Министерство транспорта Российской Федерации

Федеральное государственное автономное

образовательное учреждение высшего образования

«Российский университет транспорта (МИИТ)»

Институт транспортной техники и систем управления

Кафедра «Управление и защита информации»

Курсовая работа

по дисциплине:

«Методы программирования»

Выполнил: ст. гр. ТКИ-341

Панаргин В.М.

Вариант №5

Проверил: к.т.н., доцент Сафронов А.И.

Москва – 2024 г.

Содержание

[1 Введение 3](#_Toc166350403)

[2 Цель работы 3](#_Toc166350404)

[3 Формулировка задачи 3](#_Toc166350405)

[4 Содержательная часть работы 4](#_Toc166350406)

[4.1 Анализ печатной формы 4](#_Toc166350407)

[4.2 Описание предназначения печатной формы 4](#_Toc166350408)

[4.3 Разнообразие печатных форм и выделение различий 4](#_Toc166350409)

[4.4 Унификация печатных форм для удобстав обработки данных 4](#_Toc166350410)

[4.5 Разработка экранной формы для импорта сведений из печатной формы 5](#_Toc166350411)

[4.6 Разработка функции экспорта сведений из печатной формы в файл текстовый формата CSV 7](#_Toc166350412)

[4.7 Разработка функции импорта сведений из оригинального и изменённого текстовых файлов формата CSV с воссозданием Word 8](#_Toc166350413)

[4.8 Разработка функции экспорта сведений из экранной формы в Excel 10](#_Toc166350414)

[4.9 Графический анализ данных, содержащихся в печатных формах 11](#_Toc166350415)

[4.9.1 Показывать на столбчатой диаграмме количество уникальных соавторов у каждого автора. 11](#_Toc166350416)

[4.9.2 Показывать на столбчатой диаграмме для выбранного автора количество публикаций за каждый год. 13](#_Toc166350417)

[4.9.3 Показывать на столбчатой диаграмме для выбранного автора количества работ различного объёма в печатных страницах. 15](#_Toc166350418)

[4.10 Инструкция пользователя 17](#_Toc166350419)

[5 Заключение 19](#_Toc166350420)

[6 Выводы 19](#_Toc166350421)

# Введение

В рамках данной работы рассмотрена автоматизация чтения данных из документа Word и ее последующее эскпортирование в CSV или Excel. Так же по документам стоится статистика в виде столбчатых диаграмм. Автоматизация подобных задач является актуальной в любой сфере деятельности, так как ведение документации является обязательной частью любой деятельности.

# Цель работы

Закрепить навыки, полученные в рамках курса «Методы программирования», в том числе: подключение внешних модулей (библиотек классов) и их использоваения в разрабатываеммых программах, программируемая настройка электронных таблиц Excel и документов Word. Освоить навыки работы с элементом Chart и построением графиков.

# Формулировка задачи

1. Проанализировать печатные формы

2. Уметь воссоздавать структуру печатной формы

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

4. Уметь конвертировать печатную форму Word - Excel / Excel - Word (в зависимости от исходно выданного документа)

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

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

# Содержательная часть работы

## Анализ печатной формы

Печатная форма представляет из себя список трудов преподавателей. Ее главная составляющая – таблица, включающая в себя список работ.

## Описание предназначения печатной формы

Форма предназначена для хранения информации о списке трудов конкретного преподавателя

## Разнообразие печатных форм и выделение различий

Главное различие заключается в форматировании таблицы. В некоторых документах заголовки дублируются на каждой новой странице, в некоторых добавлены дополнительные незначащие строки. Так же в некоторых таблицах добавлены дополнительные параграфы.

## Унификация печатных форм для удобстав обработки данных

В первую очередь, из формы удалены все не несущие информации параграфы. Далее удалены заголовки и все символы, мешающие чтению текста (перенос каретки, новый параграф, разрыв страницы и т.д.)

