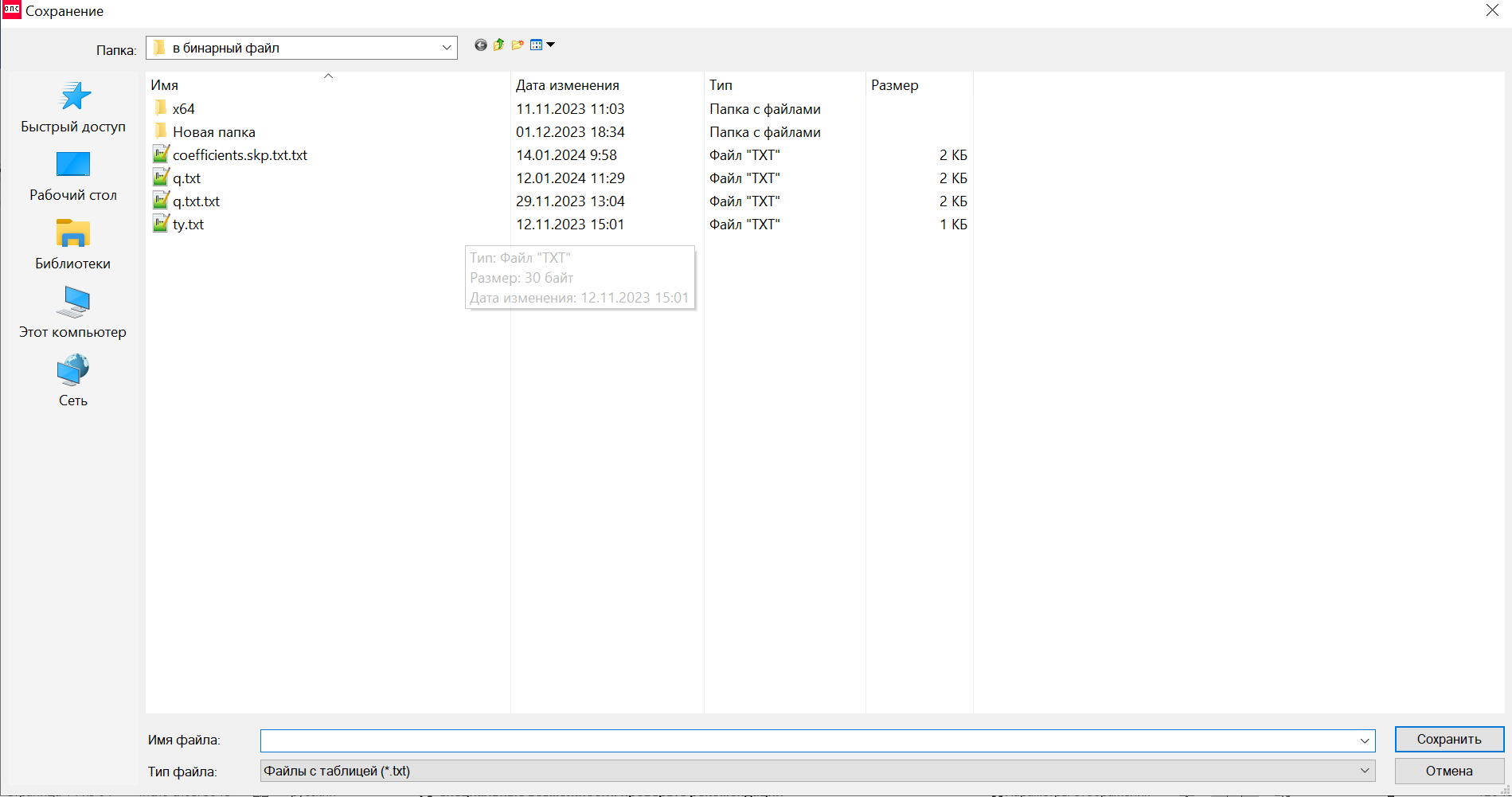
* “Обработать” дублирует кнопку на вкладке “Исходные данные”.
* “Сохранить таблицу как…” дублирует кнопку на вкладке “Результат”. При нажатии на нее, откроется диалог сохранения (рис.19):  
  

Рисунок 19 (Диалог сохранения для таблицы)

* “Сохранить график как…” дублирует кнопку на вкладке “Результат”. При нажатии на нее откроется диалог сохранения (рис.20), пользователь может выбрать следующие форматы: jpeg, gif ,bmp(рис.21)

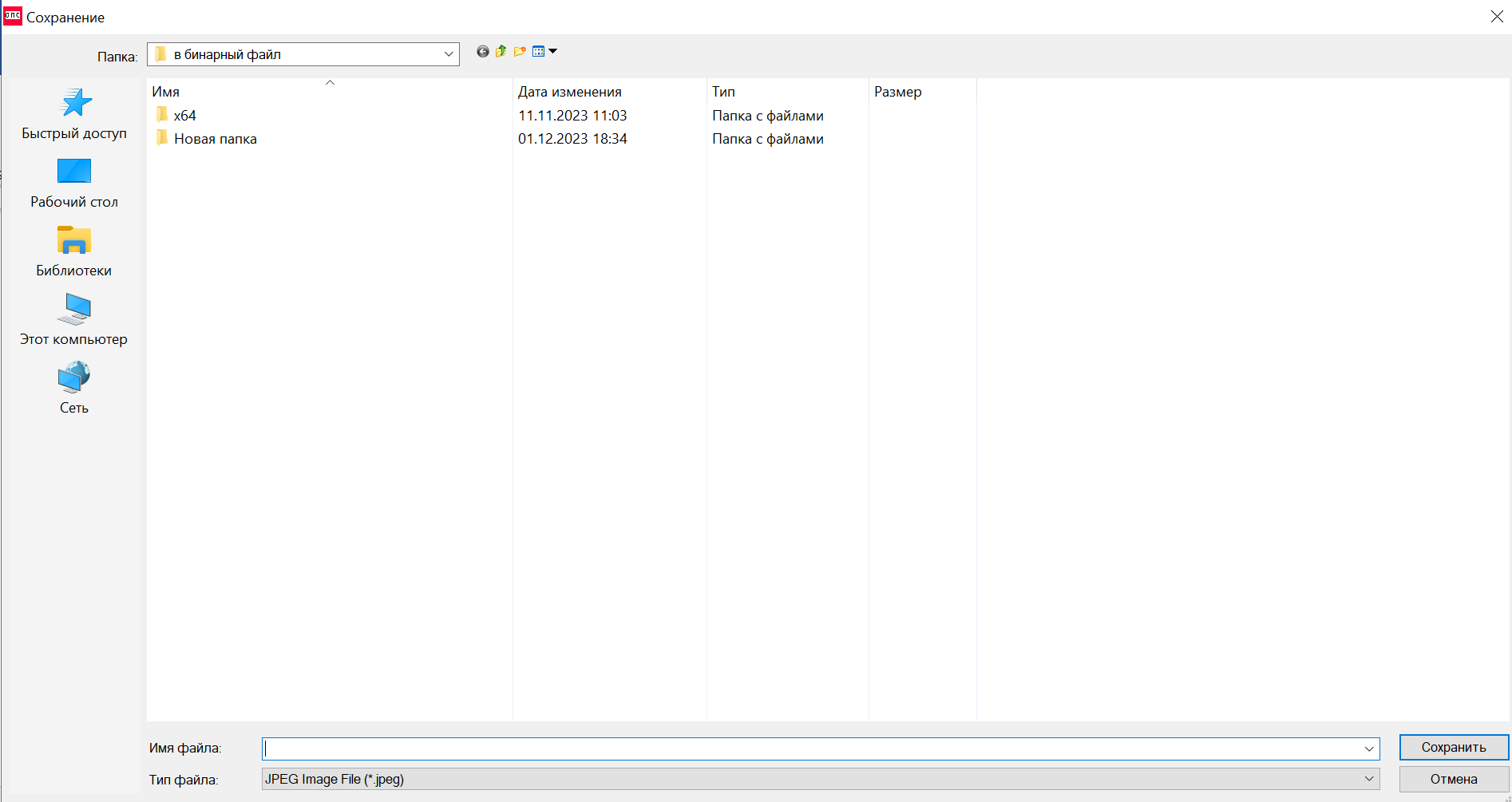


Рисунок 20 (Диалог сохранения для графика)

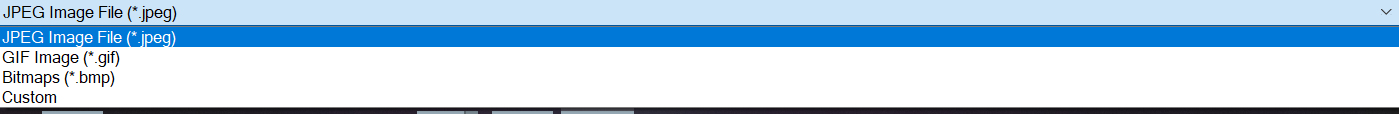


Рисунок 21 (Выбор формата графика)

# **4. Выбор среды разработки**

Выбор среды разработки С++ Builder обосновывается его наибольшей пригодностью. Поскольку предоставляется широкий набор инструментов и функциональности, специально разработанных для создания сложных приложений с графическим интерфейсом и обработкой данных. Тем самым, упрощая процесс проектирования и разработки.

# **5. Форматы данных**

### **5.1. Входные данные**

В программе доступны два метода для ввода данных:

1. Ввод необходимых данных клавиатурой.
2. Загрузка информации из внешнего файла.

Приложение использует двоичный формат данных при загрузке из файла, так как он обладает рядом преимуществ по сравнению с текстовым форматом.

1. Так как отсутствует ограничение на количество коэффициентов, то двоичный формат позволяет эффективно использовать память для хранения данных. Поскольку данные представляются в компактной структуре, что существенно сокращает размер файла и, следовательно, обеспечивает более быстрое выполнение операций чтения и записи.
2. Бинарное представление также обеспечивает более легкое написание кода.

Недостатки:

1. Теряется возможность читаемости исходных данных пользователем.

|  |  |  |
| --- | --- | --- |
| Критерий | Двоичный файл | Текстовый файл |
| Скорость работы | + | - |
| Объем занимаемой памяти | + | - |
| Легкость написания кода | + | - |
| Читаемость | - | + |

Структура файла:

Количество коэффициентов С, значения коэффициентов С, количество пар значений a и l, значения a, значения l, минимальная скорость, максимальная скорость, шаг, n, k, t.

### **5.2 Выходные данные**

Для сохранения удобочитаемости данных из таблицы, осуществляется запись в текстовый файл. Этот подход обеспечивает возможность последующего удобного восприятия информации пользователем. Одновременно график, представленный в процессе работы, сохраняется в различных форматах, таких как BMP, GIF и JPG. Глубина цвета – 24 бита.

