**НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ВЫСШАЯ ШКОЛА ЭКОНОМИКИ**

**Факультет компьютерных наук**

**Департамент программной инженерии**

***Контрольное домашнее задание***

***по дисциплине «Программирование».***

Тема работы: **РАБОТА С ФАЙЛАМИ. ОБРАБОТКА ДАННЫХ**

Выполнил: студент(ка) группы 154 (2) \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Мачнев А.Е. «26» февраля 2016 г.

тел. +7(916)-272-32-63

e-mail адрес: aemachnev@edu.hse.ru

Преподаватель: Полицын С.А.

«\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_2016 г.

Москва, 2016 год. Модуль № «3»

Оглавление

[1. УСЛОВИЕ ЗАДАЧИ 3](#_Toc444217910)

[2. ФУНКЦИИ РАЗРАБАТЫВАЕМОГО ПРИЛОЖЕНИЯ 3](#_Toc444217911)

[2.1 Варианты использования 3](#_Toc444217912)

[2.2 Описание интерфейса пользователя 3](#_Toc444217913)

[2.2.1 Вид приложения при запуске 3](#_Toc444217914)

[2.2.2 Действия для открытия файла с данными 4](#_Toc444217915)

[2.2.3 Представление данных 4](#_Toc444217916)

[2.2.4 Работа с настройками 5](#_Toc444217917)

[3. СТРУКТУРА ПРИЛОЖЕНИЯ 7](#_Toc444217918)

[3.1 Диаграмма классов 7](#_Toc444217919)

[3.1.1 Диаграмма классов проекта Homework3.Model 7](#_Toc444217920)

[3.1.2 Диаграмма классов проекта Homework3 8](#_Toc444217921)

[3.2 Описание классов, их полей, методов и методов 9](#_Toc444217922)

[4. РАСПРЕДЕЛЕНИЕ ИСХОДНОГО КОДА ПО ФАЙЛАМ ПРОЕКТА 9](#_Toc444217923)

[5. КОНТРОЛЬНЫЙ ПРИМЕР И ОПИСАНИЕ РЕЗУЛЬТАТОВ 10](#_Toc444217924)

[6. СПИСОК ЛИТЕРАТУРЫ 11](#_Toc444217925)

[7. СПИСОК ИЛЛЮСТРАЦИЙ 11](#_Toc444217926)

[ПРИЛОЖЕНИЕ. ИСХОДНЫЙ ТЕКСТ ПРОГРАММЫ 12](#_Toc444217927)

# УСЛОВИЕ ЗАДАЧИ

Необходимо написать приложение для визуализации результатов графематического анализа текстов. Необходимо реализовать открытие файла с результатами, построение диаграммы и таблицы значений по данным из открытого файла, а также сохранение некоторых настроек приложения в файле настроек и чтение их из файла. Приложение должно быть реализовано на платформе Windows Forms.

Полный текст условия задачи можно найти в LMS.

Вариант задачи: 8.

# ФУНКЦИИ РАЗРАБАТЫВАЕМОГО ПРИЛОЖЕНИЯ

## Варианты использования

Приложение можно использовать для визуализации и анализа результатов графематического анализа текста.

## Описание интерфейса пользователя

### Вид приложения при запуске

При запуске приложения окно программы имеет следующий вид (рис.1):

