МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

«САРАТОВСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИМЕНИ Н. Г. ЧЕРНЫШЕВСКОГО»

УТВЕРЖДАЮ
Зав.кафедрой,
к. ф.-м. н., доцент
_____ С. В. Миронов

ОТЧЕТ О ПРАКТИКЕ

студента 2 курса 211 группы факультета КНиИТ
Пицика Харитона Николаевича

вид практики: учебная
кафедра: математической кибернетики и компьютерных наук
курс: 2
семестр: 3
продолжительность: 18 нед., с 02.09.24 г. по 09.01.25 г.

Руководитель практики от университета,

доцент, к. ф.-м. н. _____ М. И. Сафрончик

Руководитель практики от организации (учреждения, предприятия),

доцент, к. ф.-м. н. _____ М. И. Сафрончик

Тема практики:«Разработка приложений Windows.Forms на языке C++ в среде Microsoft Visual Studio»

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Целью практики является освоение среды Visual Studio, а также отработка алгоритма построения оконных интерфейсов. В результате практики должны быть отработаны следующие навыки:

- создание нового проекта;
- добавление и конфигурация элементов управления;
- корректный ввод данных для решения поставленной задачи;
- разработка алгоритма для решения поставленной задачи;
- тестирование приложения;
- документирование кода.

1 Создание простого оконного приложения

Задание. Разработать приложение для вычисления факториала по приведенному примеру.

Создано окно приложения, содержащее два элемента TextBox, два элемента label и один элемент Button. Для отображения ошибок добавлен элемент ErrorProvider [1] (см. рисунок 1).

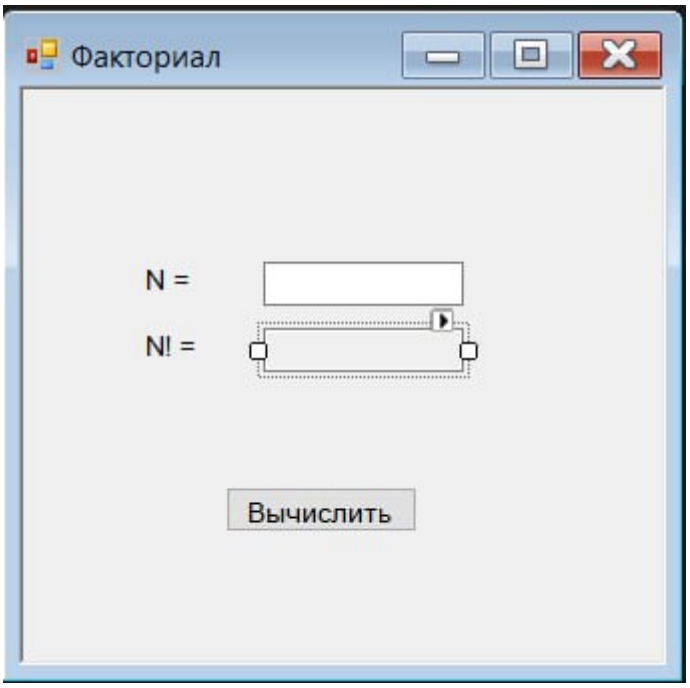

Рисунок 1 – Вид формы программы вычисления факториала

У элементов формы изменены значения некоторых свойств [2]. Значения измененных атрибутов представлены в таблице 1.

Таблица 1 – Значения атрибутов элементов формы

Свойство	Значение
Форма	
Text	Факториал
FormBorderStyle	Fixed3D
MaximizeBox	False
Первая надпись	
(Name)	label_input
Text	N =
Вторая надпись	
(Name)	label_output
Text	N! =
Первое текстовое поле	
(Name)	text_input
Второе текстовое поле	
(Name)	text_output
ReadOnly	True
Кнопка	
(Name)	btn_compute
Text	Вычислить
Обработчик ошибок	
(Name)	errorProvider1

Для работы программы был создан отдельный файл fact.h, содержащий код вычисления факториала [3]:

```
#pragma once
long long fact(long long n)
{
    //Если n < 0, то останавливаем с кодом -1
        if (n < 0)
                return -1;
    //Если на входе 0 или 1 - возвращаем 1
        else if (n == 0 || n == 1)
                return 1;
    //Иначе осуществляем рекурсивный вызов
        else
                return n * fact(n - 1);
}
```

Переменная n — число, для которого вычисляется факториал.

По нажатию кнопки «Вычислить» происходит выполнение следующего кода:

```
private: System::Void btn_compute_Click(System::Object^ sender, System::EventArgs^ e) {
    this->text_output->Text = "";
    long long input;
    bool result = Int64::TryParse(this->text_input->Text, input);
    //Если удалось прочитать число, то вычисляем
    //факториал.
    if (result)
    {
      long long output = fact(input);
      this->text_output->Text = System::Convert::ToString(output);
    }
    //В противном случае, с помощью ErrorProvider возвращаем ошибку.
    else
      errorProvider1->SetError(text_input, "Некорректные данные");
```

В случае корректных входных данных, программа посчитает факториал и выведет в text_output (см. рисунок 2).

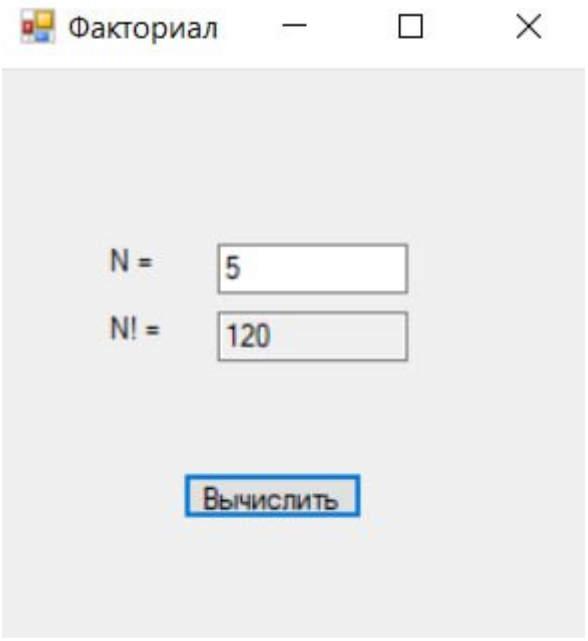

Рисунок 2 – «Факториал»: работа программы

В случае некорректных данных на входе, программа вернёт ошибку с помощью элемента errorProvider1 (см. рисунок 3).

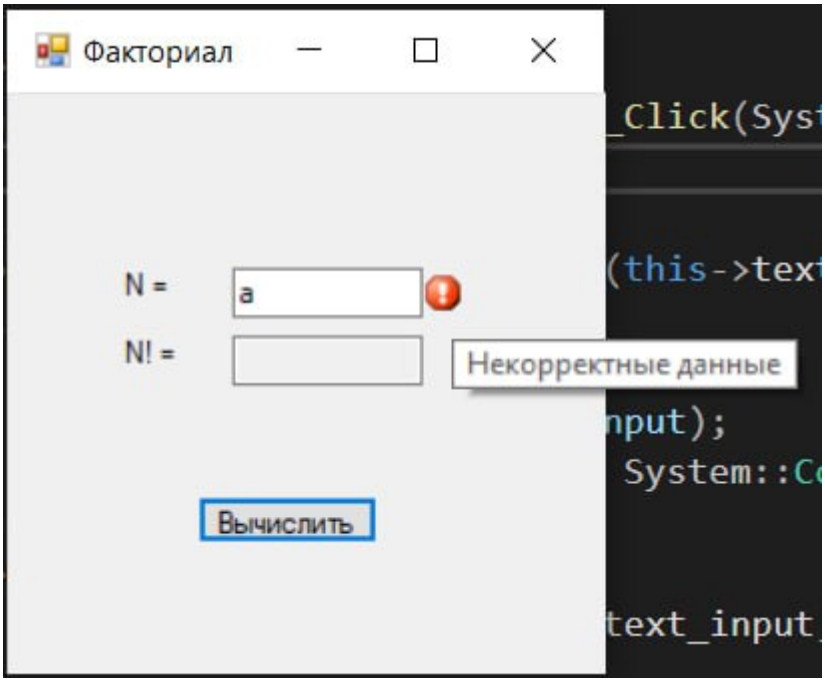

Рисунок 3 – «Факториал»: обработка ошибок

С полным кодом программы можно ознакомиться в приложении Б.

2 Обработка табличных данных

Задание. Все столбцы, содержащие минимальный элемент, заменить столбцом X.

Создано окно приложения, содержащее три элемента DataGridView и три элемента Button. Для перехвата и обработки ошибок добавлен элемент ErrorProvider (см. рисунок 4).

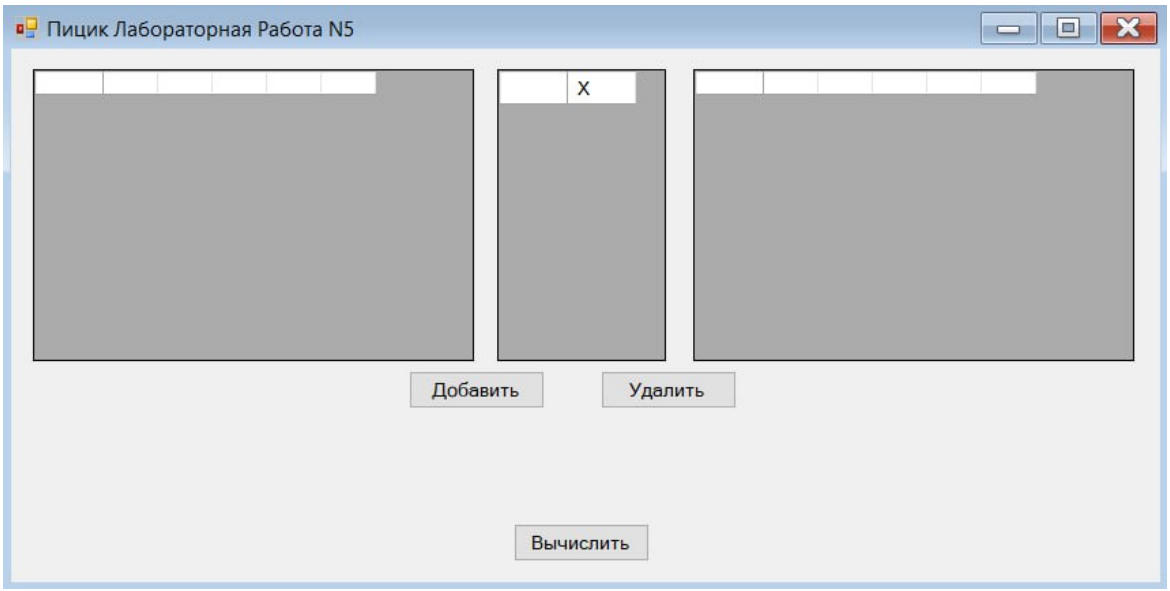

Рисунок 4 – Вид формы программы замены столбцов

У элементов формы изменены значения некоторых свойств, значения измененных атрибутов можно увидеть в таблице 2.