# **6. Структура программы**

### **6.1. Модуль Main**

Главный модуль приложения включает в себя ключевые функции, такие как загрузка и сохранение данных, обработка информации, а также предоставляет возможность редактирования таблиц. Его основная задача — обеспечить надежное управление данными приложения, включая обработку исключений для повышения устойчивости.

## **6.2. Вспомогательные типы данных**

|  |  |
| --- | --- |
| **Название типа** | **Назначение** |
| typedef double elemType | Создана для обеспечения удобства изменения типа данных, с которыми работает приложение. |

### **6.3. Константы**

|  |  |  |
| --- | --- | --- |
| **Имя** | **Тип данных** | **Назначение** |
| MIN\_COL\_WIDTH | int | Для настройки минимальной ширины |

### **6.4. Класс оконной формы Main**

Перечень размещенных во время разработки компонентов VCL

|  |  |  |
| --- | --- | --- |
| **Имя** | **Класс** | **Назначение** |
| dowload | TButton | Загрузка данных |
| bProc | TButton | Обработка данных |
| savegrid | TButton | Сохранение таблицы |
| SaveChart | TButton | Сохранение графика |
| bInsCol | TButton | Добавить коэф-т С |
| DelColC | TButton | Удалить коэф-т С |
| AddColC | TButton | Вставить коэф-т С |
| bDelRangeCol | TButton | Удаленить диапазон коэф-ов С |
| bFontC | TButton | Корректировка шрифта в таблице с коэ-ми С |
| ClearC | TButton | Полная очистка таблицы от всех коф-в С |
| bInsCol2 | TButton | Добавление одной пары коэ-тов a и l к концу таблицы |
| DelColAL | TButton | Добавить пару з-й a и l |
| AddColAL | TButton | Вставка одной пары коэ-тов a и l |
| bDelRangeColAL | TButton | Удаление диапазона коэф-ов a и l |
| bFontAL | TButton | Корректировка шрифта в таблице с коэ-ми a и l |
| ClearAL | TButton | Полная очистка таблицы от коф-ов a и l |
| SrcC | TStringGrid | Таблица с коэффициентами С |
| SrcAL | TStringGrid | Таблица с коэффициентами a и l |
| SrcNKT | TStringGrid | Таблица с коэффициентом n, k и t |
| SrcVsT | TStringGrid | Таблица со значениями скорости + шаг |
| SrcData | TTabSheet | Страница с исходными данными |
| sbInfo | TStatusBar | Статус бар |
| Panel1 | TPanel | Расположение кнопок сохранения |
| PageControl1 | TPageControl | Добавление страниц |
| PageControl1 | TTabSheet | Страница с исходными даннмыми |
| ResPage | TTabSheet | Страница с результатами |
| MainMenu1 | TMainMenu | Меню |
| N1 | TMenuItem | Вкладка “Файл” |
| N2 | TMenuItem | Создать |
| N3 | TMenuItem | Открыть |
| N2 | TMenuItem | Сохранить |
| N5 | TMenuItem | Сохранить как |
| N7 | TMenuItem | Выход |
| N8 | TMenuItem | Редактировать |
| N11 | TMenuItem | Добавить столбец |
| N12 | TMenuItem | Удалить столбец |
| N8 | TMenuItem | Вкладка “Редактирование” |
| N11 | TMenuItem | Добавить коэф-т С |
| N24 | TMenuItem | Вставить коэф-т С |
| N12 | TMenuItem | Удалить коэф-т С |
| N13 | TMenuItem | Корректировка шрифта в таблице с коэ-ми С |
| N16 | TMenuItem | Выделение диапазона |
| A2 | TMenuItem | Автоматическая  кор-я ширины |
| N17 | TMenuItem | Добавить пару з-й |
| N25 | TMenuItem | Вставить пару з-й |
| N18 | TMenuItem | Удалить пару з-й |
| N19 | TMenuItem | Корректировка шрифта в таблице с коэ-ми a и l |
| N21 | TMenuItem | Выделение диапазона  з-й a, l |
| A1 | TMenuItem | Автоматическая кор-я ширины таблицы с a, l |
| N9 | TMenuItem | Вкладка “Результат” |
| N4 | TMenuItem | Обработать данные |
| N22 | TMenuItem | Сохранить таблицу |
| N23 | TMenuItem | Сохранить график |
| cbEditAutoWidthC | TCheckBox | Автоматическая кор-я ширины таблицы C |
| cbColRangeSelect | TCheckBox | Включение режима: “Выделение диапазона”  для таблицы с коэ-ми С |
| cbEditAutoWidthAL | TCheckBox | Автоматическая кор-я ширины таблицы a и l |
| cbColRangeSelectAL | TCheckBox | Включение режима: “Выделение диапазона”  для таблицы с коэф-ми  a и l |
| SrcRes | TStringGrid | Результирующая таблица |
| QVgraph | TChart | Результирующий график |
| Panel2 | TPanel | Расположение кнопок загрузки и обработки данных |
| ODdata | TOpenDialog | Диалог загрузки |
| SaveGr | TSaveDialog | Диалог сохранения результирующей таблицы |
| SaveGraph | TSavePictureDialog | Диалог сохранения графика |

### **6.5. Функции**

|  |  |  |  |
| --- | --- | --- | --- |
| **Заголовок функции** | **Назначение** | **Параметры** | **Возвращаемое значение** |
| int CalcGridColWidth  (TStringGrid \*grid, int colIdx); | Вычисление ширины столбца, учитывая содержимое ячеек этого столбца. | 1) *grid* (входной): является исходной таблицей, для которой производится расчет ширины столбца.  2)*int colIdx*: (входной) индекс столбца, ширину которого необходимо вычислить. | int:  Представляет собой вычисленную максимальную ширину столбца с учетом содержимого ячеек и константы MIN\_COL\_WIDTH. Возвращается в пикселях. |
| void SortGridByRow(TStringGrid \*grid, int rowIdx, bool ascending, TCmpFunc cmpFunc); | Сортировка строк в таблице по указанному индексу строки (rowIdx). Сортировка производится в порядке возрастания или убывания в зависимости от значения параметра ascending. | 1)grid (входной) - указатель на таблицу, которая будет подвергнута сортировке.  2)rowIdx (входной) - индекс строки, по которой производится сортировка.  3)ascending (входной) - булево значение, определяющее порядок сортировки (true - возрастание, false - убывание).  4)cmpFunc (входной) - пользовательская функция сравнения, которая определяет порядок сортировки. Функция должна принимать две строки для сравнения и возвращать целое число: отрицательное, если первая строка меньше второй; положительное, если первая строка больше второй; ноль, если строки равны. | Нет возвращаемого значения |
| int AdjustGridColWidths(TStringGrid \*grid); | Автоматическая настройки ширины столбцов в компоненте TStringGrid так, чтобы они корректно отображали содержимое исходя из их содержимого. | 1)TStringGrid \*grid (входной): Указатель на таблицы, для которого производится коррекция ширины столбцов. | int: Возвращается общая ширина всех столбцов после их коррекции. |
| float cmpFunc1 (const UnicodeString &s1, const UnicodeString &s2); | Сравнение двух Unicode-строк, представляющих числа в виде строк, путем преобразования их в числа с плавающей запятой и последующего вычисления разности. | 1)s1 (входной): Константная ссылка на первую Unicode-строку для сравнения.  2) s2 (входной): Константная ссылка на вторую Unicode-строку для сравнения. | float;  Функция возвращает результат сравнения строк |
| bool oshib(TStringGrid\*SrcC,TStringGrid\*SrcAL,TStringGrid\*SrcNKT,TStringGrid\*SrcVsTTStatusBar\*sbInfo, TButton\*bProc); | Проверку значений в ячейках различных TStringGrid-ов и возвращает логическое значение (bool) в зависимости от наличия ошибок в данных. | 1)SrcC (входной): Указатель на таблицу с данными, которые подлежат проверке (  2)SrcAL(входной): параметр. Указатель на таблицу, содержащий данные для проверки  3)SrcNKT(входной): Указатель на таблицу с данными для проверки  4)SrcVsT(входной): Указатель на таблцу с данными для проверки  5)sbInfo: (Входной/выходной): указатель на TStatusBar, куда будет выводиться информация об ошибках  6)bProc(Входной/выходной):Указатель на TButton, состояние которого будет управляться в зависимости от наличия | bool: Логическое значение. Возвращает true, если все проверки прошли успешно (нет ошибок в данных), в противном случае возвращает false. |