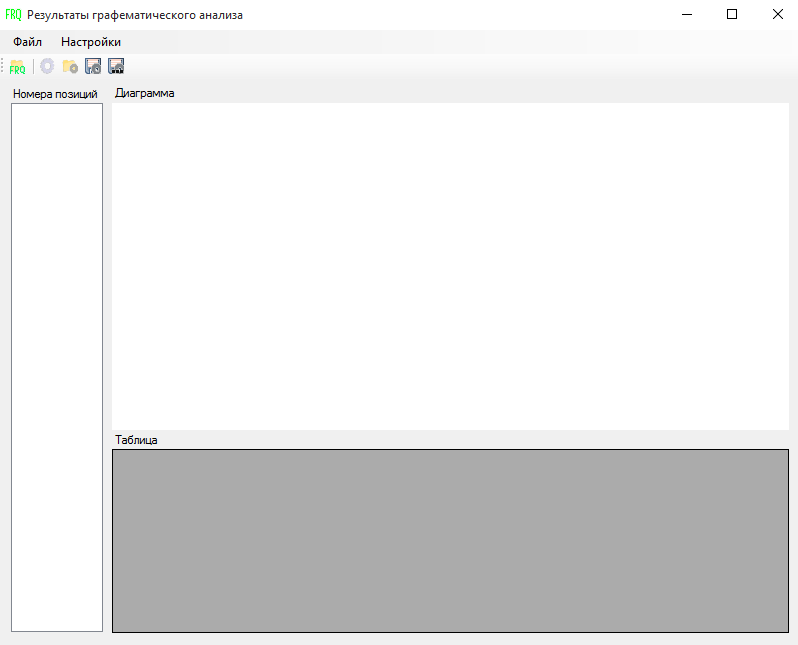


Рисунок 1 - Вид приложения при запуске

В данный момент список «Номера позиций», диаграмма «Диаграмма» и таблица «Таблица» пустые. Для того, чтобы в перечисленных элементах появились данные, необходимо открыть файл с данными (результатами графематического анализа).

### Действия для открытия файла с данными

Для того, чтобы открыть файл с результатами графематического анализа, необходимо выполнить одну из следующих последовательностей действий:

1. Нажать кнопку («открыть»), затем выбрать файл формата frq в диалоговом окне
2. Открыть меню «Файл» - «Открыть», затем выбрать файл формата frq в диалоговом окне

Если данные в открытом файле содержатся и представлены в корректном формате, эти данные будут отображены в программе; противном случае, будет выведено диалоговое окно с сообщением об ошибке.

### Представление данных

После открытия файла в списке «номера позиций» отобразятся имеющиеся в файле позиции букв в слове. Выберите интересующие вас позиции для отображения. Данные отобразятся в программе так, как показано на рис. 2.

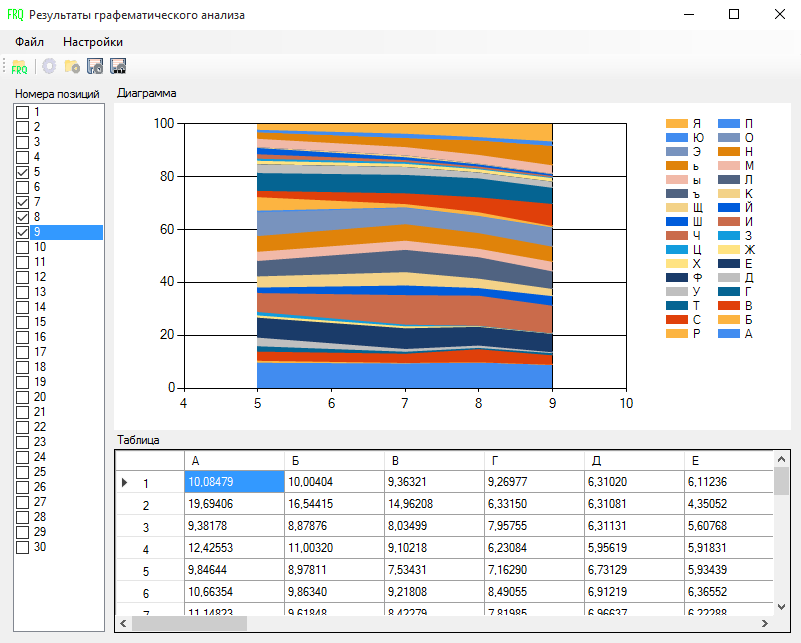


Рисунок 2 - Представление данных

Количество позиций, которые отобразятся в списке «Номера позиций», не превышает количество позиций, указанное в настройках. По умолчанию (при запуске программы) это количество равно 30.

