Министерство образования и науки Российской Федерации

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«САРАТОВСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ИМЕНИ Н. Г. ЧЕРНЫШЕВСКОГО»

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

ОТЧЕТ О ПРАКТИКЕ

студента 1 курса 151 группы факультета КНиИТ
Дергунова Дмитрия Витальевича

вид практики: учебная
кафедра: математической кибернетики и компьютерных наук
курс: 1
семестр: 2
продолжительность: 2 нед., с 30.06.2017 г. по 13.07.2017 г.

Руководитель практики от университета,

Зав. каф. техн. прогр., к. ф.-м. н. _____ И. А. Батраева

Руководитель практики от организации (учреждения, предприятия),

Зав. каф. техн. прогр., к. ф.-м. н. _____ И. А. Батраева

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

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Целью практики является освоение механизма построения оконного интерфейса приложений в среде Visual Studio [1].

В результате прохождения практики должны быть отработаны навыки

- создания нового проекта;
- добавления и настройки элементов управления;
- отладка корректного ввода данных для решения поставленной задачи;
- разработки алгоритма решения поставленной задачи с использованием оконного интерфейса;
- тестирования приложения;
- документирования разработанного кода.

1 Приложение для вычисления факториала

Задание. Реализовать приложение для вычисления факториала.

Создано окно приложения, содержащее два элемента `TextBox`, два элемента `Label` и один элемент `Button`. Для отображения сообщений об ошибках в окно добавлен элемент `ErrorProvider`. Вид окна представлен на рисунке 1.

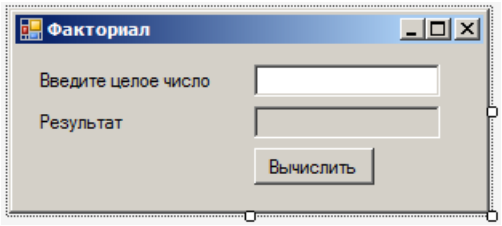

Рисунок 1 – Окно приложения «Факториал» открытое в конструкторе

У элементов изменены значения некоторых атрибутов. Значения измененных атрибутов представлены в таблице 1.

Таблица 1 – Значения атрибутов элементов в приложении «Факториал»

Наименование атрибута	Значение
Для формы	
`Text`	`Факториал`
Для первой надписи	
`(Name)`	`lblInput`
`Text`	`Введите целое число`
Для второй надписи	
`(Name)`	`lblOutput`
`Text`	`Результат`
Для первого текстового поля	
`(Name)`	`txtInput`
Для второго текстового поля	
`(Name)`	`txtOutput`
`ReadOnly`	`True`
Для кнопки	
`(Name)`	`btnCalculate`
`Text`	`Вычислить`
Для обработчика ошибок	
`(Name)`	`errorProvider`

Была написана функция вычисления факториала:

```
1 long double fact(int N) {
```

```
  // если отрицательное число
  if (N < 0)
    return 0;
  if (N == 0)
    return 1;
  else
    return N * fact(N - 1);
}
```

На нажатие кнопки «Вычислить» установлено выполнение следующего кода:

```
    int InputNumber;
    bool result = Int32::TryParse(this->txtInput->Text, InputNumber);
    if (result) {
      errorProvider->SetError(this->txtInput, String::Empty);
      if (InputNumber < 0) {
        this->txtOutput->Text = "";
        errorProvider->SetError(this->txtInput, "Введено отрицательное число");
      }
      else if (InputNumber > 17) {
        this->txtOutput->Text = "";
        errorProvider->SetError(this->txtInput, "Выход за границы типа");
      }
      else {
        double OutputNumber = fact(InputNumber);
        this->txtOutput->Text = System::Convert::ToString(OutputNumber);
      }
    }
    else {
      this->txtOutput->Text = "";
      errorProvider->SetError(this->txtInput, "Введено не целое число");
    }
```

После запуска приложения на экране появляется окно (см. рисунок 2).

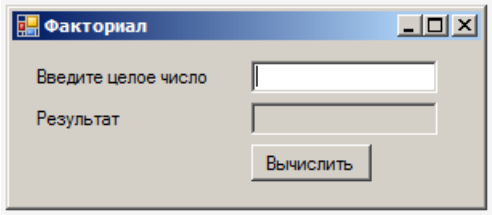

Рисунок 2 – Окно приложения «Факториал»: начальный запуск

При вводе целого числа после нажатия кнопки в поле вывода приводится результат вычисления факториала для заданного числа (см. рисунок 3).

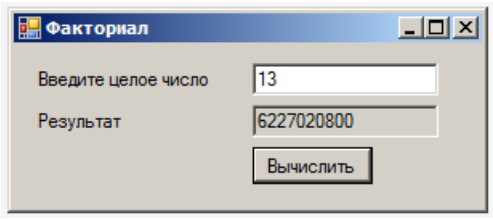

Рисунок 3 – Окно приложения «Факториал»: корректное вычисление

При вводе некорректного значения возникает сообщение об ошибке (см. рисунок 4).

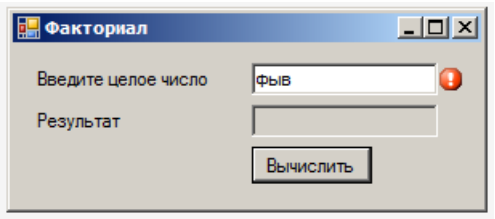

Рисунок 4 – Окно приложения «Факториал»: сообщение о некорректном вводе

При вводе слишком большого положительного числа возникает сообщение об ошибке (см. рисунок 5).

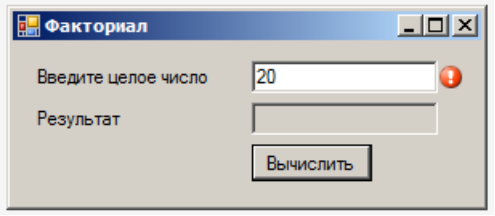

Рисунок 5 – Окно приложения «Факториал»: ввод слишком большого положительного числа

При вводе отрицательного числа возникает сообщение об ошибке (см. рисунок 6).

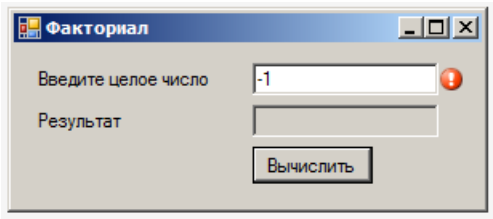

Рисунок 6 – Окно приложения «Факториал»: ввод отрицательного числа

Полный код программы приведен в приложении А.

2 Приложение для вычисления простой функции

Задание. Реализовать приложение для вычисления функции

$$\frac{\sin^2 x + y^2}{(y-4)\sqrt{x^2 - 2x + 1}}$$

Создано окно приложения, содержащее три элемента `TextBox`, три элемента `Label` и один элемент `Button`. Для отображения сообщений об ошибках в окно добавлен элемент `ErrorProvider`. Вид окна представлен на рисунке 7.

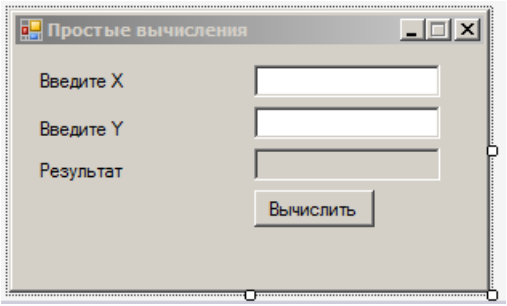

Рисунок 7 – Окно приложения «Простые вычисления» открытое в конструкторе

У элементов изменены значения некоторых атрибутов. Значения измененных атрибутов представлены в таблице 2.

Таблица 2 – Значения атрибутов элементов в приложении «Простые вычисления»

Наименование атрибута	Значение
Для формы	
`Text`	`Простые вычисления`
`MaximizeBox`	`False`
Для первой надписи	
`(Name)`	`lblInputX`
`Text`	`Введите целое число`
Для второй надписи	
`(Name)`	`lblInputY`
Для третьей надписи	
`(Name)`	`lblOutput`
`Text`	`Результат`
Для первого текстового поля	
`(Name)`	`txtInputX`
Для второго текстового поля	
`(Name)`	`txtInputY`
Для третьего текстового поля	

(Name)	txtOutput
(ReadOnly)	True
Для кнопки	
(Name)	btnCalculate
Text	Вычислить
Для обработчика ошибок	
(Name)	errorProvider

Была написана функция вычисления выражения:

```
long double count(int X, int Y) {
  //Значение функции
  return (sin(X)*sin(X) + Y*Y) / ((Y - 4)*abs(X - 1));
}
```

На нажатие кнопки «Вычислить» установлено выполнение следующего кода:

```
    int InputNumberX;
    int InputNumberY;
    bool resultX = Int32::TryParse(this->txtInputX->Text, InputNumberX);
    bool resultY = Int32::TryParse(this->txtInputY->Text, InputNumberY);

    errorProvider->SetError(this->txtInputX, String::Empty);
    errorProvider->SetError(this->txtInputY, String::Empty);
    this->txtOutput->Text = "";

    // Если x не число
    if (!resultX) {
      errorProvider->SetError(this->txtInputX, "Введено не целое число");
    }
    // Если y не число
    if (!resultY) {
      errorProvider->SetError(this->txtInputY, "Введено не целое число");
    }

    //если нет ошибок
    if (resultX && resultY) {
      // проверка деления на 0
      if (InputNumberY == 4) {
        errorProvider->SetError(this->txtInputY, "Деление на 0. Y не должен
            равняться 4");
      }
```

```
25          if (InputNumberX == 1) {
26            errorProvider->SetError(this->txtInputX, "Деление на 0. X не должен
                  равняться 1");
27          }
28
29          //если нет ошибок
30          if (InputNumberY != 4 && InputNumberX != 1) {
31            double OutputNumber = count(InputNumberX, InputNumberY);
32            this->txtOutput->Text = System::Convert::ToString(OutputNumber);
33          }
34        }
```

После запуска приложения на экране появляется окно (см. рисунок 8).

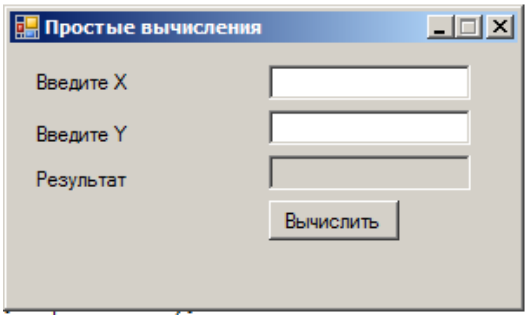

Рисунок 8 – Окно приложения «Простые вычисления»: начальный запуск

При вводе целых чисел после нажатия кнопки в поле вывода приводится результат вычисления функции для заданных чисел (см. рисунок 9).

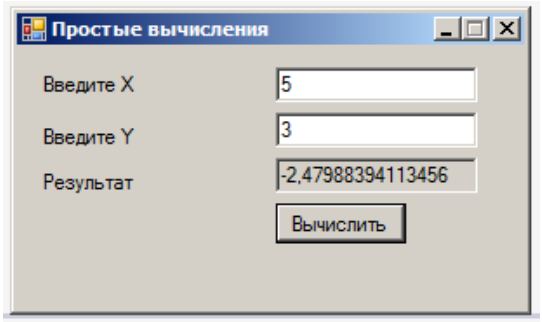

Рисунок 9 – Окно приложения «Простые вычисления»: корректное вычисление

При вводе значений, не попадающих в ОДЗ, возникает сообщение об ошибке (см. рисунок 10).

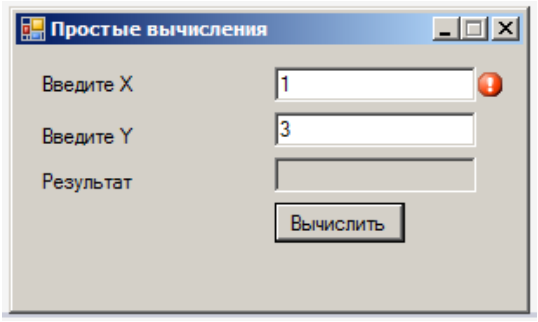

Рисунок 10 – Окно приложения «Простые вычисления»: сообщение о вводе числа, не попадающего в ОДЗ

При вводе некорректных значений возникает сообщение об ошибке (см. рисунок 11).

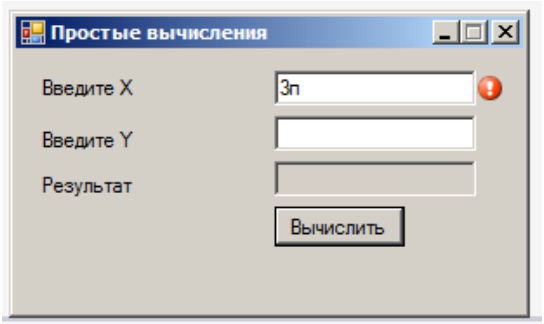

Рисунок 11 – Окно приложения «Простые вычисления»: сообщение о некорректном вводе

Полный код программы приведен в приложении А.

3 Приложение для вычисления рекурсивной функции

Задание. Реализовать приложение для вычисления рекурсивной функции

$$S = \sqrt{1 + 2\sqrt{1 + 3\sqrt{1 + 4\sqrt{1 + \cdots + n}}}}$$

Создано окно приложения, содержащее два элемента `TextBox`, два элемента `Label` и один элемент `Button`. Для отображения сообщений об ошибках в окно добавлен элемент `ErrorProvider`. Вид окна представлен на рисунке 12.

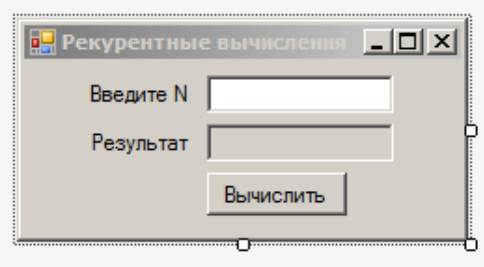

Рисунок 12 – Окно приложения «Рекуррентные вычисления» открытое в конструкторе

У элементов изменены значения некоторых атрибутов. Значения измененных атрибутов представлены в таблице 3.

Таблица 3 – Значения атрибутов элементов в приложении «Простые вычисления»

Наименование атрибута	Значение
Для формы	
`Text`	`Рекуррентные вычисления`
Для первой надписи	
`(Name)`	`lblInput`
`Text`	`Введите N`
Для второй надписи	
`(Name)`	`lblOutput`
`Text`	`Результат`
Для первого текстового поля	
`(Name)`	`txtInput`
Для второго текстового поля	
`(Name)`	`txtOutput`
`ReadOnly`	`True`
Для кнопки	
`(Name)`	`btnCalculate`
`Text`	`Вычислить`
Для обработчика ошибок	

(Name)	errorProvider

Была написана функция вычисления выражения:

```
long double recur(int N, double result = 1) {
 // если отрицательное число
 if (N < 2)
   return result;

 result = sqrt(1 + N*result);
 return recur(N - 1, result);
}
```

На нажатие кнопки «Вычислить» установлено выполнение следующего кода:

```
  int InputNumber;
  bool result = Int32::TryParse(this->txtInput->Text, InputNumber);
  //если спарсили
  if (result) {
    errorProvider->SetError(this->txtInput, String::Empty);
    if (InputNumber < 1) {
      this->txtOutput->Text = "";
      errorProvider->SetError(this->txtInput, "Введено не положительное число");
    }
    else {
      double OutputNumber = recur(InputNumber);
      this->txtOutput->Text = System::Convert::ToString(OutputNumber);
    }
  }
  else {
    this->txtOutput->Text = "";
    errorProvider->SetError(this->txtInput, "Введено не целое число");
  }
```

После запуска приложения на экране появляется окно (см. рисунок 13).

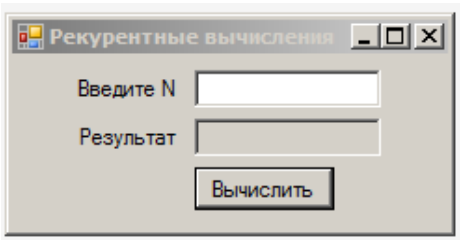

Рисунок 13 – Окно приложения «Рекуррентные вычисления»: начальный запуск

При вводе целого числа после нажатия кнопки в поле вывода приводится результат вычисления функции для заданного числа (см. рисунок 14).

