Пермский филиал федерального государственного автономного образовательного учреждения высшего образования

«Национальный исследовательский университет

«Высшая школа экономики» Пермь»

Факультет экономики, менеджмента и бизнес-информатики.

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №8

студента образовательной программы «Программная инженерия»

по направлению подготовки *09.03.04 Программная инженерия*

Чепокова Елизара Сергеевича

Руководитель:

Кандидат

технических наук

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

О.Л. Викентьева

Пермь, 2018 год

# **1.Постановка задачи:**

Написать Windows-приложение для работы с простой базой данных, хранящей информацию об объекте на внешнем носителе. Приложение должно выполнять следующие функции:

1. Создание базы данных, содержащей записи указанного формата.
2. Просмотр базы данных.
3. Удаление элементов из базы данных (по ключу/ по номеру).
4. Корректировка элементов в базе данных (по ключу / по номеру).
5. Добавление элементов в базу данных (в начало / в конец/ с заданным номером).
6. Выполнение задания, указанного в варианте.

|  |  |
| --- | --- |
| Вариант | Задание |
| 23 | Типизированный файл содержит данные о студентах 1 курса (ФИО, группа, адрес, дата рождения, итоговые оценки по дисциплинам). Программа должна   * 1. Добавлять, удалять, корректировать, позволять просматривать записи файлов.   2. Выводить:      1. список студентов, не имеющих задолженностей.      2. средний балл за сессию заданного студента. |

# **2.Анализ задачи:**

Основные функции

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| № | Функция | Функция | Входные данные | Классы входных данных | Выходные данные | Классы выходных данных |
| 1 | Создание базы данных | Создание базы данных в программе | Строка с символами типа string | string | Однонаправленный список | База данных |
| Пустая строка | string с пустой строкой | Сообщение об ошибке | Неверный ввод |
| Tab  Пробел  Несколько пробелов | string с Tab, пробелами | Замена пробелов на 0 количество символов | База данных |
| Создание базы данных из файла | Строка с символами типа string | string | Однонаправленный список | База данных |
| Пустая строка | string с пустой строкой | Сообщение об ошибке | Неверный ввод |
| Tab  Пробел  Несколько пробелов | string с Tab, пробелами | Замена пробелов на 0 количество символов | База данных |
| 2 | Корректировка элементов в базе данных | | Строка с символами типа string | string | Однонаправленный список | База данных |
| Пустая строка | string с пустой строкой | Сообщение об ошибке | Неверный ввод |
| Tab  Пробел  Несколько пробелов | string с Tab, пробелами | Замена пробелов на 0 количество символов | База данных |
| 3 | Добавление элементов в базу данных | | Строка с символами типа string | string | Однонаправленный список | База данных |
| Пустая строка | string с пустой строкой | Сообщение об ошибке | Неверный ввод |
| Tab  Пробел  Несколько пробелов | string с Tab, пробелами | Замена пробелов на 0 количество символов | База данных |

# **3.Проектирование функций:**

|  |  |  |  |
| --- | --- | --- | --- |
| **№** | **Функция** | **Входные данные** | **Выходные данные** |
|  | **Сохранение** |  |  |
|  | private void открытьИзБинарногоФайлаToolStripMenuItem\_Click | dataGridView1 | \*.bin |
|  | private void сохранитьВMicrosoftExcelToolStripMenuItem\_Click | dataGridView1 | \*.XLSX;\*.XLS |
|  | private void сохранитьВТекстовыйДокументToolStripMenuItem\_Click | dataGridView1 | \*.txt |
|  | **Загрузка** |  |  |
|  | private void сохранитьВБинарныйФайлToolStripMenuItem\_Click | \*.bin | dataGridView1 |
|  | private void загрузитьИзMicrosoftExcelToolStripMenuItem\_Click | \*.XLSX;\*.XLS | dataGridView1 |
|  | private void загрузитьИзТекстовогоДокументаToolStripMenuItem\_Click | \*.txt | dataGridView1 |
|  | **Работа с базой данных** |  |  |
|  | private void успеваемостьToolStripMenuItem\_Click | dataGridView1  string point | BadStudents  dataGridView2 |
|  | private void режимРедактированияToolStripMenuItem\_Click | dataGridView1  выбранная позиция | dataGridView1 |
|  | private void удалитьВыбранные\_Click | dataGridView1  выбранная позиция | dataGridView1 |
|  | private void удалитьВсе\_Click | dataGridView1  выбранная позиция | dataGridView1 пустой |
|  | private void добавитьДисциплинуToolStripMenuItem\_Click | NewDiscipline form = new NewDiscipline | dataGridView1  NewDiscipline |
|  | private void enter\_Click | this.name.Text = pers.name; this.surname.Text = pers.surname; this.midname.Text = pers.midname; this.group.Text = pers.group; this.date.Value = pers.date; this.address.Text = pers.address; | dataGridView1 |