## Разработка экранной формы для импорта сведений из печатной формы

Так как основную информацию в себе несет именно таблица, в рамках данной работы было решено представить информацию в элементе DataGridView. Для удобства данный пункт был выполнен после пункта 4.6, и данные считываются из CSV файла, а не из экранной формы. Пример работы данной программы представлен на рисунках 1 и 2:

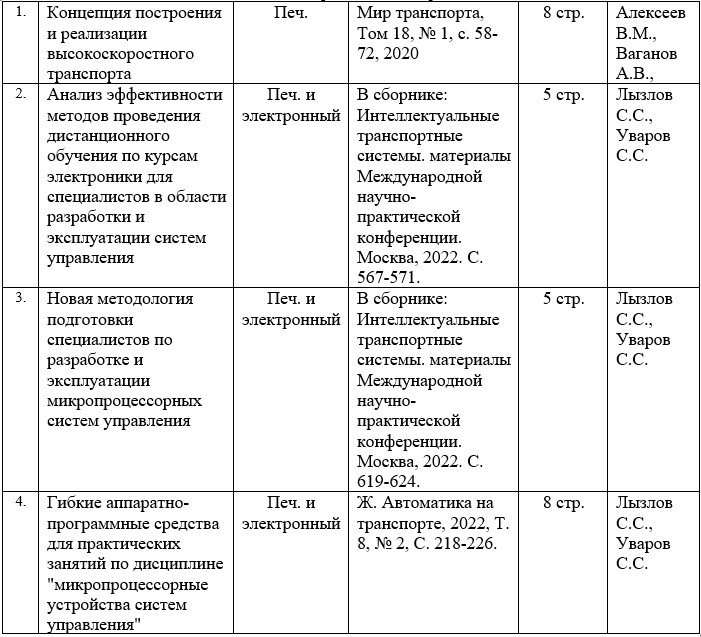


Рисунок 1 - Исходная таблица в Word

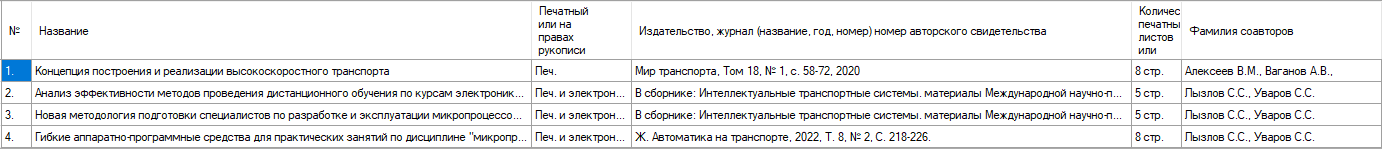


Рисунок 2 - Таблица, считанная в элемент DataGridView

Листинг кода экранной формы:

public partial class frmPreview : Form

{

    private string filename;

    public frmPreview(string filename)

    {

        this.filename = filename;

        InitializeComponent();

    }

    private void frmPreview\_Load(object sender, EventArgs e)

    {

        this.WindowState = FormWindowState.Maximized;

        this.dgwPreview.ColumnCount = 6;

        this.dgwPreview.Columns[0].HeaderText = "№";

        this.dgwPreview.Columns[0].Width = 30;

        this.dgwPreview.Columns[1].HeaderText = "Название";

        this.dgwPreview.Columns[1].Width = 500;

        this.dgwPreview.Columns[2].HeaderText = "Печатный или на правах рукописи";

        this.dgwPreview.Columns[3].HeaderText = "Издательство, журнал (название, год, номер) номер авторского свидетельства";

        this.dgwPreview.Columns[3].Width = 500;

        this.dgwPreview.Columns[4].HeaderText = "Количество печатных листов или страниц";

        this.dgwPreview.Columns[4].Width = 50;

        this.dgwPreview.Columns[5].HeaderText = "Фамилия соавторов";

        this.dgwPreview.Columns[5].Width = 200;