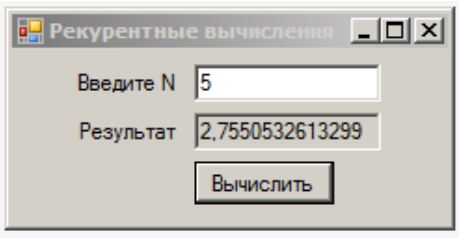

Рисунок 14 – Окно приложения «Рекуррентные вычисления»: корректное вычисление

При вводе не целого числа возникает сообщение об ошибке (см. рисунок 15).

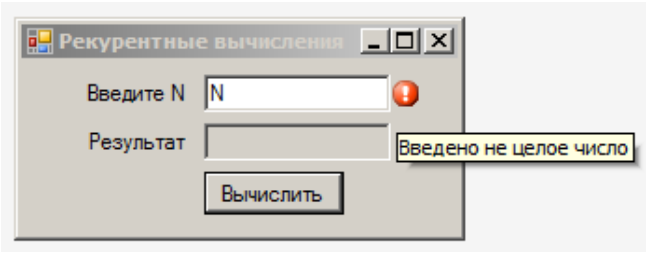

Рисунок 15 – Окно приложения «Рекуррентные вычисления»: сообщение о некорректном вводе

При вводе неположительного числа возникает сообщение об ошибке (см. рисунок 16).

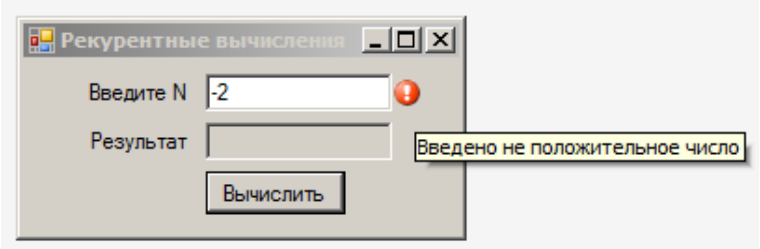

Рисунок 16 – Окно приложения «Рекуррентные вычисления»: сообщение о вводе неположительного числа

Полный код программы приведен в приложении А.

4 Обработка табличных данных. Часть 1

Задание. Дан одномерный массив, содержащий целые числа. Представить его в виде таблицы, содержащий один столбец. Предусмотреть возможность добавления и удаления строк. Найти среднее арифметическое нечетных элементов, не попадающих в заданный интервал `[a, b]`. Если таких элементов нет, вывести сообщение об этом.

Создано окно приложения, содержащее три элемента `TextBox`, четыре элемента `Label`, три элемента `Button` и один элемент `DataGridView`. Для отображения сообщений об ошибках в окно добавлен элемент `ErrorProvider`. Вид окна представлен на рисунке 17.

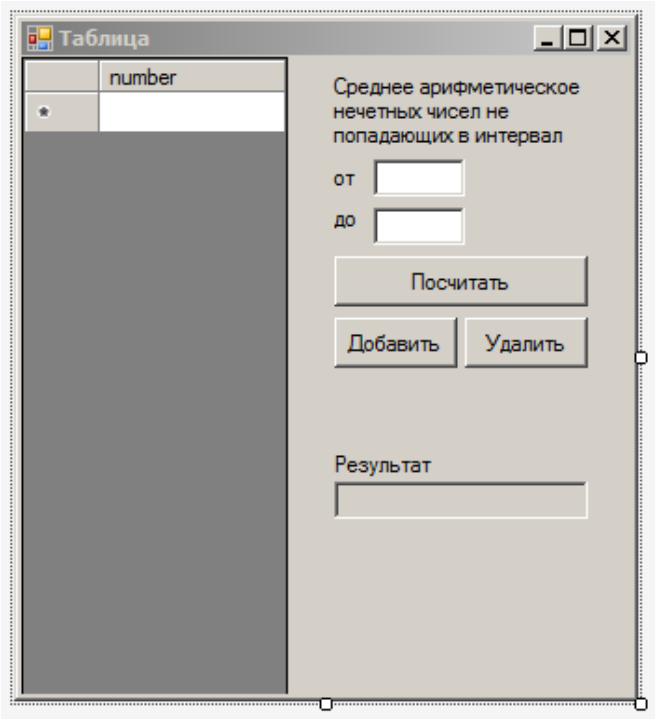

Рисунок 17 – Окно приложения «Таблица» открытое в конструкторе

У элементов изменены значения некоторых атрибутов. Значения измененных атрибутов представлены в таблице 4.

Таблица 4 – Значения атрибутов элементов в приложении «Таблица»

Наименование атрибута	Значение
Для формы	
`Text`	`Таблица`
Для первой надписи	
`(Name)`	`lblTask`
`Text`	`Среднее арифметическое нечетных чисел не попадающих в интервал`

Для второй надписи	
(Name)	lblFrom
Text	от
Для третьей надписи	
(Name)	lblTo
Text	до
Для четвертой надписи	
(Name)	lblResult
Text	Результат
Для первой таблицы	
(Name)	GridNumbers
Для первой кнопки	
(Name)	btnAdd
Text	Добавить
Для второй кнопки	
(Name)	btnRemove
Text	Удалить
Для третьей кнопки	
(Name)	btnResult
Text	Посчитать
Для первого текстового поля	
(Name)	txtFrom
Для второго текстового поля	
(Name)	txtTo
Для третьего текстового поля	
(Name)	txtResult
ReadOnly	True
Для обработчика ошибок	
(Name)	errorProvider

На нажатие кнопки «Добавить» установлено выполнение следующего кода:

```
1  // очищаем ошибки
2  errorProvider->SetError(this->txtFrom, String::Empty);
3  errorProvider->SetError(this->txtTo, String::Empty);
4  errorProvider->SetError(this->GridNumbers, String::Empty);
5  this->txtResult->Text = "";
6  this->GridNumbers->Rows->Add(1);
```

На нажатие кнопки «Удалить» установлено выполнение следующего

кода:

```
// очищаем ошибки
errorProvider->SetError(this->txtFrom, String::Empty);
errorProvider->SetError(this->txtTo, String::Empty);
errorProvider->SetError(this->GridNumbers, String::Empty);
this->txtResult->Text = "";
if (!this->GridNumbers->CurrentRow->IsNewRow) {
  int i = this->GridNumbers->CurrentRow->Index;
  this->GridNumbers->Rows->Remove(this->GridNumbers->Rows[i]);
}
```

На нажатие кнопки «Подсчитать» установлено выполнение следующего кода:

```
// очищаем ошибки
errorProvider->SetError(this->txtFrom, String::Empty);
errorProvider->SetError(this->txtTo, String::Empty);
errorProvider->SetError(this->GridNumbers, String::Empty);
this->txtResult->Text = "";

// сумма и количество
int s, k;
s = k = 0;

// блок с обработкой исключений
/* коды ошибок:
 * 1 - "От" - не число
 * 2 - "До" - не число
 * 4 - "От" больше "До"
 * 8 - в таблице пустые ячейка
 * 16 - в таблице не числа
 * 32 - нет удоавлетворяющих условию чисел
 */
// выполняем
try {
  int from, to;
  int errorCode = 0;
  bool resultFrom = Int32::TryParse(this->txtFrom->Text, from);
  bool resultTo = Int32::TryParse(this->txtTo->Text, to);
  if (!resultFrom) {
    errorCode += 1;
  }
```

```
29      if (!resultTo) {
30        errorCode += 2;
31      }
32      if (resultFrom && resultTo && (from > to)){
33        errorCode += 4;
34      }
35      for (int i = 0; i < this->GridNumbers->RowCount - 1; i++) {
36        int currentNumber;
37        //проверка на пустую строку
38        if (!(String ^)this->GridNumbers->Rows[i]->Cells[0]->Value) {
39          errorCode += 8;
40        }
41        // Проверка на целое число
42        else if (!Int32::TryParse(this->GridNumbers->Rows[i]->Cells[0]->Value->
              ToString(), currentNumber)) {
43          errorCode += 16;
44        }
45        // проверка на нечетность и попадение в интервал
46        else if (currentNumber % 2 && (currentNumber < from || currentNumber > to)) {
47          k++;
48          s += currentNumber;
49        }
50      }
51      if (!k) {
52        errorCode += 32;
53      }
54      //если код ошибки не 0, кидаем исключение
55      if (errorCode)
56        throw errorCode;
57
58      this->txtResult->Text = System::Convert::ToString((double) s / k );
59    }
60    //перехватываем исключение
61    catch (int errorCode) {
62      //парсим код ошибки
63      int n = 32;
64      while (n > 0) {
65        if (errorCode >= n) {
66          //разбираем код ошибки и выводим errorProvider'ы
67          switch (n) {
68          case 1:
```

```
69              errorProvider->SetError(this->txtFrom, "Введено не число");
70              break;
71          case 2:
72              errorProvider->SetError(this->txtTo, "Введено не число");
73              break;
74          case 4:
75              errorProvider->SetError(this->txtFrom, "\"От\" больше \"До\"");
76              errorProvider->SetError(this->txtTo, "\"От\" больше \"До\"");
77              break;
78          case 8:
79              errorProvider->SetError(this->GridNumbers, "В таблице пустые строки");
80              break;
81          case 16:
82              errorProvider->SetError(this->GridNumbers, "В таблице есть не целые
                    числа");
83              break;
84          case 32:
85              errorProvider->SetError(this->GridNumbers, "В таблице нет
                    удовлетворяющих условию чисел");
86              break;
87          }
88          errorCode -= n;
89      }
90      n /= 2;
91    }
92  }
```

После запуска приложения на экране появляется окно (см. рисунок 18).

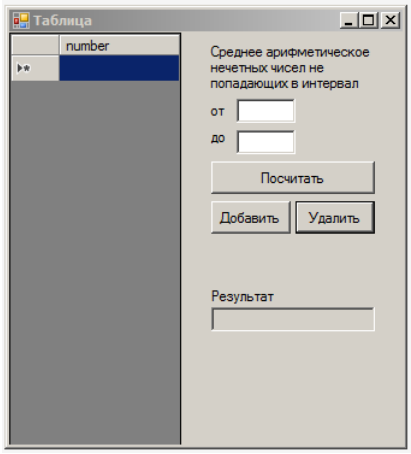

Рисунок 18 – Окно приложения «Таблица»: начальный запуск

При вводе корректных данных и нажатии кнопки «Посчитать» полу-

чаем результат (см. рисунок 19).

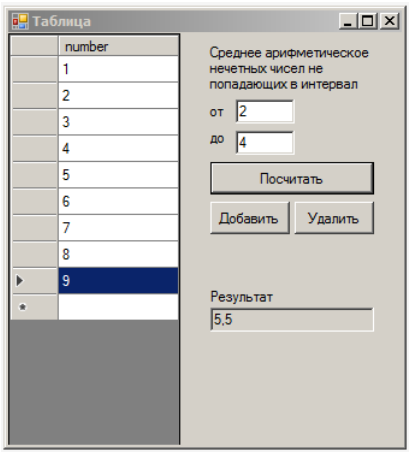

Рисунок 19 – Окно приложения «Таблица»: корректный ввод

При нажатии кнопки «Добавить» в таблицу добавляется новая строка (см. рисунок 20).

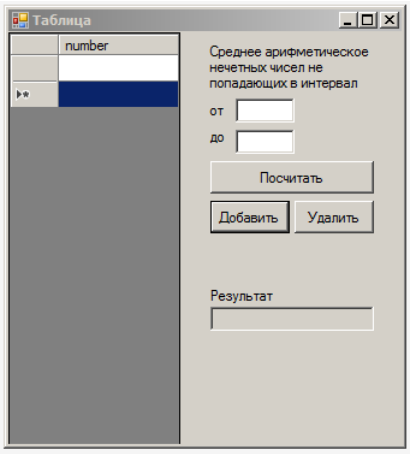

Рисунок 20 – Окно приложения «Таблица»: добавление новой строки

При нажатии кнопки «Удалить» из таблицы удаляется выбранная строка (см. рисунок 21).

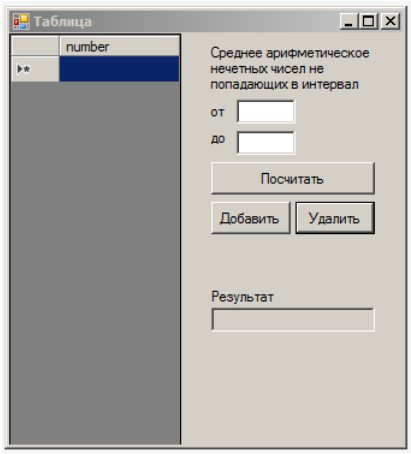

Рисунок 21 – Окно приложения «Таблица»: удаление строки

В случае, если в таблице есть пустые ячейки, получаем сообщение об ошибке (см. рисунок 22).

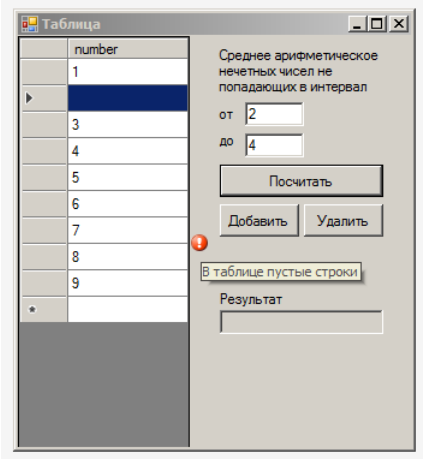

Рисунок 22 – Окно приложения «Таблица»: пустые строки

В случае, если в таблице есть ячейки, содержащие не целые числа, получаем сообщение об ошибке (см. рисунок 23).

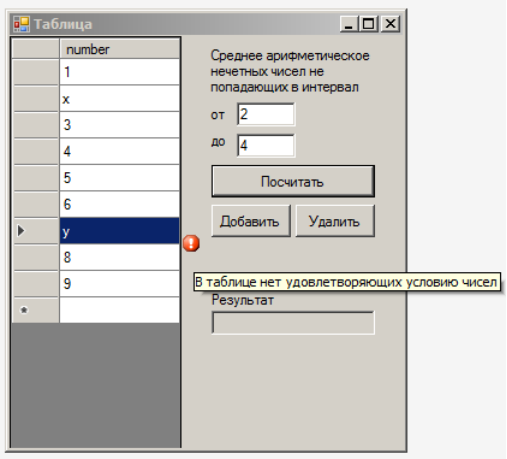

Рисунок 23 – Окно приложения «Таблица»: некорректный ввод

В случае, если в поле для левой границы интервала введено не целое число, получаем сообщение об ошибке (см. рисунок 24).

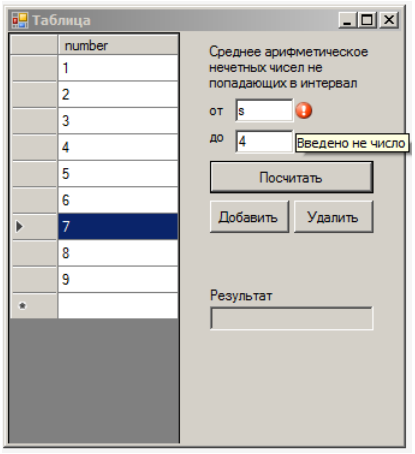

Рисунок 24 – Окно приложения «Таблица»: ошибка в левой границе интервала

В случае, если в поле для правой границы интервала введено не целое число, получаем сообщение об ошибке (см. рисунок 25).

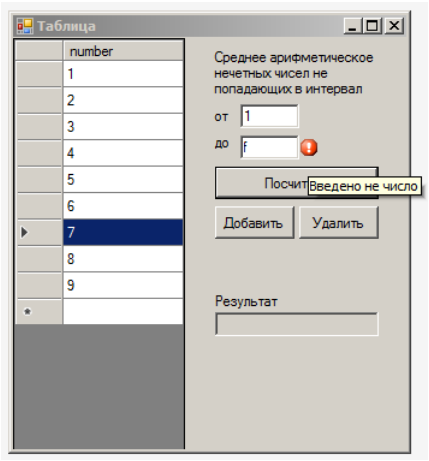

Рисунок 25 – Окно приложения «Таблица»: ошибка в правой границе интервала

В случае, если левая граница интервала больше, чем правая, получаем сообщение об ошибке (см. рисунок 26).