# **4.Тесты:**

## **Белый ящик:**

MainMenu (добавление студента)

|  |  |  |  |
| --- | --- | --- | --- |
| **Тест** | **Входные данные** | **Ожидаемый результат** | **Реальный результат** |
| 1 | Чепоков; Елизар; Сергеевич; ПИ-18-2; 28 декабря 1999; ул. Уличная; мат анализ программирование | Полная база данных | Полная база данных |
| 2 | Чепоков; Елизар; ПИ-18-2; 28 декабря 1999; мат анализ программирование | База данных без отчества | База данных без отчества |
| 3 | Чепоков; Елизар; Сергеевич; ПИ-18-2; 28 декабря 1999; ул. Уличная; | База данных без дисциплин и оценок | База данных без дисциплин и оценок |
| 4 | Сергеевич; ул. Уличная; мат анализ программирование | Сообщение об ошибке | Неверный ввод |
| 5 | пустота | Сообщение об ошибке | Неверный ввод |

MainMenu (редактирование)

|  |  |  |  |
| --- | --- | --- | --- |
| **Тест** | **Входные данные** | **Ожидаемый результат** | **Реальный результат** |
| 6 | Чепоков; Елизар; Сергеевич; ПИ-18-2; 28 декабря 1999; ул. Уличная; мат анализ программирование | Полная база данных | Полная база данных |
| 7 | Чепоков; Елизар; ПИ-18-2; 28 декабря 1999; мат анализ программирование | База данных без отчества | База данных без отчества |
| 8 | Чепоков; Елизар; Сергеевич; ПИ-18-2; 28 декабря 1999; ул. Уличная; | База данных без дисциплин и оценок | База данных без дисциплин и оценок |
| 9 | Сергеевич; 28 декабря 1999; ул. Уличная; мат анализ программирование | Сообщение об ошибке | Неверный ввод |

MainMenu (добавление студента после создания базы данных)

|  |  |  |  |
| --- | --- | --- | --- |
| **Тест** | **Входные данные** | **Ожидаемый результат** | **Реальный результат** |
| 10 | Чепоков; Елизар; Сергеевич; ПИ-18-2; 28 декабря 1999; ул. Уличная; мат анализ программирование | Полная база данных | Полная база данных |
| 11 | Чепоков; Елизар; ПИ-18-2; 28 декабря 1999; мат анализ программирование | База данных без отчества | База данных без отчества |
| 12 | Чепоков; Елизар; Сергеевич; ПИ-18-2; 28 декабря 1999; ул. Уличная; | База данных без дисциплин и оценок | База данных без дисциплин и оценок |
| 13 | Сергеевич; 28 декабря 1999; ул. Уличная; мат анализ программирование | Сообщение об ошибке | Неверный ввод |
| 14 | пустота | Сообщение об ошибке | Неверный ввод |

MainMenu (любое удаление)

|  |  |  |  |
| --- | --- | --- | --- |
| **Тест** | **Входные данные** | **Ожидаемый результат** | **Реальный результат** |
| 15 | Пустой dataGridView1 | Сообщение об ошибке | База данных пуста |
| 16 | dataGridView1 с данными  Удалить все | Пустая база данных | Пустой dataGridView1 |
| 17 | dataGridView1 с данными  Удалить с выбранной позиции | База данных баз удаленных строк | dataGridView1 баз удаленных строк |

Открытие файла (любым из способов)

|  |  |  |  |
| --- | --- | --- | --- |
| **Тест** | **Входные данные** | **Ожидаемый результат** | **Реальный результат** |
| 1 | Пустой документ | Пустая база данных | Пустой dataGridView1 |
| 2 | Закрытие сразу после открытия | Сообщение об ошибке | Вы ничего не загрузили |
| 3 | Повторное открытие удаления, после первого открытия | Сообщение об ошибке | Окно уже открыто |
| 4 | dataGridView1 с данными | База данных с данными из документа | dataGridView1 с данными из документа |