Данные отображены на диаграмме в формате многослойной линейной диаграммы, где на горизонтальной оси стоит позиция буквы в слове, каждый слой соответствует одной букве, толщина слоя зависит от частоты встречаемости этой буквы в данной позиции. Диаграмма построена, исходя из предположения о том, что сумма всех частот встречаемости букв в процентах в данной позиции равна 100. Тогда доля толщины слоя в некоторой позиции от общей толщины всех слоёв – это доля, которая приходится на букву, соответствующую этому слою в этой позиции. Если частоты встречаемости всех букв в некоторой выбранной позиции равны нулю, то все слои в этой позиции будут иметь одинаковую толщину, слои будут равномерно распределены по высоте диаграммы.

Стоит отметить, что на диаграмме верно отображаются частоты встречаемости букв только для тех позиций, которые отмечены в списке. Как правило, если позиция отмечена в списке, то на диаграмме в этой позиции будет перегиб всех слоёв. Если в списке не отметить одну или несколько позиций между двумя отмеченными позициями, то на диаграмме толщина слоёв для всех этих позиций будет рассчитываться линейно между значениями в отмеченных позициях (например, как это можно видеть на рис.2, где между отмеченными позициями 5 и 7 есть неотмеченная позиция 6).

В таблицу «Таблица» выведены значения частот встречаемости букв для каждой из имеющихся (не обязательно отмеченных) в списке «Номера позиций» позиций. Точность вывода значений в эту таблицу (количество знаков после запятой) задано в настройках. По умолчанию (при запуске программы) это 5.

### Работа с настройками

Для того, чтобы открыть настройки программы, необходимо нажать кнопку («настройки») или открыть меню «Настройки» - «Открыть настройки».

После этого откроется окно настроек, которое выглядит как показано на рис. 3.

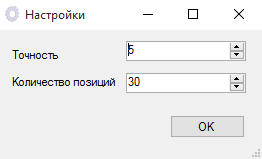


Рисунок 3 - Вид настроек

В поле «точность» вводится значение точности вывода в таблицу (количество знаков после запятой). В поле «Количество позиций» вводится количество позиций для вывода в список и в таблицу.

Для того, чтобы установить заданные настройки, нажмите «OK»; если вы не хотите, чтобы введенные вами настройки были заданные в программе, закройте окно настроек.

Заданные настройки можно сохранить, для этого нажмите кнопку («Сохранить настройки») или откройте меню «Настройки» - «Сохранить настройки». Если вы ещё ни разу не открывали настройки из файла или не сохраняли их в файл, программа предложит вам выбрать путь для сохранения через диалоговое окно «Сохранение» - выберите, куда следует сохранить ваши настройки; в противном случае настройки будет сохранены туда куда они сохранялись в последний раз.

Также имеется возможность сохранить настройки, выбрав файл, в который следует сохранять, вне зависимости от того, открывали ли вы настройки из файла или сохраняли их. Для этого нажмите кнопку («Сохранить настройки как») или откройте меню «Настройки» - «Сохранить настройки как».

Для того, чтобы открыть настройки из файла, нажмите кнопку («Открыть файл с настройками») или откройте меню «Настройки» - «Открыть файл с настройками». В открывшемся диалоговом окне «Открытие» выберите файл с настройками. Если формат выбранного файла корректен, в программе будут установлены настройки из выбранного файла, в противном случае, будет выведено диалоговое окно с сообщением об ошибке.