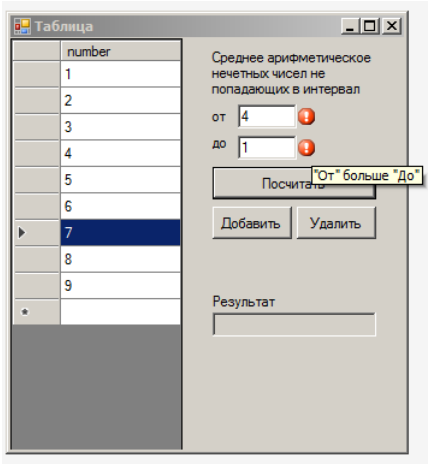

Рисунок 26 – Окно приложения «Таблица»: некорректный интервал

В случае, если в таблице нет чисел, не попадающих в заданный интервал, получаем сообщение об ошибке (см. рисунок 27).

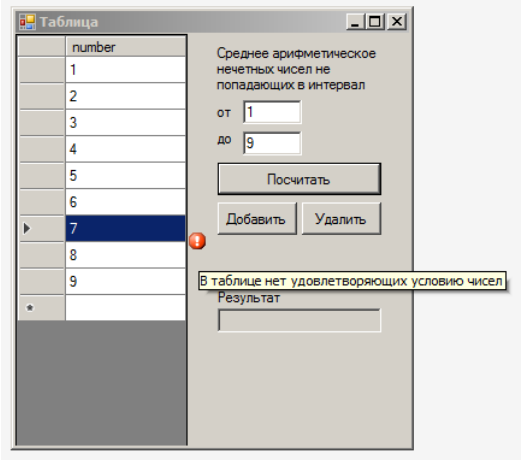

Рисунок 27 – Окно приложения «Таблица»: отсутствие удовлетворяющих условию чисел

Полный код программы приведен в приложении А.

5 Работа с табличными данными. Часть 2

Задание. Необходимо представить двумерный массив в виде таблицы `DataGridView`, содержащей произвольное количество строк и столбцов. Предусмотреть возможность добавления и удаления строк и столбцов. Провести проверку на ввод чисел. Все строки, содержащие минимальный элемент, заменить строкой `X`.

Создано окно приложения, содержащее три элемента `DataGridView`, пять элементов `Button` и три элемента `Label`. Для отображения сообщений об ошибках в окно добавлен элемент `ErrorProvider`. Вид окна представлен на рисунке 28.

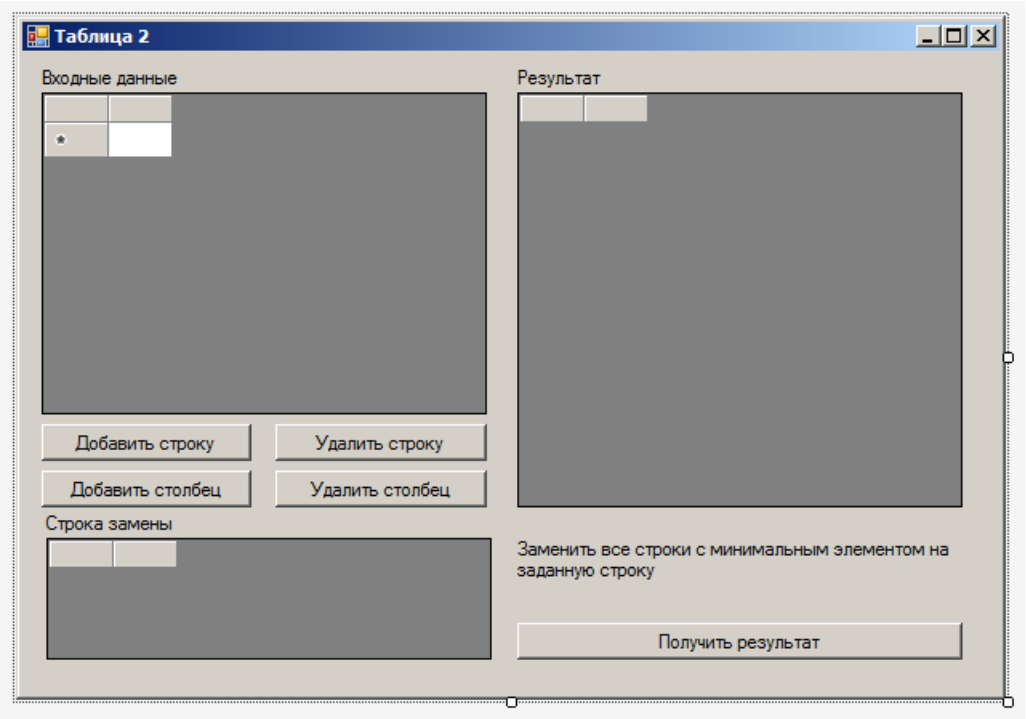

Рисунок 28 – Окно приложения «Таблица 2», открытое в конструкторе

У элементов изменены значения некоторых атрибутов. Значения измененных атрибутов представлены в таблице 5.

Таблица 5 – Значения атрибутов элементов в приложении «Таблица 2»

Наименование атрибута	Значение
Для формы	
`Text`	`Таблица 2`
Для первой надписи	
`(Name)`	`lblInput`
`Text`	`Входные данные`

Для второй надписи	
(Name)	lblOutput
Text	Результат
Для третьей надписи	
(Name)	lblTask
Text	Заменить все строки с минимальным элементом на заданную строку
Для первой таблицы	
(Name)	dataGridInput
Для второй таблицы	
(Name)	dataGridOutput
AllowUserToAddRows	False
AllowUserToDeleteRows	False
ReadOnly	True
Для третьей таблицы	
(Name)	dataGridRow
AllowUserToAddRows	False
AllowUserToDeleteRows	False
Для первой кнопки	
(Name)	btnRowAdd
Text	Добавить строку
Для второй кнопки	
(Name)	btnRowRemove
Text	Удалить строку
Для третьей кнопки	
(Name)	btnColAdd
Text	Добавить столбец
Для четвертой кнопки	
(Name)	btnColRemove
Text	Удалить столбец
Для пятой кнопки	
(Name)	btnColAdd
Text	Получить результат
Для обработчика ошибок	
(Name)	errorProvider

На нажатие кнопки «Добавить строку» установлено выполнение следующего кода:

```
1  this->dataGridInput->Rows->Add(1);
```

На нажатие кнопки «Удалить строку» установлено выполнение следующего кода:

```
  this->dataGridInput->Rows->Add(1);
}
private: System::Void btnColAdd_Click(System::Object^ sender, System::EventArgs^ e) {
  //добавляем колонки с шириной по умолчанию 40 пикселей
  int colCount = this->dataGridInput->ColumnCount;
  this->dataGridInput->Columns->Add("ColumnIn" + colCount, String::Empty);
  this->dataGridInput->Columns[colCount]->Width = 40;
  this->dataGridRow->Columns->Add("ColumnRow" + colCount, String::Empty);
  this->dataGridRow->Columns[colCount]->Width = 40;
```

На нажатие кнопки «Добавить столбец» установлено выполнение следующего кода:

```
  //добавляем колонки с шириной по умолчанию 40 пикселей
  int colCount = this->dataGridInput->ColumnCount;
  this->dataGridInput->Columns->Add("ColumnIn" + colCount, String::Empty);
  this->dataGridInput->Columns[colCount]->Width = 40;
  this->dataGridRow->Columns->Add("ColumnRow" + colCount, String::Empty);
  this->dataGridRow->Columns[colCount]->Width = 40;
```

На нажатие кнопки «Удалить столбец» установлено выполнение следующего кода:

```
  if (this->dataGridInput->ColumnCount > 1) {
    int i = 0;
    i = this->dataGridInput->ColumnCount - 1;
    this->dataGridInput->Columns->Remove(this->dataGridInput->Columns[i]);
    this->dataGridRow->Columns->Remove(this->dataGridRow->Columns[i]);
  }
```

На нажатие кнопки «Получить результат» установлено выполнение следующего кода:

```
  //чистим ошибки. очищаем вывод
  this->errorProvider->SetError(this->dataGridInput, String::Empty);
  this->errorProvider->SetError(this->dataGridRow, String::Empty);
  this->dataGridOutput->Columns->Clear();
  /* Коды ошибок
   * 1 -- нет ни одной строки
   * 2 -- в таблице есть пустые ячейки
   * 3 -- в таблице есть не целые числа
   */
```

```
10    //проверяем матрицу ввода и ищем минимум
11    int min;
12    try {
13      int errorCode;
14      //проверка на количество строк
15      if (this->dataGridInput->RowCount < 2) {
16        throw 1;
17      }
18      // min = первому элементу с проверкой на пустоту и число
19      if (!this->dataGridInput->Rows[0]->Cells[0]->Value) {
20        throw 2;
21      }
22      bool resultParse;
23      resultParse = Int32::TryParse(this->dataGridInput->Rows[0]->Cells[0]->Value->
            ToString(), min);
24      if (!resultParse) {
25        throw 3;
26      }
27      //ищем минимум
28      for (int i = 0; i < this->dataGridInput->RowCount - 1; i++) {
29        for (int j = 0; j < this->dataGridInput->ColumnCount; j++) {
30          //проверка на пустоту
31          if (!this->dataGridInput->Rows[i]->Cells[j]->Value) {
32            throw 2;
33          }
34          //проверка на число
35          int current;
36          resultParse = Int32::TryParse(this->dataGridInput->Rows[i]->Cells[j]->Value
                ->ToString(), current);
37          if (!resultParse) {
38            throw 3;
39          }
40          //проверка на минимум
41          if (current < min)
42            min = current;
43        }
44      }
45    }
46    catch (int errorCode) {
47      switch (errorCode) {
48        case 1:
```

```
49          this->errorProvider->SetError(this->dataGridInput, "В таблице нет ни одной
                строки");
50          break;
51        case 2:
52          this->errorProvider->SetError(this->dataGridInput, "В таблице есть пустые
                ячейки");
53          break;
54        case 3:
55          this->errorProvider->SetError(this->dataGridInput, "В таблице есть не
                целые числа");
56          break;
57      }
58    }
59    /*
60    * 1 --в строке замены есть пустые ячейки
61    * 2 --в строке замены есть не целые числа
62    */
63    //проверяем строку замены
64    try {
65      for (int i = 0; i < this->dataGridRow->ColumnCount; i++) {
66        bool resultParse;
67        //проверка на пустоту
68        if (!(String ^)this->dataGridRow->Rows[0]->Cells[i]->Value) {
69          throw 1;
70        }
71        //проверка на число
72        int current;
73        resultParse = Int32::TryParse(this->dataGridRow->Rows[0]->Cells[i]->Value->
              ToString(), current);
74        if (!resultParse) {
75          throw 2;
76        }
77      }
78    }
79    catch (int errorCode) {
80      switch (errorCode) {
81      case 1:
82        this->errorProvider->SetError(this->dataGridRow, "В строке есть пустые
              ячейки");
83        break;
84      case 2:
```

```
85          this->errorProvider->SetError(this->dataGridRow, "В строке есть не целые
                числа");
86          break;
87        }
88      }
89      //заменяем строки
90      try {
91        //если есть ошибки в заполнении
92        if (this->errorProvider->GetError(this->dataGridInput)->ToString() != String::
              Empty
93          || this->errorProvider->GetError(this->dataGridRow)->ToString() != String::
                Empty) {
94          throw 0;
95        }
96        //добавляем столбцы в матрице вывода
97        for (int i = 0; i < this->dataGridInput->ColumnCount; i++) {
98          this->dataGridOutput->Columns->Add("ColumnOutput" + i, String::Empty);
99          this->dataGridOutput->Columns[i]->Width = 40;
100       }
101       //добавляем строки
102       this->dataGridOutput->Rows->Add(this->dataGridInput->RowCount - 1);
103
104       //заполняем ячейки
105       for (int i = 0; i < this->dataGridInput->RowCount - 1; i++) {
106         bool f = false;
107         for (int j = 0; j < this->dataGridInput->ColumnCount; j++) {
108           if (Int32::Parse(this->dataGridInput->Rows[i]->Cells[j]->Value->ToString())
                  == min) {
109             f = true;
110           }
111           //если есть минимальный
112           if (f) {
113             //переписываем из строки для замены
114             for (int j = 0; j < this->dataGridOutput->ColumnCount; j++) {
115               this->dataGridOutput->Rows[i]->Cells[j]->Value = this->dataGridRow
                      ->Rows[0]->Cells[j]->Value;
116             }
117           }
118           else {
119             //переписываем из исходной строки
120             for (int j = 0; j < this->dataGridOutput->ColumnCount; j++) {
```

```
121                 this->dataGridOutput->Rows[i]->Cells[j]->Value = this->dataGridInput
                        ->Rows[i]->Cells[j]->Value;
122             }
123         }
124     }
125   }
126 }
127 catch (int errorCode) {
128 }
```

После запуска приложения на экране появляется окно (см. рисунок 29).

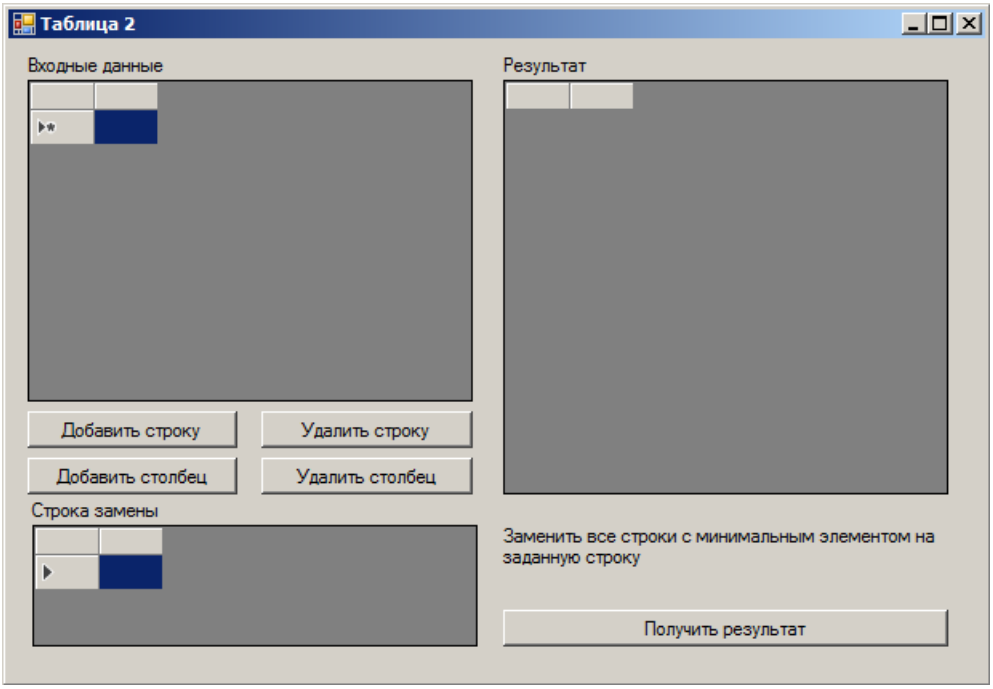

Рисунок 29 – Окно приложения «Таблица 2»: начальный запуск

При нажатии кнопки «Добавить строку» в таблицу добавляется новая строка (см. рисунок 30).

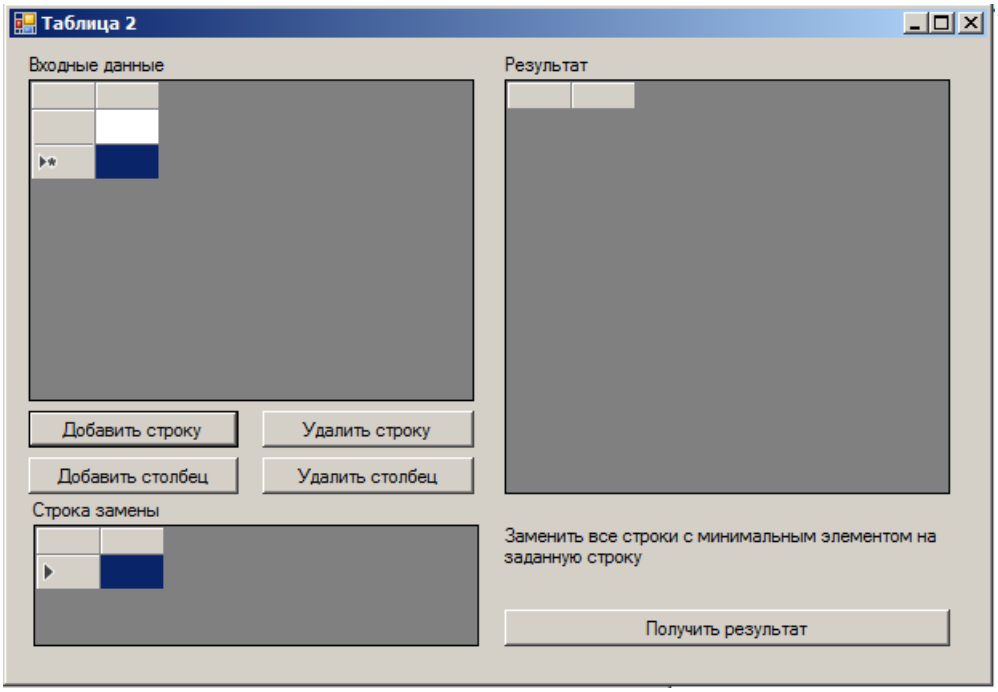

Рисунок 30 – Окно приложения «Таблица 2»: добавление новой строки

При нажатии кнопки «Удалить строку» из таблицы удаляется выбранная строка (см. рисунок 31).

