Министерство транспорта Российской Федерации

Федеральное государственное автономное образовательное

учреждение высшего образования

«Российский университет транспорта»

(ФГАОУ ВО РУТ(МИИТ), РУТ (МИИТ)

Институт транспортной техники и систем управления

Кафедра «Управление и защита информации»

Курсовая работа

по дисциплине: «Программирование и основы алгоритмизации»

на тему: «Электронный документооборот»

Выполнил: ст. гр. ТУУ-211

Голюдбин А.В.

Вариант №3

26.02.2025

(дата выполнения)

Проверил: к.т.н., доц. Сафронов А.И.

Москва – 2024 г.

# **Содержание**

[Содержание 2](#_Toc187498415)

[1. Введение 3](#_Toc187498416)

[2. Цель работы 4](#_Toc187498417)

[3. Задачи работы 4](#_Toc187498418)

[4. Содержательная часть работы 5](#_Toc187498419)

[4.1. Анализ печатной формы 5](#_Toc187498420)

[4.2. Описание предназначение печатной формы 6](#_Toc187498421)

[4.3. Разнообразие печатных форм и выявление отличий 6](#_Toc187498422)

[4.4. Унификация печатных форм для удобства обработки данных 6](#_Toc187498423)

[4.5. Разработка экранной формы для импорта сведений из печатной формы 7](#_Toc187498424)

[4.6. Разработка функций экспорта сведений из печатной формы в файл текстовый формата *CSV* 8](#_Toc187498425)

[4.6.1 Листинг 8](#_Toc187498426)

[4.6.2. Сеть Петри запрограммированного технологического процесса 11](#_Toc187498427)

[4.6.3. Блок-схема алгоритма 12](#_Toc187498428)

[4.7. Разработка функций импорта сведений из оригинального и изменённого текстовых файлов формата *CSV* с воссозданием *Word* 13](#_Toc187498429)

[4.7.1. Листинг 13](#_Toc187498430)

[4.7.2. Сеть Петри запрограммированного технологического процесса 18](#_Toc187498431)

[4.7.3. Блок-схема алгоритма 19](#_Toc187498432)

[4.8. Разработка функций экспорта сведений из экранной формы в *Excel* 22](#_Toc187498433)

[4.8.1. Листинг 22](#_Toc187498434)

[4.8.2. Сеть Петри запрограммированного технологического процесса 26](#_Toc187498435)

[4.8.3. Блок-схема алгоритма 27](#_Toc187498436)

[4.9. Графический анализ данных, содержащихся в печатной форме 32](#_Toc187498437)

[4.9.1. Пример работы графического анализа данных, содержащихся в печатных формах 34](#_Toc187498438)

[5. Инструкция пользователя 38](#_Toc187498439)

[6. Заключение 42](#_Toc187498440)

[7. Выводы 43](#_Toc187498441)

[Библиографический список 44](#_Toc187498442)

1. Введение

Электронный документооборот (ЭДО) представляет собой комплексную систему, обеспечивающую полный цикл работы с документами в цифровом формате — от создания и редактирования до хранения и анализа. Электронный документооборот удобен: не нужно распечатывать документы, передавать их на подпись и отправлять по почте. Всё можно сделать в программе за 10 минут[1]. Электронные документы — цифровые версии документов, которые создаются, обрабатываются, отправляются и хранятся с помощью компьютеров или других электронных устройств[2].

Преимущества электронного документооборота[3]:

* Ускорение бизнес-процессов. Контрагенты могут быстро отправлять и принимать документы, таким образом ускоряя поставки, выполнение работ, приёмку товара и др.
* Сокращение затрат. Применение электронных документов позволяет сэкономить на печати, доставке и хранении
* Повышение контроля. Весь процесс легко проследить в личном кабинете сервиса ЭДО: проанализировать статусы, сроки, ошибки

Специализированное программное обеспечение, разработанное в рамках данной работы, обеспечивает преобразование данных печатных форм в удобные для анализа и обработки форматы (CSV, DOCX, XLS). Это упрощает совместную работу с документами, а инструменты визуализации позволяют оперативно оценивать и анализировать ключевые показатели.

# **2. Цель работы**

В среде *Microsoft Visual Studio* на языке *C#* разработать программное решение с графическим пользовательским интерфейсом, выполняющее функцию электронного документооборота. Проект подразумевает создание удобной экранной формы, реализацию экспорта данных из печатных форм в *CSV*-файлы, а также разработку механизма импорта из *CSV* с автоматическим восстановлением информации в документе *Word*. Дополнительно необходимо обеспечить экспорт данных из разработанного интерфейса в *Excel* и возможность графического анализа полученных сведений. В рамках работы предполагается освоить навыки программного взаимодействия с *Word* и *Excel*, а также методы визуального анализа данных.

# **3. Задачи работы**

1. Проанализировать печатные формы

2. Уметь переносить содержимое печатных форм в CSV

3. Уметь воссоздавать структуру печатной формы

4. Уметь читать печатную форму и представлять информацию из неё на экранной форме

5. Уметь конвертировать печатную форму Word - Excel

6. Уметь анализировать данные, получаемые из нескольких печатных форм за различные периоды (и представлять результаты анализа в Chart-элементе)

7. Технологический процесс работы приложения отразить на карте, выполненной в нотации сетей Петри, в сочетании с классическими схемами алгоритма для этапов, отмеченных как "эффекты"

Индивидуальное задание:

1. Показывать на столбчатой диаграмме количество дней, проведённых на больничном преподавателями за весь известный период.

2. Показывать на столбчатой диаграмме количество дней, проведённых на больничном преподавателями за указанный год.

3. Показывать на столбчатой диаграмме общее количество часов, заменённых преподавателями по болезни.

4. Показывать на столбчатой диаграмме количество часов практических занятий, заменённых преподавателями по болезни.

5. Показывать на столбчатой диаграмме количество часов лабораторных работ, заменённых преподавателями по болезни.

6. Показывать на столбчатой диаграмме количество часов лекций, заменённых преподавателями по болезни.

7. Показывать на столбчатой диаграмме общее количество дней, в которые замещающие преподаватели осуществляли замены за весь известный период.

8. Показывать на столбчатой диаграмме общее количество дней, в которые замещающие преподаватели осуществляли замены за указанный год.

4. Содержательная часть работы

4.1. Анализ печатной формы

Каждая печатная форма представляет собой *Word*-документ, который содержит в себе рапорт о заменах преподавателей по болезни. В рапорте содержится информация о преподавателе, название кафедры на которой работает преподаватель, даты болезни преподавателя, подтверждающие документы (номер листка нетрудоспособности), а также информация о произведенных заменах: кто, в какие дни, какие занятия проводил. Все эти данные изменяются в зависимости от документа. Также, все печатные формы содержат информацию о том, на чье имя написан рапорт, а также об имени и должности написавшего рапорт.

4.2. Описание предназначение печатной формы

Каждая печатная форма – рапорт о заменах по болезни преподавателя. Информация, содержащаяся в печатной форме, нужна для подсчёта данных о заменах для расчета заработной платы заменяющим преподавателям и пропустившему.

4.3. Разнообразие печатных форм и выявление отличий

Печатные формы могут различаться рассматриваемым преподавателем, рассматриваемым временным периодом, наименованием дисциплин, групп, видов занятий, количеством часов, затраченных на определённую дисциплину и вид занятия, количеством и личностью заменяющих, ответственными лицами, от кого и на чье имя написан рапорт.

4.4. Унификация печатных форм для удобства обработки данных

Была произведена унификация печатных форм, заключающаяся в выделении ключевых слов (символов), которые окружают данные на форме (например, символы кавычек для названия кафедры).