Таблица 2 – Значения атрибутов элементов формы

Свойство	Значение
Форма	
Text	Пицик Лабораторная работа N5
Кнопка "Добавить"	
(Name)	btn_add
Text	Добавить
Кнопка "Удалить"	
(Name)	btn_rm
Text	Удалить
Кнопка "Вычислить"	
(Name)	btn_calc
Text	Вычислить
DataGridView для ввода	
(Name)	data_first
AllowUserToAddRows	False
AllowUserToDeleteRows	False
DataGridView для столбца X	
(Name)	data_x
AllowUserToAddRows	False
AllowUserToDeleteRows	False
DataGridView для вывода	
(Name)	data_result
AllowUserToAddRows	False
AllowUserToDeleteRows	False
ColumnHeadersHeightSizeMode	AutoSize
ReadOnly	True
Обработчик ошибок	
(Name)	error_prov

С помощью параметров столбцов элемента DataGridView, контекстное меню которых можно вызвать, нажав на стрелочку около элемента и на свойство «Правка столбцов», у элемента data_x было добавлено название столбца «X» [4].

Для добавления столбцов, в кнопке «Добавить» был разработан следующий код [5]:

```
1 private: System::Void btn_add_Click(System::Object^ sender, System::EventArgs^ e)
2   {
3     data_first->Rows->Add();
```

```
    data_x->Rows->Add();
    data_result->Rows->Add();
  }
```

Для удаления столбцов, в кнопке «Удалить» был разработан следующий код:

```
private: System::Void btn_rm_Click(System::Object^ sender, System::EventArgs^ e) {
  if (data_first->CurrentRow != nullptr && !data_first->CurrentRow->IsNewRow)
  {
    int rowIndex = data_first->CurrentRow->Index;

    if (rowIndex < data_first->Rows->Count)
    // удаляем строку из data_first
    data_first->Rows->RemoveAt(rowIndex);

    // проверяем есть ли rowIndex в data_x
    if (rowIndex < data_x->Rows->Count)
      // удаляем строку из data_x по тому же индексу
      data_x->Rows->RemoveAt(rowIndex);

    // проверяем есть ли rowIndex в data_result
    if (rowIndex < data_result->Rows->Count)
      // удаляем строку из data_result по тому же индексу
      data_result->Rows->RemoveAt(rowIndex);
  }
}
```

В переменную rowIndex записывается индекс выбранной строки DataGridView. Затем, для предотвращения ошибок, перед удалением строки из каждого элемента DataGridView, производится проверка, что индекс выбранной строки не превышает общее количество строк каждого элемента. Затем, если условие соблюдается, осуществляется удаление строк.

Для корректного отображения ошибок, была реализована вспомогательная функция ClearAll():

```
void ClearAll()
{
    error_prov->SetError(btn_calc, "");
}
```

С остальными кодами можно ознакомиться в приложении А.

При запуске приложения открывается окно(см. рисунок 5).

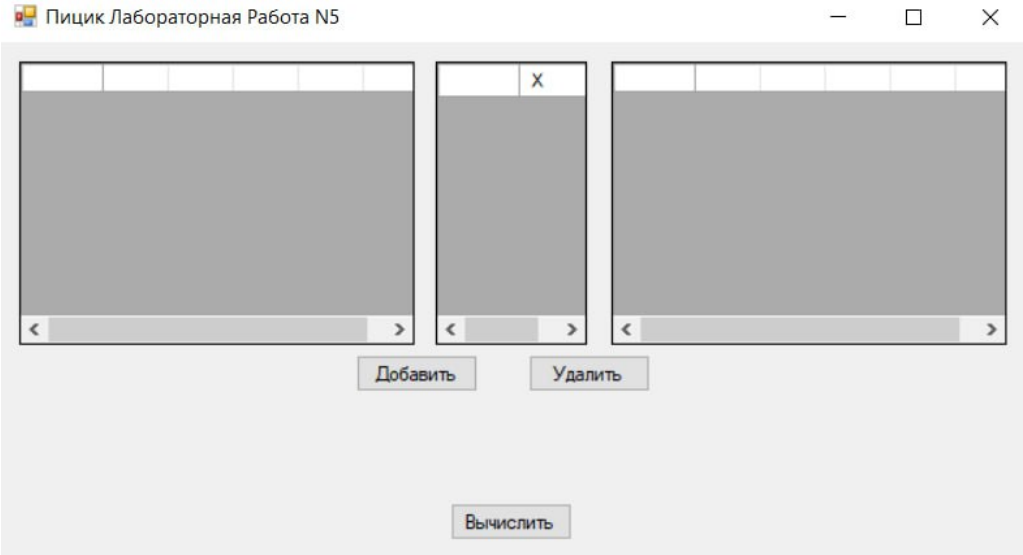

Рисунок 5 – «Табличные данные»: звпуск программы

Ввод некорректных данных обрабатывается с помощью ErrorProvider и сопровождается выводом сообщения об ошибке (см. рисунок 6).

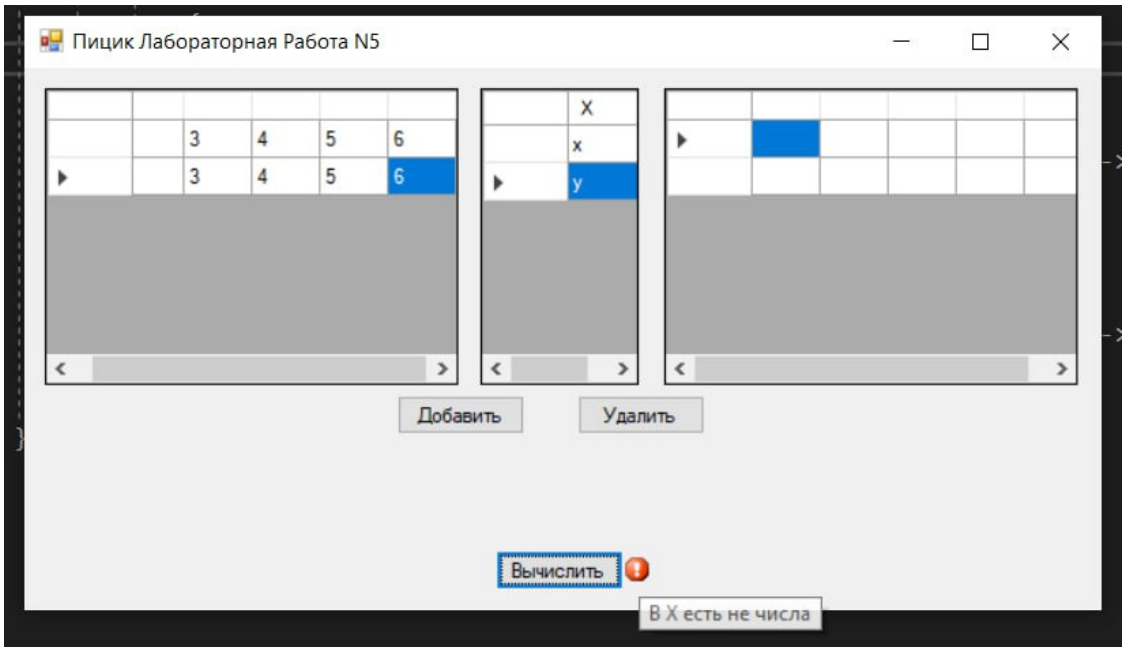

Рисунок 6 – «Табличные данные»: сообщение об ошибке

Пример корректной работы программы изображен на рисунке 7.

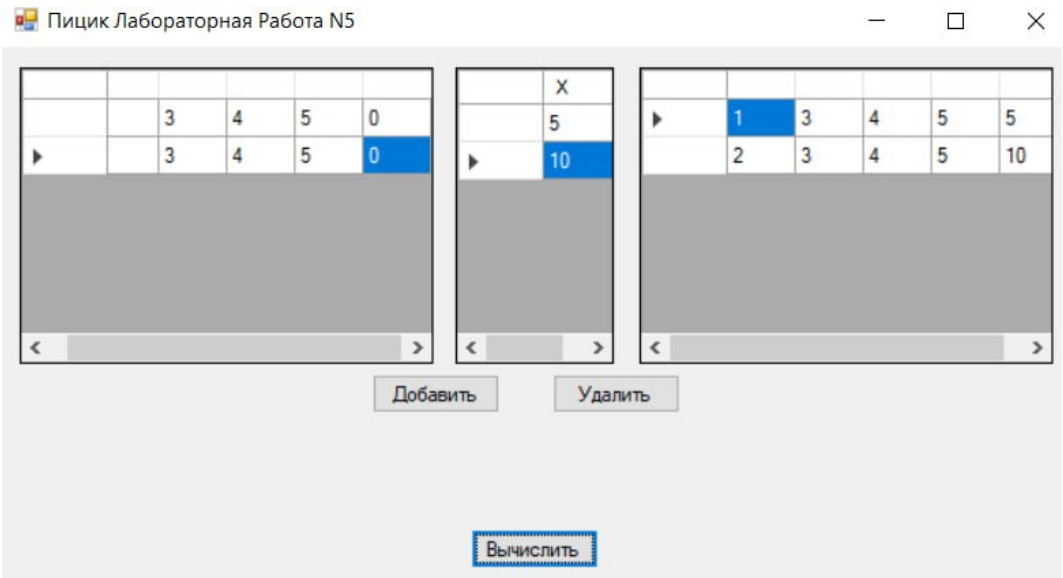

Рисунок 7 – «Табличные данные»: работа программы

Полный код программы приведён в приложении Б.

3 Использование коллекций в Windows Forms

Задание. Создать словарь, состоящий из строк. В качестве ключа выступает фамилия, в качестве значения — должность. Вывести на экран фамилии людей, занимающих данную должность. Вывести должность, занимаемую данным человеком.

Для выполнения данного задания была создана форма, содержащая один элемент DataGridView, три элемента Label, четыре элемента TextBox и три элемента Button (см. рисунок 8).

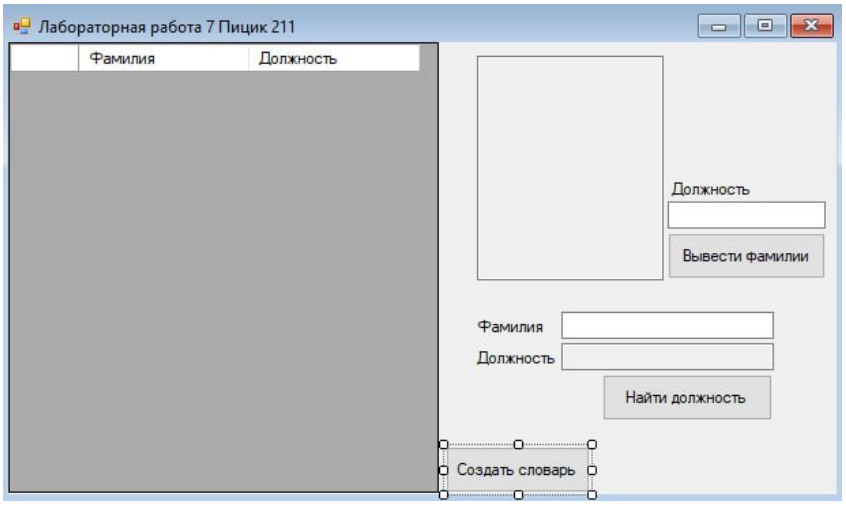

Рисунок 8 – Вид формы программы обработки отсортированного словаря

У элементов формы изменены значения некоторых свойств. С полным списком измененных свойств можно ознакомиться в таблице 3.