Сохранение файла (любым из способов)

|  |  |  |  |
| --- | --- | --- | --- |
| **Тест** | **Входные данные** | **Ожидаемый результат** | **Реальный результат** |
| 1 | Пустой dataGridView1 | Сообщение об ошибке | Нечего сохранять |
| 2 | Закрытие сразу после открытия | Сообщение об ошибке | Вы не сохранили |
| 3 | Повторное открытие удаления, после первого открытия | Сообщение об ошибке | Окно уже открыто |
| 4 | dataGridView1 с данными | Документ с данными из dataGridView1 | Документ с данными из dataGridView1 |

Добавление дисциплины (NewDiscipline)

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Тест** | **Входные данные** | | **Ожидаемый результат** | | **Реальный результат** | |
| 1 | Пустая строка | | Сообщение об ошибке | | Вы ничего не ввели | |
| 2 | Табуляция | | Сообщение об ошибке | | Неверный ввод | |
| 3 | Закрытие после открытия | | Сообщение об ошибке | | Вы ничего не ввели | |
| 4 | Повторное открытие, после первого открытия | | Документ с данными из dataGridView1 | | Окно уже открыто | |
| 5 | Наименование дисциплины | | Новая дисциплина | | Новая дисциплина | |
| 6 | Цифры | Сообщение об ошибке | | Неверный ввод | |

Вывод учеников с долгами

|  |  |  |  |
| --- | --- | --- | --- |
| **Тест** | **Входные данные** | **Ожидаемый результат** | **Реальный результат** |
| 1 | Пустой dataGridView1 | Сообщение об ошибке | Никого нет |
| 2 | Повторное открытие, после первого открытия | Сообщение об ошибке | Окно уже открыто |
| 3 | dataGridView1 с данными  Есть учащиеся с оценками < 4 | dataGridView2 с учащимися которые имеют хотя бы один долг | dataGridView2 с учащимися которые имеют хотя бы один долг |
| 4 | dataGridView1 с данными  Нет учащихся с оценками < 4 | Сообщение об ошибке | Никого нет |

## **Черный ящик:**

MainMenu:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Т1 | Т2 | Т3 | Т4 | Т5 | Т6 | Т7 | Т8 | Т9 | Т10 | Т11 | Т12 | Т13 | Т14 | Т15 | Т16 | Т17 |
| Входные данные | | | | | | | | | | | | | | | | | |
| Не введено имя |  |  |  | + | + |  |  |  | + |  |  |  | + | + |  |  |  |
| Не введена фамилия |  |  |  | + | + |  |  |  | + |  |  |  | + | + |  |  |  |
| Не введено отчество |  | + |  |  | + |  | + |  | + |  | + |  |  | + |  |  |  |
| Не выбрана группа |  |  |  | + | + |  |  |  | + |  |  |  | + | + |  |  |  |
| Не выбрана дата рождения |  |  |  | + | + |  |  |  | + |  |  |  | + | + |  |  |  |
| Не введен адрес |  |  |  |  | + |  |  |  |  |  |  |  |  | + |  |  |  |
| Нет дисциплин |  |  | + |  | + |  |  | + |  |  |  | + |  | + |  |  |  |
| Введено имя | + | + | + |  |  | + | + | + |  | + | + | + |  |  |  |  |  |
| Введена фамилия | + | + | + |  |  | + | + | + |  | + | + | + |  |  |  |  |  |
| Введено отчество | + |  | + |  |  | + |  | + |  | + |  | + |  |  |  |  |  |
| Выбрана группа | + | + | + |  |  | + | + | + |  | + | + | + |  |  |  |  |  |
| Выбрана дата рождения | + | + | + |  |  | + | + | + |  | + | + | + |  |  |  |  |  |
| Введен адрес | + | + | + |  |  | + | + | + |  | + | + | + |  |  |  |  |  |
| Есть дисциплины | + | + |  | + |  | + | + |  |  | + | + |  | + |  |  |  |  |
| Пустота |  |  |  |  | + |  |  |  |  |  |  |  |  | + |  |  |  |
| Пустая база данных |  |  |  |  | + |  |  |  |  |  |  |  |  | + | + |  |  |
| Заполненная база данных |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | + | + |
| Удалить все |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | + |  |
| Удалить выделенное |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | + |
| Выходные данные | | | | | | | | | | | | | | | | | |
| Сообщение об ошибке |  |  |  | + | + |  |  |  | + |  |  |  | + | + | + |  |  |
| Полная база данных | + |  |  |  |  | + |  |  |  | + |  |  |  |  |  |  |  |
| Не полная база данных |  | + | + |  |  |  | + | + |  |  | + | + |  |  |  |  | + |
| Пустая база данных |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | + |  |