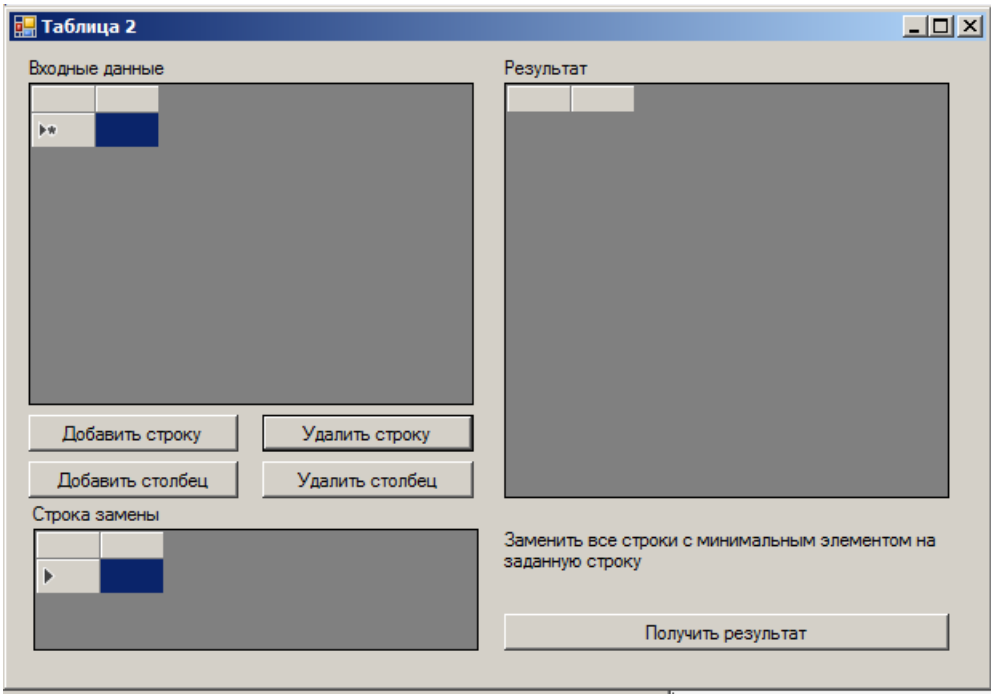

Рисунок 31 – Окно приложения «Таблица 2»: удаление строки

При нажатии кнопки «Добавить столбец» в таблицу и строку замены добавляется новый столбец (см. рисунок 32).

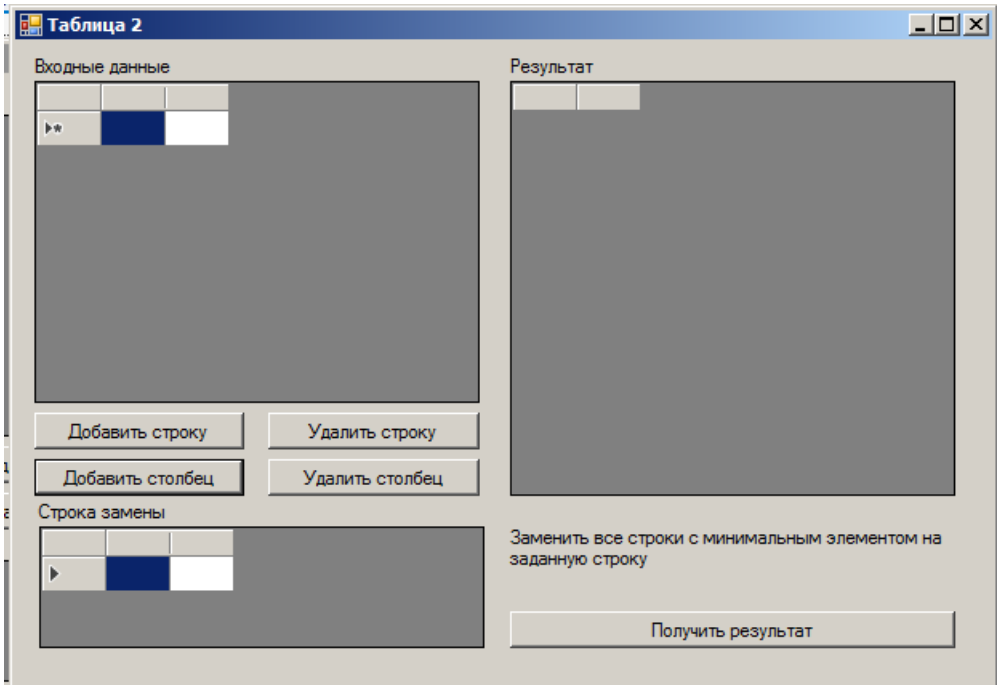

Рисунок 32 – Окно приложения «Таблица 2»: добавление столбца

При нажатии кнопки «Удалить столбец» из таблицы и из строки замены удаляется выбранный столбец (см. рисунок 33).

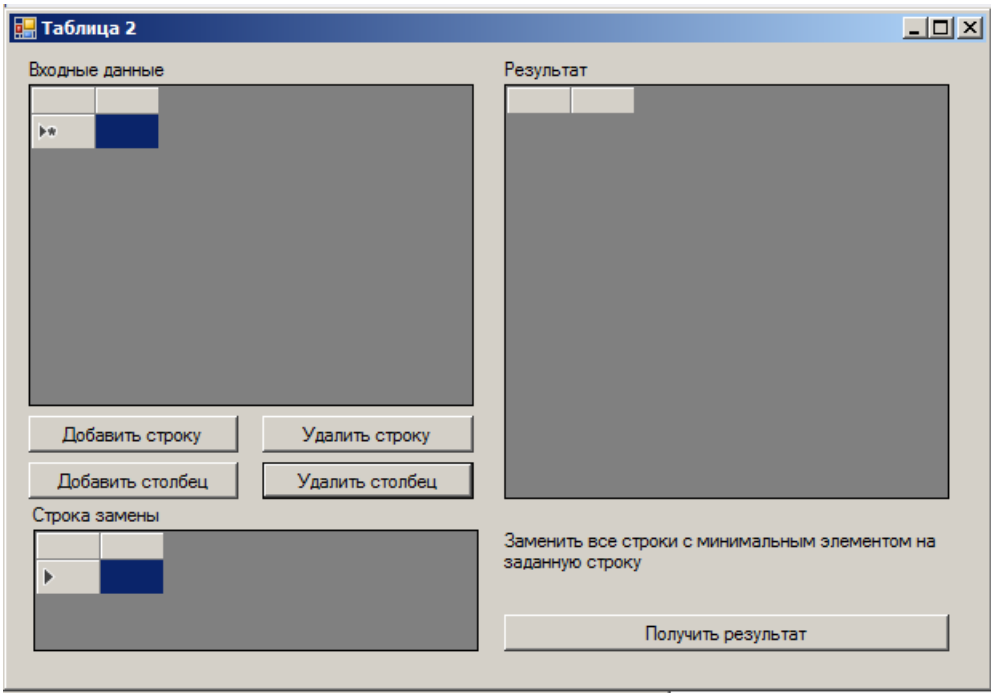

Рисунок 33 – Окно приложения «Таблица 2»: удаление столбца

При корректном вводе все строки, в которых есть минимальный элемент, заменяются строкой замены. Результат выводится в новую таблицу (см. рисунок 34).

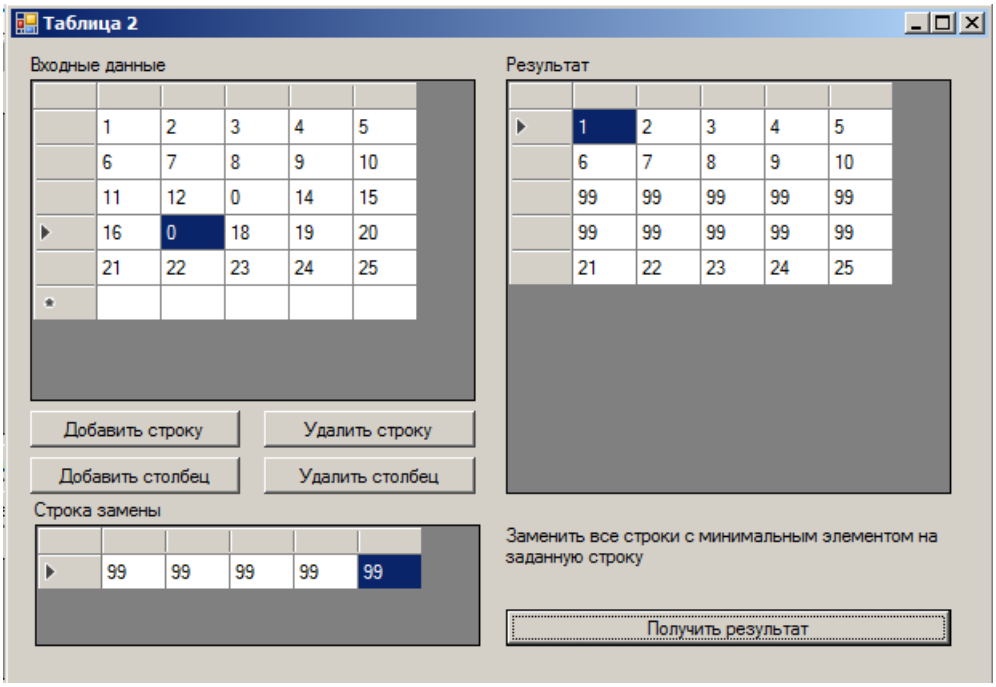

Рисунок 34 – Окно приложения «Таблица 2»: корректные данные

При попытке замены строк из таблицы, в которой нет ни одной строки, возникает сообщение об ошибке (см. рисунок 35).

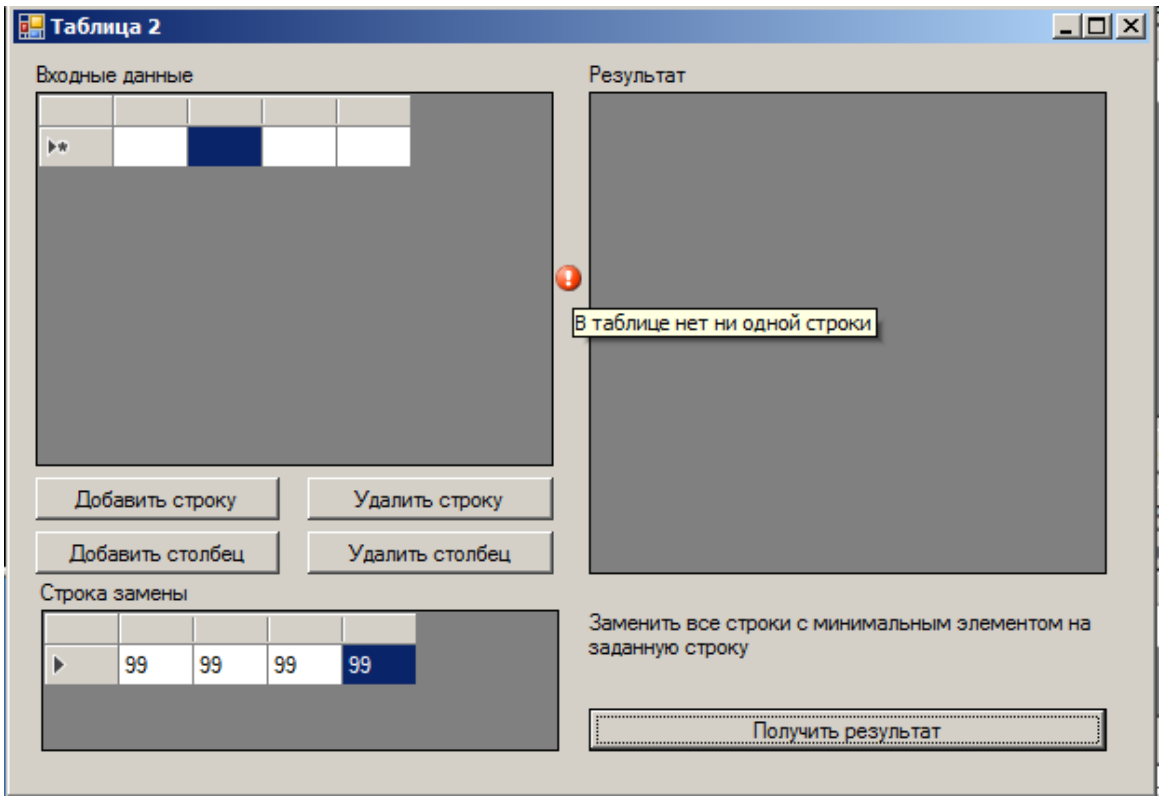

Рисунок 35 – Окно приложения «Таблица 2»: сообщение об отсутствии строк

При попытке замены строк из таблицы, в которой есть пустые ячейки, возникает сообщение об ошибке (см. рисунок 36).

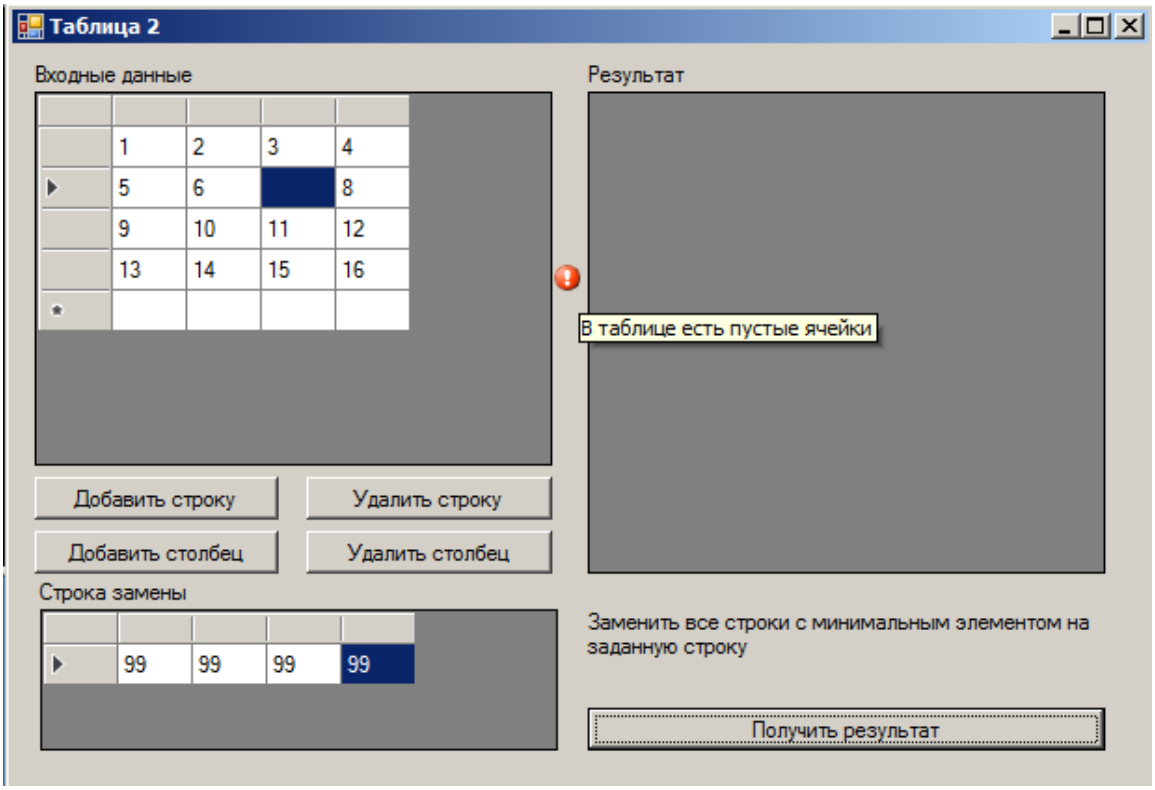

Рисунок 36 – Окно приложения «Таблица 2»: сообщение о пустых ячейках

При попытке замены строк из таблицы, в ячейках которой есть не целые числа, возникает сообщение об ошибке (см. рисунок 37).