Таблица 3 – Значения атрибутов элементов формы

Свойство	Значение
Форма	
Text	Лабораторная Работа 7 Пицик 211
Верхняя надпись «Должность»	
(Name)	label_work1
Text	Должность
Надпись «Фамилия»	
(Name)	label_surname
Text	Фамилия
Нижняя надпись «Должность»	
(Name)	label_work
Text	Должность
Текстовое поле для ввода должности	
(Name)	text_work1
Текстовое поле для вывода фамилий	
(Name)	text_list
ReadOnly	True
Multiline	True
Текстовое поле для ввода фамилии	
(Name)	text_surname
Текстовое поле для вывода должности	
(Name)	text_work
ReadOnly	True
Кнопка «Создать словарь»	
(Name)	btn_dict
Text	Создать словарь
Кнопка «Вывести фамилии»	
(Name)	btn_list
Text	Вывести фамилии
Кнопка «Найти должность»	
(Name)	btn_find
Text	Найти должность
DataGridView для колллекции	
(Name)	dgv_table
AllowUserToAddRows	False
AllowUserToDeleteRows	False
ReadOnly	True

С помощью свойства Enabled элемента button_dict осуществляется запрет

на повторное нажание кнопки [6]. Создание словаря осуществляется с помощью следующего кода:

```
System::Collections::Generic::SortedDictionary <String^, String^> dict;
```

Здесь dict — название переменной, которое будет иметь отсортированный словарь. Основные свойства и методы словаря описаны в таблице 4 [7].

Таблица 4 – Свойства и методы словаря

Свойства	
dict.Count()	возвращает количество элементов словаря
dict.Keys()	возвращает коллекцию, содержащую ключи словаря
dict.Values()	возвращает коллекцию, содержащую значения словаря
Методы	
dict.Clear()	очистка словаря
dict.Add(X)	добавление нового элемента в словарь
dict.ContainsKey(X)	возвращает true, если ключ X есть в словаре
dict.ContainsValue(X)	возвращает true, если значение X есть в словаре
dict.Remove(X)	удаляет элемент X
dict.TryGetValue(X)	возвращает true, если ключ X есть в словаре и возвращает значение

Код кнопки «Создать словарь» выглядит следующим образом:

```
private: System::Void btn_dict_Click(System::Object^ sender, System::EventArgs^ e) {
    this->btn_dict->Enabled = false;
    System::Collections::Generic::SortedDictionary <String^, String^> dict;

    //Элементы словаря
    dict.Add("Пицик", "Старший разработчик");
    dict.Add("Коновалов", "DevOps инженер");
    dict.Add("Дрождев", "Разработчик");
    dict.Add("Цой", "ML-инженер");
    dict.Add("Артамонова", "Средний разработчик");
    dict.Add("Карасёв", "DevOps инженер");
    dict.Add("Воронцов", "Prompt инженер");
    dict.Add("Гришин", "Full-stack разработчик");
    dict.Add("Николаев", "GameDev разработчик");
    dict.Add("Иванов", "Младший разработчик");

    for each (KeyValuePair<String^, String^> item in dict)
    {
```

```
19        int row = dgv_table->Rows->Add();
20        dgv_table->Rows[row]->Cells[0]->Value = item.Key;
21        dgv_table->Rows[row]->Cells[1]->Value = item.Value;
22      }
23    }
```

Создаётся словарь dict, заполняется с помощью метода dict.Add(), а затем элементы словаря переносятся в окно элемента dgv_table [8]. Для корректной работы KeyValuePair необходимо разрешить пространство имён Generic:

```
1 using namespace System::Collections::Generic;
```

С остальными кодами можно ознакомиться в приложении А.

При открытии приложения появляется окно (см. рисунок 9).

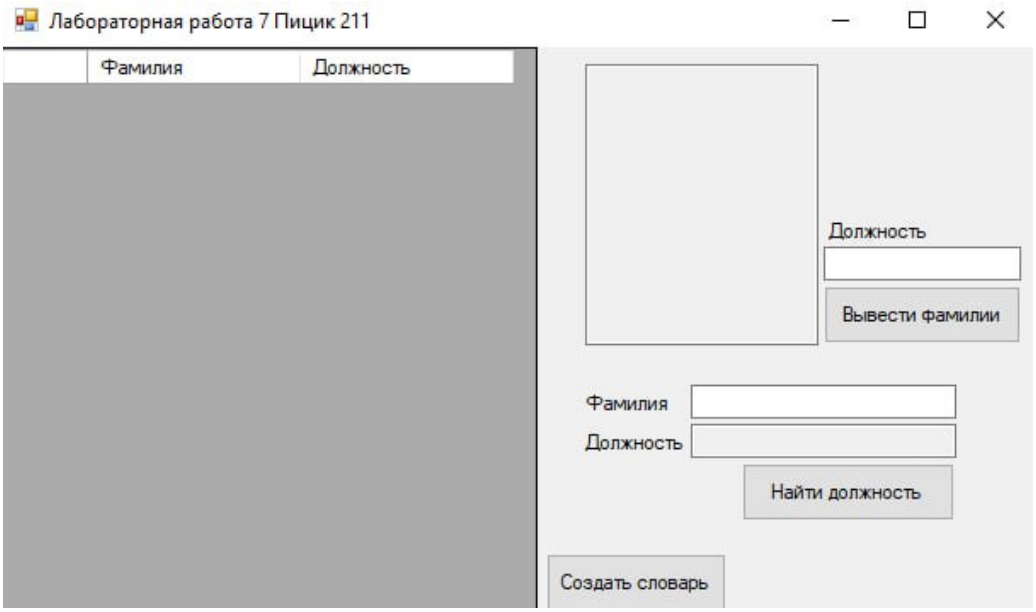

Рисунок 9 – Окно приложения «Коллекции»: начальный запуск

При нажатии кнопки «Создать словарь» обновляется содержимое DataGidView (см. рисунок 10).

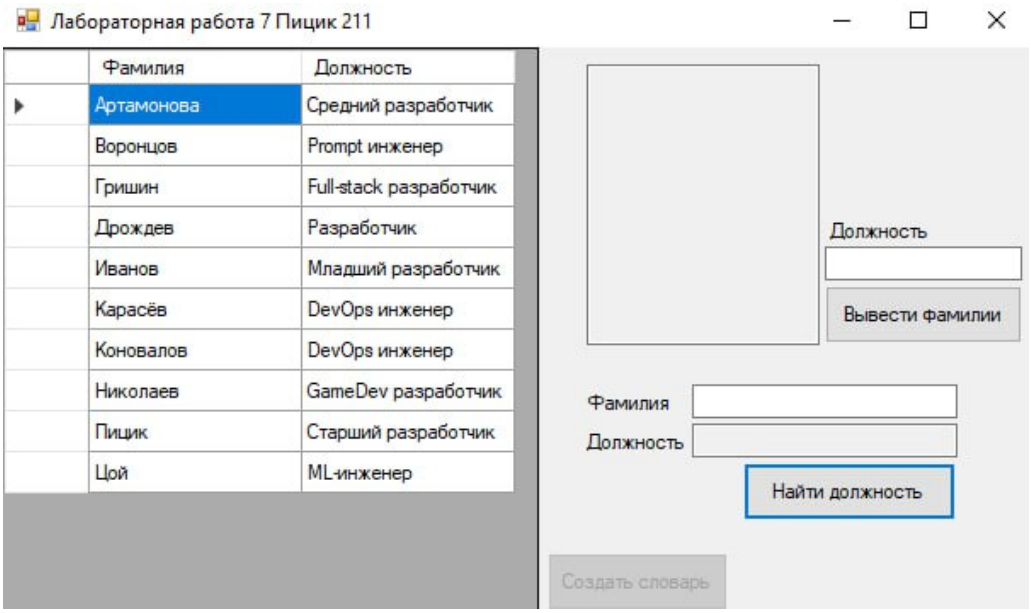

Рисунок 10 – «Коллекции»: создание словаря

При поиске фамилий по должности или должности по фамилии и корректном вводе данных, выводится соответствующий результат как показано на рисунке 11.

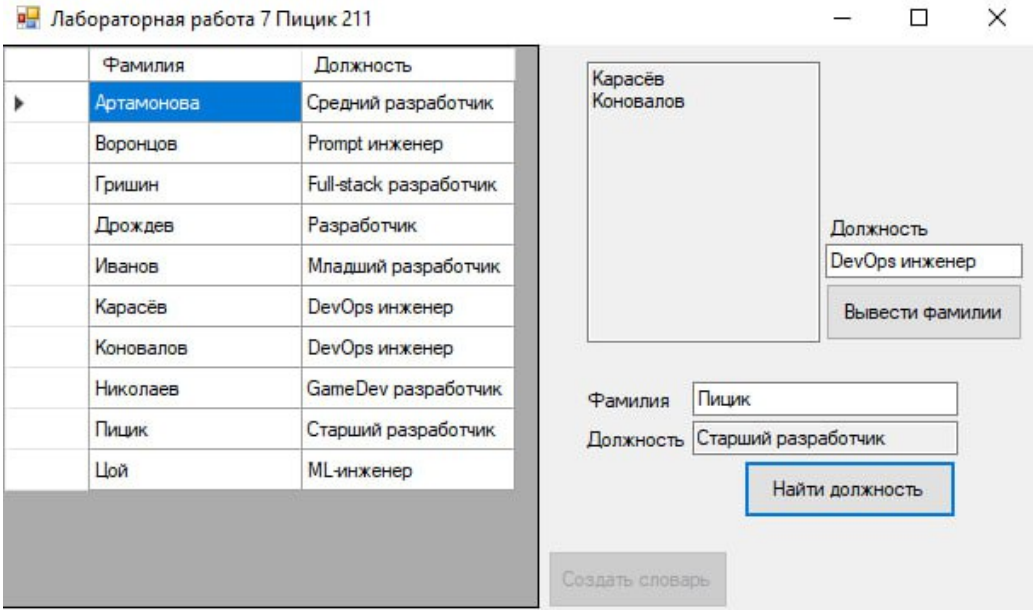

Рисунок 11 – «Коллекции»: поиск при корректных данных

При поиске с некорректными данными, выводятся соответствующие ошибки (см. рисунок 12).