Открытие файла:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Т1 | Т2 | Т3 | Т4 |
| Входные данные | | | | |
| Пустой документ | + |  |  |  |
| Заполненный документ |  |  |  | + |
| Повторное открытие |  |  | + |  |
| Закрытие сразу после открытия |  | + |  |  |
| Выходные данные | | | | |
| Пустая база данных | + |  |  |  |
| Заполненная база данных |  |  |  | + |
| Сообщение об ошибке |  | + | + |  |

Сохранение файла:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Т1 | Т2 | Т3 | Т4 |
| Входные данные | | | | |
| Пустая база данных | + |  |  |  |
| Заполненная база данных |  |  |  | + |
| Повторное открытие |  |  | + |  |
| Закрытие сразу после открытия |  | + |  |  |
| Выходные данные | | | | |
| Пустая база данных | + |  |  |  |
| Заполненная база данных |  |  |  | + |
| Сообщение об ошибке | + | + | + |  |

Добавление дисциплины:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | Т1 | Т2 | Т3 | Т4 | Т5 | Т6 |
| Входные данные | | | | | | |
| Пустая строка | + |  |  |  |  |  |
| Табуляция |  | + |  |  |  |  |
| Символы |  |  |  |  | + |  |
| Цифры |  |  |  |  |  | + |
| Закрытие сразу после открытия |  |  | + |  |  |  |
| Повторное открытие |  |  |  | + |  |  |
| Выходные данные | | | | | | |
| Название дисциплины |  |  |  |  | + |  |
| Сообщение об ошибке | + | + | + | + |  | + |

Вывод учеников с долгами:

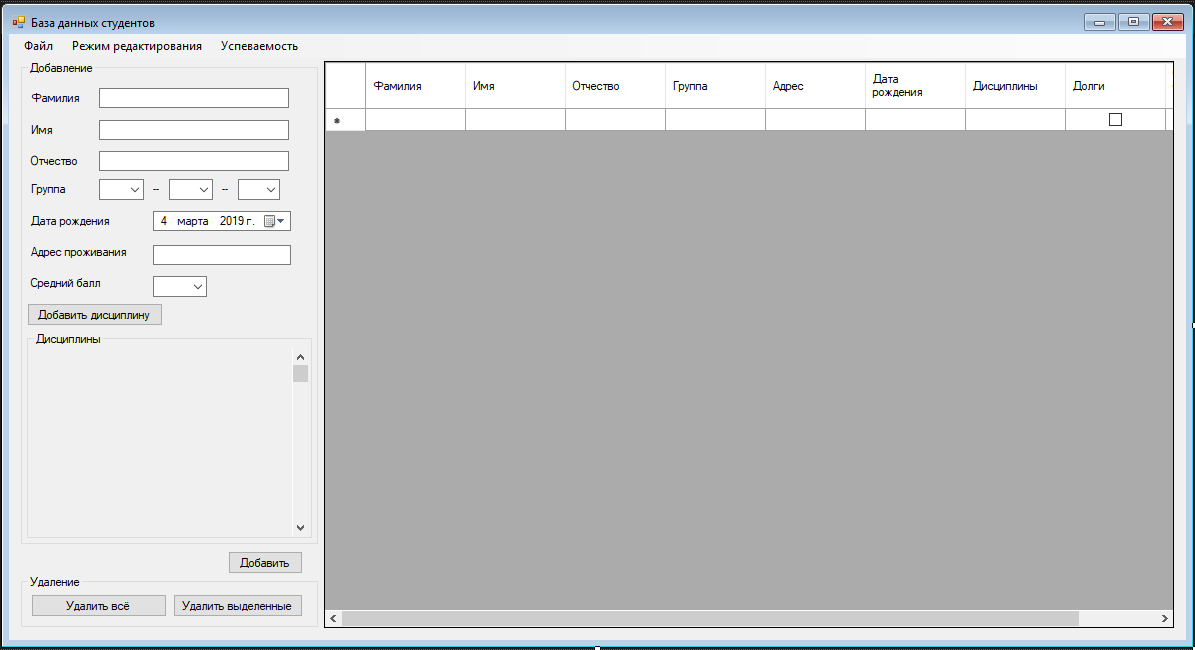
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Т1 | Т2 | Т3 | Т4 |
| Входные данные | | | | |
| Пустая база данных | + |  |  |  |
| Заполненная база данных |  |  | + | + |
| Повторное открытие |  | + |  |  |
| Есть оценки < 4 |  |  | + |  |
| Нет оценок < 4 |  |  |  | + |
| Выходные данные | | | | |
| Пустая база данных | + |  | + |  |
| Заполненная база данных |  |  |  | + |
| Сообщение об ошибке |  | + |  |  |