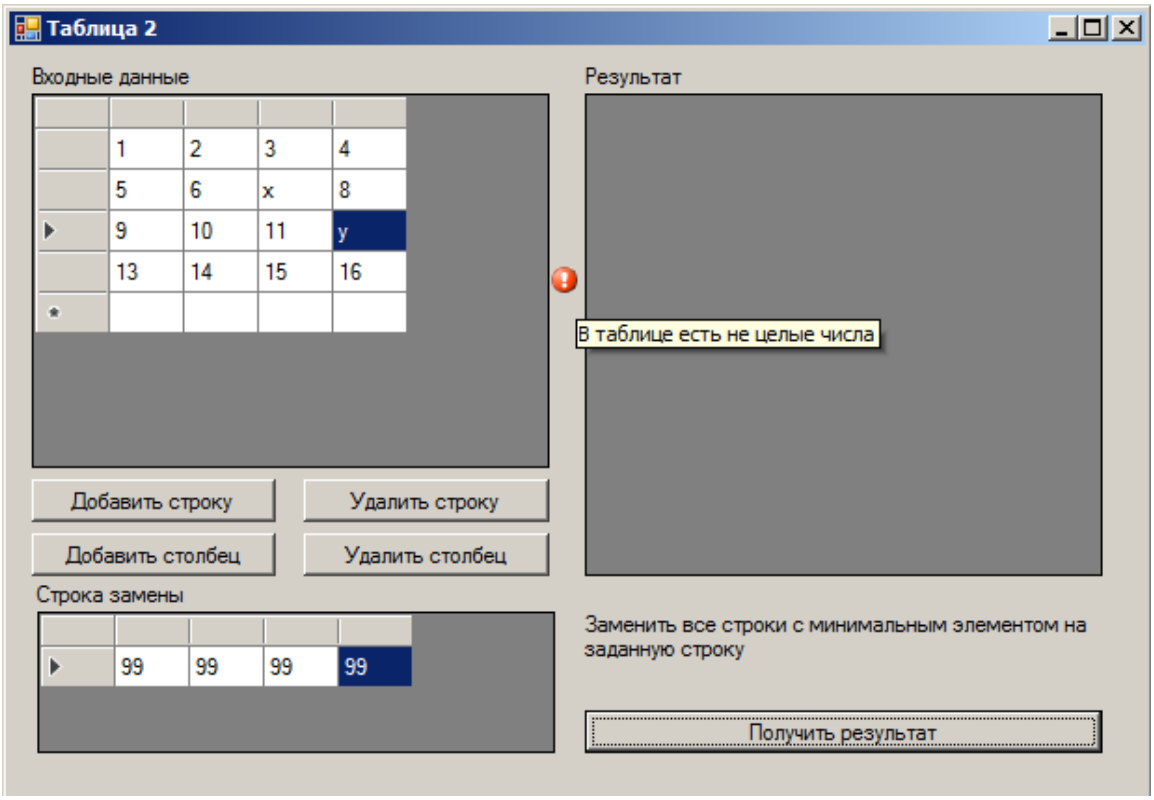

Рисунок 37 – Окно приложения «Таблица 2»: сообщение о наличии не целых чисел в строке

При попытке ввода пустых ячеек в строке замены возникает сообщение об ошибке (см. рисунок 38).

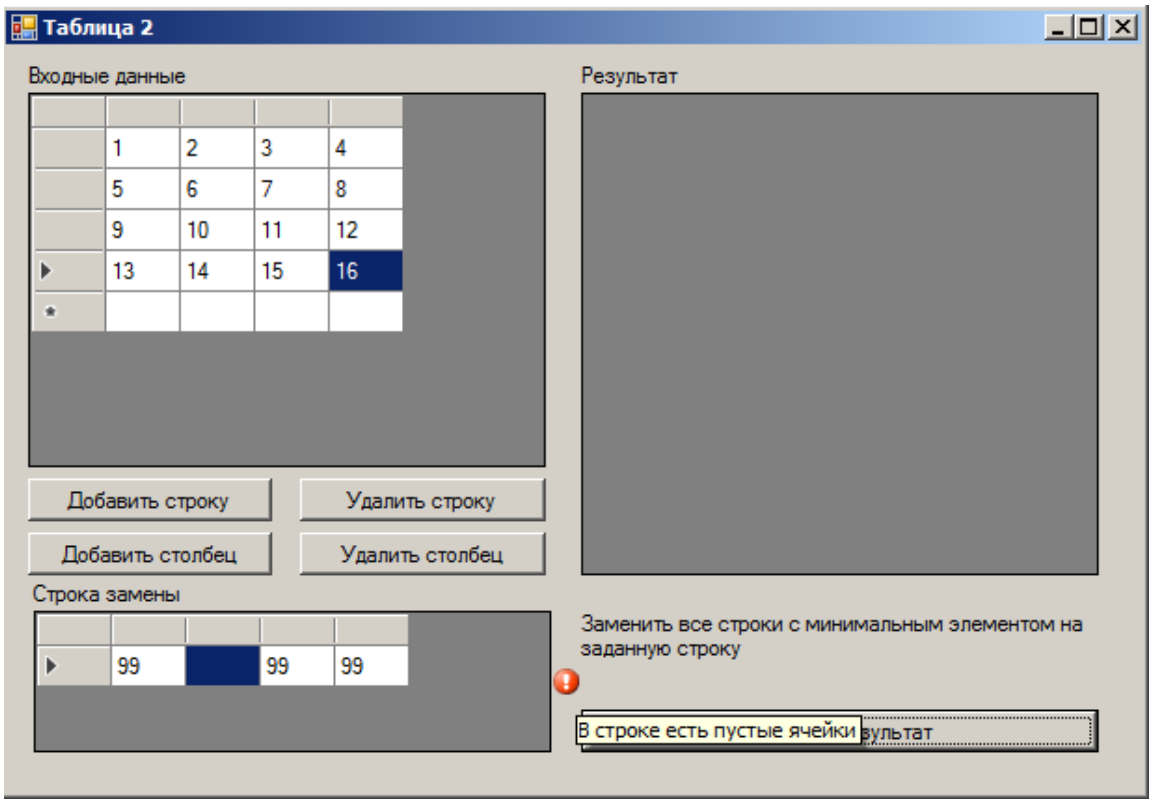

Рисунок 38 – Окно приложения «Таблица 2»: сообщение о пустых ячейках строки

При попытке ввода строки, в ячейках которой есть не целые числа, возникает сообщение об ошибке (см. рисунок 39).

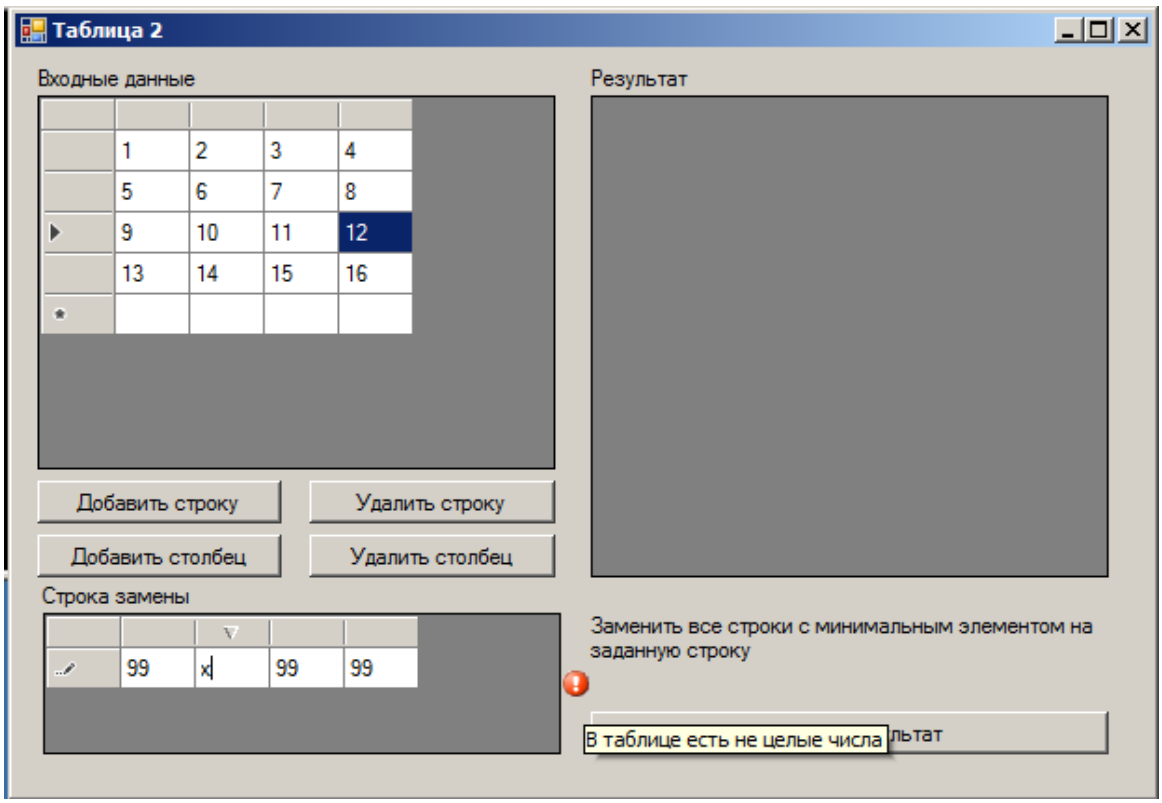

Рисунок 39 – Окно приложения «Таблица 2»: сообщение о наличии не целых чисел в строке

Полный код программы приведен в приложении А.

6 Работа с файлами

Задание. Создать приложение для создания и редактирования массива данных с заданной структурой (используя табличную обработку данных). Организовать сохранение данных в файл и загрузку в форму из файла. Создать таблицу `Travel`, содержащую следующие поля: Фамилия, имя, отчество туриста, название гостиницы, срок пребывания, дата приезда, дата отъезда. Выполнить следующие действия: считать из файла и вывести на экран данные о всех туристах, в другой файл вывести данные о туристах, отдыхающих в данный день.

Создано окно приложения, содержащее два элемента `DataGridView`, шесть элементов `Button`, один элемент `TextBox` и три элемента `Label`. Для отображения сообщений об ошибках в окно добавлен элемент `ErrorProvider`. Для файловых диалогов в окно добавлен элемент `saveFileDialog` и `openFileDialog`. Вид окна представлен на рисунке 40.

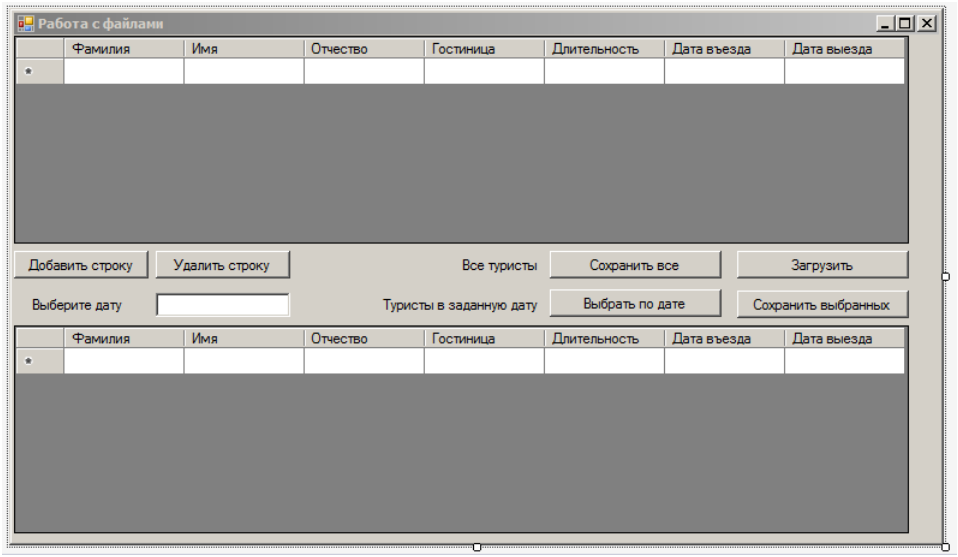

Рисунок 40 – Окно приложения «Работа с файлами», открытое в конструкторе

У элементов изменены значения некоторых атрибутов. Значения измененных атрибутов представлены в таблице 6.

Таблица 6 – Значения атрибутов элементов в приложении «Работа с файлами»

Наименование атрибута	Значение
Для формы	
`Text`	`Работа с файлами`
Для первой надписи	
`(Name)`	`lblAll`

Text	Все туристы
Для второй надписи	
(Name)	lblOnlyDate
Text	Туристы в заданную дату
Для третьей надписи	
(Name)	lblChoose
Text	Выберите дату
Для первого текстового поля	
(Name)	txtDate
Для первой таблицы	
(Name)	Travel
Для второй таблицы	
(Name)	TravelFilter
ReadOnly	True
Для первой кнопки	
(Name)	btnAdd
Text	Добавить строку
Для второй кнопки	
(Name)	btnRemove
Text	Удалить строку
Для третьей кнопки	
(Name)	btnSave
Text	Сохранить все
Для четвертой кнопки	
(Name)	btnLoad
Text	Загрузить
Для пятой кнопки	
(Name)	btnSave
Text	Выбрать по дате
Для шестой кнопки	
(Name)	btnSaveChoose
Text	Сохранить выбранных
Для обработчика ошибок	
(Name)	errorProvider
Для файлового диалога сохранения	
(Name)	saveFileDialog
Filter	Текстовые файлы (*.txt)|*.txt|Все файлы (*.*)|*.*
Для файлового диалога открытия	
(Name)	openFileDialog
Filter	Текстовые файлы (*.txt)|*.txt|Все файлы (*.*)|*.*

На нажатие кнопки «Добавить строку» установлено выполнение следующего кода:

```
this->TravelFilter->Rows->Clear();
this->Travel->Rows->Add(1);
```

На нажатие кнопки «Удалить строку» установлено выполнение следующего кода:

```
this->TravelFilter->Rows->Clear();
if (!this->Travel->CurrentRow->IsNewRow) {
  int i = this->Travel->CurrentRow->Index;
  this->Travel->Rows->Remove(this->Travel->Rows[i]);
}
```