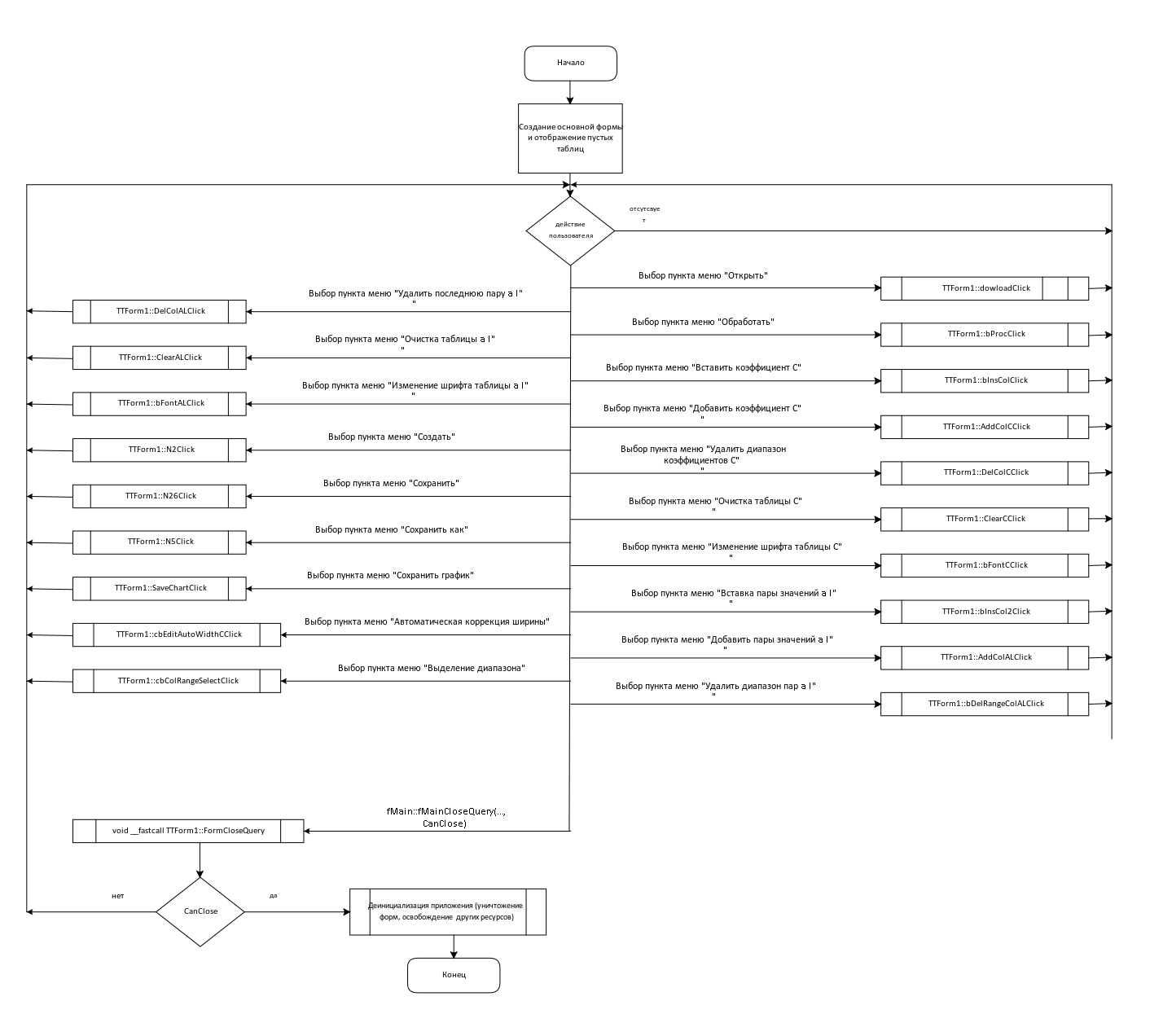
### **6.6. Переменные**

**Глобальные**

|  |  |
| --- | --- |
| **Имя** | **Тип данных** |
| TTForm1 | \*TForm1 |

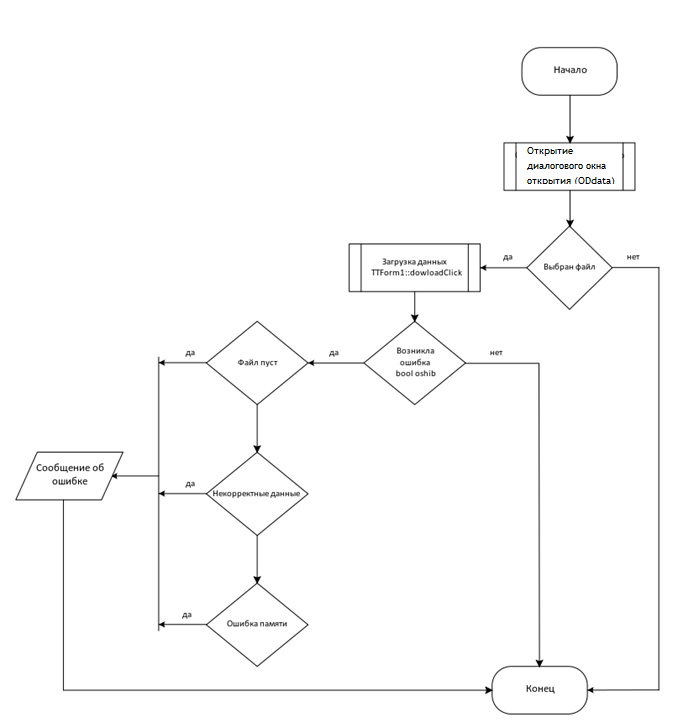
# **7. Описание алгоритмов**

### **7.1. Блок-схема укрупненного алгоритма работы пользовательского интерфейса приложения**



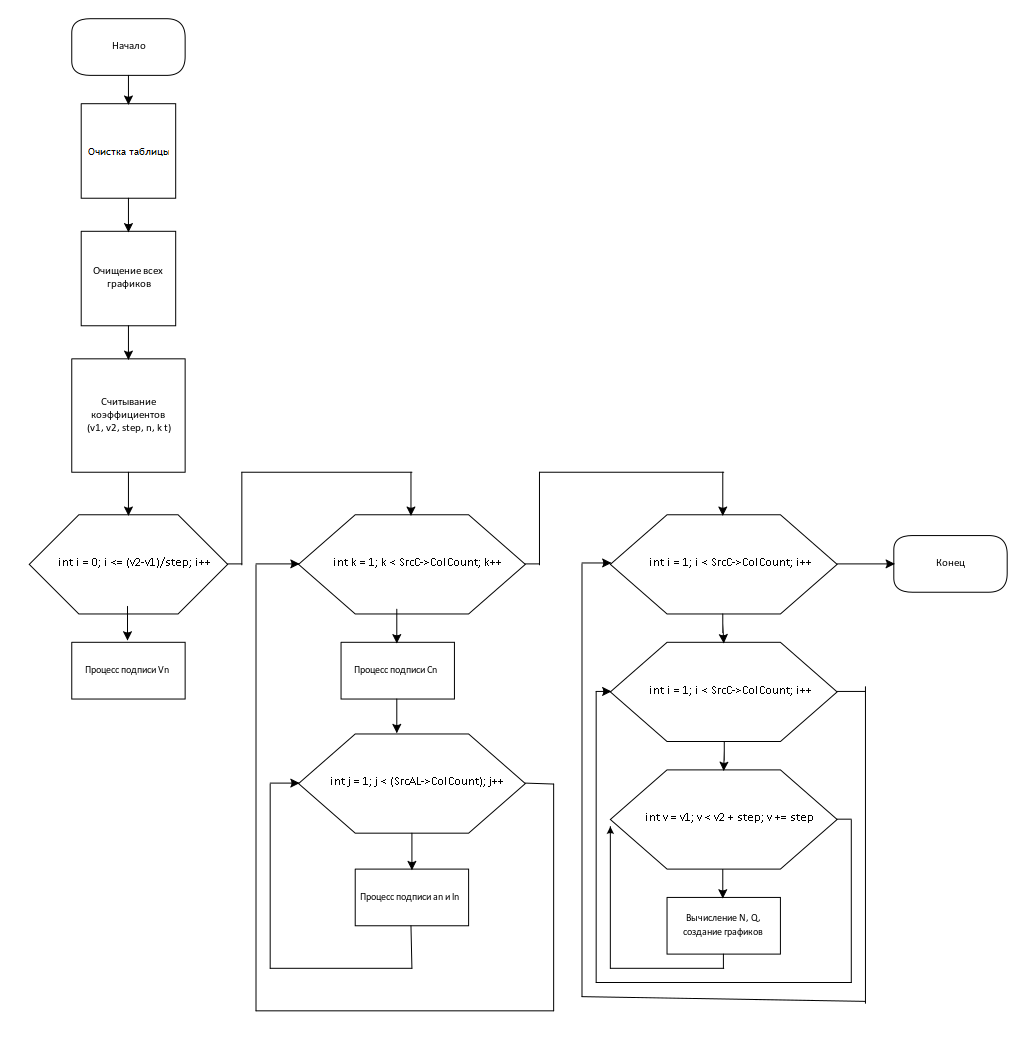
### **7.2. Блок-схема алгоритма выбора пункта меню “Открыть”**

### **[void \_\_fastcall TTForm1::dowloadClick(TObject \*Sender)]**



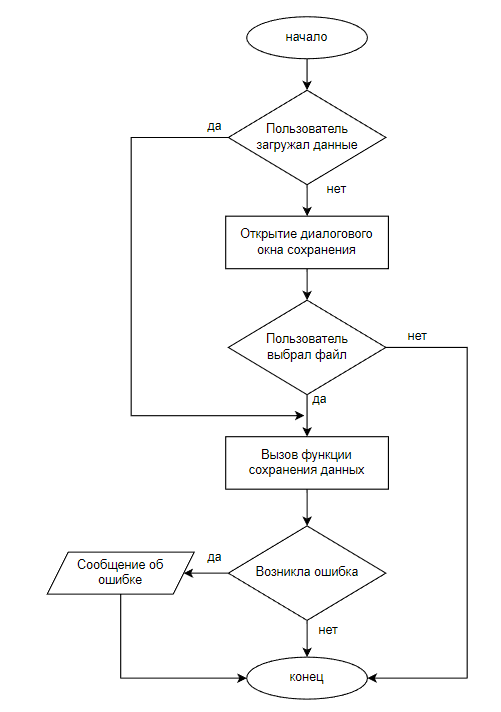
### **7.3. Блок-схема алгоритма выбора пункта меню “Обработать”**

### **[void \_\_fastcall TTForm1::bProcClick(TObject \*Sender)]**

****

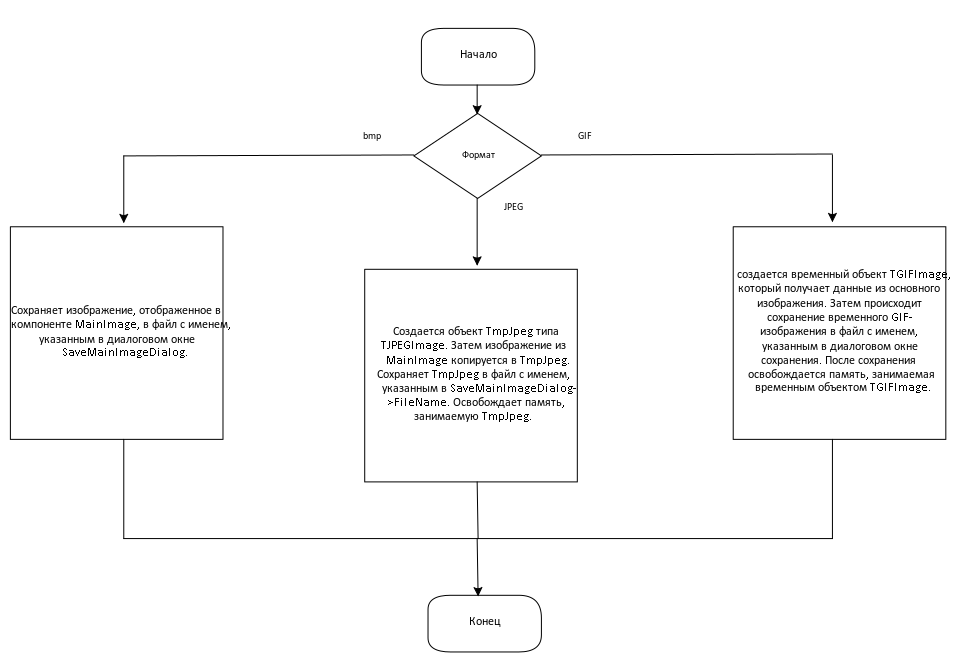
### **7.4. Блок-схема алгоритма выбора пункта меню “Сохранить таблицу как …”**