        StreamReader sr = new StreamReader(this.filename);

        string temp;

        while ((temp = sr.ReadLine()) != null)

        {

            this.dgwPreview.Rows.Add(temp.Remove(temp.Length-1).Split('!'));

        }

        sr.Close();

    }

}

## Разработка функции экспорта сведений из печатной формы в файл текстовый формата CSV

Данные считываются из печатной формы и записываются в CSV файл с разделителем «!». Пример работы представлен на рисунках 1 и 3:

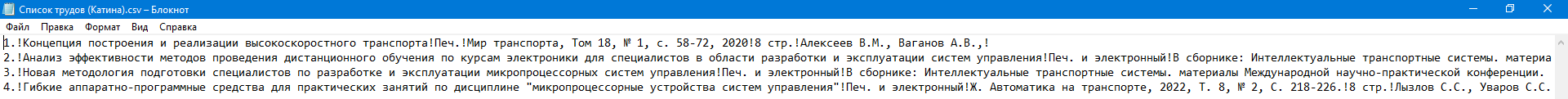


Рисунок 3 - Генерируемый программой CSV файл

Листинг кода функции экспорта в CSV файл:

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

{

    wordapp = new Word.Application();

    wd = wordapp.Documents.Open(this.files[this.cmbFiles.SelectedIndex], ReadOnly:true);

    StreamWriter sw = new StreamWriter(Application.StartupPath + "\\temp\\" + ChngExtension(this.cmbFiles.SelectedItem.ToString(), ".csv"));

    Word.Table wt = wd.Tables[1];

    string temp;

    foreach (Word.Row row in wt.Rows)

    {

        foreach (Word.Cell cl in row.Cells)

        {

            temp = cl.Range.Text;

            temp = temp.Replace("\a", string.Empty).Replace("\n", string.Empty).Remove(temp.Length-2);

            Console.WriteLine(temp);

            sw.Write(temp);

            sw.Write("!");

        }

        Console.WriteLine("N");

        sw.Write("\n");

    }

    sw.Close();

    wordapp.Quit();

}

## Разработка функции импорта сведений из оригинального и изменённого текстовых файлов формата CSV с воссозданием Word

Данные построчно считываются из CSV файла и записываются в таблицу Word. Пример работы представлен на рисунках 3 и 4:

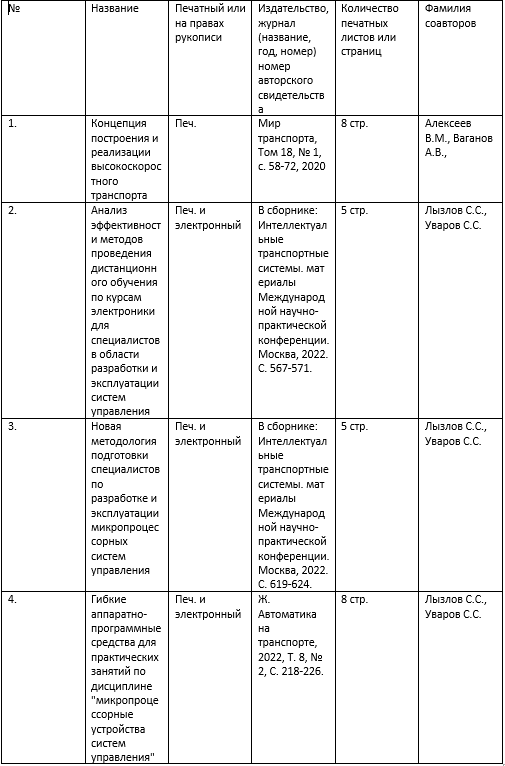


Рисунок 4 - Востановленный из CSV файла документ

Листинг кода функции импорта из CSV:

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

{

    wordapp = new Word.Application();

    wd = wordapp.Documents.Add();

    Word.Paragraph wp = wd.Paragraphs[1];

    Word.Table wt = wd.Tables.Add(wp.Range, 1, 6);

    wt.set\_Style("Сетка таблицы");