# СТРУКТУРА ПРИЛОЖЕНИЯ

## Диаграмма классов

### 3.1.1 Диаграмма классов проекта Homework3.Model

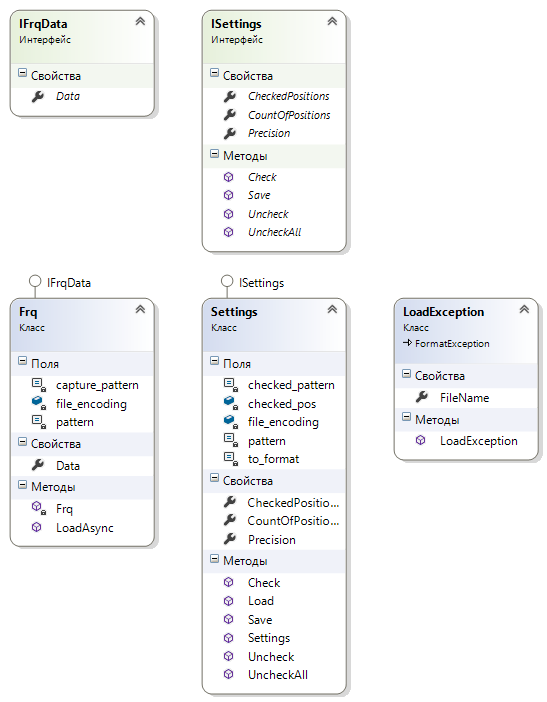


Рисунок 4 - Диаграмма классов проекта Homework3.Model

### 3.1.2 Диаграмма классов проекта Homework3

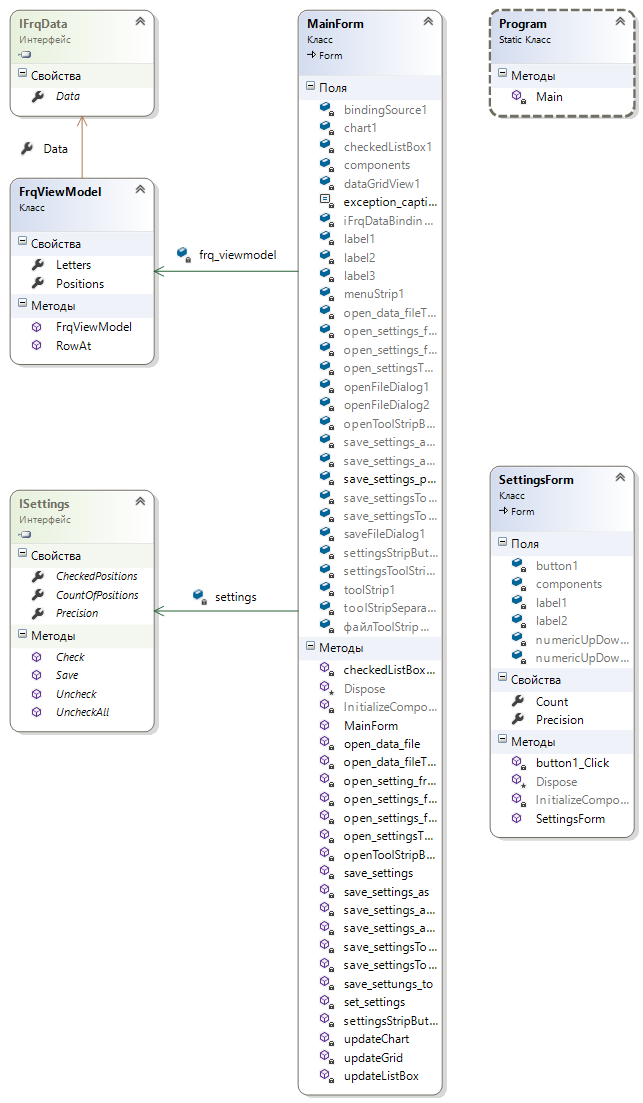


Рисунок 5 - Диаграмма классов проекта Homework3

## Описание классов, их полей, методов и методов

Описание классов, их полей, методов и свойств, содержится в xml-документации к исходному коду программы, которая создается при написании комментариев в специальном формате; описание вспомогательных методов и свойств содержится в обычных комментариях к этим полям, методам и свойствам в исходном коде программы. Исходный код программы содержится в *Приложении.*

# РАСПРЕДЕЛЕНИЕ ИСХОДНОГО КОДА ПО ФАЙЛАМ ПРОЕКТА

Функционал каждого файла проекта описан в комментариях в самой верхней части файла. Исходный код программы содержится в приложении.