На нажатие кнопки «Сохранить все» установлено выполнение следующего кода:

```
//Очищаем ошибки
this->errorProvider->SetError(this->txtDate, String::Empty);
this->errorProvider->SetError(this->Travel, String::Empty);
this->TravelFilter->Rows->Clear();
//Строка для записи в файл
String^ out = "";

//проверка таблицы
try {
  if (this->Travel->RowCount < 2) {
    throw 1;
  }
  //проверяем строки
  for (int i = 0; i < this->Travel->RowCount - 1; i++) {
    //Проверка на пустоту
    for (int j = 0; j < 7; j++) {
      if (!(System::String ^)this->Travel->Rows[i]->Cells[j]->Value) {
        throw 2;
      }
    }
    //Дата прибытия
    if (this->Travel->Rows[i]->Cells[5]->Value->ToString()->Length != 10
      || !cor(s2d(this->Travel->Rows[i]->Cells[5]->Value->ToString()))) {
      throw 3;
    }
    //Дата отбытия
```

```
27          if (this->Travel->Rows[i]->Cells[6]->Value->ToString()->Length != 10
28            || !cor(s2d(this->Travel->Rows[i]->Cells[6]->Value->ToString()))) {
29            throw 4;
30          }
31          //Дата прибытия больше даты отбытия
32          if (!aNotMoreThenB(
33            s2d(this->Travel->Rows[i]->Cells[5]->Value->ToString()),
34            s2d(this->Travel->Rows[i]->Cells[6]->Value->ToString()))) {
35            throw 5;
36          }
37          //Длительность
38          int longDate;
39          if (!Int32::TryParse(this->Travel->Rows[i]->Cells[4]->Value->ToString(),
              longDate)) {
40            throw 6;
41          }
42          //равенство пребывания и интервала дат
43          if (longDate !=
44            hmd(s2d(this->Travel->Rows[i]->Cells[5]->Value->ToString()),
45              s2d(this->Travel->Rows[i]->Cells[6]->Value->ToString()))) {
46            throw 7;
47          }
48          //пишем значения ячеек в строку через разделитель
49          for (int j = 0 ; j < 7; j++) {
50            if (this->Travel->Rows[i]->Cells[j]->Value->ToString()->Contains(";")) {
51              throw 8;
52            }
53            out += this->Travel->Rows[i]->Cells[j]->Value->ToString() + ";";
54          }
55        }
56      }
57      //отлавливаем ошибки
58      catch (int errorCode) {
59        switch (errorCode) {
60        case 1:
61          this->errorProvider->SetError(this->Travel, "Пустая таблица");
62          break;
63        case 2:
64          this->errorProvider->SetError(this->Travel, "Есть пустые ячейки");
65          break;
66        case 3:
```

```
67          this->errorProvider->SetError(this->Travel, "Некорректная дата прибытия");
68          break;
69        case 4:
70          this->errorProvider->SetError(this->Travel, "Некорректная дата отбытия");
71          break;
72        case 5:
73          this->errorProvider->SetError(this->Travel, "Дата прибытия больше даты
                отбытия");
74          break;
75        case 6:
76          this->errorProvider->SetError(this->Travel, "Некорректная длительность
                пребывания");
77          break;
78        case 7:
79          this->errorProvider->SetError(this->Travel, "Неверная длительность
                пребывания");
80          break;
81        case 8:
82          this->errorProvider->SetError(this->Travel, "Ячейки не должны содержать
                точку с запятой");
83          break;
84        }
85      }
86
87      System::IO::Stream^ myStream;
88      //если нет ошибок таблицы
89      if ((String ^)this->errorProvider->GetError(this->Travel) == String::Empty
90        && this->saveFileDialog->ShowDialog() == System::Windows::Forms::DialogResult::
              OK) {
91        if ((myStream = saveFileDialog->OpenFile()) != nullptr) {
92          System::IO::StreamWriter^ sw =
93            gcnew System::IO::StreamWriter(myStream, System::Text::Encoding::GetEncoding
                  (1251));
94          //Запись в файл
95          sw->Write(out);
96          sw->Close();
97        }
98      }
```

На нажатие кнопки «Загрузить» установлено выполнение следующего кода:

```
1     //чистим ошибки и поля
```

```
2   this->TravelFilter->Rows->Clear();
3   this->errorProvider->Clear();
4   this->txtDate->Text = String::Empty;
5   //открываем файл
6   System::IO::Stream^ myStream;
7   if (this->openFileDialog->ShowDialog() == System::Windows::Forms::DialogResult::
        OK) {
8     if ((myStream = openFileDialog->OpenFile()) != nullptr) {
9       System::IO::StreamReader^ sw = gcnew System::IO::StreamReader(myStream,
10        System::Text::Encoding::GetEncoding(1251));
11      System::String ^str1 = "";
12      str1 = sw->ReadToEnd();
13      //разбиваем по разделителю
14      array<System::String ^>^ str = str1->Split(';');
15      int i = 0;
16      int j = 0;
17      //чистим таблицу
18      this->Travel->Rows->Clear();
19      //заполняем таблицу
20      for each (String ^ s in str) {
21        //Добавляем строку
22        if (!j) {
23          this->Travel->Rows->Add(1);
24        }
25        //Заполняем ячейки
26        this->Travel->Rows[i]->Cells[j++]->Value = s;
27        if (j > 6) {
28          j = 0;
29          i++;
30        }
31      }
32      //Удаляем лишнюю строку в конце
33      this->Travel->Rows->RemoveAt(this->Travel->RowCount - 2);
34      //Закрываем файловый поток
35      sw->Close();
36    }
37  }
```

На нажатие кнопки «Выбрать по дате» установлено выполнение следующего кода:

```
1   this->errorProvider->SetError(this->txtDate, String::Empty);
2   this->errorProvider->SetError(this->Travel, String::Empty);
```

```
3   this->TravelFilter->Rows->Clear();
4   //Строка для записи
5   String^ out = "";
6
7   //проверка таблицы
8   try {
9     if (this->Travel->RowCount < 2) {
10      throw 1;
11    }
12    //проверяем строки
13    for (int i = 0; i < this->Travel->RowCount - 1; i++) {
14      //Проверка на пустоту
15      for (int j = 0; j < 7; j++) {
16        if (!(System::String ^)this->Travel->Rows[i]->Cells[j]->Value) {
17          throw 2;
18        }
19      }
20      //Дата прибытия
```

На нажатие кнопки «Сохранить выбранных» установлено выполнение следующего кода:

```
1   //чистим ошибки
2   this->errorProvider->SetError(this->txtDate, String::Empty);
3   this->errorProvider->SetError(this->Travel, String::Empty);
4   this->txtDate->Text = String::Empty;
5   //Строка для записи в файл
6   String^ out = "";
7
8   try {
9     //Если пустая таблица
10    if (this->TravelFilter->RowCount < 2) {
11      throw 1;
12    }
13    //пишем в строку через разделитель
14    for (int i = 0; i < this->TravelFilter->RowCount - 1; i++) {
15      for (int j = 0; j < 7; j++) {
16        out += this->TravelFilter->Rows[i]->Cells[j]->Value->ToString() + ";";
17      }
18    }
19  }
20  catch (int errorCode) {
21    switch (errorCode) {
```

```
22      case 1:
23        this->errorProvider->SetError(this->Travel, "Пустая_таблица");
24        break;
25      }
26    }
27
28    System::IO::Stream^ myStream;
29    //если нет ошибок таблицы
30    if ((String ^)this->errorProvider->GetError(this->Travel) == String::Empty
31      && (String ^)this->errorProvider->GetError(this->txtDate) == String::Empty
32      && this->saveFileDialog->ShowDialog() == System::Windows::Forms::DialogResult::
            OK) {
33      if ((myStream = saveFileDialog->OpenFile()) != nullptr) {
34        System::IO::StreamWriter^ sw =
35          gcnew System::IO::StreamWriter(myStream, System::Text::Encoding::GetEncoding
                (1251));
36        //Запись в файл
37        sw->Write(out);
38        sw->Close();
39      }
40    }
```

После запуска приложения на экране появляется окно (см. рисунок 41).

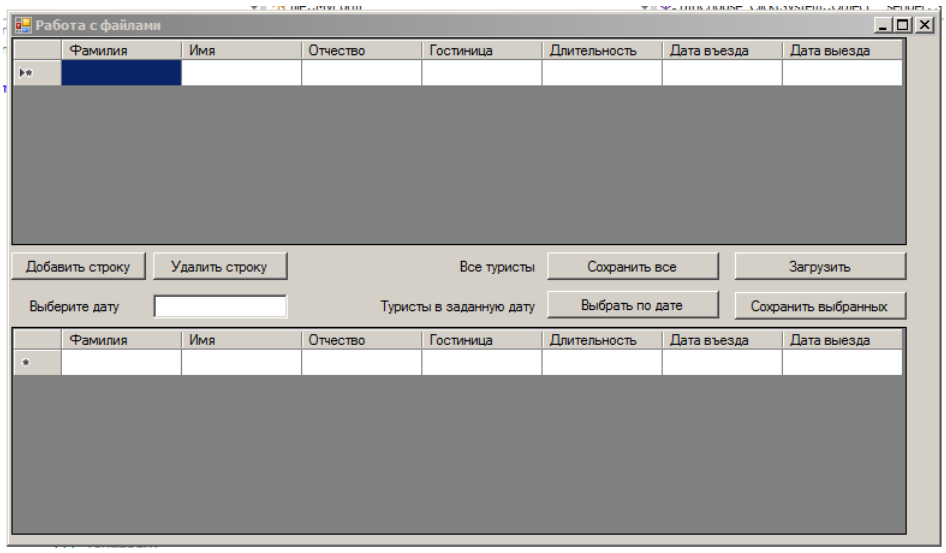

Рисунок 41 – Окно приложения «Работа с файлами»: начальный запуск

При нажатии кнопки «Добавить строку» в таблицу добавляется новая строка (см. рисунок 42).

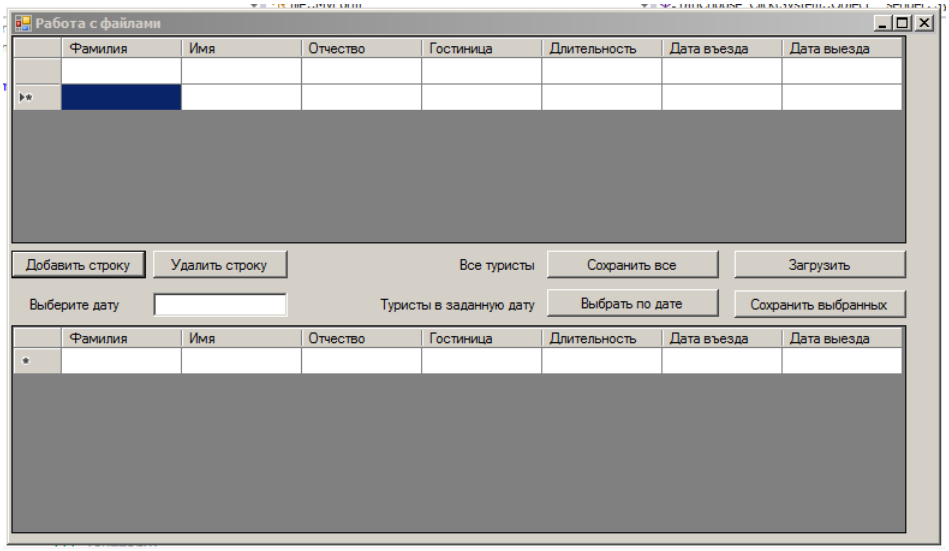

Рисунок 42 – Окно приложения «Работа с файлами»: добавление новой строки

При нажатии кнопки «Удалить строку» из таблицы удаляется выбранная строка (см. рисунок 43).

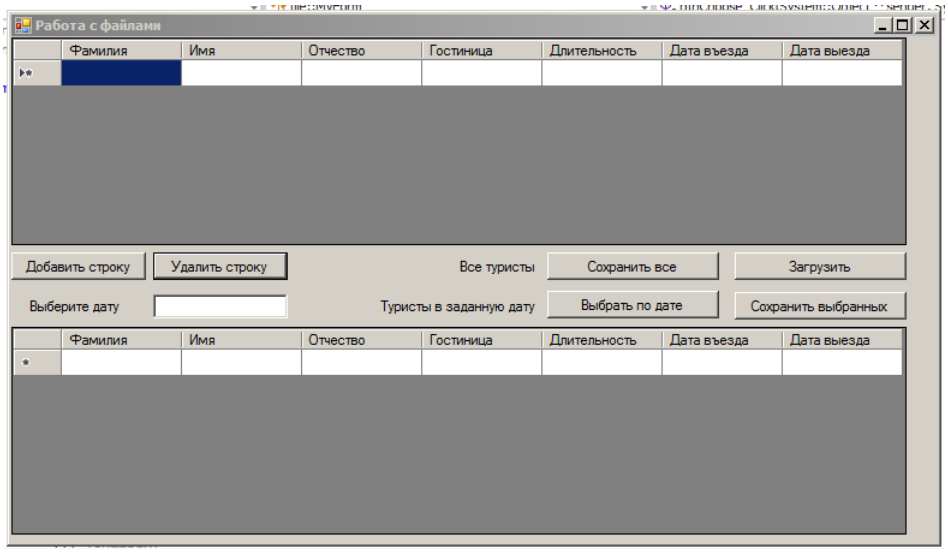

Рисунок 43 – Окно приложения «Работа с файлами»: удаление строки

При нажатии кнопки «Сохранить все» происходит проверка введенных данных и открывается файловый диалог для сохранения файла с данными из верхней таблицы.

При нажатии кнопки «Загрузить» открывается файловый диалог для открытия файла с данными для верхней таблицы.

При нажатии кнопки «Выбрать по дате» в нижнюю таблицу переписываются удовлетворяющие введенным данным записи (см. рисунок 44).

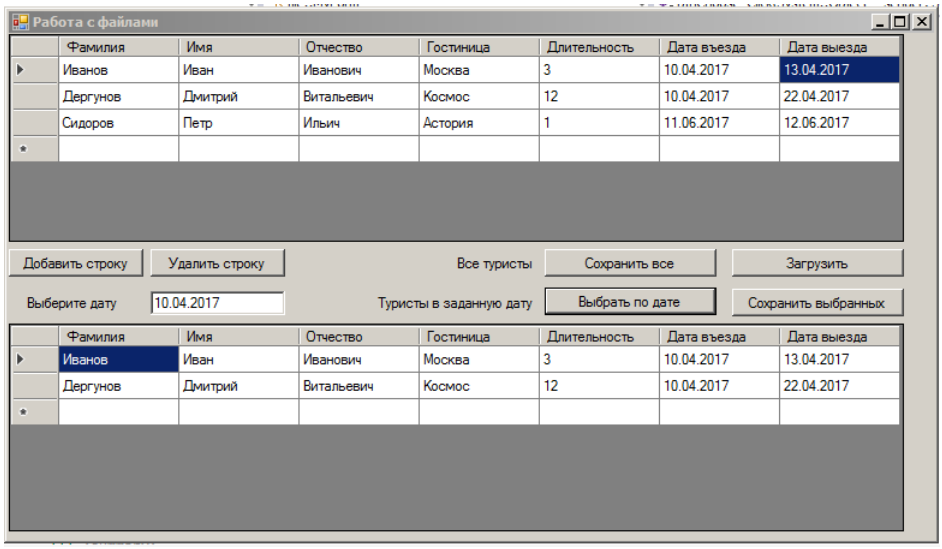

Рисунок 44 – Окно приложения «Работа с файлами»: выбор по дате

При нажатии кнопки «Сохранить выбранных» открывается файловый диалог для сохранения файла с данными из нижней таблицы.

Если таблица пустая, возникает сообщение об ошибке (см. рисунок 45).

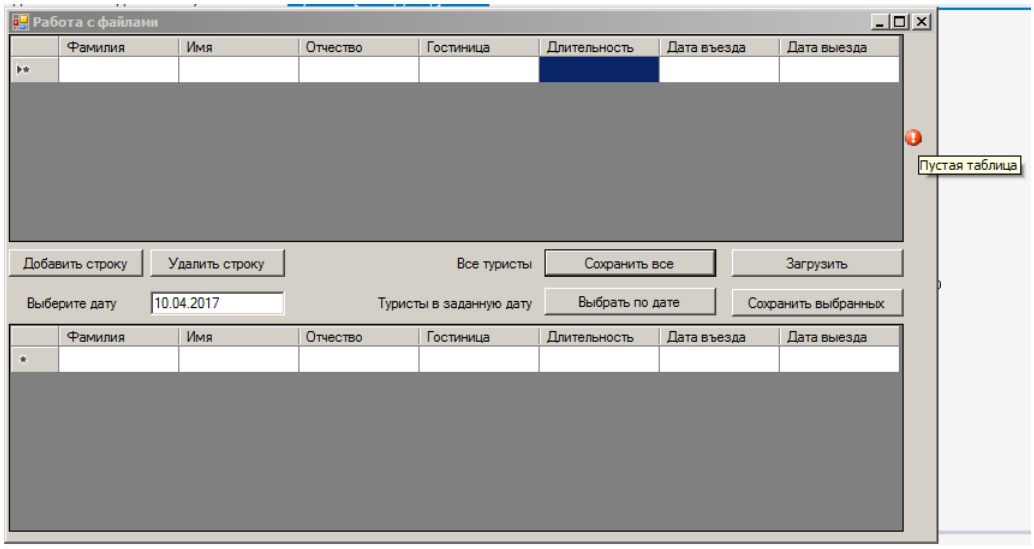

Рисунок 45 – Окно приложения «Работа с файлами»: пустая таблица

Если в таблице есть пустые ячейки, возникает сообщение об ошибке (см. рисунок 46).