    StreamReader sr = new StreamReader(Application.StartupPath + "\\temp\\" + ChngExtension(this.cmbFiles.SelectedItem.ToString(), ".csv"));

    int counter1 = 1;

    int counter2 = 1;

    string temp;

    string[] atemp;

    string[] top = { "№", "Название", "Печатный или на правах рукописи", "Издательство, журнал (название, год, номер) номер авторского свидетельства", "Количество печатных листов или страниц", "Фамилия соавторов" };

    foreach (string line in top)

    {

        wt.Cell(1, counter2).Range.Text = line;

        counter2++;

    }

    counter1++;

    while ((temp = sr.ReadLine()) != null)

    {

        atemp = temp.Remove(temp.Length - 1).Split('!');

        counter2 = 1;

        wt.Rows.Add();

        foreach (string line in atemp)

        {

            wt.Cell(counter1, counter2).Range.Text = line;

            counter2++;

        }

        counter1++;

    }

    sr.Close();

    wd.SaveAs2(Application.StartupPath + "\\output\\" + this.cmbFiles.SelectedItem.ToString());

    wordapp.Quit();

}

## Разработка функции экспорта сведений из экранной формы в Excel

Для удобства данные для записи в Excel файл считываются из CSV файла. Пример работы представлен на рисунках 3 и 5:

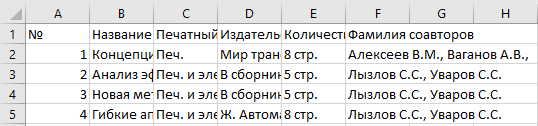


Рисунок 5 - Генерируемый программой файл Excel

Листинг кода функции экспорта в Excel:

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

{

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

    exapp.Workbooks.Add();

    Excel.Worksheet workSheet = (Excel.Worksheet)exapp.ActiveSheet;

    StreamReader sr = new StreamReader(Application.StartupPath + "\\temp\\" + ChngExtension(this.cmbFiles.SelectedItem.ToString(), ".csv"));

    string temp;

    string[] atemp;

    int counter1 = 1;

    int counter2 = 1;

    string[] top = { "№", "Название", "Печатный или на правах рукописи", "Издательство, журнал (название, год, номер) номер авторского свидетельства", "Количество печатных листов или страниц", "Фамилия соавторов" };

    foreach (string line in top)

    {

        workSheet.Cells[counter1, counter2] = line;

        counter2++;

    }

    counter1++;

    while ((temp = sr.ReadLine()) != null)

    {

        atemp = temp.Remove(temp.Length - 1).Split('!');

        counter2 = 1;

        foreach (string line in atemp)

        {

            workSheet.Cells[counter1, counter2] = line;

            counter2++;

        }

        counter1++;

    }

    sr.Close();

    exapp.ActiveWorkbook.SaveAs(Application.StartupPath + "\\output\\" + ChngExtension(this.cmbFiles.SelectedItem.ToString(), ".xlsx"));

    exapp.Quit();

}

## Графический анализ данных, содержащихся в печатных формах

### Показывать на столбчатой диаграмме количество уникальных соавторов у каждого автора.

Для решения данной задачи колонка, содержащая соавторов делится по разделителю запятая, а затем в словарь записывается количество уникальных соавторов для каждого преподавателя. Пример работы на рисунке 6:

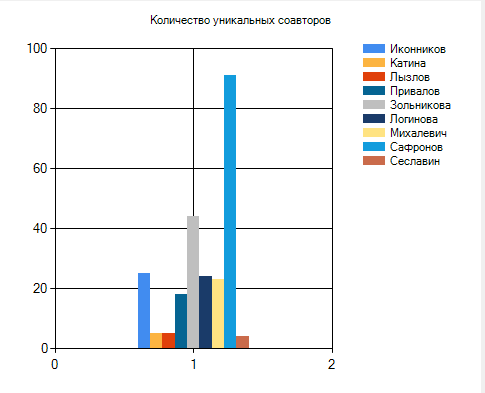


Рисунок 6 - Диаграмма количества уникальных соавторов

Листинг кода для построения данной диаграммы:

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

{

    string[] files = Directory.GetFiles(Application.StartupPath + "\\temp");

    Dictionary<string, int> soauth = new Dictionary<string, int>();

    foreach (string filename in files)

    {

        List<string> authors = new List<string>();

        StreamReader sr = new StreamReader(filename);

        string temp;

        string autst;

        string[] atemp;

        int counter = 0;

        while ((temp = sr.ReadLine()) != null)

        {

            autst = temp.Remove(temp.Length - 1).Split('!')[5];

            if (autst == "") { }

            else

            {

                atemp = autst.Replace(" ", string.Empty).Split(',');

                foreach (string author in atemp)

                {

                    if (!(authors.Contains(author))) authors.Add(author);

                }

            }

        }

        foreach (string author in authors) counter++;

        string name = Regex.Match(filename, "\\(\\w+\\)").Value.Replace("(", string.Empty).Replace(")", string.Empty);

        soauth[name] = counter;

        sr.Close();

    }

    foreach (KeyValuePair<string, int> author in soauth)

    {

        Console.WriteLine($"Author: {author.Key}, Soauthors: {author.Value.ToString()}");

    }

    this.chrtStat.Series.Clear();

    this.chrtStat.Titles.Clear();

    this.chrtStat.Titles.Add("Количество уникальных соавторов");

    foreach (KeyValuePair<string, int> author in soauth)

    {

        Series series = this.chrtStat.Series.Add(author.Key);

        series.Points.Add(author.Value);

    }

}

### Показывать на столбчатой диаграмме для выбранного автора количество публикаций за каждый год.

Для решения данной задачи из колонки 4 извлекается год с использованием регулярных выражений. Затем в словарь записывается количество работ для каждого года. Пример работы на рисунке 7:

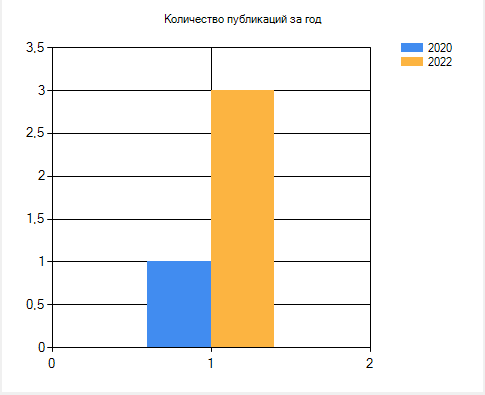


Рисунок 7 - Диаграмма количества публикаций за каждый год

Листинг кода для построения данной диаграммы:

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

{

    StreamReader sr = new StreamReader(this.afilename);

    string temp;

    string pub, year;

    Dictionary<string, int> py = new Dictionary<string, int>();

    while ((temp = sr.ReadLine()) != null)

    {

        pub = temp.Remove(temp.Length - 1).Split('!')[3];

        year = Regex.Match(pub, "20[0-2]\\d").Value;

        if (year == "") year = "NONE";

        if (py.ContainsKey(year)) py[year]++;

        else py[year] = 1;

    }

    foreach (KeyValuePair<string, int> author in py)

    {

        Console.WriteLine($"Year: {author.Key}, Count: {author.Value.ToString()}");

    }

    this.chrtStat.Series.Clear();

    this.chrtStat.Titles.Clear();

    this.chrtStat.Titles.Add("Количество публикаций за год");

    foreach (KeyValuePair<string, int> author in py)

    {

        Series series = this.chrtStat.Series.Add(author.Key);

        series.Points.Add(author.Value);

    }

    sr.Close();