4.5. Разработка экранной формы для импорта сведений из печатной формы

На Рисунке 1 показана главная экранная форма для импорта сведений из печатаной формы:



Рисунок 1 – Экранная форма для импорта сведений из печатной формы

Экранная форма для импорта сведений из печатной формы содержит 2 кнопки в левой части формы, предназначенные для конвертации данных в разные типы: по нажатию на кнопку «Импортировать документ»открывается диалоговое окно с выбором печатной формы для импорта и конвертации в CSV-файл и в Excel-документ, а по нажатию на кнопку «Восстановить Word из CSV» открывается диалоговое окно выбора CSV-файла для восстановления печатной формы из CSV-файла. Для наглядности, над элементом *Chart*, предназначенным для вывода нужных диаграмм, располагается элемент *Label* «Графический анализ данных», а под ним располагается необходимый элемент *ComboBox*, предназначенный для выбора варианта построения диаграмм. Справа от элемента ComboBox находится элемент NumericUpDown для ввода года при необходимости.

4.6. Разработка функций экспорта сведений из печатной формы в файл текстовый формата *CSV*

4.6.1 Листинг

private void ConvertToCSV(string filePath)

{

if (!File.Exists(filePath))

{

MessageBox.Show("Выберите файл!");

return;

}

string pathToCSV = Path.GetDirectoryName(filePath) + "\\" + Path.GetFileNameWithoutExtension(filePath) + ".csv";

Word.Application oWord = new Word.Application();

Word.Document oDoc = null;

oDoc = oWord.Documents.Open(filePath, ReadOnly: true);

using (StreamWriter writer = new StreamWriter(pathToCSV, false, Encoding.UTF8))

{

//Создаем оглавление CSV

string[] headers = { "\"Кому", "По болезни", "Период", "Кафедра", "Подпись", "Б/л", "Заменяющий", "Дата", "Занятие", "Группа\"" };

writer.WriteLine(string.Join("\",\t\"", headers));

//Записываем весь файл в строку

StringBuilder sb = new StringBuilder();

foreach (Word.Paragraph p in oDoc.Paragraphs)

if (!string.IsNullOrEmpty(p.Range.Text.Trim()))

sb.AppendLine(p.Range.Text.Trim());

string text = sb.ToString();

//Заполнение информации, содержащейся в одной строке

writer.Write("\"" + text.Substring(0, text.ToLower().IndexOf("рапорт")).Replace(Environment.NewLine, " ").Trim() + "\",\t");

writer.Write("\"" + Regex.Replace(GetTextPart(text, "В связи с болезнью ", " с "), " в период", "") + "\",\t");

writer.Write("\"" + GetTextPart(GetTextPart(text, "В связи с болезнью ", "(") + "(", " с ", " (") + "\",\t");

writer.Write("\"" + GetTextPart(text, "«", "»") + "\",\t");

string signature = text.Split('\n').Last(s => !string.IsNullOrWhiteSpace(s) && s.Trim() != "\a").Trim('\a', '\r').Trim();

writer.Write("\"" + signature + "\",\t");

//Сохранение больничных листов

List<long> b\_lists = GetTextPart(text, "№", ")").Split(',').Select(s => long.Parse(Regex.Replace(s.Replace("№", ""), @"\s", "").Split('о')[0])).ToList();

writer.Write(b\_lists[0] + ",\t");

//Сохранение данных о заменах

List<Substitution> substitutions = new List<Substitution>();

string[] substitutionsText = text.Substring(text.IndexOf("зки:") + 4).Split('\n');

string person = "";

foreach (string s in substitutionsText)

{

if (s.Length < 2)

continue;

if (char.IsDigit(s[1]))

{

DateTime date;

if (!DateTime.TryParse(s.Substring(0, 8), out date))

{

MessageBox.Show("Некорректная дата (" + s.Substring(0, 8) + ") пропущена");

continue;

}

string type = s.Substring(10).Split(' ').First(\_s => !string.IsNullOrEmpty(\_s) && char.IsLetter(\_s[0]));

substitutions.Add(new Substitution(person, date, type, Regex.Match(s, @"\b[А-Я]{3}-\d{1,}\b").Value));

continue;

}

if (s.StartsWith("Всего:"))

continue;

person = Regex.Replace(s, @"\d.", "").Trim();

}

string[] \_n = Path.GetFileNameWithoutExtension(filePath).Substring(9).Split('\_');

string name = \_n[0] + " " + \_n[1] + ". " + \_n[2] + ".";

if (!AllData.ContainsKey(name))

AllData.Add(name, new List<Substitution>());

AllData[name].AddRange(substitutions);

writer.Write(substitutions[0].Person + ",\t" + substitutions[0].date.ToShortDateString() + ",\t" + substitutions[0].ClassType + ",\t" + substitutions[0].Group + "\n");

int rows\_amount = Math.Max(b\_lists.Count, substitutions.Count);

for (int i = 1; i < rows\_amount; i++)

writer.WriteLine(",\t,\t,\t,\t,\t" + (b\_lists.Count > i ? b\_lists[i].ToString() : "") + ",\t"

+ (substitutions.Count > i ? (substitutions[i].Person + ",\t" + substitutions[i].date.ToShortDateString() + ",\t" + substitutions[i].ClassType + ",\t" + substitutions[i].Group + "")

: ",\t,\t,\t"));

}

MessageBox.Show("Данные успешно записаны в CSV!");

oDoc.Close();

oWord.Quit();