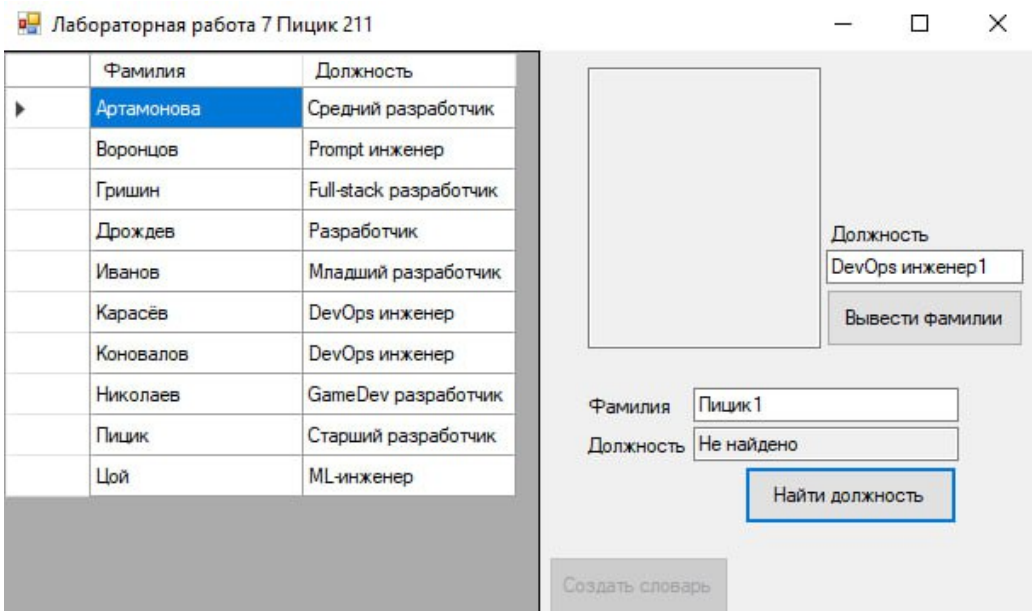

Рисунок 12 – «Коллекции»: вывод ошибок

Полный код программы доступен в приложении Б.

4 Файловые диалоги и работа с файлами

Задание. Создать таблицу Student. В другой файл вывести студентов, у которых сдана сессия. (Вариант 1)

Для выполнения данного задания была разработна форма, содержащая два элемента DataGridView, шесть элементов Button, элементы OpenFileDialog и SaveFileDialog для работы с файлами, и элемент ErrorProvider (см. рисунок 13).

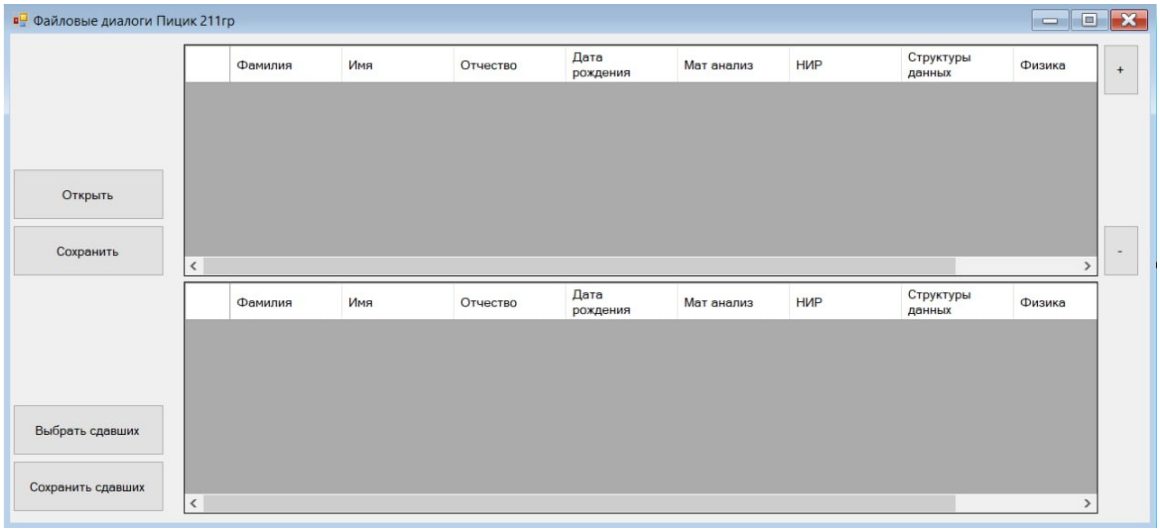

Рисунок 13 – Итоговый вид окна

У элементов формы изменены значения некоторых свойств. С полным списком измененных свойств можно ознакомиться в таблице 5.

Таблица 5 – Значение атрибутов элементов формы

Форма	
Text	Файловые диалоги Пицик 211гр.
DataGridView для ввода	
(Name)	data_inuput
AllowUserToAddRows	False
AllowUserToDeleteRows	False
DataGridView для вывода	
(Name)	data_output
AllowUserToAddRows	False
AllowUserToDeleteRows	False
Кнопка «Открыть»	
(Name)	btn_open
Text	Открыть
Кнопка «Сохранить»	
(Name)	btn_save
Text	Сохранить
Кнопка «Выбрать сдавших»	
(Name)	btn_choose
Text	Выбрать сдавших
Кнопка «Сохранить сдавших»	
(Name)	btn_savechoose
Text	Сохранить сдавших
Кнопка «+»	
(Name)	btn_add
Text	+
Кнопка « - »	
(Name)	btn_remove
Text	-
Файловые диалог «Открыть»	
(Name)	open_fd
Title	Открыть файл
Filter	Текстовые файлы (*.txt) | *.txt | Все файлы (*.*) | *.*
Файловые диалог «Сохранить»	
(Name)	save_fd
Title	Сохранить файл
Filter	Текстовые файлы (*.txt) | *.txt | Все файлы (*.*) | *.*

При нажатии кнопки «Открыть» открывается окно, позволяющее выбрать файлы форматов, заданных в свойстве Filter для элемента OpenFileDialog (см.

рисунок 14) [9].

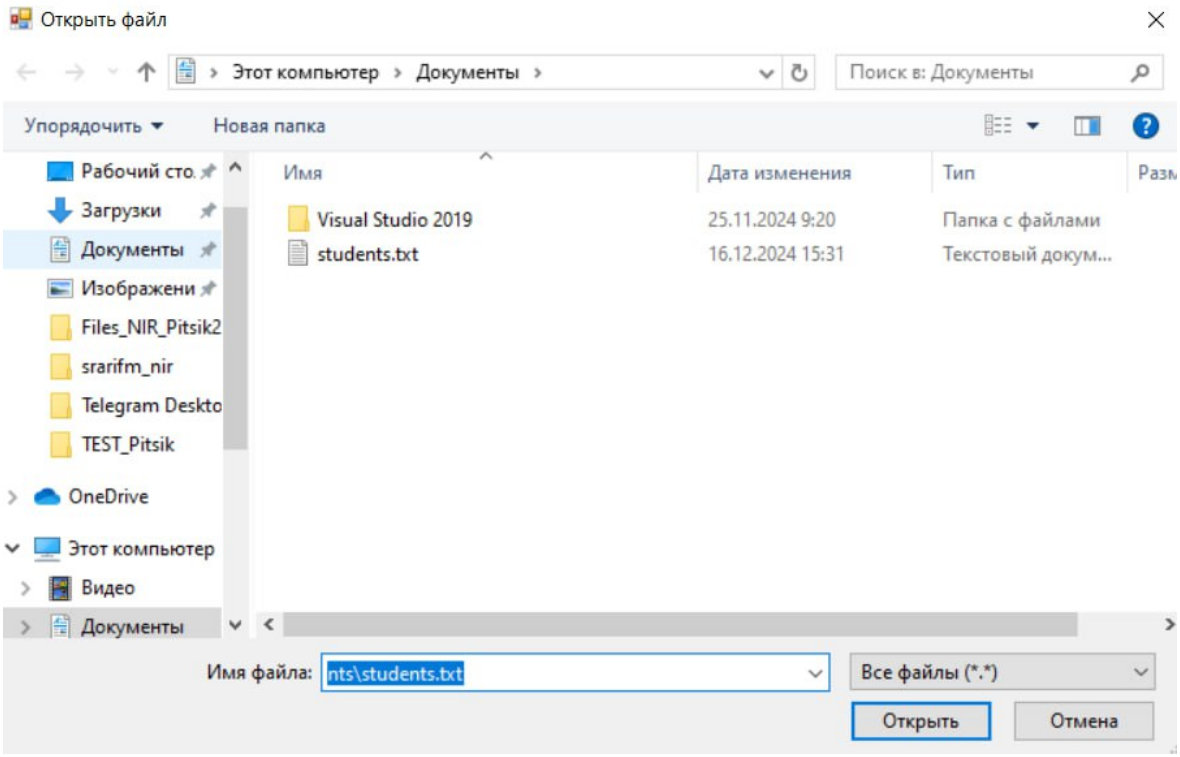

Рисунок 14 – Открытие файла

Для кнопнки «Открыть» был разработан следующий код:

```
private: System::Void btn_open_Click(System::Object^ sender, System::EventArgs^ e) {
    error_prov->Clear();

//открываем поток
    System::IO::Stream^ myStream;
    if (this->openFileDialog->ShowDialog() == System::Windows::Forms::DialogResult::OK)
  //если успешно найден файл
      if ((myStream = openFileDialog->OpenFile()) != nullptr)
      {
//задаем кодировку чтения
        System::IO::StreamReader^ sr = gcnew System::IO::StreamReader(
          myStream, System::Text::Encoding::GetEncoding(1251)
        );
//заполняем dgv
        int row = 0;
        System::String^ str1 = "";
        while (sr->Peek() > -1)
        {
          str1 = sr->ReadLine();
```

```
20          array<String^>^ words = str1->Split( ' ');
21          dgv_input->Rows->Add(words);
22        }
23  //закрываем поток
24        sr->Close();
25      }
26    }
```

В случае сохранения файла, необходимо нажать кнопку «Сохранить», фрагмент кода которой представлен ниже [10].

```
1  System::IO::Stream^ myStream;
2  //открываем окно сохранения файла
3       if (this->saveFileDialog->ShowDialog() == System::Windows::Forms::DialogResult::OK)
4       {
5         if ((myStream = saveFileDialog->OpenFile()) != nullptr)
6         {
7  //запись файла
8           System::IO::StreamWriter^ sw = gcnew System::IO::StreamWriter(myStream,
            ↪  System::Text::Encoding::GetEncoding(1251));
9  //считываем с итоговой таблицы
10          for (int i = 0; i < dgv_output->RowCount; ++i)
11          {
12            String^ s;
13            for (int j = 0; j < dgv_output->ColumnCount; ++j)
14              s += System::Convert::ToString(dgv_output->Rows[i]->Cells[j]->Value) + " ";
15 //записываем строку
16            sw->WriteLine(s);
17          }
18 //закрываем поток
19          sw->Close();
20        }
21      }
```

С остальными кодами можно ознакомиться в приложении А.