}

### Показывать на столбчатой диаграмме для выбранного автора количества работ различного объёма в печатных страницах.

Для решения данной задачи количество печатных страниз извлекается из колонки 5. Затем в словарь записывается количество работ для каждого объема. Пример работы на рисунке 8:

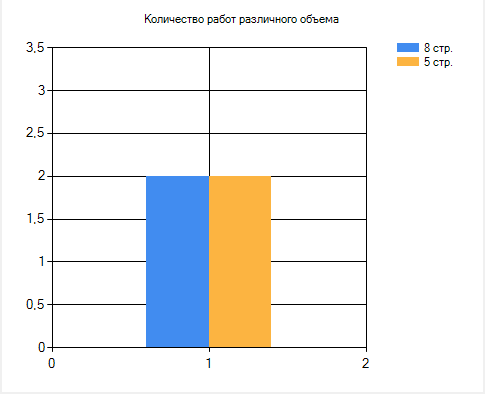


Рисунок 8 - Диаграмма количества работ по количеству страниц

Листинг кода для построения данной диаграммы:

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

{

    StreamReader sr = new StreamReader(this.afilename);

    string temp;

    string stc;

    Dictionary<string, int> volume = new Dictionary<string, int>();

    while ((temp = sr.ReadLine()) != null)

    {

        stc = temp.Remove(temp.Length - 1).Split('!')[4];

        if (stc == "") stc = "NONE";

        if (volume.ContainsKey(stc)) volume[stc]++;

        else volume[stc] = 1;

    }

    foreach (KeyValuePair<string, int> author in volume)

    {

        Console.WriteLine($"STC: {author.Key}, Count: {author.Value.ToString()}");

    }

    this.chrtStat.Series.Clear();

    this.chrtStat.Titles.Clear();

    this.chrtStat.Titles.Add("Количество работ различного объема");

    foreach (KeyValuePair<string, int> author in volume)

    {

        Series series = this.chrtStat.Series.Add(author.Key);

        series.Points.Add(author.Value);

    }

    sr.Close();

}

## Инструкция пользователя

Перед открытием программы убедитесь, что в папке с программой имеются: папка works с печатными формами, папка temp и папка output.

При открытии программы появится интерфейс главной формы:

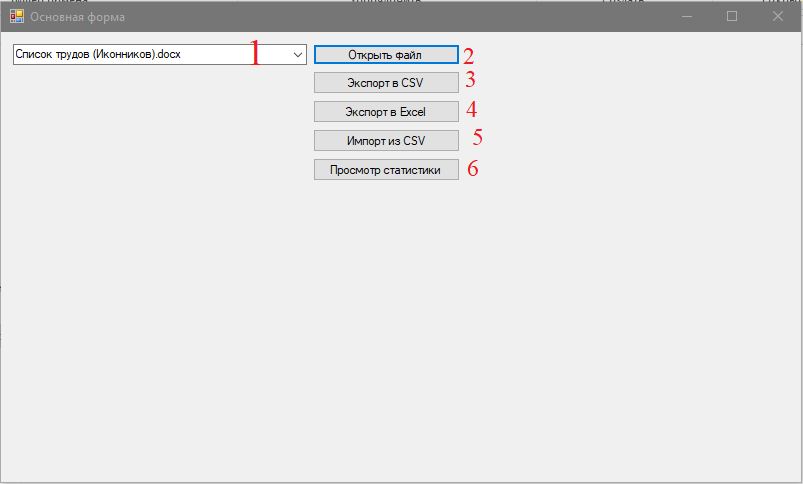


Рисунок 9 - Интерфейс главной формы

На данной форме присутствует 6 основных элементов управления в виде одного ComboBox (1) и пяти кнопок.

Для начала выберите нужный вам файл в ComboBox (1). Затем, нажмите кнопку «Экспорт в CSV» (3). Теперь программа готова к работе.

При нажатии на кнопку «Открыть файл» (2) появится форма с DataGridView, содержащим информацию из файла. Пример представлен на рисунке 2.