# **5.Листинг:**

public static string text = "";

public static int count\_courses;

public const int start\_left = 12, start\_text\_left = start\_left + 150;

 public string course = "";

public MainMenu()

{

InitializeComponent();

table.AllowUserToAddRows = false;

table.DefaultCellStyle.WrapMode = DataGridViewTriState.True;

table.AutoSizeRowsMode = DataGridViewAutoSizeRowsMode.AllCells;

MainMenu\_AnyResize(new Object(), null);

WritingInGrid();

enter.Enabled = false;

string[] lessons = Core.AllLessons();

count\_courses = lessons.Length;

date.Value = new DateTime(2000, 01, 01);

Label label = new Label

{

Location = new Point(start\_text\_left, label7.Top),

Name = "label\_mark",

Text = "Оценки"

};

this.Controls.Add(label);

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

{

int start\_top = label7.Top + (label7.Height + 10) + i \* (17 + 5);

CheckBox checkBox = new CheckBox

{

Location = new Point(start\_left, start\_top),

Name = "checkBox" + (i + 1),

Size = new Size(80, 17),

Text = lessons[i],

AutoSize = true

};

checkBox.CheckedChanged += new EventHandler(lessonCheckBox\_CheckedChanged);

TextBox textBox = new TextBox

{

Location = new Point(start\_text\_left, start\_top),

Name = "mark" + (i + 1),

Enabled = false

};

textBox.TextChanged += new EventHandler(this.CheckNum);

this.Controls.Add(textBox);

this.Controls.Add(checkBox);

}

if (Core.student != null)

{

Person pers = Core.student;

this.Text = "Изменение информации о студенте";

this.name.Text = pers.name;

this.surname.Text = pers.surname;

this.midname.Text = pers.midname;

this.group.Text = pers.group;

this.date.Value = pers.date;

this.address.Text = pers.address;

enter.Enabled = true;

Mark help = pers.marks;

while (help != null)

{

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

{

if ((this.Controls["checkBox" + (i + 1)] as CheckBox).Text == help.name)

{

CheckBox checkbox = this.Controls["checkBox" + (i + 1)] as CheckBox;

checkbox.Checked = true;

TextBox textbox = this.Controls["mark" + (i + 1)] as TextBox;

textbox.Text = Convert.ToString(help.mark);

}

}

help = help.next;

}

}

}

private void button1\_Click(object sender, EventArgs e)

{

string surname = this.surname.Text;

string name = this.name.Text;

string middle\_name = midname.Text;

string group = group1.Text + "-" + group2.Text + "-" + group3.Text;

string adress = address.Text;

string date = this.date.Text;

string point = comboBox4.Text;

bool dolg = false;

dataGridView1.Rows.Add(surname, name, middle\_name, group, adress, date, dolg, point);

}

private void dataGridView1\_RowPrePaint\_1(object sender, DataGridViewRowPrePaintEventArgs e)

{

int index = e.RowIndex;

string indexStr = (index + 1).ToString();

object header = this.dataGridView1.Rows[index].HeaderCell.Value;

if (header == null || !header.Equals(indexStr))

this.dataGridView1.Rows[index].HeaderCell.Value = indexStr;

}

//загрузка и схранение

#region

private void загрузитьИзТекстовогоДокументаToolStripMenuItem\_Click(object sender, EventArgs e)

{

Stream mystr = null;

openFileDialog1.Filter = "Текстовый документ (\*.txt)|\*.txt";

openFileDialog1.Title = "Укажите директорию и имя файла";

openFileDialog1.FilterIndex = 2;

openFileDialog1.RestoreDirectory = true;

if (openFileDialog1.ShowDialog() == DialogResult.OK)

{

if ((mystr = openFileDialog1.OpenFile()) != null)