# КОНТРОЛЬНЫЙ ПРИМЕР И ОПИСАНИЕ РЕЗУЛЬТАТОВ

Запустите программу. Откройте файл, данный в качестве примера к заданию («all\_2.frq»). Задайте в настройках «Точность» 2, «Количество позиций» 20. В списке «Номера позиций», а также в Таблице должны отобразиться позиции с 1й по 20ю. Все значения в таблице должны иметь не больше двух знаков после запятой.

Выберите в списке «Номера позиций» позиции 3, 5, 8, 9. Представление должно иметь вид, как на рис. 4.

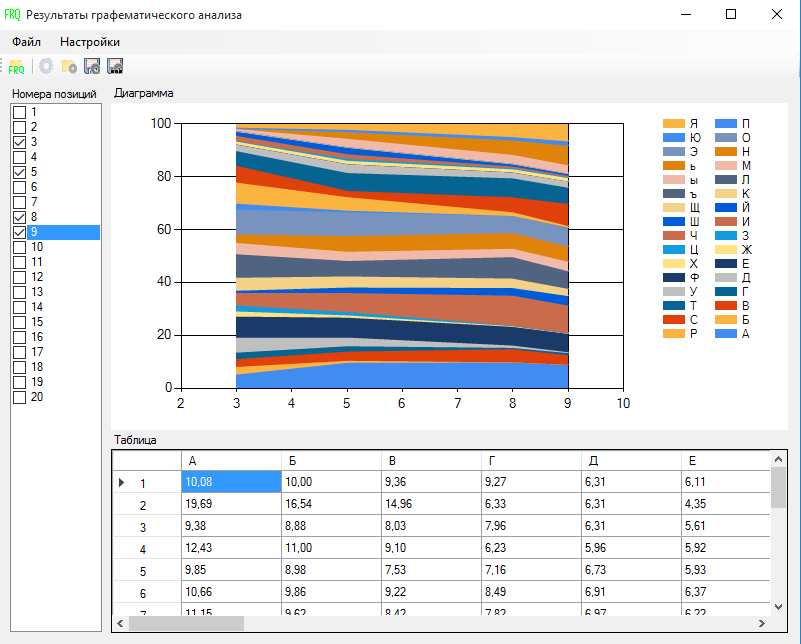


Рисунок 6 - Представление результатов контрольного примера

Теперь сохраните настройки в некоторый файл. Закройте программу и запустите её заново. Откройте настройки из файла, в который вы их только что сохранили, затем, откройте настройки. В открывшемся окне настроек значения должны следующие: «Точность»: 2; «Количество позиций»: 20.

# СПИСОК ЛИТЕРАТУРЫ

x

|  |  |
| --- | --- |
|  | LMS HSE [Электронный ресурс] // LMS: [сайт]. [2016]. URL: http:/​/​lms.hse.ru/ (дата обращения: 25.02.2016.февраль.2016). |
|  | Windows Forms [Электронный ресурс] // MSDN: [сайт]. URL: https:/​/​msdn.microsoft.com/​ru-ru/​library/​dd30h2yb(v=vs.110).aspx (дата обращения: 24.02.2016.февраль.2016). |
|  | Петцольд Ч. Программирование с использованием Microsoft Windows Forms. Русская Редакция, Питер, 2006. 432 pp. |
|  | Фаулер М. UML Основы. 3rd ed. Санкт-Петербург: Символ, 2005. 192 pp. |
|  | Шилдт Г. C#: полное руководство. Москва, Санкт-Петербург, Киев: Издательский дом "Вильямс", 2011. 1056 pp. |

x

# СПИСОК ИЛЛЮСТРАЦИЙ

[Рисунок 1 - Вид приложения при запуске 3](#_Toc444215977)

[Рисунок 2 - Представление данных 4](#_Toc444215978)

[Рисунок 3 - Вид настроек 5](#_Toc444215979)

[Рисунок 4 - Диаграмма классов проекта Homework3.Model 7](#_Toc444215980)

[Рисунок 5 - Диаграмма классов проекта Homework3 8](#_Toc444215981)

[Рисунок 6 - Представление результатов контрольного примера 10](#_Toc444215982)

# ПРИЛОЖЕНИЕ. ИСХОДНЫЙ ТЕКСТ ПРОГРАММЫ

Файл Homework3.Model/IFrqData.cs

//

// Дисциплина: Программирование

// Студент: Мачнев Алексей

// Работа: Контрольное Домашнее задание за 3й модуль

// Дата написаниия: 24 февраля 2016 г.