### **[void \_\_fastcall TTForm1::savegridClick(TObject \*Sender)]**



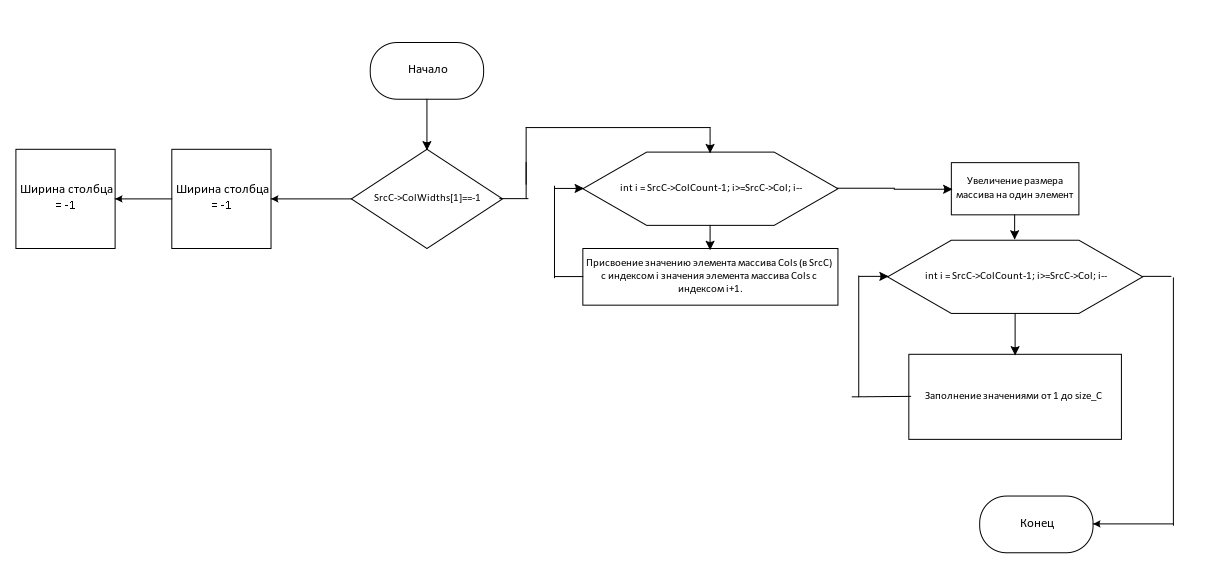
### **7.5. Блок-схема алгоритма выбора пункта меню “Сохранить график как…”**

### **[void \_\_fastcall TTForm1::SaveChartClick(TObject \*Sender)]**



### **7.6. Блок-схема алгоритма метода выбора пункта меню “Вставить столбец”**

### **[void \_\_fastcall TTForm1::bInsColClick(TObject \*Sender)]**



## **8. Текст программы с комментариями**

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

#include <memory>

#include <fstream>

#include <vcl.h>

#pragma hdrstop

#include "Main.h"

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

#pragma package(smart\_init)

#pragma resource "\*.dfm"

TTForm1 \*TForm1;

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

\_\_fastcall TTForm1::TTForm1(TComponent\* Owner)

: TForm(Owner)

{

//QVgraph->Series[1]->Active = true;

SrcC->Cells[0][0] = "№";

SrcC->Cells[0][1] = "C";

SrcAL->Cells[0][0] = "№";

SrcAL->Cells[1][0] = "1";

SrcAL->Cells[0][1] = "a";

SrcAL->Cells[0][2] = "l";

SrcVsT->Cells[0][0] = "v min";

SrcVsT->Cells[1][0] = "v max";

SrcVsT->Cells[2][0] = "шаг";

SrcNKT->Cells[0][0] = "n";

SrcNKT->Cells[1][0] = "k";

SrcNKT->Cells[2][0] = "t";

if(SrcC->ColCount==2)

SrcC->ColWidths[1] = -1;

if(SrcAL->ColCount==2)

SrcAL->ColWidths[1] = -1;

// SrcC -> SetFocus();

// SrcAL -> SetFocus();

FSortedRowNbr = 0;

if(a)

bProc->Enabled=true;

else

bProc->Enabled=false;

// if (SrcNKT->Cells[2][1] != "")

// bProc -> Enabled=True;

}

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

void \_\_fastcall TTForm1::AddColCClick(TObject \*Sender)

{

if(SrcC->ColWidths[1]==-1){

SrcC->ColWidths[1] = SrcC -> DefaultColWidth;

cbColRangeSelectClick(NULL);

}

else

SrcC -> ColCount++;

SrcC -> Col = SrcC -> ColCount-1;

SrcC -> Cols[SrcC->Col]-> Clear();

SrcC -> SetFocus();

DelColC->Enabled = true;

bDelRangeCol->Enabled = true;

bFontC->Enabled=true;

ClearC->Enabled=true;

size\_C++;

for(int i = 0; i < size\_C; i++)

SrcC->Cells[i+1][0] = IntToStr(i+1);

}

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

void \_\_fastcall TTForm1::DelColCClick(TObject \*Sender)

{

if(SrcC->ColWidths[1]==-1)

return;

if(SrcC->ColCount>2){

SrcC -> ColCount--;

for(int i = SrcC->ColCount; i<SrcC->ColCount; i++){

SrcC->Cols[i] = SrcC -> Cols[i+1];

}

}

else{

SrcC->ColWidths[1]=-1;

SrcC->Options = SrcC->Options>>goEditing;

bProc->Enabled=false;

DelColC->Enabled = false;

bDelRangeCol->Enabled = false;

bFontC->Enabled=false;

ClearC->Enabled=false;

}

size\_C--;

for(int i = 0; i < size\_C; i++)

SrcC->Cells[i+1][0] = IntToStr(i+1);

}

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

void \_\_fastcall TTForm1::AddColALClick(TObject \*Sender)

{

if(SrcAL->ColWidths[1]==-1){

SrcAL->ColWidths[1] = SrcAL -> DefaultColWidth;

cbColRangeSelectALClick(NULL);

}

else

SrcAL -> ColCount++;

SrcAL->Cells[(SrcAL->ColCount)-1][0] = IntToStr(SrcAL->ColCount);

SrcAL -> Col = SrcAL -> ColCount-1;

SrcAL -> Cols[SrcAL->Col]-> Clear();

SrcAL -> SetFocus();

DelColAL->Enabled = true;

bDelRangeColAL->Enabled = true;

bFontAL->Enabled=true;

ClearAL->Enabled=true;

size\_al++;

for(int i = 0; i < size\_al; i++)

SrcAL->Cells[i+1][0] = IntToStr(i+1);

}

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

void \_\_fastcall TTForm1::DelColALClick(TObject \*Sender)

{

if(SrcAL->ColWidths[1]==-1)

return;

if(SrcAL->ColCount>2){

SrcAL -> ColCount--;

for(int i = SrcAL->ColCount; i < SrcAL->ColCount; i++){

SrcAL->Cols[i] = SrcAL -> Cols[i+1];

}

}

else{

SrcAL->ColWidths[1]=-1;

SrcAL->Options = SrcAL->Options>>goEditing;

bProc->Enabled=false;

DelColAL->Enabled = false;

bDelRangeColAL->Enabled = false;

bFontAL->Enabled=false;

ClearAL->Enabled=false;

}

size\_al--;

for(int i = 0; i < size\_al; i++)

SrcAL->Cells[i+1][0] = IntToStr(i+1);

}

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

void \_\_fastcall TTForm1::ClearCClick(TObject \*Sender) //очстка не работает

{

if (Application->MessageBox(

UnicodeString(L"Вы действительно хотите полностью очистить таблицу коэф-тов С?").w\_str(),

L"Определитель пропускной способности", MB\_YESNO | MB\_ICONQUESTION | MB\_DEFBUTTON2) != IDYES)

return;

for (int i = 1; i < SrcC->ColCount; i++) {

SrcC ->Cells[i][0] = L"";

//delete SrcC->Objects[i][1];

}

//SrcC->ColCount=1;

SrcC->ColCount=2;

SrcC->ColWidths[1]=-1;

bProc->Enabled=false;

DelColC->Enabled = false;

bDelRangeCol->Enabled = false;

bFontC->Enabled=false;

ClearC->Enabled=false;

}

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

void \_\_fastcall TTForm1::ClearALClick(TObject \*Sender)

{

for (int i = 1; i < SrcAL->ColCount; i++) {

SrcAL ->Cells[i][0] = L"";

//delete SrcC->Objects[i][1];

}

//SrcC->ColCount=1;

SrcAL->ColCount=2;

SrcAL->ColWidths[1]=-1;

bProc->Enabled=false;

DelColAL->Enabled = false;

bDelRangeColAL->Enabled = false;

bFontAL->Enabled=false;

ClearAL->Enabled=false;

}

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

void \_\_fastcall TTForm1::dowloadClick(TObject \*Sender)

{

if(!ODdata->Execute()){

return;

}

//bProc->Enabled=false;

//FileName = SaveGr->FileName;

//sbInfo->SimpleText = FileName;

//String fn = (eFileName -> Text);

SrcC->ColWidths[1] = SrcC -> DefaultColWidth;

SrcAL->ColWidths[1] = SrcAL -> DefaultColWidth;

ifstream fn;

fn.open(AnsiString(ODdata->FileName).c\_str(),ios::binary);

if(!fn.is\_open()){

sbInfo->SimpleText=

String(L"Ошибка открытия файла: " + (ODdata->FileName) + L", \nкод ошибки: ") + errno;

return;

}

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

int n= fn.tellg();

Memo1 -> Text = IntToStr(n);

if (n==0){

fn.close();

sbInfo->SimpleText= String(L"Нет данных в\"" + (ODdata->FileName));

return;

}

bool a;

if (n < sizeof(size\_al) || ((n-sizeof(size\_al)-sizeof(size\_C)) % sizeof(double))) //р-р маленький и не кратный

a= true;

else {

fn.seekg(0);

fn.read((char\*)&size\_C,sizeof(size\_C));

fn.read((char\*)&size\_al,sizeof(size\_al));

if(size\_C\*sizeof(double)+ size\_al\*sizeof(double)\*2 + 6\*sizeof(double)!=

n-sizeof(size\_C)-sizeof(size\_al))

a = true;

}

if (a) {

fn.close();

sbInfo->SimpleText= String(L"Целостность содержимого файла нарушена");

}

SrcC -> ColCount = size\_C + 1;

for(int i = 0; i < size\_C; i++){

SrcC->Cells[i+1][0] = IntToStr(i+1);

fn.read((char\*)&elem,sizeof(elemType));

SrcC->Cells[i+1][1]= FloatToStr(elem);

}

int tt=0;

SrcAL -> ColCount = size\_al + 1;

for(int i = 1; i < 3; i++){

for(int j=0; j < size\_al; j++){

SrcAL->Cells[tt+1][0]= tt+1;

tt++;

fn.read((char\*)&elem,sizeof(elemType));

SrcAL->Cells[j+1][i]= FloatToStr(elem);