При запуске программы открывается окно (см. рисунок 15).

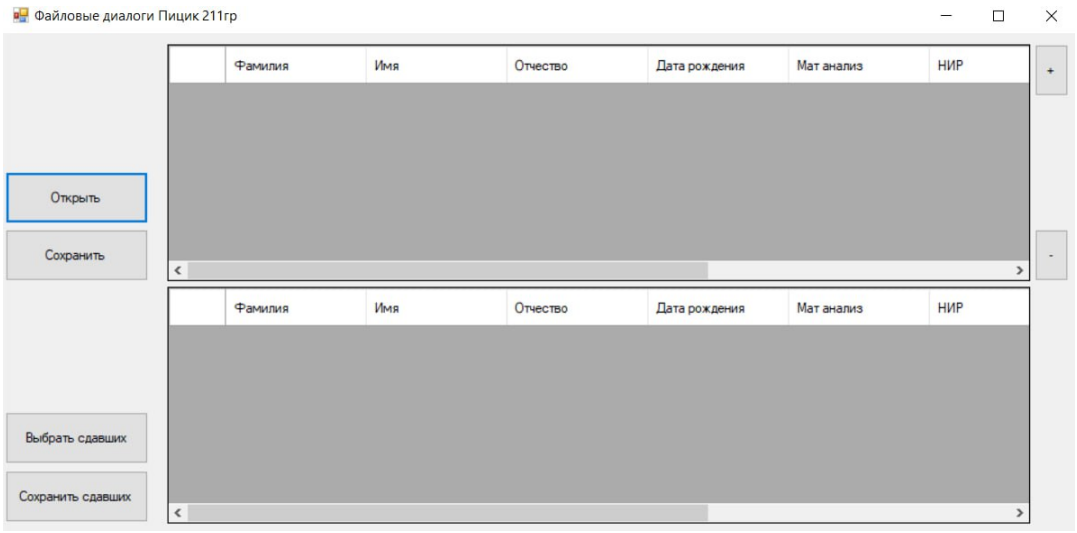

Рисунок 15 – Окно приложения «Файловые диалоги»: запуск программы

При нажатии на кнопку «Выбрать сдавших» в нижний элемент DataGridView выводится список сдавших студентов (см. рисунок 16).

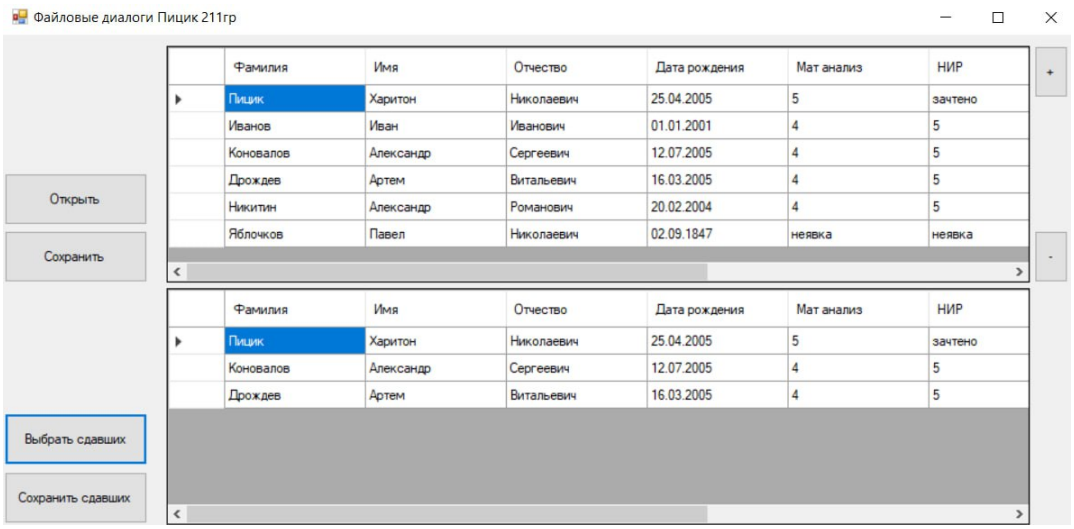

Рисунок 16 – «Файловые диалоги»: вид окна после выбора сдавших

При попытке добавить в исходную таблицу строку, содержащую некорректное значение даты или оценки, выводится ошибка (см. рисунок 17).

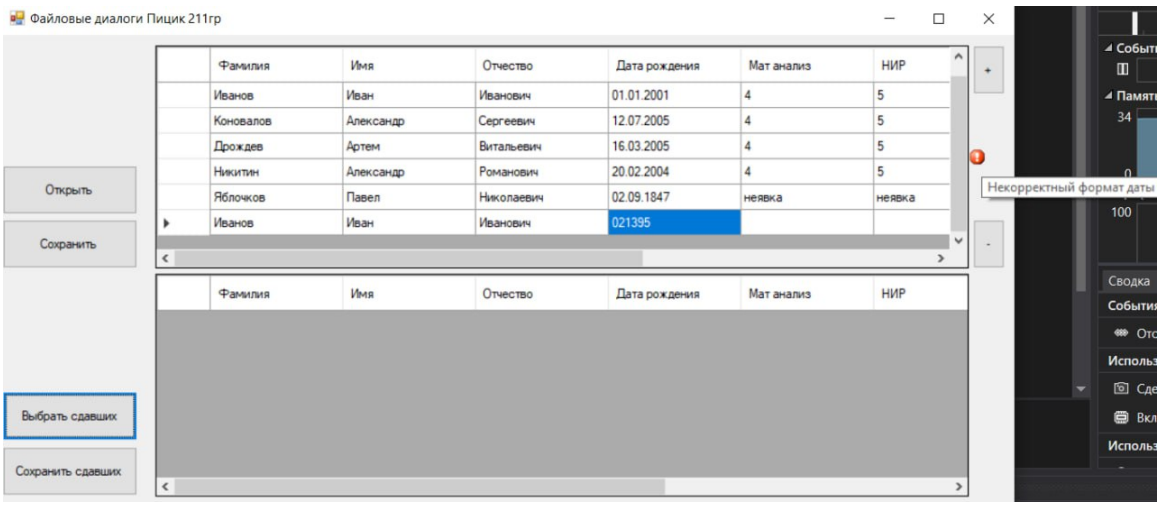

Рисунок 17 – «Файловые диалоги»: вывод ошибки в окне программы

Полный код программы приведён в приложении Б.

5 Приложение ТЕСТ

Задание. Создать приложение для проведения тестирования. Должно содержать:

1. Набор вопросов по какой-то теме (и вопросы и ответы должны быть реальные) — не менее 10
2. Вопросы должны выбираться случайным образом.
3. Вопросы должны быть нескольких типов — "Да/нет Выбор одного ответа, Выбор нескольких ответов, Короткий ответ.
4. Необходимо создать сообщения для правильного и неправильного ответа (Молодец, Не правильно и т.д.)
5. Необходимо подсчитать количество правильных ответов и вывести результат.

Для выполнения данного задания была разработана форма, содержащая два элемента TextBox, четыре элемента GroupBox, шесть элементов RadioButton, четыре элемента CheckBox и один элемент Button (см. рисунок 18).

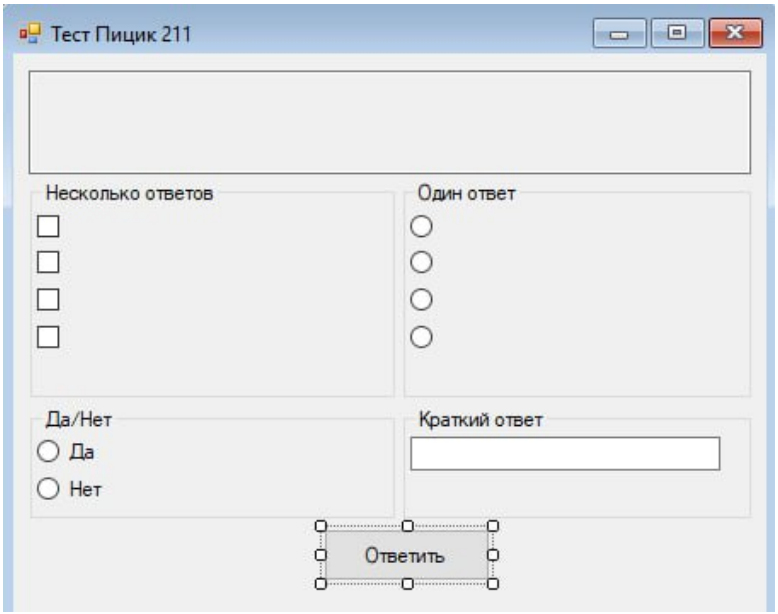

Рисунок 18 – Окно приложения «Тест» открытое в конструкторе

У элементов формы изменены значения некоторых свойств. Значения изменённых атрибутов представлены в таблицах 6 и 7.

Таблица 6 – Значение артибутов элементов формы, часть 1

Форма	
Text	Тест Пицик 211
Окно вывода вопросов	
(Name)	text_questions
ReadOnly	True
Группа для нескольких ответов	
(Name)	group_multiple
Text	Несколько ответов
Несколько ответов: 1-й вариант	
(Name)	check_1
Text	
Несколько ответов: 2-й вариант	
(Name)	check_2
Text	
Несколько ответов: 3-й вариант	
(Name)	check_3
Text	
Несколько ответов: 4-й вариант	
(Name)	check_4
Text	
Группа для одного ответа	
(Name)	group_single
Text	Один ответ
Один ответ: 1-й вариант	
(Name)	radio_single1
Text	
Один ответ: 2-й вариант	
(Name)	radio_single2
Text	
Один ответ: 3-й вариант	
(Name)	radio_single3
Text	
Один ответ: 4-й вариант	
(Name)	radio_single4
Text	

Таблица 7 – Значение артибутов элементов формы, часть 2

Группа для Да/Нет	
(Name)	group_yesno
Text	Да/нет
Да/нет: да	
(Name)	radio_yes
Text	Да
Да/нет: нет	
(Name)	radio_no
Text	Нет
Группа для краткого ответа	
(Name)	group_txt
Text	Краткий ответ
Краткий ответ: текст	
(Name)	text_answer
Кнопка для ответа	
(Name)	btn_answer
Text	Ответить

Были созданы отдельные структуры для отслеживания типа вопроса, вариантов ответа и правильного ответа:

```
// структура для Теста
public enum struct Test
    {
        yes_no,
        single,
        multiple,
        txt_answer
    };

    //структура вопросов
    public ref struct Questions
    {
        System::String^ text; //формулировка вопроса
        array<System::String^>^ answers; //варианты ответа
        Test type; //тип ответа
        System::Object^ answer; //ответ
    };
```

Была создана отдельная функция для перемешивания всех вопросов:

```
// функция для перемешивания вопросов
void randomize_questions(array<Questions^>^ arr) {
        Random^ rand = gcnew Random();
        for (int i = arr->Length - 1; i > 0; --i)
        {
            int j = rand->Next(0, i + 1);

            Questions^ temp = arr[i];
            arr[i] = arr[j];
            arr[j] = temp;
        }
    }
```

Фрагмент массива, содержащего все вопросы для теста:

```
//массив всех вопросов
array<Questions^>^ test_dataset() {
    array<Questions^>^ q = gcnew array<Questions^>(12);

    q[0] = gcnew Questions();
    q[0]->text = "C++, Python, C - это всё _______ (полное словосочетание)";
    q[0]->type = Test::txt_answer;
    q[0]->answer = "языки программирования";

    q[1] = gcnew Questions();
    q[1]->text = "Какой тип данных описывает целочисленный тип?";
    q[1]->type = Test::single;
    q[1]->answers = gcnew array<String^>{ "integer", "float", "boolean", "string" };
    q[1]->answer = 0;
```

С остальными кодами можно ознакомиться в приложении А.