При нажатии на кнопку «Экспорт в CSV» (3) информация из печатной формы экспортируется в .csv файл. Данный файл будет помещен в папку temp. Пример представлен на рисунке 3.

При нажатии на кнопку «Экспорт в Excel» (4) информация из печатной формы экспортируется в .xlsx файл. Данный файл будет помещен в папку output. Пример представлен на рисунке 5.

При нажатии на кнопку «Импорт из CSV» (5) информация из CSV файла будет считана и будет воссоздан .docx файл. Данный файл будет помещен в папку output. Пример представлен на рисунке 4.

При нажатии на кнопку «Просмотр статистики», откроется форма с возможностью просмотра различных графиков. Интерфейс данной формы представлен на рисунке 10:

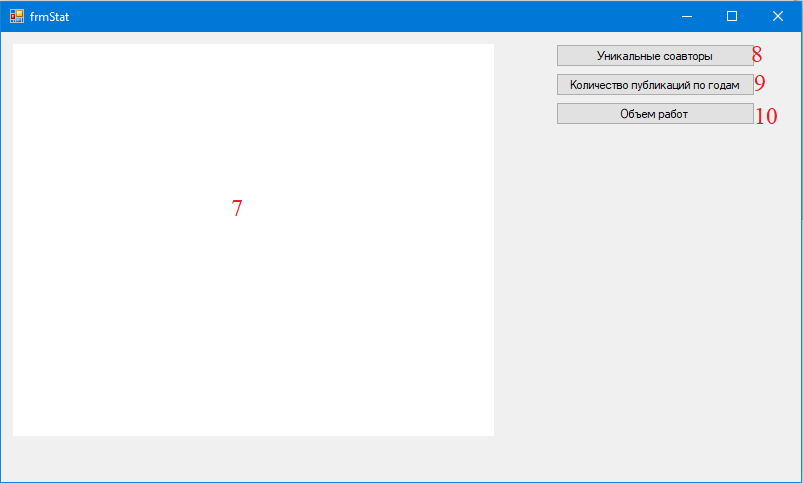


Рисунок 10 - Форма для просмотра статистики

В данной форме присутствует поле для отрисовки графиков Chart (7) и три кнопки управления.

При нажатии на кнопку «Уникальные соавторы» (8) на поле графика будет отрисована столбчатая диаграмма уникального количество соавторов для преподавателей. Учитывайте, что данная диаграмма рисуется по CSV файлам из папки temp, поэтому, если вам необходима статистика по всем преподавателям необходимо произвести экспорт в CSV для каждого файла. Пример представлян на рисунке 6.

При нажатии на кнопку «Количество публикаций по годам» (9) на поле графика будет отрисована столбчатая диаграмма количества публикций по годам для преподавателя, выбранного в ComboBox (1). Пример представлен на рисунке 7.

При нажатии на кнопку «Объем работ» (10) на поле графика будет отрисована столбчатая диаграмма количества публикций по количеству страниц для преподавателя, выбранного в ComboBox (1). Пример представлен на рисунке 8.

# Заключение

В рамках данной работы были выполнены задачи:

* Анализ печатной формы
* Описание предназначения печатной формы
* Выявление различий в печатных формах
* Унификация печатных форм
* Разработка экранной формы для импорта сведений из печатной формы
* Разработка функций экспорта сведений из печатной формы в файл тексторый формата CSV
* Разработка функций импорта сведений из оригинального и изменённого текстовых файлов формата CSV с воссозданием Word
* Разработка функций экспорта сведений из экранной формы в Excel
* Графический анализ данных, содержащихся в печатной форме
* Создание инструкции пользователя

# Выводы

Язык программирования C# в сочетании с Windows Forms позволяет создать удобное Desktop приложение, для автоматизации работы с любого вида документацией в рамках приложений Microsoft Office. Написание данной программы довольно трудоемкий процесс, однако в дальнейшем написанное ПО значительно сократит время, затрачиваемое на составление документов.