}

}

for(int i=0; i < 3; i++){

fn.read((char\*)&elem,sizeof(elemType));

SrcVsT->Cells[i][1]= FloatToStr(elem);

}

for(int i=0; i < 3; i++){

fn.read((char\*)&elem,sizeof(elemType));

SrcNKT->Cells[i][1]= FloatToStr(elem);

}

if(oshib(SrcC, SrcAL, SrcNKT, SrcVsT, sbInfo, bProc)){

bProc->Enabled=True;

}

// if (SrcNKT->Cells[2][1] != "")

// bProc -> Enabled=True;

// fn.close();

// ShowMessage(L"Файл содержит некорректные данные.");

// return;

// }

cbColRangeSelectClick(NULL);

DelColC->Enabled = true;

bFontC->Enabled=true;

ClearC->Enabled=true;

DelColAL->Enabled = true;

bDelRangeColAL->Enabled = true;

bFontAL->Enabled=true;

ClearAL->Enabled=true;

bProc->Enabled=true;

FSortedRowNbr = 0;

// if(cbEditAutoWidth->Checked)

// AdjustGridColWidths(SrcC);

if(cbEditAutoWidthC->Checked){

AdjustGridColWidths(SrcC);

SrcC->Options = SrcC->Options >> goColSizing;

}

else

SrcC->Options = SrcC->Options << goColSizing;

if(cbEditAutoWidthAL->Checked){

AdjustGridColWidths(SrcAL);

SrcAL->Options = SrcAL->Options >> goColSizing;

}

else

SrcAL->Options = SrcAL->Options << goColSizing;

a=true;

b=true;

c=true;

d=true;

bDelRangeCol->Enabled = true;

}

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

void \_\_fastcall TTForm1::bProcClick(TObject \*Sender)

{

ResPage->TabVisible=true;

PageControl1->Pages[1]->Enabled = true;

for(int i = 0; i <= SrcRes->ColCount; i++){

for(int j = 0; j <= SrcRes->RowCount; j++){

SrcRes->Cells[i][j] = L"";

}

}

// Очистка всех графиков

while (QVgraph->SeriesList->Count > 0) {

TChartSeries \*series = QVgraph->SeriesList->Items[QVgraph->SeriesList->Count - 1];

series->Clear();

QVgraph->SeriesList->Delete(QVgraph->SeriesList->Count - 1);

delete series;

}

v1= (SrcVsT->Cells[0][1]).ToDouble();

v2 = (SrcVsT->Cells[1][1]).ToDouble();

step = (SrcVsT->Cells[2][1]).ToDouble();

n = (SrcNKT->Cells[0][1]).ToDouble();

k = (SrcNKT->Cells[1][1]).ToDouble();

t = (SrcNKT->Cells[2][1]).ToDouble();

SrcRes -> RowCount = (v2-v1)/step +2;

for (int i = 0; i <= (v2-v1)/step; i++) {

SrcRes->Cells[0][i+1]=L"v="+FloatToStr(v1+step\*i); /\*\*/

}

SrcRes -> ColCount = (SrcAL->ColCount)\*((SrcC->ColCount)-1);

int i = 1;

for (int k = 1; k < SrcC->ColCount; k++) {

SrcRes->Cells[i - 1][0] = "C" + IntToStr(k) + " = " + SrcC->Cells[k][1];

i++;

for (int j = 1; j < (SrcAL->ColCount); j++) {

SrcRes->Cells[i - 1][0] = "a= "+SrcAL->Cells[j][1]

+" l=" + SrcAL->Cells[j][2];

i++;

}

}

TLineSeries \*series = new TLineSeries(QVgraph);

// double N, Q;

for (int i = 1; i < SrcC->ColCount; i++) {

for (int j = 1; j < SrcAL->ColCount; j++) {

TLineSeries \*series = new TLineSeries(QVgraph);

QVgraph->AddSeries(series);

AnsiString legendText = "С" + IntToStr(i) + " "+ SrcRes->Cells[j][0];

series->Title = legendText;

for (int v = v1; v < v2 + step; v += step) {

N = 3600 \* v1 / (StrToFloat(SrcAL->Cells[j][1]) +

StrToFloat(SrcAL->Cells[j][2]) +

v \* t + StrToFloat(SrcC->Cells[i][1]) \* v \* v);

N=N-fmod(N,0.001);

Q = N \* n \* k;

//SrcRes->Cells[j + (SrcAL->ColCount) \* i - (SrcAL->ColCount)][(v - v1) / step + 1] = N;

//SrcRes->Cells[j + (SrcAL->ColCount) \* i - (SrcAL->ColCount)][(v - v1) / step + 1] = N;

SrcRes->Cells[j + (SrcAL->ColCount) \* i - (SrcAL->ColCount)][(v - v1) / step + 1] =

" " + FormatFloat("#0.00", N);

series->AddXY(v, Q);

}

}

}

if(cbEditAutoWidthRes->Checked){

AdjustGridColWidths(SrcRes);

SrcRes->Options = SrcRes->Options >> goColSizing;

}

else

SrcRes->Options = SrcRes->Options << goColSizing;

sbInfo ->SimpleText = L"Данные обработаны" ;

PageControl1->ActivePage = ResPage;

}

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

void \_\_fastcall TTForm1::SrcCKeyPress(TObject \*Sender, System::WideChar &Key)

{

if (!SrcC->EditorMode)

return;

if (Key==VK\_ESCAPE && SrcC->EditorMode) {

SrcC->Cells[SrcC->Col][SrcC->Row]=FCurCellText;

return;

}

if(SrcC->EditorMode){

oshib(SrcC, SrcAL, SrcNKT, SrcVsT, sbInfo, bProc);

return;

}

if(!((Key >= L'0' && Key <= L'9') || Key==L'-' || Key==L'+' || Key == FormatSettings.DecimalSeparator ||

Key ==L'e' || Key ==L'E') && Key!=VK\_BACK && Key!=VK\_RETURN && Key!=3 && Key!=22 &&

Key!= 24 && Key!= 26){

//bProc1->Enabled=false;

sbInfo ->SimpleText = L"Недопустимый символ '" + UnicodeString(Key) + L'\'';

Key = 0;

if(oshib(SrcC, SrcAL, SrcNKT, SrcVsT, sbInfo, bProc)){

bProc->Enabled=True;

}

}

else {

sbInfo->SimpleText=L"";

bProc->Enabled=true;

//a=true;

}

}

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

bool \_\_fastcall TTForm1::savegridClick(TObject \*Sender)

{

WideString str;

TStringList \*sl = new TStringList;

for(int i = 0 ; i<SrcRes->RowCount ;++i)

{

if(i!=1){

str = "\t";

sl->Add(str);

}

for(int j = 0; j< SrcRes->ColCount;++j)

str +=SrcRes->Cells[j][i]+ "\t";

sl->Add(str);

str = "\r" ;

str = "";

}

if (SaveGr->Execute())

sl->SaveToFile(SaveGr->FileName + ".txt");

delete sl;

// String svdFileName = SaveGr->FileName;

//

// if(!SaveGr->Execute())

// return false;

//

// try{

// mText->Lines->SaveToFile(SaveGr->FileName);

// }

// catch (Exception &e){

// Application -> MessageBox(Format(L"Ошибка сохранения файла \"%s\": \"%s\"",

// ARRAYOFCONST((SaveGr->FileName, e.Message))).w\_str(),

// Application->Title.w\_str(),

// MB\_OK|MB\_ICONERROR);

// SaveGr->FileName = svdFileName;

// return false;

// }

//

// savegrid->Enabled = false;

// Caption = S\_EDITOR+ L" - " + (SaveGr->FileName);

// return true;

}

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

void \_\_fastcall TTForm1::SaveChartClick(TObject \*Sender)

{

if (!SaveGr->Execute())

return;

String FileName = String(SaveGr -> FileName);

if (BMP->Checked) {

QVgraph->SaveToBitmapFile(UnicodeString(SaveGr -> FileName) + L".bmp");

}

if (JPEG->Checked) {

TCanvas \*Source = new TCanvas();

Source->Handle = GetDC(QVgraph->Handle);

Graphics::TBitmap \*bmp = new Graphics::TBitmap();

bmp->SetSize(QVgraph->ClientWidth, QVgraph->ClientHeight);

RECT src = QVgraph->ClientRect;

RECT dst = {0, 0, QVgraph->ClientWidth, QVgraph->ClientHeight};

bmp->Canvas->CopyRect(dst, Source, src);

TJPEGImage \*jpeg = new TJPEGImage();

jpeg->Assign(bmp);

jpeg->SaveToFile(FileName);

jpeg->Free();

bmp->Free();

ReleaseDC(QVgraph->Handle, Source->Handle);

Source->Free();

}

if (GIF->Checked) {

TCanvas \*Source = new TCanvas();

Source->Handle = GetDC(QVgraph->Handle);

Graphics::TBitmap \*bmp = new Graphics::TBitmap();

bmp->SetSize(QVgraph->ClientWidth, QVgraph->ClientHeight);

RECT src = QVgraph->ClientRect;

RECT dst = {0, 0, QVgraph->ClientWidth, QVgraph->ClientHeight};

bmp->Canvas->CopyRect(dst, Source, src);

TGIFImage\* gif= new TGIFImage();

gif->Assign(bmp);

gif->SaveToFile(String(SaveGr -> FileName));

gif->Free();

bmp->Free();

ReleaseDC(QVgraph->Handle, Source->Handle);

Source->Free();

FileName += ".gif";

}

}

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

void \_\_fastcall TTForm1::SrcCSetEditText(TObject \*Sender, int ACol, int ARow, const UnicodeString Value)

{

float d;

if (!TryStrToFloat(SrcC->Cells[ACol][ARow], d)) {

sbInfo -> SimpleText = UnicodeString(L"\"" + SrcC -> Cells[ACol][ARow] + L"\" не является корректным значением");

bProc->Enabled=false;

return;

}

if(oshib(SrcC, SrcAL, SrcNKT, SrcVsT, sbInfo, bProc)){

bProc->Enabled=True;

}

//bProc->Enabled = true;

}

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

void \_\_fastcall TTForm1::SrcALSetEditText(TObject \*Sender, int ACol, int ARow, const UnicodeString Value)

{

float d;

if (!TryStrToFloat(SrcAL->Cells[ACol][ARow], d)) {

sbInfo -> SimpleText = UnicodeString(L"\"" + SrcAL -> Cells[ACol][ARow] + L"\" не является корректным значением");

bProc->Enabled=false;

return;

}

b = true;

//bProc->Enabled = true;

}

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

void \_\_fastcall TTForm1::SrcVsTSetEditText(TObject \*Sender, int ACol, int ARow, const UnicodeString Value)