После запуска приложения появляется окно (см. рисунок 19).

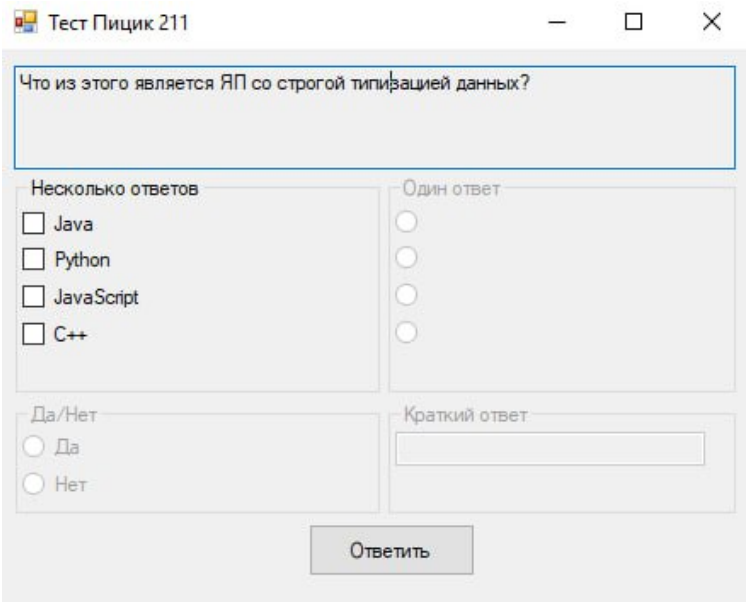

Рисунок 19 – Окно приложения «Тест»: начальный запуск

При выборе правильного варианта ответа появляется MessageBox с надписью (см. рисунок 20).

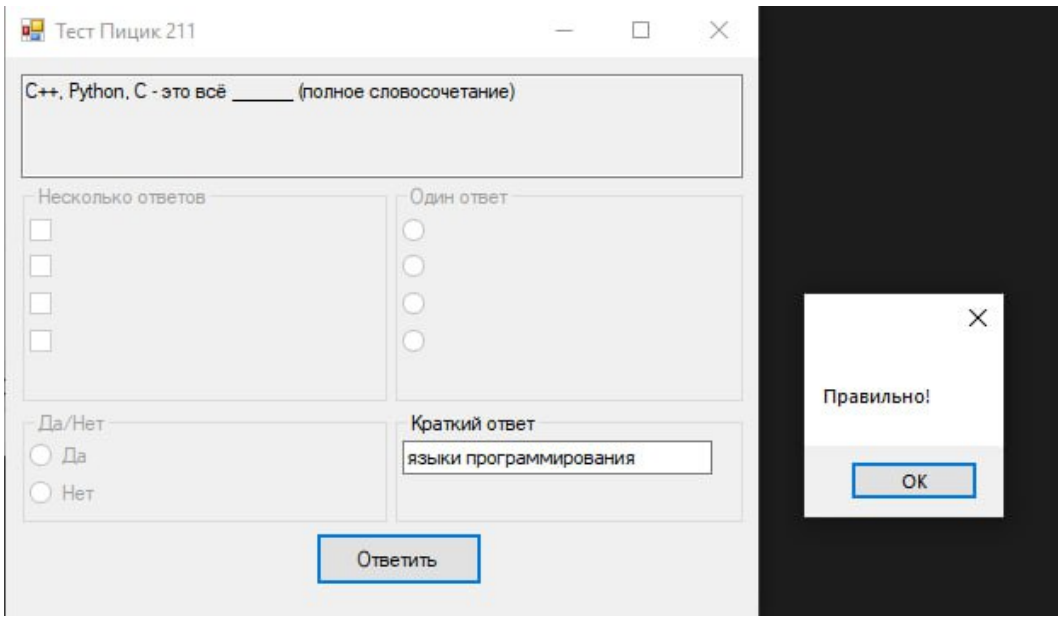

Рисунок 20 – «Тест»: выбор правильного ответа

При выборе неправильного варианта, выводится другое сообщение (см. рисунок 21).

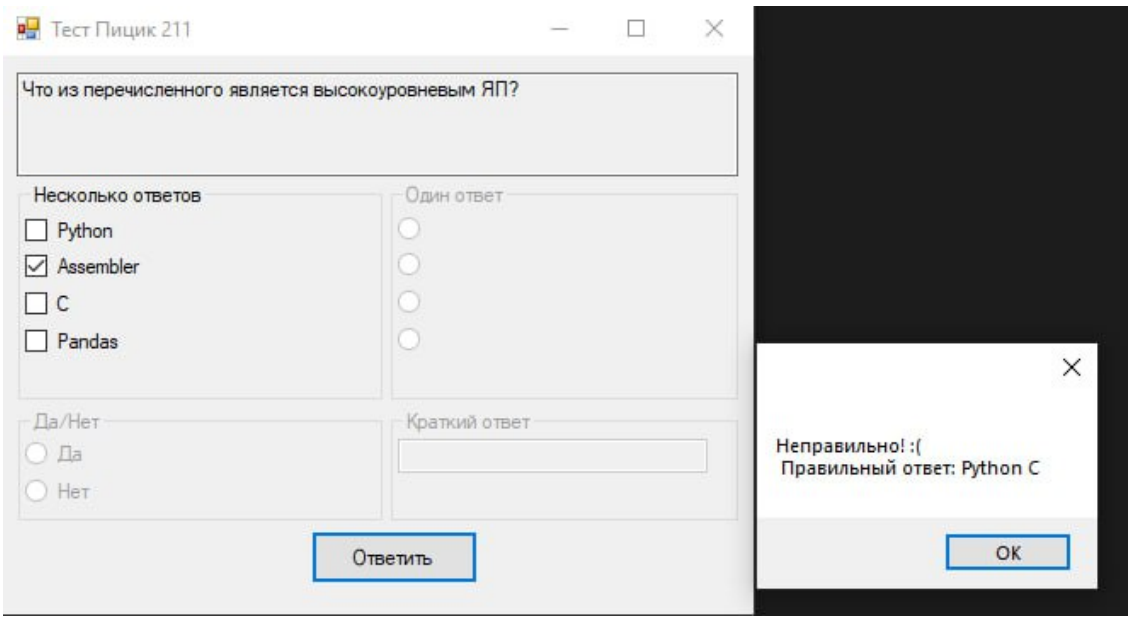

Рисунок 21 – «Тест»: выбор неправильного ответа

По прохождению теста выводится сообщение о завершении теста, а также счёт пользователя (см. рисунок 22).

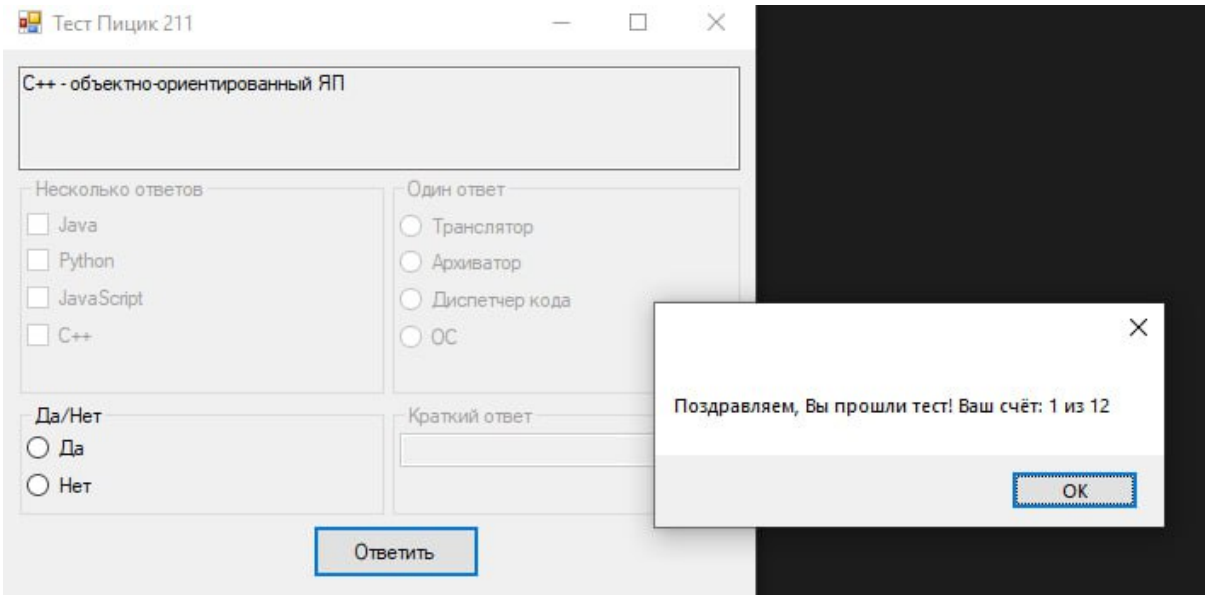

Рисунок 22 – «Тест»: завершение теста

С полным кодом программы можно ознакомиться в приложении Б.

ЗАКЛЮЧЕНИЕ

В ходе практики было реализовано несколько приложений на языке C++ в среде Microsoft Visual Studio с целью закрепления навыков построения оконного интерфейса. На практике разработаны приложения, содержащие такие элементы интерфейса как TextBox, Label, Button, DataGridView, OpenFileDialog, SaveFileDialog, ErrorProvider.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1 ErrorProvider Класс [Электронный ресурс]. — URL: https://learn.microsoft.com/en-us/dotnet/api/system.windows.forms.errorprovider?view=windowsdesktop-8.0 (Дата обращения 01.10.2024). Загл. с экр. Яз. анг.

2 Руководство. Создание приложения Windows Forms с помощью .NET [Электронный ресурс]. — URL: https://learn.microsoft.com/ru-ru/dotnet/desktop/winforms/get-started/create-app-visual-studio?view=netdesktop-9.0 (Дата обращения 01.10.2024). Загл. с экр. Яз. рус.