{

StreamReader readtxt = new StreamReader(mystr);

string[] str;

int num = 0;

try

{

string[] str1 = readtxt.ReadToEnd().Split('\n');

num = str1.Count();

dataGridView1.RowCount = num;

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

{

str = str1[i].Split('$');

for (int j = 0; j < dataGridView1.ColumnCount; j++)

{

try

{

string data = str[j].Replace("[symbol]", "$");

dataGridView1.Rows[i].Cells[j].Value = data;

}

catch { }

}

}

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

}

finally

{

readtxt.Close();

}

}

}

}

private void сохранитьВТекстовыйДокументToolStripMenuItem\_Click(object sender, EventArgs e)

{

Stream myStr;

SaveFileDialog saveFileDialog1 = new SaveFileDialog()

{

Filter = "Текстовый документ (\*.txt)|\*.txt",

DefaultExt = "\*.txt",

Title = "Укажите директорию и имя файла для сохранения"

};

if (saveFileDialog1.ShowDialog() == DialogResult.OK)

{

if ((myStr = saveFileDialog1.OpenFile()) != null)

{

StreamWriter writetxt = new StreamWriter(myStr);

try

{

for (int i = 0; i < dataGridView1.RowCount - 1; i++)

{

for (int j = 0; j < dataGridView1.ColumnCount; j++)

{

string data = dataGridView1.Rows[i].Cells[j].Value.ToString().Replace("$", "[simbol]");

writetxt.Write(data + "$");

}

writetxt.WriteLine();

}

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

}

finally

{

writetxt.Close();

}

}

}

}

private void загрузитьИзMicrosoftExcelToolStripMenuItem\_Click(object sender, EventArgs e)

{

OpenFileDialog openDialog = new OpenFileDialog();

openDialog.Filter = "Файл Excel|\*.XLSX;\*.XLS";

openDialog.ShowDialog();

try

{

Excel.Application ObjExcel = new Excel.Application();

Excel.\_Workbook ObjWorkBook = ObjExcel.Workbooks.Open(openDialog.FileName);

Excel.Worksheet ObjWorkSheet = ObjExcel.ActiveSheet as Excel.Worksheet;

Excel.Range rg = null;

Int32 row = 1;

dataGridView1.Rows.Clear();

List<String> arr = new List<string>();

while (ObjWorkSheet.get\_Range("a" + row, "a" + row).Value != null)

{

rg = ObjWorkSheet.get\_Range("a" + row, "u" + row);

foreach (Excel.Range item in rg)

{

try

{

arr.Add(item.Value.ToString().Trim());

}

catch { arr.Add(""); }

}

dataGridView1.Rows.Add(arr[0], arr[1], arr[2], arr[3], arr[4], arr[5], arr[6], arr[7], arr[8]);

arr.Clear();

row++;

}

MessageBox.Show("Файл успешно считан!", "Считывание файла", MessageBoxButtons.OK, MessageBoxIcon.Information);

}

catch (Exception ex)

{

MessageBox.Show("Ошибка: " + ex.Message, "Ошибка при считывании excel файла", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

private void releaseObject(object obj)

{

try

{

System.Runtime.InteropServices.Marshal.ReleaseComObject(obj);

obj = null;

}

catch (Exception ex)

{

obj = null;

MessageBox.Show("Unable to release the object " + ex.ToString());

}

finally

{

GC.Collect();

}

}

private void сохранитьВMicrosoftExcelToolStripMenuItem\_Click(object sender, EventArgs e)

{

Excel.Application excelapp = new Excel.Application();

Excel.Workbook workbook = excelapp.Workbooks.Add();

Excel.Worksheet worksheet = workbook.ActiveSheet;

for (int i = 1; i < dataGridView1.RowCount + 1; i++)

{

for (int j = 1; j < dataGridView1.ColumnCount + 1; j++)

{

worksheet.Rows[i].Columns[j] = dataGridView1.Rows[i - 1].Cells[j - 1].Value;

}

}

SaveFileDialog sfd = new SaveFileDialog()

{

Filter = "MS Excel dosuments (\*.xlsx)|\*.xlsx",

DefaultExt = "\*.xlsx",

Title = "Укажите директорию и имя файла для сохранения"

};

if (sfd.ShowDialog() == DialogResult.OK)

{

workbook.SaveAs(sfd.FileName);

}

else

{

excelapp.Quit();

}

excelapp.Quit();