}

4.6.2. Сеть Петри запрограммированного технологического процесса

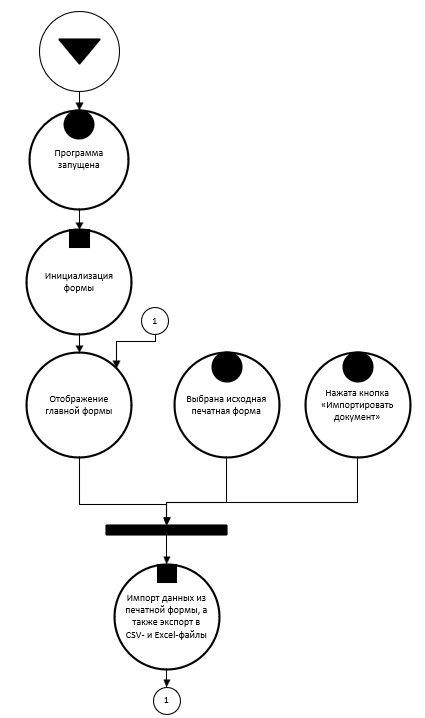


Рисунок 2 – Сеть Петри запрограммированного технологического процесса

4.6.3. Блок-схема алгоритма

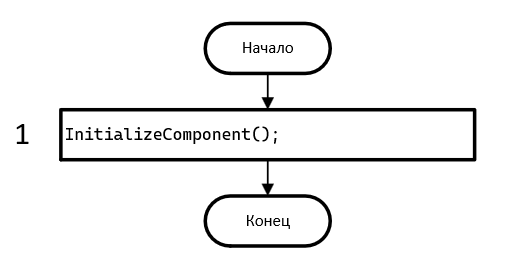


Рисунок 3 – Инициализация формы

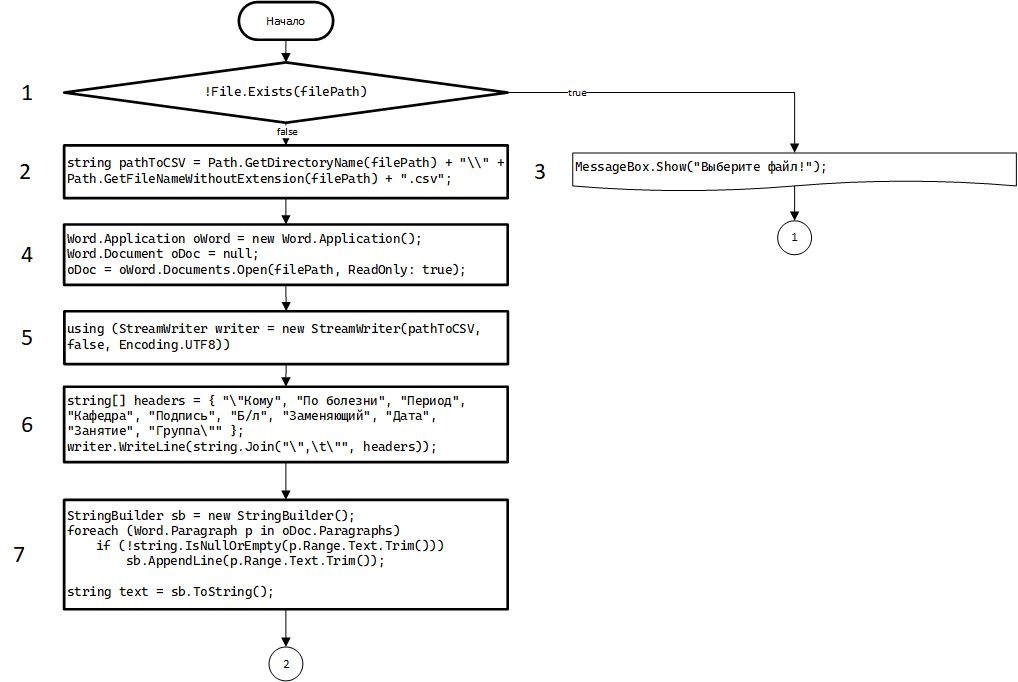


Рисунок 4 – Импорт данных из печатной формы, а также экспорт в CSV-файл, часть 1

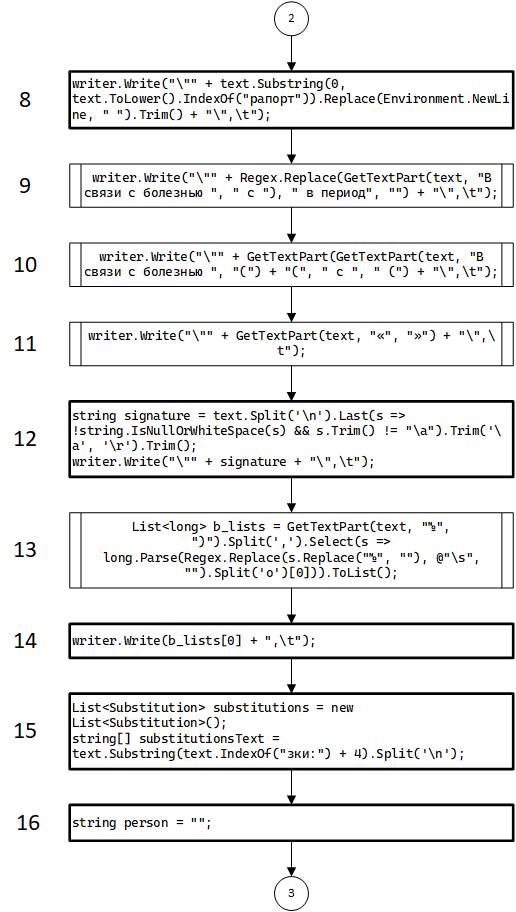


Рисунок 5 – Импорт данных из печатной формы, а также экспорт в CSV-файл, часть 2

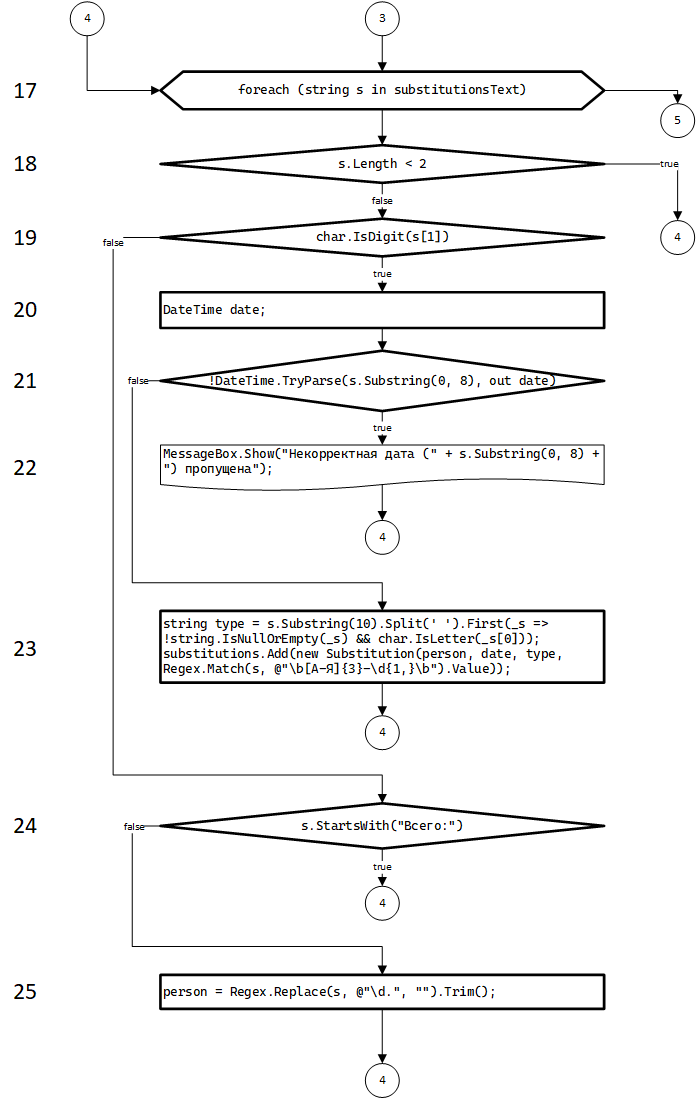


Рисунок 6 – Импорт данных из печатной формы, а также экспорт в CSV-файл, часть 3

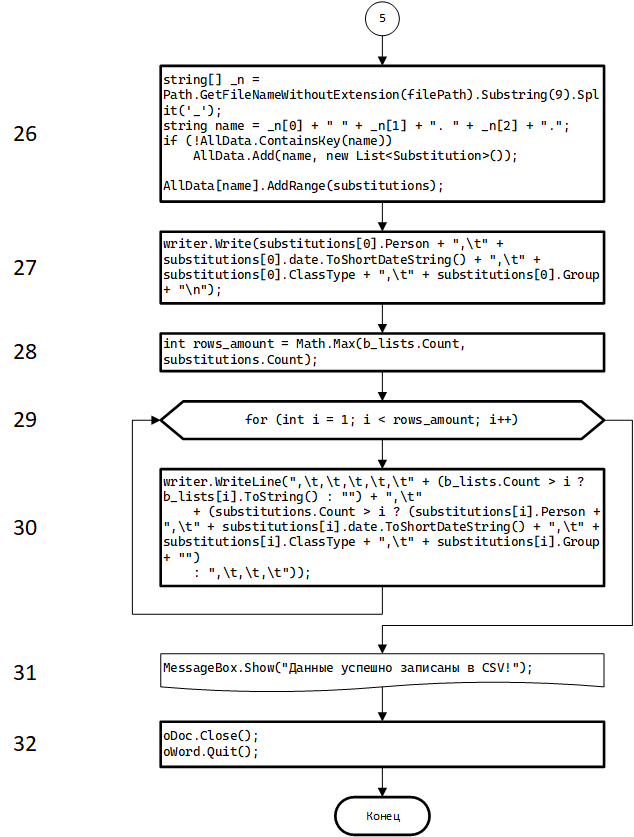
**

Рисунок 7 – Импорт данных из печатной формы, а также экспорт в CSV-файл, часть 4

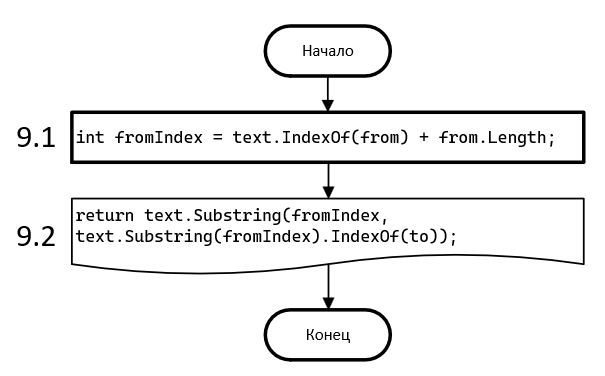
**

Рисунок 8 – Метод GetTextPart

4.7. Разработка функций импорта сведений из оригинального и изменённого текстовых файлов формата *CSV* с воссозданием *Word*

4.7.1. Листинг

private void ConvertCsvToWord(string filePath)

{

if (!File.Exists(filePath))

{

MessageBox.Show("Файл не выбран!");

return;

}

string pathToWord = Path.GetDirectoryName(filePath) + "\\" + Path.GetFileNameWithoutExtension(filePath) + "\_restored.docx";

Word.Application oWord = new Word.Application();

Word.Document oDoc = oWord.Documents.Add();

//Записываем весь файл в строку

string text = File.ReadAllText(filePath, Encoding.UTF8);

string[][] cells = text.Split('\n').Select(s => new Regex(",(?=(?:[^\"]\*\"[^\"]\*\")\*(?![^\"]\*\"))").Split(s).Select(s1 => s1.Trim().Trim('\"')).ToArray()).ToArray();

Word.Paragraph oPr = oDoc.Paragraphs.Add();

oPr.Range.Font.Size = 12;

oPr.Range.Text = cells[1][0];

oPr.Alignment = Word.WdParagraphAlignment.wdAlignParagraphRight;

oPr.Range.Text += "\n\n\n";

oPr.Alignment = Word.WdParagraphAlignment.wdAlignParagraphCenter;

oPr.Range.Font.Size = 12;

oPr.Range.InsertParagraphAfter();