3 *Н.Б.Культин,* // Microsoft Visual C++ в задачах и примерах. — БХВ-Петербург, 2018.

4 *В.В.Зиборов,* // MS Visual C++ 2010 в среде .NET. — Библиотека программиста, 2012.

5 *Г.Хогенсон,* // C++/CLI: язык Visual C++ для среды : NET. — Диалектика / Вильямс, 2007.

6 Свойство Enabled (Microsoft Forms) [Электронный ресурс]. — URL: https://learn.microsoft.com/ru-ru/office/vba/language/reference/user-interface-help/enabled-property-microsoft-forms (Дата обращения 14.11.2024). Загл. с экр. Яз. рус.

7 SortedDictionary<TKey,TValue> Класс [Электронный ресурс]. — URL: https://learn.microsoft.com/ru-ru/dotnet/api/system.collections.generic.sorteddictionary-2?view=net-8.0 (Дата обращения 28.10.2024). Загл. с экр. Яз. рус.

8 DataGridView Класс [Электронный ресурс]. — URL: https://learn.microsoft.com/ru-ru/dotnet/api/system.windows.forms.datagridview?view=windowsdesktop-9.0 (Дата обращения 14.10.2024). Загл. с экр. Яз. рус.

9 *Б.Пахомов,* // C/C++ и MS Visual C++ 2012 для начинающих. — БХВ-Петербург, 2015.

10 *Т.А.Панюкова,* // Языки и методы программирования: Создание простых GUI-приложений с помощью Visual C++. — Либроком, 2013.

ПРИЛОЖЕНИЕ А

Примеры кода

Обработка табличных данных

Код кнопки «Вычислить»:

```
private: System::Void btn_calc_Click(System::Object^ sender, System::EventArgs^ e) {
  ClearAll();
  if (data_x->Rows->Count != data_first->Rows->Count)
    error_prov->SetError(btn_calc, "Размерности исходной матрицы и X не совпадают!");
  else
  {
    int min_el = 1000000; //условное значение для поиска минимума
    //первый цикл - ищем минимальный элемент в принципе

    bool flag_for_data = true;
    for (int i = 0; i < data_first->Columns->Count; i++)
      for (int j = 0; j < data_first->Rows->Count; j++)
      {
        int data_el;
        bool data_fl =
          ↪ Int32::TryParse(System::Convert::ToString(data_first->Rows[j]->Cells[i]->Value),
          ↪ data_el);
        if (!data_fl)
        {
          error_prov->SetError(btn_calc, "В матрице есть не числа");
          flag_for_data = false; //исходная таблица не прошла проверку
        }
        if (data_el < min_el)
          min_el = data_el;
      }
    bool flag_for_x = true;

    //проверяем корректность данных в X
    for (int k = 0; k < data_x->Rows->Count; k++)
    {
      int x_el;
      bool x_fl =
        ↪ Int32::TryParse(System::Convert::ToString(data_x->Rows[k]->Cells[0]->Value),
        ↪ x_el);
      if (!x_fl)
      {
```

```
        //выводим ошибку
        error_prov->SetError(btn_calc, "В X есть не числа");
        flag_for_x = false;//x не прошел проверку
      }
    }
    //если все данные прошли проверку
    if (flag_for_x && flag_for_data)
    {
      bool is_swapped; //если столбец уже нужно поменять на X
      for (int i = 0; i < data_first->Columns->Count; i++)
      {
        is_swapped = false;
        for (int j = 0; j < data_first->Rows->Count; j++)
        {
          int data_el =
          ↪  Int32::Parse(System::Convert::ToString(data_first->Rows[j]->Cells[i]->Value));
          if (data_el == min_el) //если в столбце найден мин. элемент
          {
            is_swapped = true; //говорим, что столбец нужно менять
            for (int k = 0; k < data_x->Rows->Count; k++)
            {
              //в результирующую таблицу вносим столбец X
              data_result->Rows[k]->Cells[i]->Value = data_x->Rows[k]->Cells[0]->Value;
            }
            break;
          }
          //иначе вносим исходную
          if (!is_swapped) //иначе
            data_result->Rows[j]->Cells[i]->Value = data_first->Rows[j]->Cells[i]->Value;
        }
      }
    }
  }
}
```

Использование коллекций в Windows Forms

Для кнопки «Найти должность» разработан следующий код:

```
private: System::Void btn_find_Click(System::Object^ sender, System::EventArgs^ e) {
  String^ surname = this->text_surname->Text; //считываем введенную фамилию
  System::Collections::Generic::SortedDictionary <String^, String^> dict;

```

```
  for (int i = 0; i < dgv_table->RowCount; ++i) //заполняем словарь для поиска
    dict.Add(System::Convert::ToString(dgv_table->Rows[i]->Cells[0]->Value),
      System::Convert::ToString(dgv_table->Rows[i]->Cells[1]->Value));

  bool find_flag = false; //флаг чтобы установить, нашли или нет
  for each (KeyValuePair<String^, String^> item in dict)
    //если найден, то выводим в окно и выходим из текста
    if (item.Key == surname)
    {
      find_flag = true;
      this->text_work->Text = item.Value;
      break;
    }
    //иначе выводим, что не нашли
  if (!find_flag)
    this->text_work->Text = "Не найден";
}
```

Для кнопки «Вывести фамилии» разработан следующий код:

```
private: System::Void btn_list_Click(System::Object^ sender, System::EventArgs^ e) {
  this->text_list->Text = ""; //очистим результат прошлых вызовов
  String^ work = this->text_work1->Text; //считаем введнную должность
  System::Collections::Generic::SortedDictionary <String^, String^> dict;

 //проходимся по таблице и заполняем словарь для поиска
  for (int i = 0; i < dgv_table->RowCount; ++i)
    dict.Add(System::Convert::ToString(dgv_table->Rows[i]->Cells[0]->Value),
      System::Convert::ToString(dgv_table->Rows[i]->Cells[1]->Value));

  //ищем нужные фамилии, и выводим
  for each (KeyValuePair<String^, String^> item in dict)
    if (item.Value == work)
      this->text_list->Text += item.Key + "\r\n";
}
```

Файловые диалоги и работа с файлами

Полный код кнопки «Сохранить»:

```
    private: System::Void btn_save_Click(System::Object^ sender, System::EventArgs^ e) {
    error_prov->Clear();
    bool date_flag = true; //для проверки даты
```

```
4       for (int i = 0; i < dgv_input->RowCount; ++i)
5       {
6         String^ format = "dd.MM.yyyy"; //формат даты
7         DateTime date;
8         String^ date_to_parse =
            ↪ System::Convert::ToString(dgv_input->Rows[i]->Cells[3]->Value);
9         date_flag = DateTime::TryParseExact(date_to_parse, format,
            ↪ System::Globalization::CultureInfo::InvariantCulture,
10          DateTimeStyles::None, date); //пытаемся считать дату
11
12        if (!date_flag) //если не вышло, то дальше выведем ошибку и остановим операцию
13          break;
14      }
15      if (!date_flag)
16        error_prov->SetError(dgv_input, "Некорректный формат даты");
17      else
18      {
19        bool flag = true; //для проверки корректности оценок
20        for (int i = 0; i < dgv_input->RowCount; ++i)
21        {
22          for (int j = 4; j < 9; ++j)
23          {
24            String^ s = System::Convert::ToString(dgv_input->Rows[i]->Cells[j]->Value);
25            int num;
26            bool parse = Int32::TryParse(s, num);
27
28            if (parse)
29              if (num > 5 || num < 0) //если оценка некорректная
30                flag = false;
31          }
32          if (!flag)
33            break;
34        }
35
36        if (!flag)
37          error_prov->SetError(dgv_input, "Некорректный формат оценки");
38        else
39        {
40          System::IO::Stream^ myStream; //открываем поток для сохранения
41          if (this->saveFileDialog->ShowDialog() ==
              ↪ System::Windows::Forms::DialogResult::OK)
```

```
42          {
43            if ((myStream = saveFileDialog->OpenFile()) != nullptr)
44            {
45              System::IO::StreamWriter^ sw = gcnew System::IO::StreamWriter(myStream,
                ↪  System::Text::Encoding::GetEncoding(1251));
46              for (int i = 0; i < dgv_input->RowCount; ++i)
47              {
48                String^ s; //с помощью строки s записываем таблицу в файл построчно
49                for (int j = 0; j < dgv_input->ColumnCount; ++j)
50                  s += System::Convert::ToString(dgv_input->Rows[i]->Cells[j]->Value) + " ";
51
52                sw->WriteLine(s); //сформировав строку, записываем её
53              }
54
55              sw->Close(); //закрываем поток
56            }
57          }
58        }
59      }
60    }
```

Код кнопки «Выбрать сдавших»:

```
1   private: System::Void btn_choose_Click(System::Object^ sender, System::EventArgs^ e) {
2  error_prov->Clear();
3  dgv_output->Rows->Clear(); //очистим таблицу с предыдущего вызова
4  bool date_flag = true;
5  for (int i = 0; i < dgv_input->RowCount; ++i)
6  {
7    String^ format = "dd.MM.yyyy";
8    DateTime date;
9    String^ date_to_parse =
      ↪  System::Convert::ToString(dgv_input->Rows[i]->Cells[3]->Value);
10   date_flag = DateTime::TryParseExact(date_to_parse, format,
      ↪  System::Globalization::CultureInfo::InvariantCulture,
11     DateTimeStyles::None, date);
12
13   if (!date_flag)
14     break;
15 }
16 if (!date_flag)
17   error_prov->SetError(dgv_input, "Некорректный формат даты");
```

```
18    else
19    {
20      bool flag = true;
21      for (int i = 0; i < dgv_input->RowCount; ++i)
22      {
23        for (int j = 4; j < 9; ++j)
24        {
25          String^ s = System::Convert::ToString(dgv_input->Rows[i]->Cells[j]->Value);
26          int num;
27          bool parse = Int32::TryParse(s, num);
28
29          if (parse) //если это число, то проверим его
30            if (num > 5  num < 0)
31              flag = false;
32          else //иначе проверим случай, если это строка
33            if (s != "неявка" && s != "незачтено" && s != "зачтено")
34        }
35        if (!flag)
36          break;
37      }
38
39      if (!flag)
40        error_prov->SetError(dgv_input, "Некорректный формат оценок");
41      else
42      {
43        int row_fill = 0;
44        for (int i = 0; i < dgv_input->RowCount; ++i)
45        {
46          bool session_flag = true; //сдал ли студент сессию
47          for (int j = 4; j < 9; ++j)
48          {
49            String^ s = System::Convert::ToString(dgv_input->Rows[i]->Cells[j]->Value);
50            if (s == "зачтено"  s =="незачет"  s == "2"  s == "1" || s == "0")
51              session_flag = false; //не сдал
52          }
53          if (session_flag) //если сдал
54          {
55            dgv_output->Rows->Add(); //записываем в таблицу
56            for (int l = 0; l < dgv_output->ColumnCount; ++l)
57              dgv_output->Rows[row_fill]->Cells[l]->Value =
                ↪   dgv_input->Rows[i]->Cells[l]->Value;
```

```
58            row_fill++; //количество сдавших (для записи в таблицу)
59          }
60        }
61      }
62    }
63  }
```