{

float d;

if (!TryStrToFloat(SrcVsT->Cells[ACol][ARow], d)) {

sbInfo -> SimpleText = UnicodeString(L"\"" + SrcVsT -> Cells[ACol][ARow]

+ L"\" не является корректным значением");

bProc->Enabled=false;

return;

}

d = true;

//bProc->Enabled = true;

}

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

void \_\_fastcall TTForm1::SrcNKTSetEditText(TObject \*Sender, int ACol, int ARow, const UnicodeString Value)

{

float d;

if (!TryStrToFloat(SrcNKT->Cells[ACol][ARow], d)) {

sbInfo -> SimpleText = UnicodeString(L"\"" + SrcNKT -> Cells[ACol][ARow]

+ L"\" не является корректным значением");

bProc->Enabled=false;

return;

}

c = true;

bProc->Enabled = true;

}

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

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

void \_\_fastcall TTForm1::cbColRangeSelectClick(TObject \*Sender)

{

if(Sender==cbColRangeSelect)

N16->Checked = cbColRangeSelect->Checked;

else{

cbColRangeSelect->OnClick = NULL;

try{

cbColRangeSelect->Checked = N16->Checked;

}

\_\_finally{

cbColRangeSelect->OnClick = cbColRangeSelectClick;

}

}

if(cbColRangeSelect->Checked){

SrcC->Options = SrcC->Options >> goEditing;

}

else

if(SrcC->ColWidths[1] == -1)

SrcC->Options=SrcC->Options >> goEditing /\*>> goRowSelect\*/;

else

SrcC->Options=SrcC->Options << goEditing /\*<< goRowSelect\*/;

}

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

void \_\_fastcall TTForm1::cbColRangeSelectALClick(TObject \*Sender)

{

if(Sender==cbColRangeSelectAL)

N21->Checked = cbColRangeSelectAL->Checked;

else{

cbColRangeSelectAL->OnClick = NULL;

try{

cbColRangeSelectAL->Checked = N21->Checked;

}

\_\_finally{

cbColRangeSelectAL->OnClick = cbColRangeSelectALClick;

}

}

if(cbColRangeSelectAL->Checked){

SrcAL->Options = SrcAL->Options >> goEditing;

SrcAL -> SetFocus();

}

else

if(SrcAL->ColWidths[1] == -1)

SrcAL->Options=SrcAL -> Options >> goEditing /\*>> goRowSelect\*/;

else

SrcAL->Options=SrcAL -> Options << goEditing /\*<< goRowSelect\*/;

}

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

void \_\_fastcall TTForm1::bInsColClick(TObject \*Sender)

{

if(SrcC->ColWidths[1]==-1) {

SrcC->ColWidths[1] = SrcC -> DefaultColWidth;

cbColRangeSelectClick(NULL);

}

else

SrcC -> ColCount++;

for(int i = SrcC->ColCount-1; i>=SrcC->Col; i--){

SrcC->Cols[i+1] = SrcC -> Cols[i];

// SrcC -> ColCount++;

}

SrcC->Cols[SrcC->Col] -> Clear();

SrcC->SetFocus();

DelColC->Enabled = true;

bDelRangeCol->Enabled = true;

bFontC->Enabled=true;

ClearC->Enabled=true;

size\_C++;

for(int i = 0; i < size\_C; i++)

SrcC->Cells[i+1][0] = IntToStr(i+1);

}

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

void \_\_fastcall TTForm1::bInsCol2Click(TObject \*Sender)

{

if(SrcAL->ColWidths[1]==-1) {

SrcAL->ColWidths[1] = SrcAL -> DefaultColWidth;

cbColRangeSelectALClick(NULL);

}

else

SrcAL -> ColCount++;

for(int i = SrcAL->ColCount-1; i>=SrcAL->Col; i--){

SrcAL->Cols[i+1] = SrcAL -> Cols[i];

// SrcC -> ColCount++;

}

SrcAL->Cols[SrcAL->Col] -> Clear();

SrcAL->SetFocus();

DelColAL->Enabled = true;

bDelRangeColAL->Enabled = true;

bFontAL->Enabled=true;

ClearAL->Enabled=true;

size\_al++;

for(int i = 0; i < size\_al; i++)

SrcAL->Cells[i+1][0] = IntToStr(i+1);

}

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

void \_\_fastcall TTForm1::bDelRangeColClick(TObject \*Sender)

{

if(SrcC->ColWidths[1]==-1)

return;

//// if (Application->MessageBox(Format(L"Вы действительно хотите удалить \"запись\" №%d?",

//// ARRAYOFCONST((SrcC->Col))).c\_str(),

//// MB\_YESNO | MB\_ICONQUESTION | MB\_DEFBUTTON2) != ID\_YES)

////

//// return;

//

//

if (Application->MessageBox((UnicodeString(L"Вы действительно хотите удалить выделенные строки (") +

SrcC->Selection.Left + L'-' +

SrcC->Selection.Right + L")?").w\_str(),

L"Определитель пропускной способности", MB\_YESNO | MB\_ICONQUESTION | MB\_DEFBUTTON2) != IDYES)

return;

int n = SrcC->Selection.Right - SrcC->Selection.Left + 1;

if(SrcC->ColCount-n>=2){

int a = SrcC->Selection.Right;

for (int i = SrcC->Selection.Left; i <= SrcC->Selection.Right+1; i++){

if (SrcC->Selection.Left == a)

SrcC->Cells[i][1] = SrcC->Cells[i+1][1];

else

SrcC->Cells[i][1] = SrcC->Cells[i + n][1];

}

SrcC->ColCount -= n;

}

else {

SrcC->ColCount=2;

SrcC->ColWidths[1]=-1;

SrcC->Options=SrcC->Options>>goEditing;

bProc->Enabled=false;

DelColC->Enabled = false;

bDelRangeCol->Enabled = false;

bFontC->Enabled=false;

ClearC->Enabled=false;;

}

}

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

void \_\_fastcall TTForm1::bDelRangeColALClick(TObject \*Sender)

{

if(SrcAL->ColWidths[1]==-1){

bProc->Enabled=false;

return;

}

if (Application->MessageBox((UnicodeString(L"Вы действительно хотите удалить выделенные строки (") +

SrcAL->Selection.Left + L'-' +

SrcAL->Selection.Right + L")?").w\_str(),

L"Определитель пропускной способности", MB\_YESNO | MB\_ICONQUESTION | MB\_DEFBUTTON2) != IDYES)

return;;

int n = SrcAL->Selection.Right - SrcAL->Selection.Left + 1;

if(SrcAL->ColCount-n>=2){

for (int i = SrcAL->Selection.Left; i <= SrcAL->Selection.Right+1; i++){

if (SrcAL->Selection.Left == SrcAL->Selection.Right){

SrcAL->Cells[i][1] = SrcAL->Cells[i+1][1];

SrcAL->Cells[i][2] = SrcAL->Cells[i+1][2];

}

else{

SrcAL->Cells[i][1] = SrcAL->Cells[i + n][1];

SrcAL->Cells[i][2] = SrcAL->Cells[i + n][2];

}

}

SrcAL->ColCount -= n;

}

else {

SrcAL->ColCount=2;

SrcAL->ColWidths[1]=-1;

SrcAL->Options=SrcAL->Options>>goEditing;

bProc->Enabled=false;

DelColAL->Enabled = false;

bDelRangeColAL->Enabled = false;

bFontAL->Enabled=false;

ClearAL->Enabled=false;

}

}

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

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

//

//

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

void \_\_fastcall TTForm1::SrcCDrawCell(TObject \*Sender, int ACol, int ARow, TRect &Rect,

TGridDrawState State)

{

// TCanvas \*a = ((TStringGrid\*)Sender)->Canvas;

// float d;

//

// // Проверка на выделение ячейки

// if (State.Contains(gdSelected))

// {

// // Установка цвета фона для выделенных ячеек (например, светло-голубой)

// a->Brush->Color = clSkyBlue;

// }

// else

// {

// // Установка цвета фона ячейки в белый по умолчанию

// a->Brush->Color = clWhite;

//

// // Если ячейка является зафиксированной, установите цвет фона на зеленый

// if (State.Contains(gdFixed))

// a->Brush->Color = clMoneyGreen;

//

// // Если ячейка не является зафиксированной и пустая, установите цвет фона на светло-желтый

// else if (((TStringGrid\*)Sender)->Cells[ACol][ARow] == L"")

// a->Brush->Color = clCream; // TColor(RGB(251, 251, 230));

// }

//

// // Заполнение фона ячейки выбранным цветом

// a->FillRect(Rect);

//

// // Определение прямоугольника для отображения текста ячейки

// TRect r = Rect;

// r.left += 2;

// r.top += 2;

//

// // Рисование текста ячейки по центру

// DrawText(a->Handle, ((TStringGrid\*)Sender)->Cells[ACol][ARow].t\_str(), -1, (TRect\*)&r, DT\_CENTER);

}

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

void \_\_fastcall TTForm1::bFontCClick(TObject \*Sender)

{

fdGrid->Font=SrcC->Font;

if(!fdGrid->Execute())

return;

SrcC->Font=fdGrid->Font;

SrcC->Canvas->Font->Assign(SrcC->Font);

SrcC->DefaultRowHeight=SrcC->Canvas->TextHeight(L"hy")+4;

if(cbEditAutoWidthC->Checked){

AdjustGridColWidths(SrcC);

SrcC->Options = SrcC->Options >> goColSizing;

}

else

SrcC->Options = SrcC->Options << goColSizing;

}

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

//void \_\_fastcall sgFontChanged(TObject \*Sender){

//

// SrcC->Canvas->Font->Assign(SrcC->Font);

// SrcC->DefaultRowHeigh=SrcC->Canvas->TextHeight(L"hy")+4;

// //if()

//}

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

//\_\_fastcall TForm1::TForm1(TComponent\* Owner)

// : TForm(Owner)

//{

//

// SrcC->Font->OnChange= sgFontChanged;

//

//}

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

void \_\_fastcall TTForm1::bFontALClick(TObject \*Sender)

{

fdGrid->Font=SrcAL->Font;

if(!fdGrid->Execute())

return;

SrcAL->Font=fdGrid->Font;

SrcAL->Font=fdGrid->Font;

SrcAL->Canvas->Font->Assign(SrcAL->Font);

SrcAL->DefaultRowHeight=SrcAL->Canvas->TextHeight(L"hy")+4;

if(cbEditAutoWidthAL->Checked){

AdjustGridColWidths(SrcAL);

SrcAL->Options = SrcAL->Options >> goColSizing;

}

else

SrcAL->Options = SrcAL->Options << goColSizing;

}

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

void \_\_fastcall TTForm1::ApplicationEvents1hint(TObject \*Sender)

{

sbInfo->SimpleText=Application->Hint;

}

void \_\_fastcall TTForm1::ApplicationEvents1ActionExecute(TBasicAction \*Action, bool &Handled)