oPr.Range.Text = "Рапорт\n";

oPr.Range.Font.Size = 12;

oPr.Alignment = Word.WdParagraphAlignment.wdAlignParagraphJustify;

oPr.Range.InsertParagraphAfter();

List<long> b\_lists = new List<long>();

for (int i = 1; i < cells.Length - 1; i++)

{

if (!long.TryParse(cells[i][5], out \_))

continue;

b\_lists.Add(long.Parse(cells[i][5]));

}

oPr.Range.Text = "\tВ связи с болезнью " + cells[1][1] + " в период с " + cells[1][2]

+ " (больничные листы: " + b\_lists.Select(bl => bl.ToString("№000 000 000 000")).Aggregate((s1, s2) => s1 + ", " + s2) + ")"

+ " на кафедре «" + cells[1][3] + "» производились замены занятий в счет бюджетной нагрузки:";

List<Substitution> substitutions = new List<Substitution>();

for (int i = 1; i < cells.Length - 1; i++)

{

substitutions.Add(new Substitution(cells[i][6], DateTime.Parse(cells[i][7]), cells[i][8], cells[i][9]));

}

substitutions = substitutions.OrderBy(sub => sub.Person).ToList();

int subPersonCount = 0;

string person = string.Empty;

for(int i = 0; i < substitutions.Count; i++)

{

if (person != substitutions[i].Person)

{

if (person != string.Empty)

oPr.Range.Text += "\t\t "

+ "Всего: " + (substitutions.Where(sub => sub.Person == person).Count() \* 2);

person = substitutions[i].Person;

subPersonCount++;

oPr.Range.Font.Size = 12;

oPr.Range.InsertParagraphAfter();

oPr.Range.Text = "\n\t\t" + subPersonCount + ". " + substitutions[i].Person;

oPr.Alignment = Word.WdParagraphAlignment.wdAlignParagraphLeft;

}

oPr.Range.Text += "\t\t "

+ substitutions[i].date.ToString("dd.MM.yy") + "г. " + substitutions[i].ClassType + " " + substitutions[i].Group + " – 2ч.";

}

oPr.Range.Text += "\t\t "

+ "Всего: " + (substitutions.Where(sub => sub.Person == person).Count() \* 2) + "\n";

oPr.Alignment = Word.WdParagraphAlignment.wdAlignParagraphRight;

oPr.Range.InsertParagraphAfter();

oPr.Range.Text = cells[1][4];

oDoc.SaveAs2(pathToWord);

oWord.Quit();

MessageBox.Show("Документ восстановлен!");