Код кнопки «Сохранить сдавших»:

```
1     private: System::Void btn_savechoose_Click(System::Object^ sender, System::EventArgs^
        ↪ e) {
2   error_prov->Clear();
3
4   bool date_flag = true;
5   for (int i = 0; i < dgv_output->RowCount; ++i)
6   {
7     String^ format = "dd.MM.yyyy"; //формат даты
8     DateTime date;
9     String^ date_to_parse =
        ↪ System::Convert::ToString(dgv_output->Rows[i]->Cells[3]->Value);
10    date_flag = DateTime::TryParseExact(date_to_parse, format,
        ↪ System::Globalization::CultureInfo::InvariantCulture,
11      DateTimeStyles::None, date); //пытаемся считать дату
12
13    if (!date_flag)
14      break;
15  }
16  if (!date_flag) //если не удалось
17    error_prov->SetError(dgv_output, "Некорректный формат даты");
18  else
19  {
20    bool flag = true; //флаг для проверки оценки
21    for (int i = 0; i < dgv_output->RowCount; ++i)
22    {
23      for (int j = 4; j < 9; ++j)
24      {
25        String^ s = System::Convert::ToString(dgv_output->Rows[i]->Cells[j]->Value);
26        int num;
27        bool parse = Int32::TryParse(s, num);
28
29        if (parse)
30          if (num > 5 || num < 0)
```

```
31              flag = false;
32            else
33              if (s != "незачтено" && s != "неявка" && s != "зачтено")
34            }
35            if (!flag)
36              break;
37          }
38
39          if (!flag) //если некорректные оценки
40            error_prov->SetError(dgv_output, "Некорректный формат оценок");
41          else
42          {
43            System::IO::Stream^ myStream; //открываем поток
44            if (this->saveFileDialog->ShowDialog() == System::Windows::Forms::DialogResult::OK)
45            {
46              if ((myStream = saveFileDialog->OpenFile()) != nullptr)
47              {
48                System::IO::StreamWriter^ sw = gcnew System::IO::StreamWriter(myStream,
                  ↪  System::Text::Encoding::GetEncoding(1251));
49                for (int i = 0; i < dgv_output->RowCount; ++i)
50                {
51                  String^ s; //формируем строку
52                  for (int j = 0; j < dgv_output->ColumnCount; ++j)
53                    s += System::Convert::ToString(dgv_output->Rows[i]->Cells[j]->Value) + " ";
54
55                  sw->WriteLine(s); //записываем в файл
56                }
57
58                sw->Close(); //закрываем поток
59              }
60            }
61          }
62        }
63      }
```

Приложение ТЕСТ

Полный массив, содержащий все вопросы теста:

```
1       //массив всех вопросов
2     array<Questions^>^ test_dataset() {
3       array<Questions^>^ q = gcnew array<Questions^>(12);
4
```

```
5         //вопрос 1
6         q[0] = gcnew Questions();
7         q[0]->text = "C++, Python, C - это всё _______ (полное словосочетание)";
8         q[0]->type = Test::txt_answer;
9         q[0]->answer = "языки программирования";
10
11        //вопрос 2
12        q[1] = gcnew Questions();
13        q[1]->text = "Какой тип данных описывает целочисленный тип?";
14        q[1]->type = Test::single;
15        q[1]->answers = gcnew array<String^>{ "integer", "float", "boolean", "string" };
16        q[1]->answer = 0;
17
18        //вопрос 3
19        q[2] = gcnew Questions();
20        q[2]->text = "C++ является надстройкой над C";
21        q[2]->type = Test::yes_no;
22        q[2]->answer = 0;
23
24        //вопрос 4
25        q[3] = gcnew Questions();
26        q[3]->text = "Каким языком является Python?";
27        q[3]->type = Test::single;
28        q[3]->answers = gcnew array<String^>{"Функциональным", "Интерпретируемым",
          ↪  "Транслируемым", "Компилируемым"};
29        q[3]->answer = 1;
30
31        //вопрос 5
32        q[4] = gcnew Questions();
33        q[4]->text = "C++ - объектно-ориентированный ЯП";
34        q[4]->type = Test::yes_no;
35        q[4]->answer = 0;
36
37        //вопрос 6
38        q[5] = gcnew Questions();
39        q[5]->text = "Что из перечисленного является высокоуровневым ЯП?";
40        q[5]->type = Test::multiple;
41        q[5]->answers = gcnew array<String^>{"Python", "Assembler", "C", "Pandas"};
42        q[5]->answer = gcnew array<int>{0, 2};
43
44        //вопрос 7
```

```
45          q[6] = gcnew Questions();
46          q[6]->text = "if в языках программирования является оператором _______";
47          q[6]->type = Test::txt_answer;
48          q[6]->answer = "условия";
49
50          //вопрос 8
51          q[7] = gcnew Questions();
52          q[7]->text = "Что из этого является языком для управления данными через
            ↪ запросы?";
53          q[7]->type = Test::single;
54          q[7]->answers = gcnew array < String^> {"PHP", "Python", "R", "SQL"};
55          q[7]->answer = 3;
56
57          //вопрос 9
58          q[8] = gcnew Questions();
59          q[8]->text = "Как называется программа перевода языка высокого уровня в
            ↪ машинный код?";
60          q[8]->type = Test::single;
61          q[8]->answers = gcnew array<String^> {"Транслятор", "Архиватор", "Диспетчер
            ↪ кода", "ОС"};
62          q[8]->answer = 0;
63
64          //вопрос 10
65          q[9] = gcnew Questions();
66          q[9]->text = "Выберите парадигмы программирования";
67          q[9]->type = Test::multiple;
68          q[9]->answers = gcnew array<String^> {"Функциональная", "Абстрактная",
            ↪ "Процедурная", "Объектно-ориентированная"};
69          q[9]->answer = gcnew array<int>{0, 2, 3};
70
71          //вопрос 11
72          q[10] = gcnew Questions();
73          q[10]->text = "Что из этого является ЯП со строгой типизацией данных?";
74          q[10]->type = Test::multiple;
75          q[10]->answers = gcnew array<String^> {"Java", "Python", "JavaScript", "C++"};
76          q[10]->answer = gcnew array<int>{0, 3};
77
78          //вопрос 12
79          q[11] = gcnew Questions();
80          q[11]->text = "HTML - язык программирования";
81          q[11]->type = Test::yes_no;
```

```
82        q[11]->answer = 1;
83
84      return q;
85    }
```

Код кнопки «Ответить»:

```
1     static int counter_of_correct = 0; //общий счётчик правильных ответов
2   private: System::Void btn_answer_Click(System::Object^ sender, System::EventArgs^ e) {
3     if (current_question < questions->Length) //если остались вопросы
4     {
5       test::Questions^ q = questions[current_question];
6       switch (q->type) //в зависимости от типа вопроса
7       {
8       //Да/нет
9       case test::Test::yes_no:
10        if ((radio_yes->Checked && (int)q->answer == 0)  (radio_no->Checked &&
          ↪  (int)q->answer == 1))
11        {
12          counter_of_correct++;
13          MessageBox::Show("Правильно!");
14        }
15        else
16          MessageBox::Show("Неправильно! :(");
17        break;
18
19      //Краткий ответ
20      case test::Test::txt_answer:
21        if (text_answer->Text == System::Convert::ToString(q->answer))
22        {
23          counter_of_correct++;
24          MessageBox::Show("Правильно!");
25        }
26        else
27          MessageBox::Show("Неправильно! :(\n Правильный ответ: " +
            ↪  System::Convert::ToString(q->answer));
28        break;
29
30      //Один ответ
31      case test::Test::single:
32        if ((radio_single1->Checked && (int)q->answer == 0)  (radio_single2->Checked &&
          ↪  (int)q->answer == 1)
```

```
33          (radio_single3->Checked && (int)q->answer == 2)  (radio_single4->Checked &&
              ↪  (int)q->answer == 3))
34        {
35          counter_of_correct++;
36          MessageBox::Show("Правильно!");
37        }
38        else
39          MessageBox::Show("Неправильно! :( \n Правильный ответ: " +
              ↪  (q->answers[(int)q->answer]));
40        break;
41
42      //Несколько ответов
43      case test::Test::multiple:
44        bool is_checked_only_correct = true;
45        array<int>^ correct = (array<int>^)q->answer;
46        for (int i = 0; i < correct->Length; ++i)
47        {
48          int cor = correct[i];
49          if ((cor == 0 && !check_1->Checked)  (cor == 1 && !check_2->Checked)
50            (cor == 2 && !check_3->Checked) || (cor == 3 && !check_4->Checked))
51            is_checked_only_correct = false;
52        }
53
54        //защита от случая, когда пользователь выбрал все ответы
55        if (check_1->Checked && Array::IndexOf(correct, 0) == -1) {
56          is_checked_only_correct = false;
57        }
58        if (check_2->Checked && Array::IndexOf(correct, 1) == -1) {
59          is_checked_only_correct = false;
60        }
61        if (check_3->Checked && Array::IndexOf(correct, 2) == -1) {
62          is_checked_only_correct = false;
63        }
64        if (check_4->Checked && Array::IndexOf(correct, 3) == -1) {
65          is_checked_only_correct = false;
66        }
67
68        if (is_checked_only_correct)
69        {
70          counter_of_correct++;
71          MessageBox::Show("Правильно!");
```

```
72          }
73          else
74          {
75            // строка для вывода
76            String^ string_to_print = "Неправильно! :( \n Правильный ответ: ";
77            for (int i = 0; i < correct->Length; i++)
78              string_to_print += System::Convert::ToString(q->answers[correct[i]]) + " ";
79
80            MessageBox::Show(string_to_print);
81          }
82          break;
83        }
84      }
85
86
87      current_question++;
88      if (current_question < questions->Length) {
89        display_question(questions[current_question]);
90      }
91      else //если вопросы закончились
92        MessageBox::Show("Поздравляем, Вы прошли тест! Ваш счёт: " +
          ↪  System::Convert::ToString(counter_of_correct) + " из " +
          ↪  System::Convert::ToString(questions->Length));
93    }
```

ПРИЛОЖЕНИЕ Б

Flash-накопитель с отчетом о выполненной работе

На приложенном flash-накопителе можно ознакомиться со следующими файлами:

Папка report — LaTeX- вариант отчета о практике;

Папка factorial — задание №1;

Папка dgv_tables — задание №2;

Папка collections — задание №3;

Папка files — задание №4;

Папка test — задание №5;

Файл report.pdf — отчет о практике;

Файл screenshots.pdf — скриншоты результатов работы программ.