}

#endregion

//

private void button1\_Click\_1(object sender, EventArgs e)

{

NewDiscipline form = new NewDiscipline

{

Owner = this

};

form.ShowDialog();

int start\_top = label7.Top + (label7.Height + 10) + count\_courses \* (17 + 5);

CheckBox checkBox = new CheckBox

{

Location = new Point(start\_left, start\_top),

Name = "checkBox" + ++count\_courses,

Size = new Size(80, 17),

Text = course,

AutoSize = true

};

checkBox.CheckedChanged += new EventHandler(lessonCheckBox\_CheckedChanged);

TextBox textBox = new TextBox

{

Location = new Point(start\_text\_left, start\_top),

Name = "mark" + count\_courses,

Enabled = false

};

textBox.TextChanged += new EventHandler(this.CheckNum);

this.Controls.Add(textBox);

this.Controls.Add(checkBox);

}

private void режимРедактированияToolStripMenuItem\_Click(object sender, EventArgs e)

{

ToolStripMenuItem btn = (ToolStripMenuItem)sender;

if (btn.Text == "Перейти в режим редактирования")

{

this.table.ReadOnly = false;

this.id.ReadOnly = true;

btn.Text = "Выйти из режима редактирования";

}

else

{

this.table.ReadOnly = true;

this.id.ReadOnly = true;

btn.Text = "Перейти в режим редактирования";

}

}

public NewDiscipline()

{

InitializeComponent();

if (Core.new\_file)

{

label1.Text = "Введите название создаваемого файла (без расширения)";

this.Text = "Создание нового файла";

}

}

private void button1\_Click(object sender, EventArgs e)

{

NewDiscipline form = (NewDiscipline)Owner;

if (Core.new\_file)

{

MainMenu.text = textBox1.Text;

}

else

{

form.course = textBox1.Text;

}

this.DialogResult = DialogResult.OK;

this.Dispose();

}

private void button2\_Click(object sender, EventArgs e)

{

this.DialogResult = DialogResult.Cancel;

this.Dispose();

}

private void вКонецToolStripMenuItem\_Click(object sender, EventArgs e)

{

Core.core\_id = -1;

AddStud f = new AddStud

{

Owner = this

};

f.ShowDialog();

}

public void WritingInGrid()

{

bool ok = false;

Person[] pers = new Person[0];

try { pers = Core.ReadAllStudents(Core.core\_file); ok = true; }

catch { MessageBox.Show("В файле сохранены данные другого типа", "Ошибка открытия файла", MessageBoxButtons.OK); }

if (ok)

{

while (table.Rows.Count != 0)

{

table.Rows.Remove(table.Rows[table.Rows.Count - 1]);

}

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

{

if (pers[i] != null && !pers[i].delete)

table.Rows.Add(pers[i].id, pers[i].surname, pers[i].name, pers[i].midname, pers[i].group, pers[i].date.ToString("dd.MM.yyyy"), pers[i].address, pers[i].marks.ToString());

}

}

}

private void удалитьToolStripMenuItem\_Click(object sender, EventArgs e)

{

Core.DeleteDataBase(Core.core\_file);

while (table.Rows.Count != 0)

{

table.Rows.Remove(table.Rows[table.Rows.Count - 1]);

}

}

private void MainMenu\_AnyResize(object sender, EventArgs e)

{

this.table.Width = this.Width - table.Left \* 3;

this.table.Height = this.Height - table.Top \* 3;

int width = (this.table.Width - this.id.Width - 40) / 7;

this.surname.Width = width;

this.name.Width = width;

this.midname.Width = width;

this.group.Width = width;

this.date.Width = width;

this.address.Width = width;

this.marks.Width = width;

}

private void выбранногоToolStripMenuItem\_Click(object sender, EventArgs e)

{

if (table.SelectedCells.Count != 0)

{

int selRowNum = table.SelectedCells[0].RowIndex;

Core.DeleteStudent(Convert.ToInt32(table.Rows[selRowNum].Cells[0].Value.ToString()));

table.Rows.Remove(table.Rows[selRowNum]);

if (Core.DeletedMoreThanHalfCount()) Core.CleanFile();

}

}

private void поIDToolStripMenuItem\_Click(object sender, EventArgs e)

{

InputID form = new InputID();

form.ShowDialog();

if (form.DialogResult == DialogResult.OK) Core.DeleteStudent(Core.core\_id);

WritingInGrid();