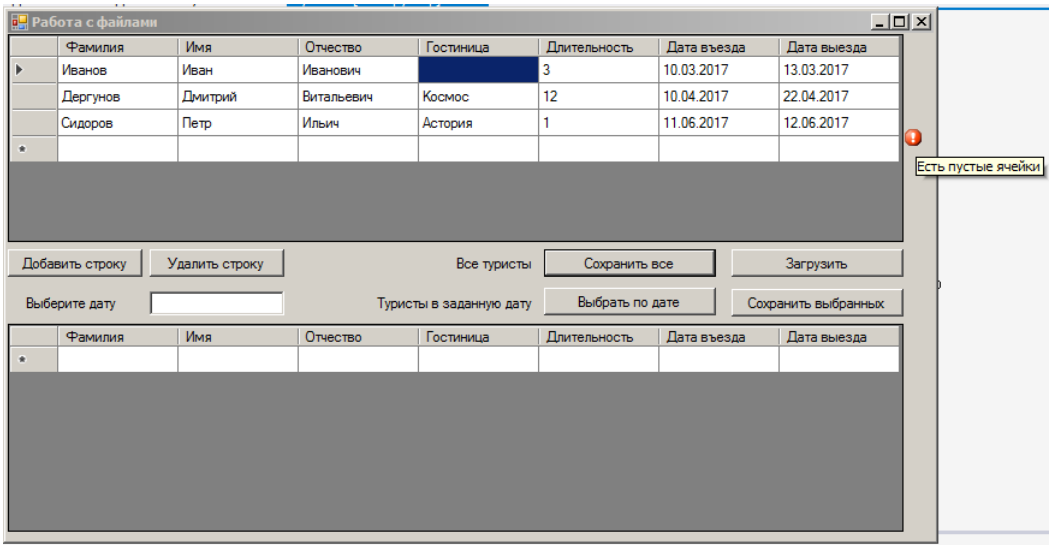

Рисунок 46 – Окно приложения «Работа с файлами»: пустые ячейки

В случае ввода некорректной даты прибытия возникает сообщение об ошибке (см. рисунок 47).

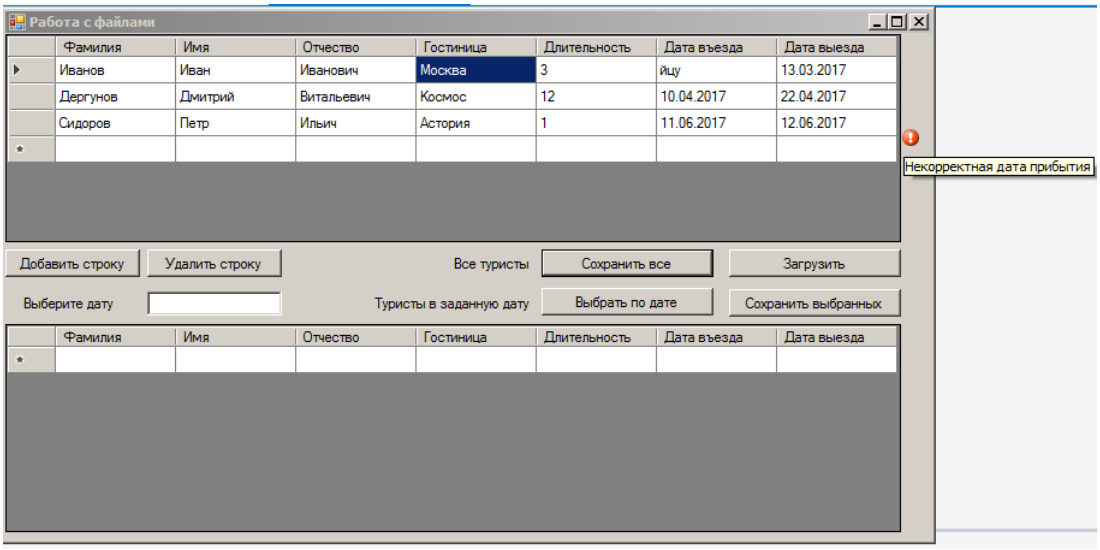

Рисунок 47 – Окно приложения «Работа с файлами»: некорректная дата прибытия

В случае ввода некорректной даты отбытия возникает сообщение об ошибке (см. рисунок 48).

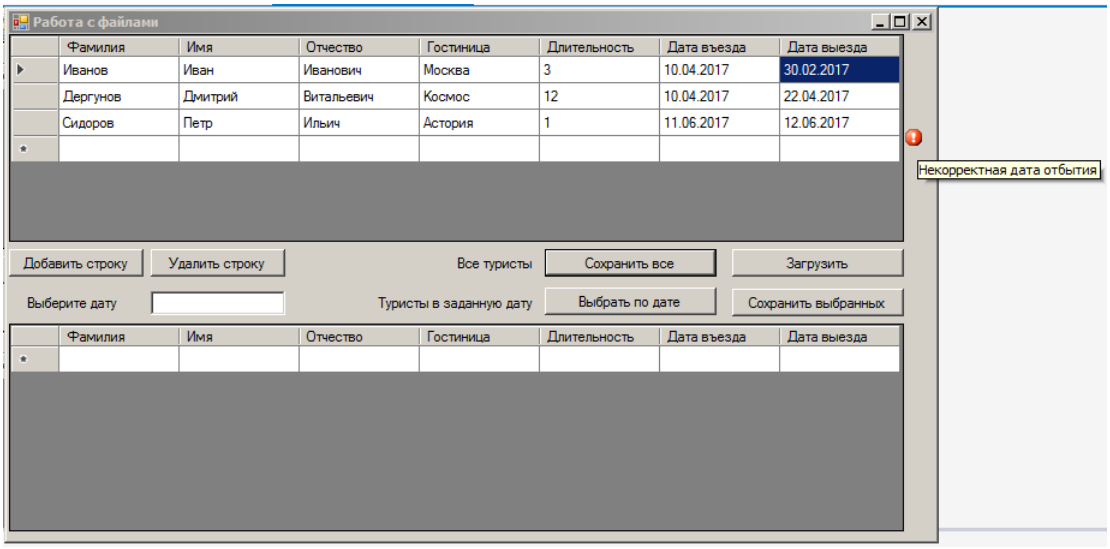

Рисунок 48 – Окно приложения «Работа с файлами»: некорректная дата отбытия

В случае, если дата прибытия больше даты отбытия, возникает сообщение об ошибке (см. рисунок 49).

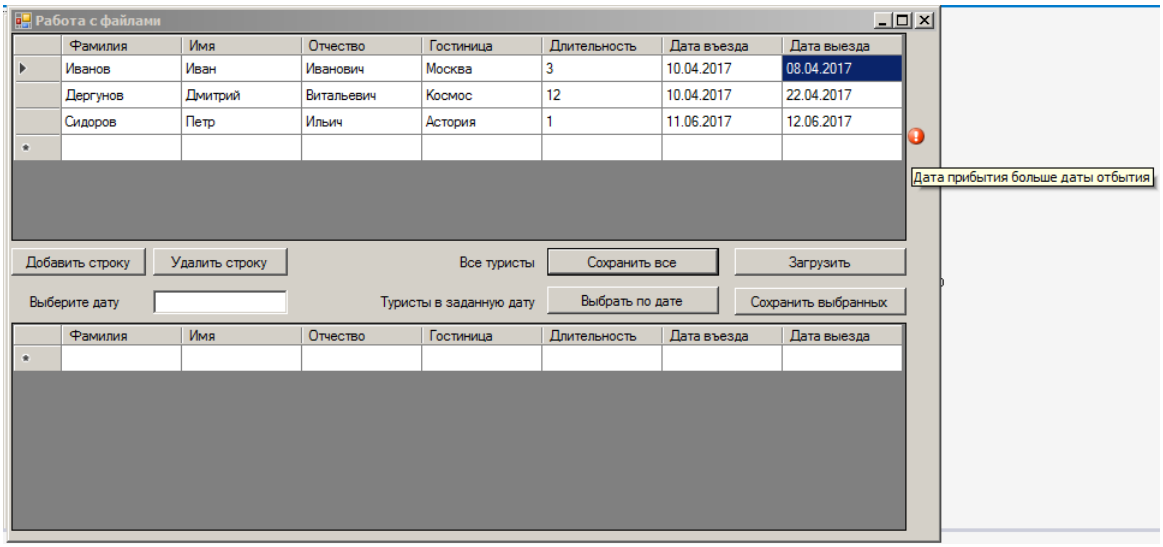

Рисунок 49 – Окно приложения «Работа с файлами»: дата прибытия больше даты отбытия

В случае ввода неверной длительности пребывания возникает сообщение об ошибке (см. рисунок 50).

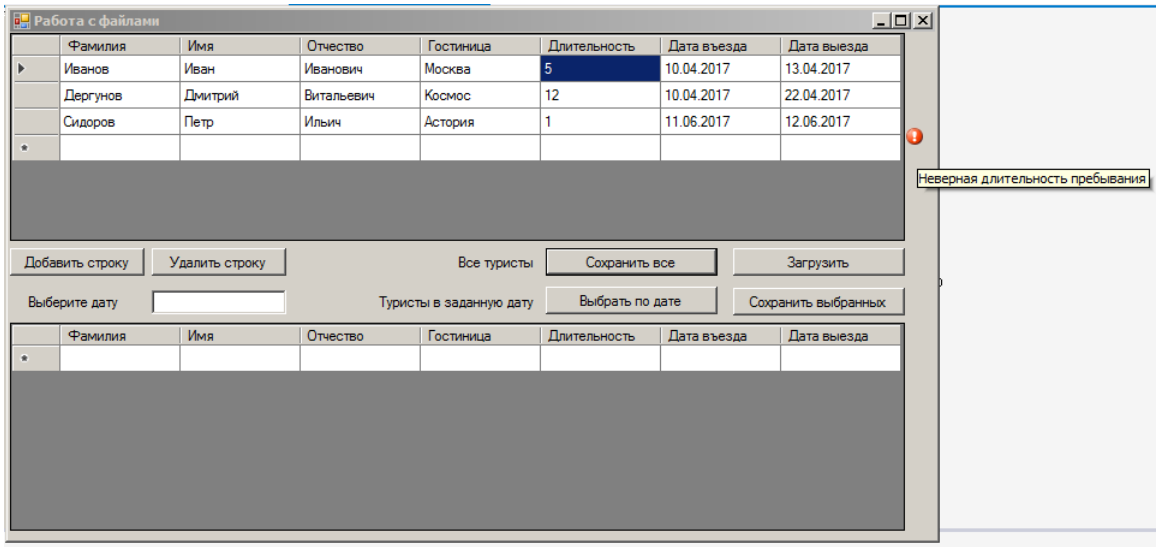

Рисунок 50 – Окно приложения «Работа с файлами»: неверная длительность пребывания

В случае ввода некорректной даты пребывания возникает сообщение об ошибке (см. рисунок 51).

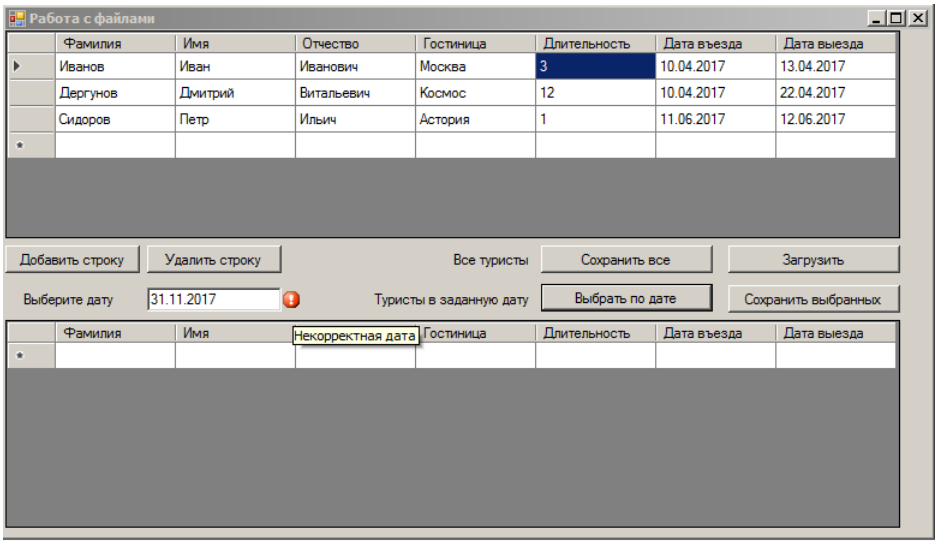

Рисунок 51 – Окно приложения «Работа с файлами»: некорректная дата пребывания

Полный код программы приведен в приложении А.

7 Использование коллекций

Задание. Используя коллекции, создать приложение для работы с двусвязным списком. Создать двусвязный список, состоящий из целых чисел. Предусмотреть возможность создания списка из набора чисел, добавления одного элемента, удаления одного элемента, вывода результата на экран, удаления всех элементов с помощью кнопок. Найти сумму элементов, больших минимального нечетного элемента. Получить новый список, удалив все максимальные элементы.

Создано окно приложения, содержащее семь элементов `Button`, пять элементов `TextBox` и пять элементов `Label`. Для отображения сообщений об ошибках в окно добавлен элемент `ErrorProvider`. Вид окна представлен на рисунке 52.

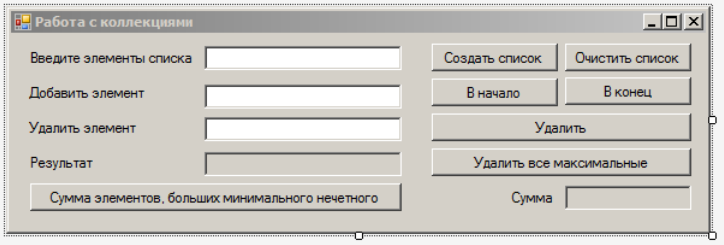

Рисунок 52 – Окно приложения «Работа с коллекциями», открытое в конструкторе

У элементов изменены значения некоторых атрибутов. Значения измененных атрибутов представлены в таблице 7.

Таблица 7 – Значения атрибутов элементов в приложении «Работа с коллекциями»

Наименование атрибута	Значение
Для формы	
`Text`	`Работа с коллекциями`
Для первой надписи	
`(Name)`	`lblInput`
`Text`	`Введите элементы списка`
Для второй надписи	
`(Name)`	`lblAdd`
`Text`	`Добавить элемент`
Для третьей надписи	
`(Name)`	`lblRemove`
`Text`	`Удалить элемент`
Для четвертой надписи	
`(Name)`	`lblResult`

Text	Результат
Для пятой надписи	
(Name)	lblSum
Text	Сумма
Для первого текстового поля	
(Name)	txtInput
Для второго текстового поля	
(Name)	txtAdd
Для третьего текстового поля	
(Name)	txtRemove
Для четвертого текстового поля	
(Name)	txtResult
ReadOnly	True
Для пятого текстового поля	
(Name)	txtSum
ReadOnly	True
Для первой кнопки	
(Name)	btnCreate
Text	Создать список
Для второй кнопки	
(Name)	btnClear
Text	Очистить список
Для третьей кнопки	
(Name)	btnAddFront
Text	В начало
Для четвертой кнопки	
(Name)	btnAddBack
Text	В конец
Для пятой кнопки	
(Name)	btnRemove
Text	Удалить
Enabled	False
Для шестой кнопки	
(Name)	btnRemoveAllMax
Text	Удалить все максимальные
Enabled	False
Для седьмой кнопки	
(Name)	btnSum
Text	Сумма элементов, больших минимального нечетного
Enabled	False

Для обработчика ошибок	
(Name)	errorProvider

На нажатие кнопки «Создать список» установлено выполнение следующего кода:

```
1   //чистим ошибки
2   this->errorProvider->Clear();
3    list .Clear();
4   //создаем список
5   try {
6     //Проверка на пустоту строки
7     if (this->txtInput->Text == String::Empty) {
8       throw 1;
9     }
10    for each (String ^ str in this->txtInput->Text->Trim()->Split('␣')) {
11      //на случай если идет несколько разделителей(пробелов) подряд
12      if (str == String::Empty) {
13        continue;
14      }
15      //проверка на числа
16      int num;
17      if (!Int32::TryParse(str, num)) {
18        throw 2;
19      }
20      //добавление в конец списка
21       list .AddLast(num);
22    }
23    //показываем результат
24    this->showResult();
25
26  }
27  //отлавливаем ошибки
28  catch (int errorCode) {
29    switch (errorCode) {
30    case 1:
31      errorProvider->SetError(this->txtInput, "Пустая␣строка");
32      break;
33    case 2:
34      errorProvider->SetError(this->txtInput, "Есть␣не␣целые␣числа");
35      break;
```

```
36      }
37    }
```

На нажатие кнопки «Очистить список» установлено выполнение следующего кода:

```
1     list .Clear();
2    this->showResult();
```