}

4.7.2. Сеть Петри запрограммированного технологического процесса

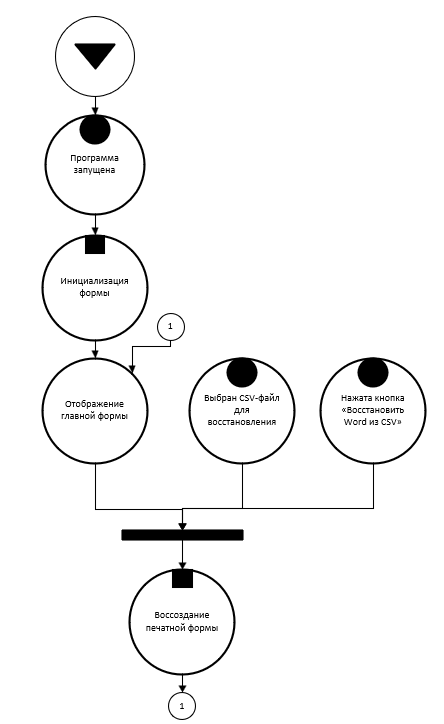


Рисунок 9 – Сеть Петри запрограммированного технологического процесса

4.7.3. Блок-схема алгоритма

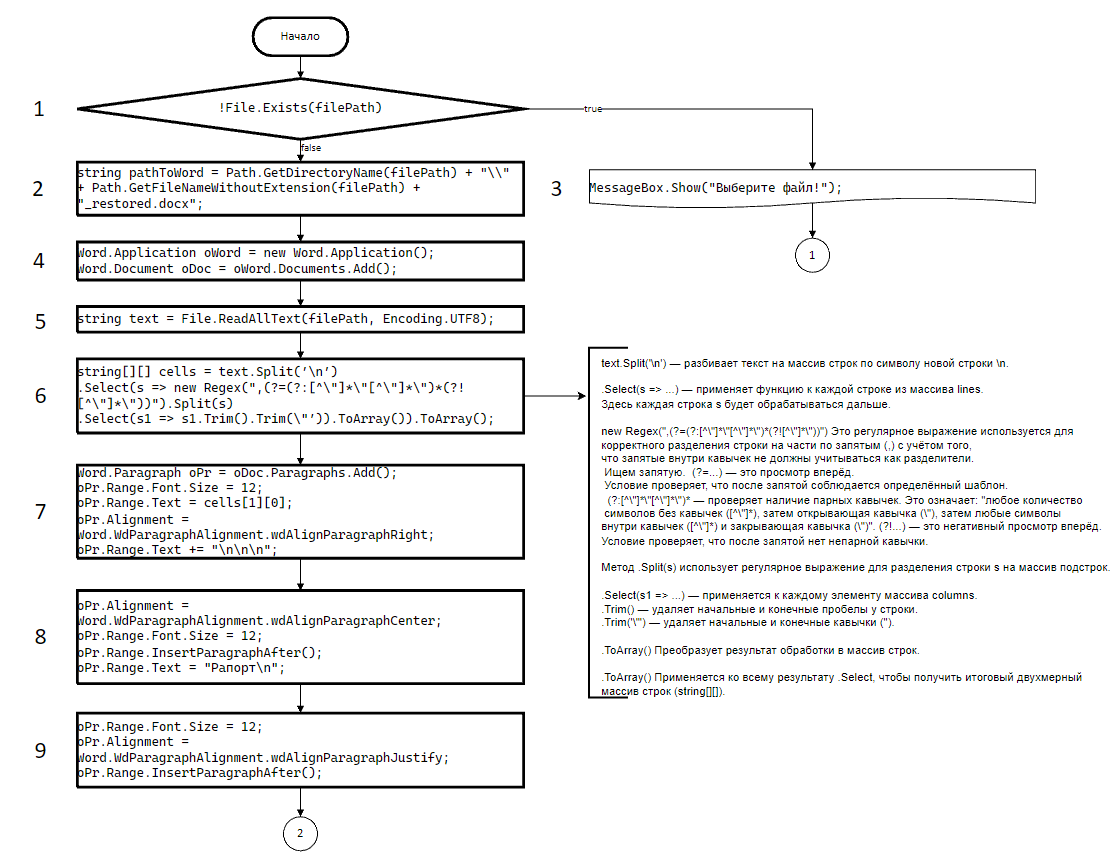


Рисунок 10 – Воссоздание печатной формы, часть 1

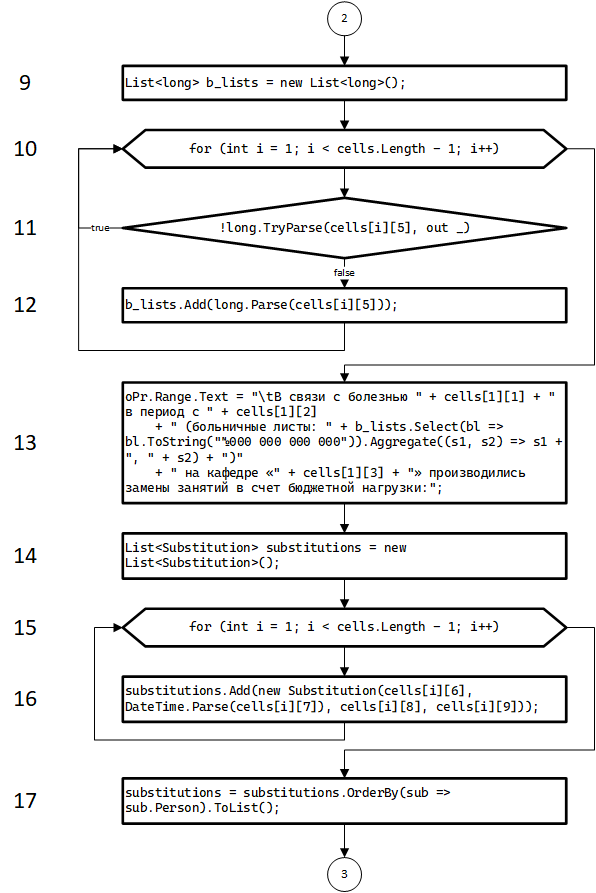


Рисунок 11 – Воссоздание печатной формы, часть 2

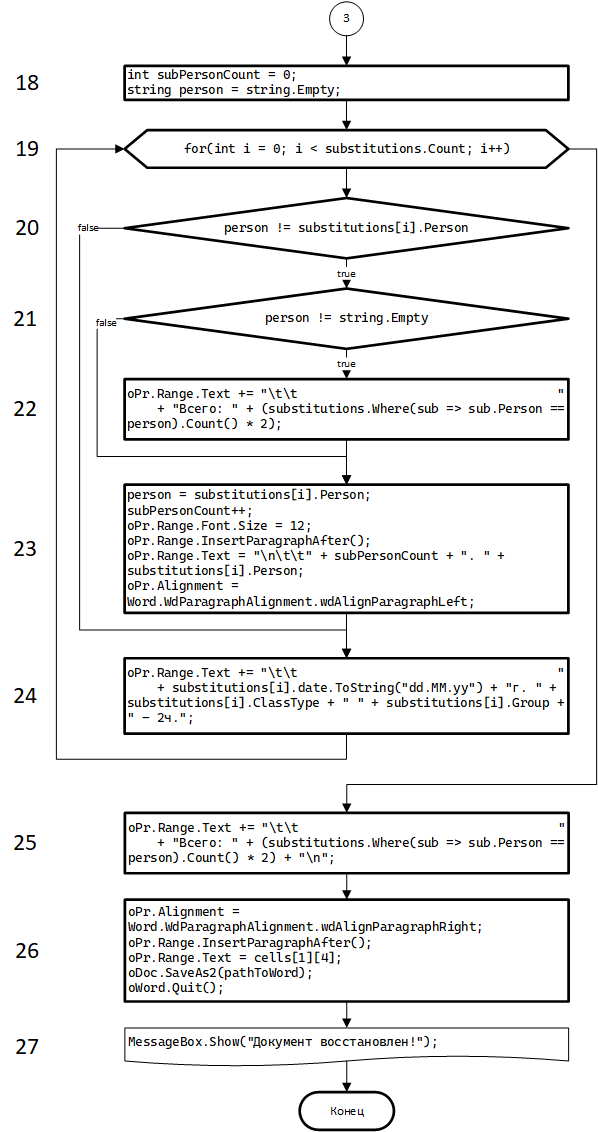


Рисунок 12 – Воссоздание печатной формы, часть 3

4.8. Разработка функций экспорта сведений из экранной формы в *Excel*

4.8.1. Листинг

private void ConvertCSVToExcel(string filePath)

{

if (!File.Exists(filePath))

{

MessageBox.Show("Выберите файл!");

return;

}

string pathToExcel = Path.GetDirectoryName(filePath) + "\\" + Path.GetFileNameWithoutExtension(filePath) + ".xlsx";

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

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

Excel.Worksheet worksheet = workbook.Worksheets[1];

//Записываем весь файл в строку

string text = File.ReadAllText(filePath, Encoding.UTF8);

string[][] cells = text.Split('\n').Select(s => new Regex(",(?=(?:[^\"]\*\"[^\"]\*\")\*(?![^\"]\*\"))").Split(s)).ToArray();

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

for(int j = 0; j < cells.Max(col => col.Length); j++)

{

try

{

worksheet.Cells[i + 1, j + 1] = cells[i][j].Trim('\"', '\t');

}

catch (Exception e) { }

}

File.Delete(pathToExcel);

// Сохранение Excel файла

workbook.SaveAs(pathToExcel);

// Закрываем книгу без сохранения

workbook.Close(false);

oExcel.Quit();

MessageBox.Show("Данные успешно записаны в Excel!");