}

private void включитьИзмененияToolStripMenuItem\_Click(object sender, EventArgs e)

{

ToolStripMenuItem btn = (ToolStripMenuItem)sender;

if (btn.Text == "Перейти в режим редактирования")

{

this.table.ReadOnly = false;

this.id.ReadOnly = true;

btn.Text = "Выйти из режима редактирования";

}

else

{

this.table.ReadOnly = true;

this.id.ReadOnly = true;

btn.Text = "Перейти в режим редактирования";

}

}

private void вУказаннуюПозициюToolStripMenuItem\_Click(object sender, EventArgs e)

{

InputNum form = new InputNum

{

Owner = this

};

form.ShowDialog();

if (form.DialogResult == DialogResult.OK)

{

AddStud f = new AddStud

{

Owner = this

};

f.ShowDialog();

}

}

private void table\_CellBeginEdit(object sender, DataGridViewCellCancelEventArgs e)

{

int selRowNum = table.SelectedCells[0].RowIndex;

int id = Convert.ToInt32(table.Rows[selRowNum].Cells[0].Value.ToString());

Core.student = Core.FindID(id);

Core.core\_id = Core.NumInFile(id);

AddStud f = new AddStud

{

Owner = this

};

f.Show();

}

private void студентыСЗадолженностьюToolStripMenuItem\_Click(object sender, EventArgs e)

{

BadStud form = new BadStud();

form.ShowDialog();

}

private void дисциплиныСЗадолженностьToolStripMenuItem\_Click(object sender, EventArgs e)

{

MessageBox.Show(Core.BadLessons(), "Дисциплины с задолженностью", MessageBoxButtons.OK);

}

private void открытьToolStripMenuItem\_Click(object sender, EventArgs e)

{

OpenFileDialog OPF = new OpenFileDialog();

OPF.Filter = "Файлы bin|\*.bin";

if (OPF.ShowDialog() == DialogResult.OK)

{

foreach (string file in OPF.FileNames)

{

bool ok = false;

try { System.IO.FileStream f = new System.IO.FileStream(file, System.IO.FileMode.Open, System.IO.FileAccess.ReadWrite); ; ok = true; f.Close(); }

catch (UnauthorizedAccessException){ MessageBox.Show("Отказано в доступе", "Ошибка доступа", MessageBoxButtons.OK); }

catch { MessageBox.Show("Неизвестная ошибка", "Ошибка", MessageBoxButtons.OK); }

if (ok)

{

Core.core\_file = file;

WritingInGrid();

}

}

}

}

private void создатьToolStripMenuItem\_Click(object sender, EventArgs e)

{

FolderBrowserDialog FBD = new FolderBrowserDialog();

FBD.ShowNewFolderButton = false;

if (FBD.ShowDialog() == DialogResult.OK)

{

string str = FBD.SelectedPath;

Core.new\_file = true;

AddCourse form = new AddCourse();

DialogResult dr = form.ShowDialog();

if (dr == DialogResult.OK)

{

str += @"\" + MainMenu.text + ".bin";

Core.core\_file = str;

WritingInGrid();

}

}

}

private void MainMenu\_FormClosed(object sender, FormClosedEventArgs e)

{

Core.CleanFile();

}

public BadStudents()

{

InitializeComponent();

table.DefaultCellStyle.WrapMode = DataGridViewTriState.True;

table.AutoSizeRowsMode = DataGridViewAutoSizeRowsMode.AllCells;

BadStud\_AnyResize(new object(), null);

WritingInGrid();

}

private void BadStud\_AnyResize(object sender, EventArgs e)

{

this.table.Width = this.Width - table.Left \* 3;

this.table.Height = this.Height - table.Top \* 3;

int width = (this.table.Width - this.id.Width - 40) / 7;

this.surname.Width = width;

this.name.Width = width;

this.midname.Width = width;

this.group.Width = width;

this.date.Width = width;

this.address.Width = width;

this.marks.Width = width;

}

public void WritingInGrid()

{

while (table.Rows.Count != 0)

{

table.Rows.Remove(table.Rows[table.Rows.Count - 1]);

}

Person[] pers = Core.ReadBadStudents();

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

{

if (pers[i] != null && !pers[i].delete)

table.Rows.Add(pers[i].id, pers[i].surname, pers[i].name, pers[i].midname, pers[i].group, pers[i].date.ToString("dd.MM.yyyy"), pers[i].address, pers[i].marks.ToString());

}

}