На нажатие кнопки «В начало» установлено выполнение следующего кода:

```
1    try {
2      //проверка на пустоту
3      if (this->txtAdd->Text == String::Empty) {
4        throw 1;
5      }
6      //проверка на число
7      int num;
8      if (!Int32::TryParse(this->txtAdd->Text, num)) {
9        throw 2;
10     }
11     //добавляем в начало
12     list .AddFirst(num);
13     this->showResult();
14   }
15   //отлавливаем ошибки
16   catch (int errorCode) {
17     switch (errorCode)
18     {
19     case 1:
20       errorProvider->SetError(this->txtAdd, "Введите число");
21       break;
22     case 2:
23       errorProvider->SetError(this->txtAdd, "Введено не целое число");
24       break;
25     }
26   }
```

На нажатие кнопки «В конец» установлено выполнение следующего кода:

```
1    //чистим ошибку
2    this->errorProvider->Clear();
3    try {
```

```
4       //проверка на пустоту
5       if (this->txtAdd->Text == String::Empty) {
6         throw 1;
7       }
8       //проверка на числа
9       int num;
10      if (!Int32::TryParse(this->txtAdd->Text, num)) {
11        throw 2;
12      }
13      //Добавляем в конец списка
14      list.AddLast(num);
15      this->showResult();
16    }
17    //отлавливаем ошибки
18    catch (int errorCode) {
19      switch (errorCode)
20      {
21      case 1:
22        errorProvider->SetError(this->txtAdd, "Введите число");
23        break;
24      case 2:
25        errorProvider->SetError(this->txtAdd, "Введено не целое число");
26        break;
27      }
28    }
```

На нажатие кнопки «Удалить» установлено выполнение следующего кода:

```
1     try {
2       //проверка на пустоту
3       if (this->txtRemove->Text == String::Empty) {
4         throw 1;
5       }
6       //проверка на число
7       int num;
8       if (!Int32::TryParse(this->txtRemove->Text, num)) {
9         throw 2;
10      }
11      //удаляем первое вхождение
12      if (!list.Remove(num)) {
13        throw 3;
14      }
```

```
15      this->showResult();
16    }
17    //отлавливаем ошибки
18    catch (int errorCode) {
19      switch (errorCode)
20      {
21      case 1:
22        errorProvider->SetError(this->txtRemove, "Введите число");
23        break;
24      case 2:
25        errorProvider->SetError(this->txtRemove, "Введено не целое число");
26        break;
27      case 3:
28        errorProvider->SetError(this->txtRemove, "Нет такого числа в списке");
29        break;
30      }
31    }
```

На нажатие кнопки «Удалить все максимальные» установлено выполнение следующего кода:

```
1   System::Collections::Generic::LinkedListNode<int>^ node = list.First;
2   int max = node->Value;
3   //идем прямо. ищем максимум
4   while ( node = node->Next) {
5     if (node->Value > max) {
6       max = node->Value;
7     }
8   }
9   //удаляем максимумы
10  while(list.Remove(max));
11  //выводим
12  this->showResult();
```

На нажатие кнопки «Сумма элементов, больших минимального нечетного» установлено выполнение следующего кода:

```
1   this->errorProvider->Clear();
2   try {
3     int min = 0;
4     System::Collections::Generic::LinkedListNode<int>^ node = list.First;
5     //идем прямо. ищем минимум
6     do {
7       if ((node->Value % 2) && ((node->Value < min) || !min)) {
```

```
8           min = node->Value;
9         }
10      } while (node = node->Next);
11      //если нет минимального
12      if (!min) {
13        throw 1;
14      }
15      //считаем сумму
16      int s = 0;
17      node = list.First;
18      do {
19        if (node->Value > min) {
20          s += node->Value;
21        }
22      } while (node = node->Next);
23      //Выводим сумму
24      this->txtSum->Text = System::Convert::ToString(s);
25    }
26    //отлавниваем ошибки
27    catch (int errorCode) {
28      switch (errorCode) {
29      case 1: this->errorProvider->SetError(this->btnSum, "Нет нечетных элементов");
30        break;
31      }
32    }
```

После запуска приложения на экране появляется окно (см. рисунок 53).

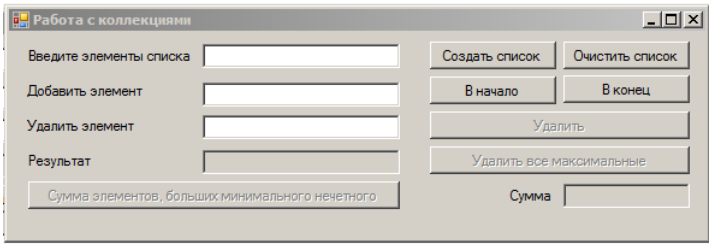

Рисунок 53 – Окно приложения «Работа с коллекциями»: начальный запуск

При нажатии кнопки «Создать список» создается новый список из элементов, введенных в соответствующее текстовое поле (см. рисунок 54).

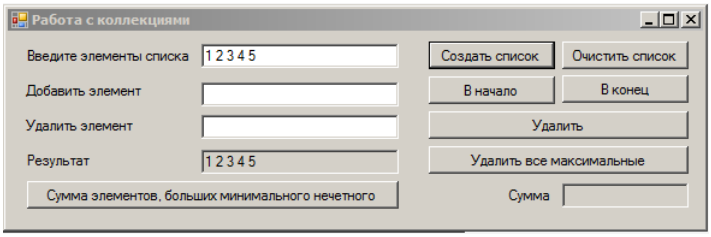

Рисунок 54 – Окно приложения «Работа с коллекциями»: создание списка

При нажатии кнопки «Очистить список» удаляется текущий список (см. рисунок 55).

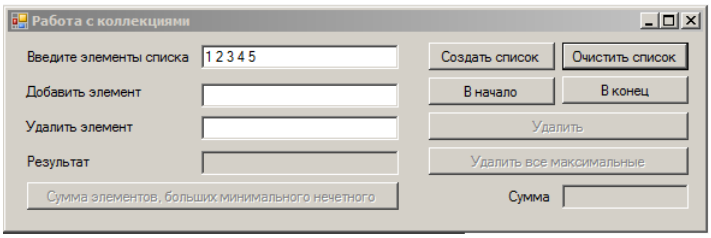

Рисунок 55 – Окно приложения «Работа с коллекциями»: очистка списка

При нажатии кнопки «В начало» в начало списка добавляется новый элемент из соответствующего текстового поля (см. рисунок 56).

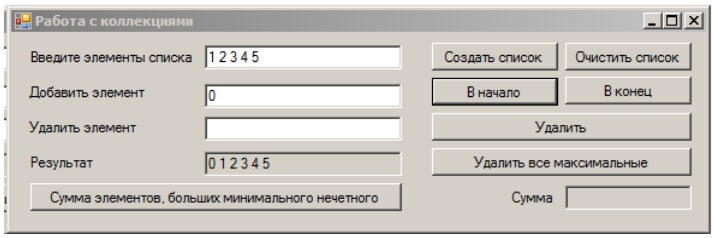

Рисунок 56 – Окно приложения «Работа с коллекциями»: добавление в начало

При нажатии кнопки «В конец» в конец списка добавляется новый элемент из соответствующего текстового поля (см. рисунок 57).

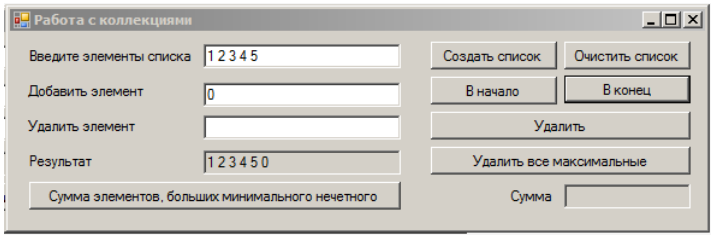

Рисунок 57 – Окно приложения «Работа с коллекциями»: добавление в конец

При нажатии кнопки «Удалить» из списка удаляется один элемент с указанным значением (см. рисунок 58).

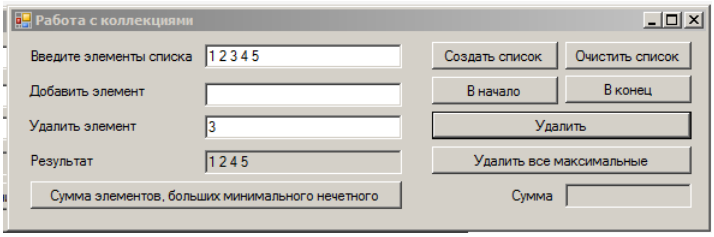

Рисунок 58 – Окно приложения «Работа с коллекциями»: удаление элемента

При нажатии кнопки «Удалить все максимальные» из списка удаляются все максимальные элементы (см. рисунок 59).

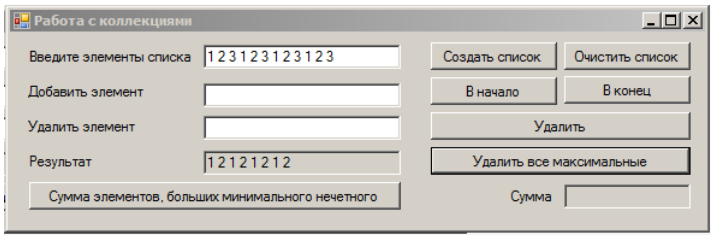

Рисунок 59 – Окно приложения «Работа с коллекциями»: удаление максимальных элементов

При нажатии кнопки «Сумма элементов, больших минимального нечетного» в соответствующее текстовое поле выводится сумма всех элементов, больших минимального нечетного (см. рисунок 60).

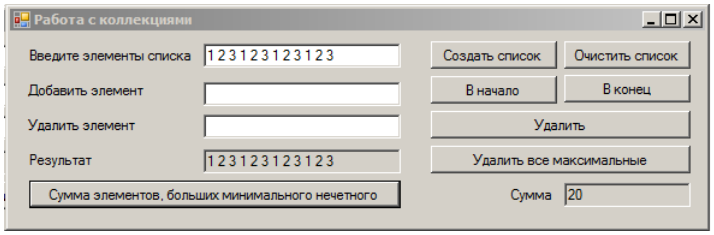

Рисунок 60 – Окно приложения «Работа с коллекциями»: сумма элементов, больших минимального нечетного

При попытки создания пустого списка возникает сообщение об ошибке (см. рисунок 61).

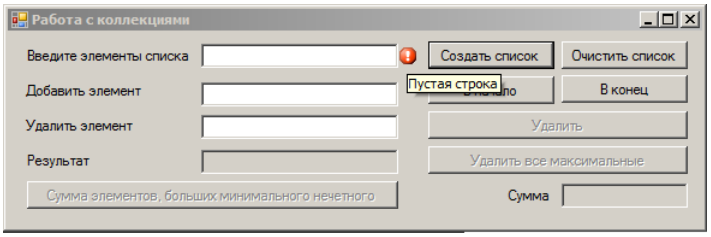

Рисунок 61 – Окно приложения «Работа с коллекциями»: пустой список

При попытки списка, состоящего не только из целых чисел, возникает сообщение об ошибке (см. рисунок 62).

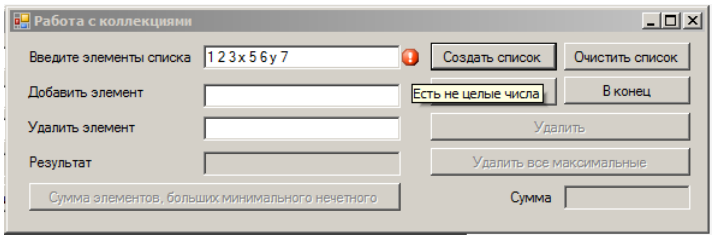

Рисунок 62 – Окно приложения «Работа с коллекциями»: не целые числа

При попытки добавления пустого элемента в начало или конец возникает сообщение об ошибке (см. рисунок 63).

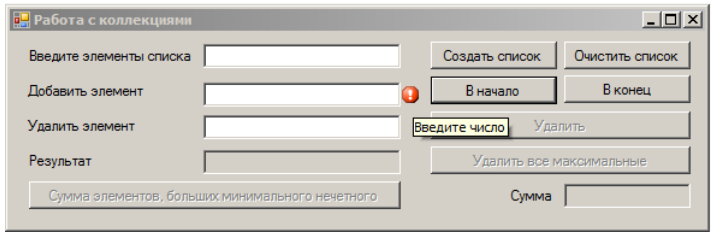

Рисунок 63 – Окно приложения «Работа с коллекциями»: добавление пустого элемента

При попытки добавления не целого числа в начало или конец возникает сообщение об ошибке (см. рисунок 64).

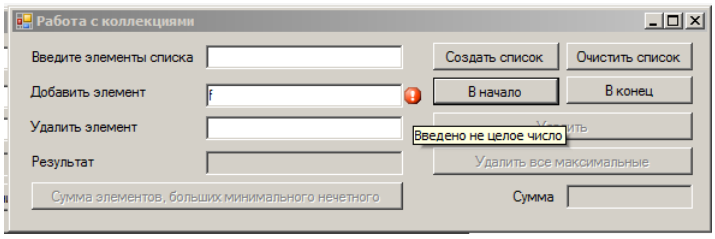

Рисунок 64 – Окно приложения «Работа с коллекциями»: добавление не целого числа

При попытки удаления элемента, значение которого не введено, возникает сообщение об ошибке (см. рисунок 65).

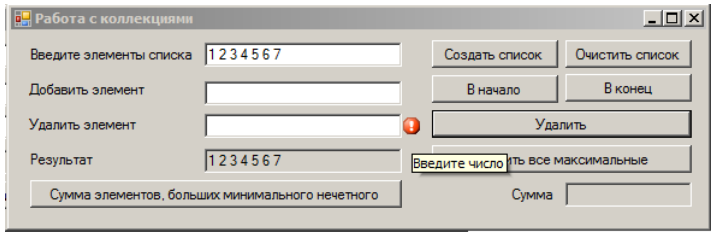

Рисунок 65 – Окно приложения «Работа с коллекциями»: удаление невведенного элемента

При попытки удаления элемента, значение которого не является целым числом, возникает сообщение об ошибке (см. рисунок 66).

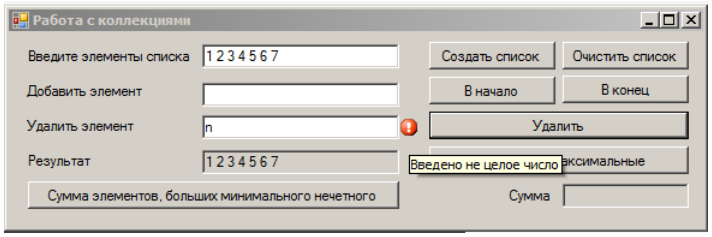

Рисунок 66 – Окно приложения «Работа с коллекциями»: удаление не целочисленного элемента

При попытки удаления несуществующего элемента возникает сообщение об ошибке (см. рисунок 67).

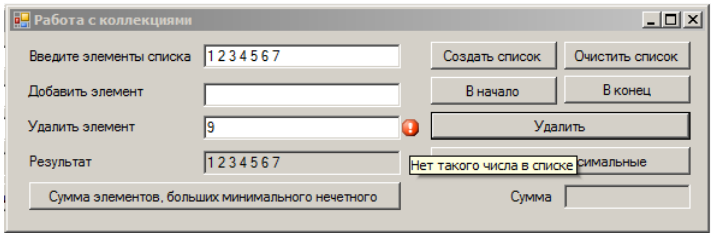

Рисунок 67 – Окно приложения «Работа с коллекциями»: удаление несуществующего элемента

Полный код программы приведен в приложении А.

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 Конструктор Windows Forms [Электронный ресурс]. — URL: https://msdn.microsoft.com/ru-ru/library/e06hs424.aspx (Дата обращения 13.07.2017). Загл. с экр. Яз. рус.

ПРИЛОЖЕНИЕ А

USB-накопитель с отчетом о выполненной работе

На приложенном USB-накопителе можно ознакомиться со следующими файлами:

Папка `report` — LaTeX- вариант отчета о практике;

Папка `factorial` — задание №1;

Папка `simpleCount` — задание №2, вариант №6;

Папка `recursion` — задание №3, вариант №13;

Папка `table1` — задание №4, вариант №7;

Папка `table2` — задание №5, вариант №9;

Папка `file` — задание №6, вариант №10;

Папка `collections` — задание №7, вариант №9;

`report.pdf` — отчет о практике.