{

//

}

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

typedef float TCmpFunc(const UnicodeString &s1, const UnicodeString &s2);

float cmpFunc1 (const UnicodeString &s1, const UnicodeString &s2){

return StrToFloat(s1) - StrToFloat(s2);

}

const int MIN\_COL\_WIDTH = 20;

void \_\_fastcall TTForm1::SrcCDblClick(TObject \*Sender)

{

TPoint p;

GetCursorPos(&p);

p=((TStringGrid\*)Sender)->ScreenToClient(p);

//p=((TStringGrid\*)Sender)->CalCursorPos();

if(GetCursor()==Screen->Cursors[crHSplit]){

TGridCoord gc = ((TStringGrid\*)Sender)->MouseCoord(p.X-4, p.Y);

int w = CalcGridColWidth((TStringGrid\*)Sender, gc.X);

if(((TStringGrid\*)Sender)->ColWidths[gc.X]!=w)

((TStringGrid\*)Sender)->ColWidths[gc.X]=w;

else

((TStringGrid\*)Sender)->ColWidths[gc.X]= MIN\_COL\_WIDTH;

}

else{

//

TGridCoord gc = ((TStringGrid\*)Sender)->MouseCoord(p.X, p.Y);

if (gc.X==0) { //заголовок строки

int i = abs(FSortedRowNbr)-1;

//

if(FSortedRowNbr){

((TStringGrid\*)Sender)->Cells[0][i] =

((TStringGrid\*)Sender)->Cells[0][i].SubString(1,

((TStringGrid\*)Sender)->Cells[0][i].Length()-2);

}

if(cbEditAutoWidthC->Checked)

((TStringGrid\*)Sender)->ColWidths[i]=

CalcGridColWidth(((TStringGrid\*)Sender), i);

// индекс нового столбца

if(FSortedRowNbr && i==gc.Y)

FSortedRowNbr = -FSortedRowNbr;

else

FSortedRowNbr = gc.Y+1;

//

i=gc.Y;

//

if(FSortedRowNbr < 0)

((TStringGrid\*)Sender)->Cells[0][i]=

((TStringGrid\*)Sender)->Cells[0][i] + L" \x25b6";

else

((TStringGrid\*)Sender)->Cells[0][i]=

((TStringGrid\*)Sender)->Cells[0][i] + L" \x25c0";

if(cbEditAutoWidthC->Checked)

((TStringGrid\*)Sender)->ColWidths[i]=

CalcGridColWidth(((TStringGrid\*)Sender), i);

((TStringGrid\*)Sender)->Cols[((TStringGrid\*)Sender)->Col]->Objects[0] =

(TObject\*)true;

SortGridByRow((TStringGrid\*)Sender, i, FSortedRowNbr>0,

cmpFunc1);

/\*? cmpFunc1 : cmpFunc2\*/

//

for (int i = 1; i < ((TStringGrid\*)Sender)->ColCount; i++) {

if (((TStringGrid\*)Sender)->Cols[i]->Objects[0]) {

((TStringGrid\*)Sender)->Cols[i]->Objects[0] = (TObject\*)false;

((TStringGrid\*)Sender)->Col = i;

break;

}

}

}

//

//

}

}

int CalcGridColWidth(TStringGrid \*grid, int colIdx){

if(!grid || colIdx<0 || colIdx >= grid->ColCount)

return 0;

grid -> Canvas -> Font->Assign(grid->Font);

int maxWidth = 0;

for(int i = 0; i < grid->ColCount; i++){

int textWidth = grid->Canvas->TextWidth(grid->Cells[colIdx][i]);

if (textWidth > maxWidth)

maxWidth = textWidth;

}

return maxWidth + MIN\_COL\_WIDTH;

}

void SortGridByRow(TStringGrid \*grid, int rowIdx, bool ascending, TCmpFunc cmpFunc) {

TStrings \*row = grid->Rows[rowIdx];

TStrings \*tmpCol = new TStringList();

try {

for (int i = grid->FixedCols; i < grid->ColCount - 1; i++) {

for (int j = i + 1; j < grid->ColCount; j++) {

if ((cmpFunc(row->Strings[i], row->Strings[j]) > 0 && ascending) ||

(cmpFunc(row->Strings[i], row->Strings[j]) < 0 && !ascending)) {

tmpCol->Assign(grid->Cols[i]);

grid->Cols[i]->Assign(grid->Cols[j]);

grid->Cols[j]->Assign(tmpCol);

}

}

}

} \_\_finally {

delete tmpCol;

}

}

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

void \_\_fastcall TTForm1::cbEditAutoWidthCClick(TObject \*Sender)

{

if(Sender==cbEditAutoWidthC)

A2->Checked = cbEditAutoWidthC->Checked;

else{

cbEditAutoWidthC->OnClick = NULL;

try{

cbEditAutoWidthC->Checked = A2->Checked;

}

\_\_finally{

cbEditAutoWidthC->OnClick = cbEditAutoWidthCClick;

}

}

if(cbEditAutoWidthC->Checked){

AdjustGridColWidths(SrcC);

SrcC->Options = SrcC->Options >> goColSizing >> goRowSizing;

}

else

SrcC->Options = SrcC->Options << goColSizing << goRowSizing;

}

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

int AdjustGridColWidths(TStringGrid \*grid){

if(!grid)

return 0;

int totalWidth = 0;

for (int j=0; j<grid->ColCount; j++) {

int w = CalcGridColWidth(grid, j);

grid->ColWidths[j] = w;

totalWidth += w;

}

if(!(grid->Options \* (TGridOptions() << goFixedVertLine << goVertLine)).Empty())

totalWidth += grid->ColCount \* grid->GridLineWidth;

return totalWidth;

}

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

void \_\_fastcall TTForm1::cbEditAutoWidthALClick(TObject \*Sender)

{

if(Sender==cbEditAutoWidthAL)

A1->Checked = cbEditAutoWidthAL->Checked;

else{

cbEditAutoWidthAL->OnClick = NULL;

try{

cbEditAutoWidthAL->Checked = A1->Checked;

}

\_\_finally{

cbEditAutoWidthAL->OnClick = cbEditAutoWidthALClick;

}

}

if(cbEditAutoWidthAL->Checked){

AdjustGridColWidths(SrcAL);

SrcAL->Options = SrcAL->Options >> goColSizing >> goRowSizing;

}

else

SrcAL->Options = SrcAL->Options << goColSizing << goRowSizing;

}

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

void \_\_fastcall TTForm1::cbEditAutoWidthResClick(TObject \*Sender)

{

if(cbEditAutoWidthRes->Checked){

AdjustGridColWidths(SrcRes);

SrcRes->Options = SrcRes->Options >> goColSizing >> goRowSizing;

}

else

SrcRes->Options = SrcRes->Options << goColSizing << goRowSizing;

}

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

void \_\_fastcall TTForm1::N6Click(TObject \*Sender)

{

Close();

}

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

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

void \_\_fastcall TTForm1::N2Click(TObject \*Sender)

{

if (Application->MessageBox(

UnicodeString(L"Вы действительно хотите очистить все таблицы?").w\_str(),

L"Определитель пропускной способности", MB\_YESNO | MB\_ICONQUESTION | MB\_DEFBUTTON2) != IDYES)

return;

//...................................................

for (int i = 1; i < SrcAL->ColCount; i++) {

SrcAL ->Cells[i][0] = L"";

//delete SrcC->Objects[i][1];

}

//SrcC->ColCount=1;

SrcAL->ColCount=2;

SrcAL->ColWidths[1]=-1;

bProc->Enabled=false;

DelColAL->Enabled = false;

bDelRangeColAL->Enabled = false;

bFontAL->Enabled=false;

ClearAL->Enabled=false;

////////////////////////////////////////////

for (int i = 1; i < SrcC->ColCount; i++) {

SrcC ->Cells[i][0] = L"";

//delete SrcC->Objects[i][1];

}

SrcC->ColCount=2;

SrcC->ColWidths[1]=-1;

bProc->Enabled=false;

DelColC->Enabled = false;

bDelRangeCol->Enabled = false;

bFontC->Enabled=false;

ClearC->Enabled=false;

for (int i = 0; i < SrcVsT->ColCount; i++)

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

for (int i = 0; i < SrcNKT->ColCount; i++)

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

bProc->Enabled=false;

DelColAL->Enabled = false;

bDelRangeColAL->Enabled = false;

bFontAL->Enabled=false;

ClearAL->Enabled=false;

DelColC->Enabled = false;

bDelRangeCol->Enabled = false;

bFontC->Enabled=false;

ClearC->Enabled=false;

}

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

void \_\_fastcall TTForm1::CheckBox2Click(TObject \*Sender)

{

if(CheckBox2->Checked == false)

QVgraph->View3D = false;

if(CheckBox2->Checked==true)

QVgraph->View3D = true;

}

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

void \_\_fastcall TTForm1::SrcCGetEditText(TObject \*Sender, int ACol, int ARow, UnicodeString &Value)

{

FCurCellText = Value;

}

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

void \_\_fastcall TTForm1::SrcALKeyPress(TObject \*Sender, System::WideChar &Key)

{

if (!SrcAL->EditorMode)

return;

if (Key==VK\_ESCAPE && SrcAL->EditorMode) {

SrcAL->Cells[SrcAL->Col][SrcAL->Row]=FCurCellText2;

return;

}

if(!((Key >= L'0' && Key <= L'9') || Key==L'-' || Key==L'+' || Key == FormatSettings.DecimalSeparator ||

Key ==L'e' || Key ==L'E') && Key!=VK\_BACK && Key!=VK\_RETURN && Key!=3 && Key!=22 &&

Key!= 24 && Key!= 26){

//bProc1->Enabled=false;

sbInfo ->SimpleText = L"Недопустимый символ '" + UnicodeString(Key) + L'\'';

Key = 0;

}

else {

sbInfo->SimpleText=L"";

bProc->Enabled=true;

}

}

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

void \_\_fastcall TTForm1::SrcALGetEditText(TObject \*Sender, int ACol, int ARow, UnicodeString &Value)

{

FCurCellText2 = Value;

}

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

void \_\_fastcall TTForm1::SrcNKTGetEditText(TObject \*Sender, int ACol, int ARow, UnicodeString &Value)

{

FCurCellText3 = Value;

}

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

void \_\_fastcall TTForm1::SrcNKTKeyPress(TObject \*Sender, System::WideChar &Key)

{

if (!SrcNKT->EditorMode)

return;

if (Key==VK\_ESCAPE && SrcNKT->EditorMode) {

SrcNKT->Cells[SrcNKT->Col][SrcNKT->Row]=FCurCellText3;

return;

}

if(!((Key >= L'0' && Key <= L'9') || Key==L'-' || Key==L'+' || Key == FormatSettings.DecimalSeparator ||