}

4.8.2. Сеть Петри запрограммированного технологического процесса

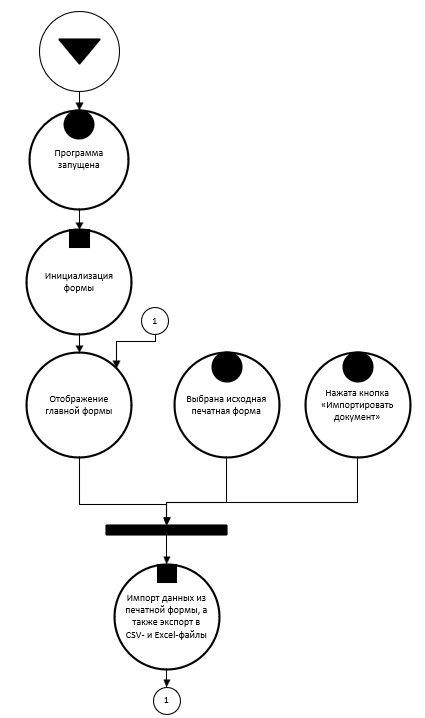


Рисунок 13 – Сеть Петри запрограммированного технологического процесса

4.8.3. Блок-схема алгоритма

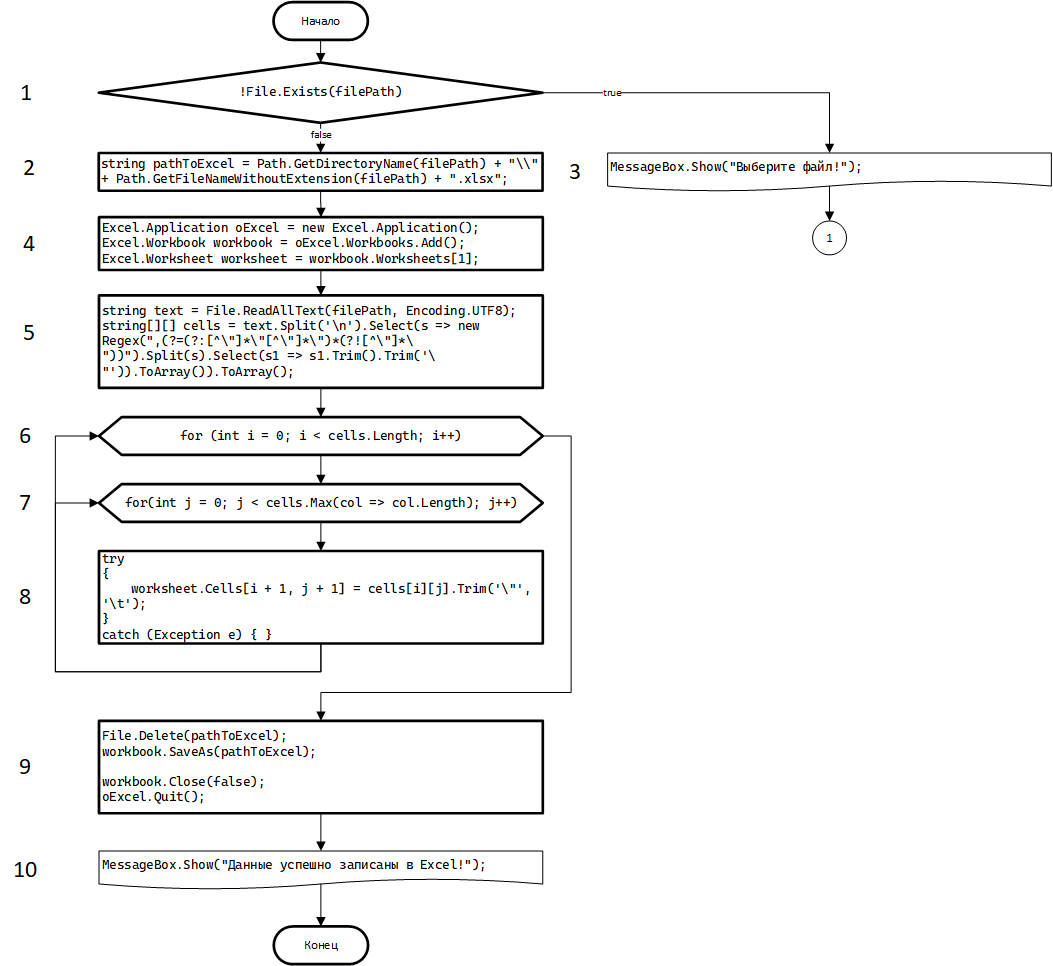


Рисунок 14 – Экспорт данных из CSV-файла в Excel

4.9. Графический анализ данных, содержащихся в печатной форме

Режимы диаграмм, отображающихся на элементе экранной формы *Chart*, выбираются при помощи выпадающего списка (*ComboBox*). Для демонстрации работы диаграмм были импортированы первые пять печатных форм.

Пример работы режима диаграммы «Количество дней, проведённых на больничном преподавателями за весь известный период» на Рисунке 15:

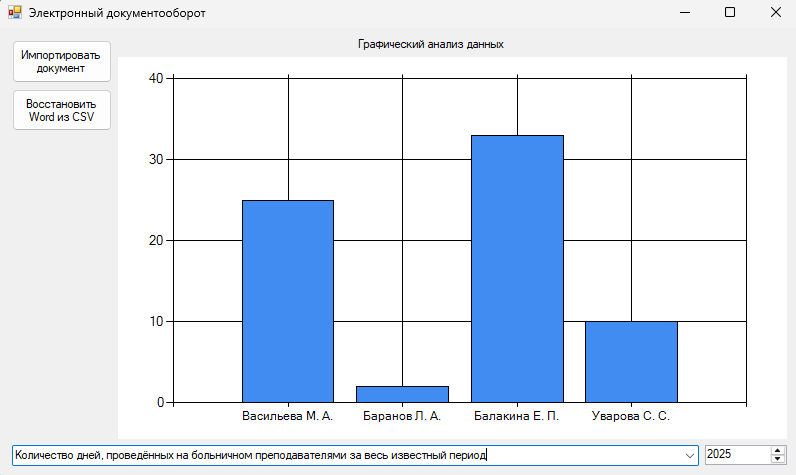


Рисунок 15 – Пример работы первого режима диаграммы

Пример работы режима диаграммы «Количество дней, проведённых на больничном преподавателями за указанный год» (выбранный год – 2020) на Рисунке 16:

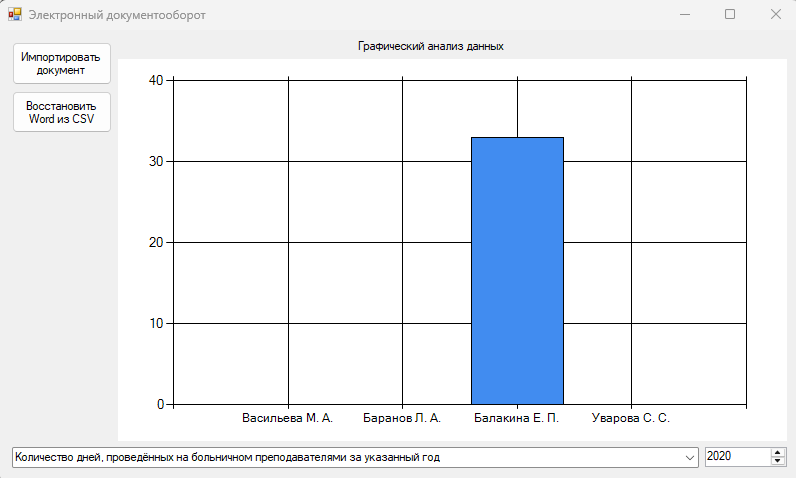


Рисунок 16 – Пример работы второго режима диаграммы

Пример работы режима диаграммы «Общее количество часов, заменённых преподавателями по болезни» на Рисунке 17:

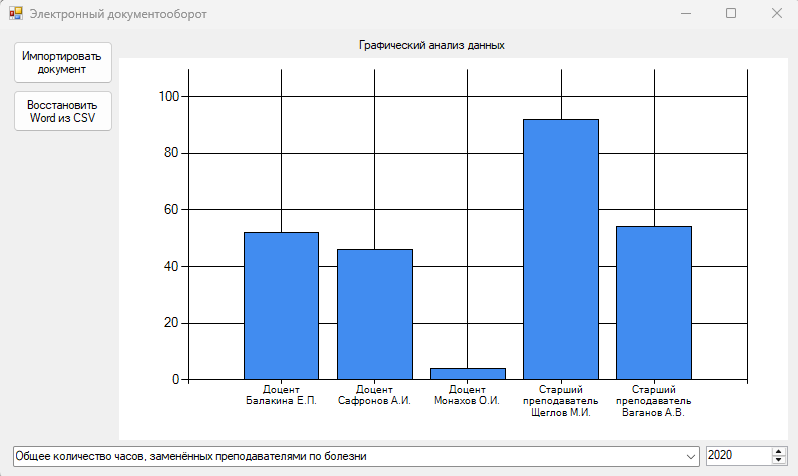


Рисунок 17 – Пример работы третьего режима диаграммы

Пример работы режима диаграммы «Количество часов практических занятий, заменённых преподавателями по болезни» на Рисунке 18:

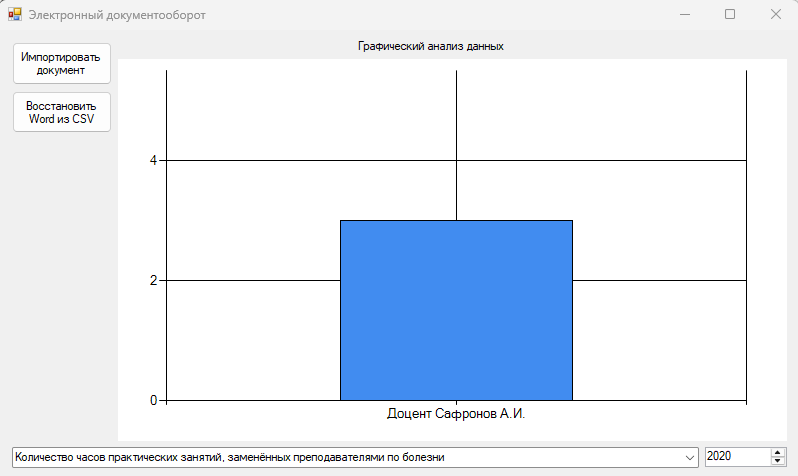


Рисунок 18 – Пример работы четвёртого режима диаграммы

Пример работы режима диаграммы «Количество часов лабораторных работ, заменённых преподавателями по болезни» на Рисунке 19:

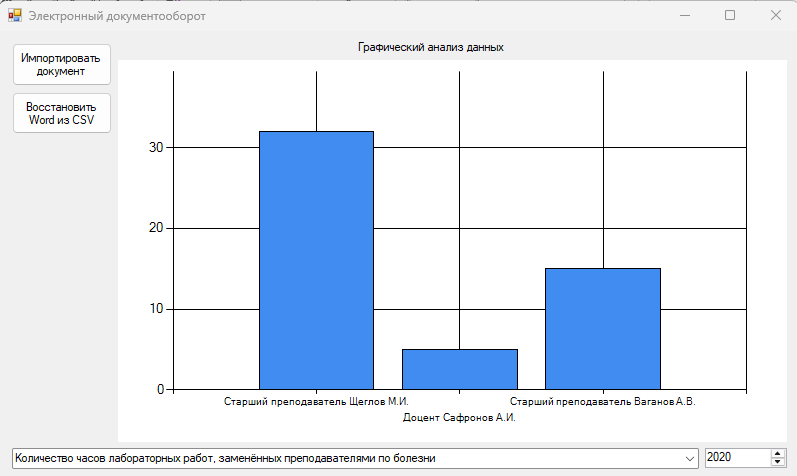


Рисунок 19 – Пример работы пятого режима диаграммы

Пример работы режима диаграммы «Количество часов лекций, заменённых преподавателями по болезни» на Рисунке 20:

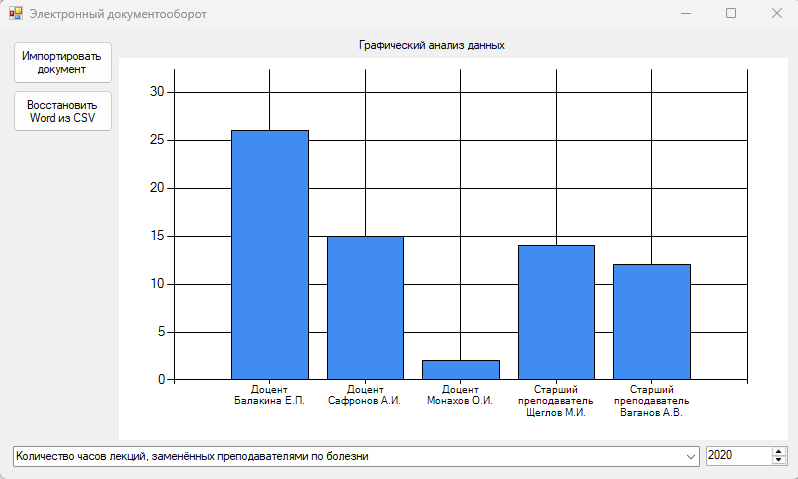


Рисунок 20 – Пример работы шестого режима диаграммы

Пример работы режима диаграммы «Общее количество дней, в которые замещающие преподаватели осуществляли замены за весь известный период» на Рисунке 21:

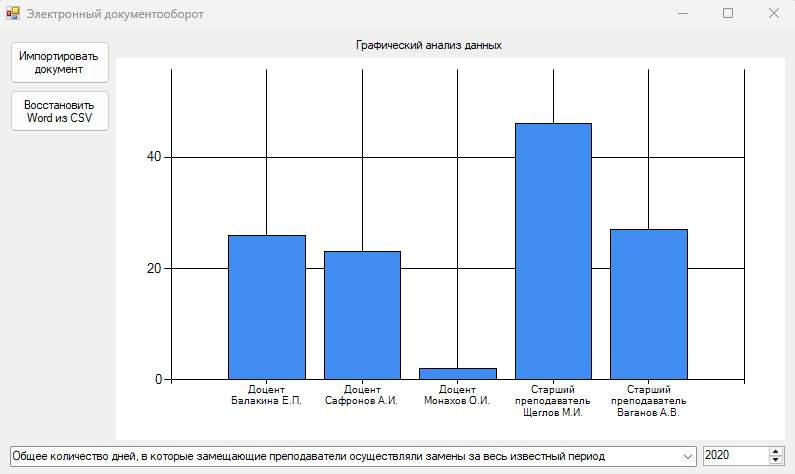


Рисунок 21 – Пример работы седьмого режима диаграммы

Пример работы режима диаграммы «Общее количество дней, в которые замещающие преподаватели осуществляли замены за указанный год.» (выбранный год – 2017) на Рисунке 22:

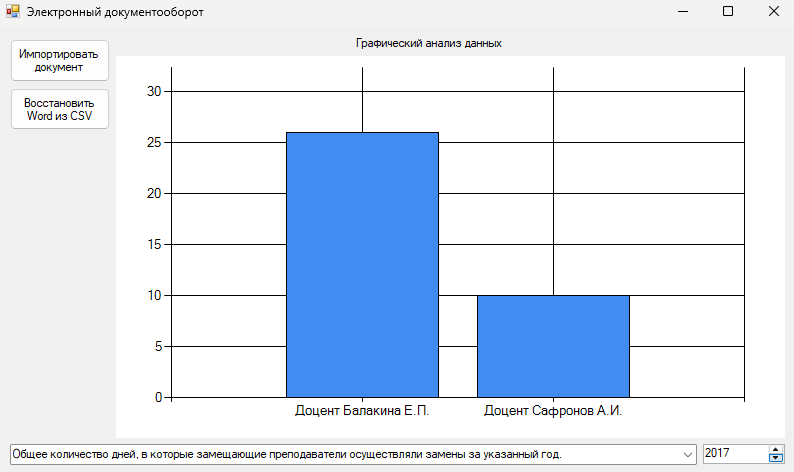


Рисунок 22 – Пример работы восьмого режима диаграммы

4.9.1. Листинг

private void UpdateChart(Tuple<string, float>[] data)

{

if (data.Length > 0)

{

float max = data.Select(d => d.Item2).Max();

MainChart.ChartAreas[0].AxisY.Maximum = max + (max / 6) + 2;

}

Series SL = new Series

{

IsValueShownAsLabel = false,

ChartType = SeriesChartType.Column,

BorderColor = Color.Black,

BorderWidth = 1,

IsVisibleInLegend = false

};

MainChart.ChartAreas[0].AxisX.CustomLabels.Clear();

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

{

MainChart.ChartAreas[0].AxisX.CustomLabels.Add(i + 0.5, i + 1.5, data[i].Item1);

SL.Points.AddXY(i + 1, data[i].Item2);

}

MainChart.Series.Clear();

MainChart.Series.Add(SL);

}

4.9.2. Сеть Петри запрограммированного технологического процесса

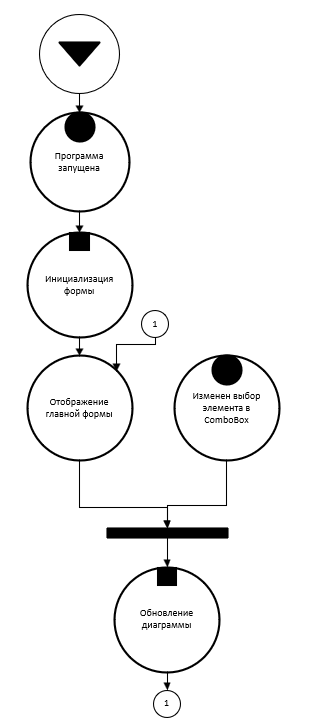
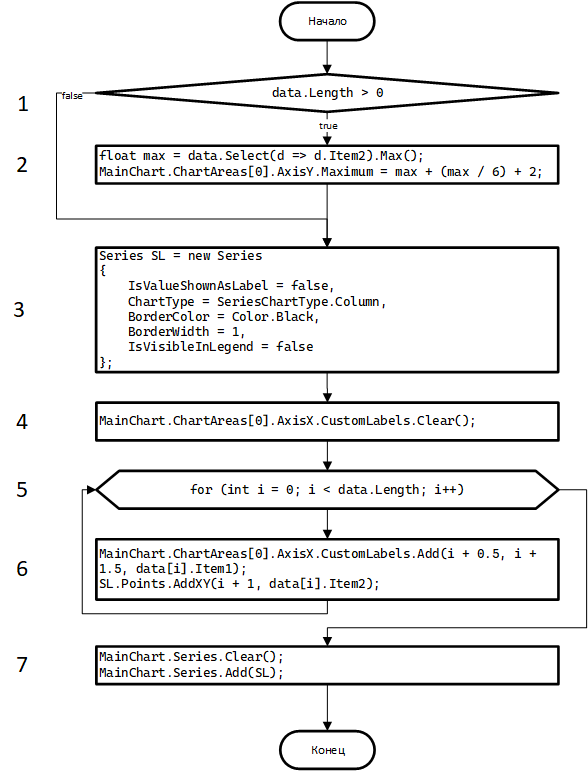


Рисунок 23 – Сеть Петри запрограммированного технологического процесса

4.9.3. Блок-схема алгоритма

 Рисунок 24 – Обновление диаграммы

5. Инструкция пользователя

Интерфейс главного окна программы на Рисунке 25:



Рисунок 25 – Интерфейс главного окна программы

Группы элементов экранной формы, отвечающих за выбор и генерацию документов:

1. Функции импорта:
   1. Кнопка «Импортировать документ»: позволяет выбрать печатную форму из файловой системы для импорта данных на экранную форму, а также для экспорта в CSV и Excel.

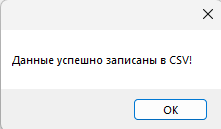


Рисунок 26 – Всплывающее сообщение «Данные успешно записаны в CSV!»

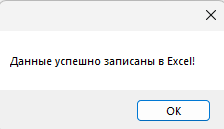


Рисунок 27 – Всплывающее сообщение «Данные успешно записаны в Excel!»

1. Функции экспорта:
   1. Кнопка «Восстановить в Word из CSV»: инициализирует процесс генерации *Word*-документа, основываясь на данных из *CSV­*-файла. По завершению этого процесса появится всплывающее сообщение о выполнении процесса экспорта (Рисунок 30):

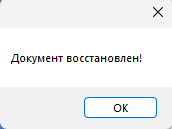


Рисунок 28 – Всплывающее сообщение «Документ восстановлен!»

Группа элементов экранной формы, отвечающих за графический анализ данных, содержащихся в печатной форме:

1. Выпадающий список позволяет выбрать режим отображаемой диаграммы.
2. Числовое поле выбора года предназначено для 1 и 8 режимов построения диаграммы, где необходимо указать год.
3. Элемент экранной формы «*Chart*»: отображает выбранную диаграмму.

6. Заключение

В результате выполненной работы было создано программное решение для электронного документооборота, включающее в себя функции конвертации данных в различные форматы (DOCX, XLSX, CSV), а также поддержку графического анализа информации из печатных форм. Разработанная система обеспечивает эффективную обработку электронных документов, облегчает их анализ и экономит время пользователя за счет автоматизации рутинных процессов. Таким образом, повышается общая эффективность при работе с электронными документами.

7. Выводы

В процессе выполнения работы был разработан эргономичный пользовательский интерфейс, который позволяет пользователю легко пользоваться элементами и функциями программного обеспечения, предназначенными для электронного документооборота. В процессе выполнения работы был освоен навык работы с пространством имен *Microsoft.Office.Interop*, использование его для работы с *Excel, Word* программным путём, также был повышен уровень навыка работы с *CSV* файлами. Дополнительно был освоен навык работы с *Chart*-элементом экранной формы (настройка осей диаграммы и серий данных диаграммы). Закреплен навык чтения и записи данных в файлы, а также улучшен навык написания алгоритмов.

Библиографический список

1. Почему ЭДО — это удобно, и как организовать его в компании [Электронный источник]: unisender.com *URL: https://www.unisender.com/ru/blog/kak-organizovat-edo-elektronnyj-dokumentooborot-v-kompanii/* (дата обращения: 10.01.2025).

2. Электронный документооборот (ЭДО): для чего он нужен и как организовать его в компании [Электронный источник]: *skillbox.ru URL:https://skillbox.ru/media/management/elektronnyy-dokumentooborot-edo-dlya-chego-on-nuzhen-i-kak-organizovat-ego-v-kompanii/* (дата обращения: 10.01.2025).

3. Применение электронного документооборота. Для чего бизнесу нужен ЭДО? [Электронный источник]: *esphere.ru URL: https://www.esphere.ru/blog/chto-takoe-elektronnyj-dokumentooborot/* (дата обращения: 10.01.2025).