Key ==L'e' || Key ==L'E') && Key!=VK\_BACK && Key!=VK\_RETURN && Key!=3 && Key!=22 &&

Key!= 24 && Key!= 26){

//bProc1->Enabled=false;

sbInfo ->SimpleText = L"Недопустимый символ '" + UnicodeString(Key) + L'\'';

Key = 0;

}

else {

sbInfo->SimpleText=L"";

bProc->Enabled=true;

}

}

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

void \_\_fastcall TTForm1::SrcVsTKeyPress(TObject \*Sender, System::WideChar &Key)

{

if (!SrcVsT->EditorMode)

return;

if (Key==VK\_ESCAPE && SrcVsT->EditorMode) {

SrcVsT->Cells[SrcVsT->Col][SrcVsT->Row]=FCurCellText4;

return;

}

if(!((Key >= L'0' && Key <= L'9') || Key==L'-' || Key==L'+' || Key == FormatSettings.DecimalSeparator ||

Key ==L'e' || Key ==L'E') && Key!=VK\_BACK && Key!=VK\_RETURN && Key!=3 && Key!=22 &&

Key!= 24 && Key!= 26){

//bProc1->Enabled=false;

sbInfo ->SimpleText = L"Недопустимый символ '" + UnicodeString(Key) + L'\'';

Key = 0;

}

else {

sbInfo->SimpleText=L"";

bProc->Enabled=true;

}

}

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

void \_\_fastcall TTForm1::SrcVsTGetEditText(TObject \*Sender, int ACol, int ARow, UnicodeString &Value)

{

FCurCellText4 = Value;

}

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

void \_\_fastcall TTForm1::FormAfterMonitorDpiChanged(TObject \*Sender, int OldDPI, int NewDPI)

{

// if(a and b and c and d)

// bProc->Enabled=true;

// else

// bProc->Enabled=false;

}

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

bool oshib(TStringGrid\*SrcC,TStringGrid\*SrcAL,TStringGrid\*SrcNKT,TStringGrid\*SrcVsT,

TStatusBar\*sbInfo, TButton\*bProc){

int count=0,count2=0;

String uncor,unacc;

float alCheck,сСheck,VTcheck,grafCheck;

for (int i=0;i<SrcC->ColCount-1;i++){

if(SrcC->Cells[i+1][0]==L""){

continue;

}

try{

сСheck=StrToFloat(SrcC->Cells[i+1][0]);

if (сСheck<=0){

unacc+=L"C["+IntToStr(0)+L"]"+L"["+IntToStr(i+2)+L"] ";

count2++;

}

}

catch(EConvertError &e){

uncor+=L"C["+IntToStr(0)+L"]"+L"["+IntToStr(i+2)+L"] ";

count++;

}

}

for (int i=0;i<SrcAL->ColCount-1;i++){

for (int j=0;j<SrcAL->RowCount;j++){

try{

alCheck=StrToFloat(SrcAL->Cells[i+1][j]);

if (alCheck<0){

unacc+=L"al["+IntToStr(j+1)+L"]"+L"["+IntToStr(i+2)+L"] ";

count2++;

}

}

catch(EConvertError &e){

uncor+=L"al["+IntToStr(j+1)+L"]"+L"["+IntToStr(i+2)+L"] ";

count++;

}

}

}

for (int i=0;i<SrcNKT->ColCount-1;i++){

try{

VTcheck=StrToFloat(SrcNKT->Cells[i][1]);

if(VTcheck<=0){

unacc+=L"Интервал,шаг,время["+IntToStr(i)+L"]"+L"["+IntToStr(1)+L"] ";

count2++;

}

}

catch(EConvertError &e){

uncor+=L"Интервал,шаг,время["+IntToStr(i)+L"]"+L"["+IntToStr(1)+L"] ";

count++;

}

}

for (int i=0;i<SrcVsT->ColCount-1;i++){

try{

grafCheck=StrToFloat(SrcVsT->Cells[i][1]);

if(grafCheck<0){

unacc+=L"График["+IntToStr(i)+L"]"+L"["+IntToStr(1)+L"] ";

count2++;

}

}

catch(EConvertError &e){

uncor+=L"График["+IntToStr(i)+L"]"+L"["+IntToStr(1)+L"] ";

count++;

}

}

if(count==0 && count2==0){

bProc->Enabled=True;

return true;

}

if(count2!=0 && count==0){

sbInfo->SimpleText=L"Недопустимые значения в ячейках : "+unacc;

bProc->Enabled=False;

return false;

}

sbInfo->SimpleText=L"Некорректные значения в ячейках : "+uncor;

bProc->Enabled=False;

return false;

}

bool TTForm1::CheckChangesAndSave() {

// if (MSave -> Enabled) {

// String fname;

// if (!SaveGr->FileName.IsEmpty())

// fname = SaveGr->FileName;

// else

// fname = L"No\_Name";

//

// switch (Application->MessageBox((UnicodeString(L"Сохранить изменения в \"") + fname + L"\"?").w\_str(),

// Application->Title.w\_str(), MB\_YESNOCANCEL|MB\_ICONQUESTION)) {

// case ID\_YES:

// if (!SaveData(SaveGr->FileName.IsEmpty()))

// return false;

// break;

// case ID\_CANCEL:

// return false;

// }

// }

// return true;

}

void \_\_fastcall TTForm1::SaveGraphTypeChange(TObject \*Sender)

{

//

}

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

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

void \_\_fastcall TTForm1::ODdataShow(TObject \*Sender)

{

//

}

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

void \_\_fastcall TTForm1::N26Click(TObject \*Sender)

{

if(SaveGr->FileName.IsEmpty()){

sbInfo->SimpleText = "hello";

}

SaveData(SaveGr->FileName.IsEmpty());

}

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

bool \_\_fastcall TTForm1::SaveData(bool showDialog) {

String sfname = SaveGr->FileName;

SaveGr->InitialDir = "D:\\";

if (showDialog) {

if (!SaveGr->Execute())

return false;

}

try {

fstream f;

f.open(AnsiString(SaveGr->FileName).c\_str(), std::ios\_base::out | std::ios\_base::binary);

size\_C = SrcC -> ColCount-1;

f.write((char\*)&size\_C,sizeof(size\_C));

size\_al = SrcAL -> ColCount-1;

f.write((char\*)&size\_al,sizeof(size\_al));

for(int i = 0; i < size\_C; i++){

elem = StrToFloat(SrcC->Cells[i + 1][1]);

f.write((char\*)&elem,sizeof(elemType));

}

for (int i = 1; i < 3; i++) {

for (int j = 0; j < size\_al; j++) {

elem = StrToFloat(SrcAL->Cells[j + 1][i]);

f.write(reinterpret\_cast<const char\*>(&elem), sizeof(elemType));

}

}

for (int i = 0; i < 3; i++) {

elem = StrToFloat(SrcVsT->Cells[i][1]);

f.write(reinterpret\_cast<const char\*>(&elem), sizeof(elemType));

}

for (int i = 0; i < 3; i++) {

elem = StrToFloat(SrcNKT->Cells[i][1]);

f.write(reinterpret\_cast<const char\*>(&elem), sizeof(elemType));

}

f.close();

//

} catch (Exception &e) {

Application->MessageBox(Format(L"Ошибка сохранения файла \"%s\": \"%s\"",

ARRAYOFCONST((SaveGr->FileName, e.Message))).w\_str(),

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

SaveGr->FileName = sfname;

return false;

}

// mSave->Enabled = false;

// MemoForDataOutput -> Color = clWindow;

// Caption = "Редактирование - " + (SaveDialog1->FileName);

return true;

}

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

void \_\_fastcall TTForm1::N5Click(TObject \*Sender)

{

SaveData(true);

}

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

void \_\_fastcall TTForm1::N15Click(TObject \*Sender)

{

//

}

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

void \_\_fastcall TTForm1::BMPClick(TObject \*Sender)

{

GIF->Checked = false;

BMP->Checked = true;

JPEG->Checked = false;

//String FileName = String(SaveDialog1 -> FileName);

}

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

void \_\_fastcall TTForm1::SrcDataContextPopup(TObject \*Sender, TPoint &MousePos, bool &Handled)

{

//

}

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

void \_\_fastcall TTForm1::JPEGClick(TObject \*Sender)

{

GIF->Checked = false;

BMP->Checked = false;

JPEG->Checked = true;

}

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

void \_\_fastcall TTForm1::GIFClick(TObject \*Sender)

{

GIF->Checked = true;

BMP->Checked = false;

JPEG->Checked = false;

}

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

void \_\_fastcall TTForm1::FormCreate(TObject \*Sender)

{

int panelWidth = Panel3->Width;

bDelRangeCol->Left = (panelWidth - bDelRangeCol->Width) / 2;

DelColC->Left = (panelWidth - DelColC->Width) / 2;

ClearC->Left = DelColC->Left + DelColC->Width + 20;

bFontC ->Left = DelColC->Left + DelColC->Width + 20;

int distanceFromRight = Panel3->Width - (ClearC->Left + ClearC->Width);

bInsCol->Left = distanceFromRight;

AddColC->Left = distanceFromRight;

////////////

bDelRangeColAL->Left = (panelWidth - bDelRangeColAL->Width) / 2;

DelColAL->Left = (panelWidth - DelColAL->Width) / 2;

ClearAL->Left = DelColC->Left + DelColC->Width + 20;

bFontAL->Left = DelColC->Left + DelColC->Width + 20;

int distanceFromRight1 = Panel4->Width - (ClearAL->Left + ClearAL->Width);

bInsCol2->Left = distanceFromRight;

AddColAL->Left = distanceFromRight;

////////////

// Расстояние между нижней границей Panel3 и нижней границей формы

int distanceFromBottom3 = TForm1->ClientHeight - Panel3->Top - Panel3->Height;

// Расположение Panel4 так, чтобы она занимала середину между нижней границей Panel3 и нижней границей формы

Panel4->Top = Panel3->Top + Panel3->Height + (distanceFromBottom3 - Panel4->Height) / 2 - 60;

////////////

//SrcAL->Parent = Panel4;

// Устанавливаем выравнивание таблицы по верхней границе панели

//SrcAL->Align = alTop;

// Устанавливаем расстояние от верхней границы панели до верхней границы таблицы (если нужно)

SrcAL->Top = Panel4->Top + Panel4->Height;

}

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

void \_\_fastcall TTForm1::N22Click(TObject \*Sender)

{

//ТАБЛИЦА КАК...

}

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

void \_\_fastcall TTForm1::N23Click(TObject \*Sender)

{

// ..

}

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

void \_\_fastcall TTForm1::ExitClick(TObject \*Sender)

{

Close();

}

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

void \_\_fastcall TTForm1::SaveGrShow(TObject \*Sender)

{

//

}

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