// Файл содержит описание интерфейса IFrqData

//

using System.Collections.Generic;

namespace Homework3.Model

{

/// <summary>

/// Интерфейс описывает результаты графематического анализа текстов

/// </summary>

public interface IFrqData

{

/// <summary>

/// Возвращает коллекцию значений

/// результотов графематического анализа

/// где для каждой позиции в слове определены

/// частоты каждой буквы в данной позиции

/// Ключ - позиция в слове

/// Значение - коллекция буква - частота встречаемости в данной позиции

/// </summary>

Dictionary<uint, Dictionary<char, decimal>> Data { get; }

}

}

Файл Homework3.Model/Frq.cs

//

// Дисциплина: Программирование

// Студент: Мачнев Алексей

// Работа: Контрольное Домашнее задание за 3й модуль

// Дата написаниия: 24 февраля 2016 г.

// Файл содержит описание класса Frq

//

using System.Collections.Generic;

using System.Text;

using System.Threading.Tasks;

using System.IO;

using System.Text.RegularExpressions;

namespace Homework3.Model

{

/// <summary>

/// Класс является реализацией интерфейса IFrqData

/// </summary>

public class Frq : IFrqData

{

// Шаблон записи, описывающей одну позицию букв в слове

const string pattern = @"Позиция\s\*№\s\*(\d+)\s\*[\r\n]+\s\*Буква\s\*%\s\*[\r\n]+(\s\*\w\s+\d+[\.,]?\d\*[\r\n]+)+";

// Шаблон записи, описывающей одну букву в данной позиции

const string capture\_pattern = @"(\w)\s+(\d+[\.,]?\d\*)[\r\n]+";

// Кодировка файла

static readonly Encoding file\_encoding = Encoding.GetEncoding("windows-1251");

private Frq(Dictionary<uint, Dictionary<char, decimal>> data)

{

this.Data = data;

}

/// <summary>

/// Возвращает коллекцию значений

/// результотов графематического анализа

/// где для каждой позиции в слове определены

/// частоты каждой буквы в данной позиции

/// Ключ - позиция в слове

/// Значение - коллекция буква - частота встречаемости в данной позиции

/// </summary>

public Dictionary<uint, Dictionary<char, decimal>> Data { get; private set; }

/// <summary>

/// Выполняет асинхронное чтение данных из файла, путь к которому указан в path

/// </summary>

/// <param name="path">Путь к файлу с данными</param>

/// <returns>Объект класса Frq, представляющий результаты графематического анализа</returns>

public static async Task<Frq> LoadAsync(string path)

{

Frq frq = new Frq(new Dictionary<uint, Dictionary<char, decimal>>());

using (StreamReader reader = new StreamReader(path, file\_encoding))

{

string content = await reader.ReadToEndAsync();

if (!(await Task.Run(() => Regex.IsMatch(content, pattern, RegexOptions.IgnoreCase))))

{

throw new LoadException("Неправильный формат файла 'frq'", path);

}

// Ищем и обрабатываем записи о каждой позиции в слове

foreach (Match m in await Task.Run(() => Regex.Matches(content, pattern, RegexOptions.IgnoreCase)))

{

// Номер позиции в слове

uint num = uint.Parse(m.Groups[1].Value);

var \_dict\_for\_num = new Dictionary<char, decimal>();

// Обрабатываем записи о каждой букве в данной позиции

foreach (Capture c in m.Groups[2].Captures)

{

Match capt\_m = Regex.Match(c.Value, capture\_pattern);

// Буква

char letter = capt\_m.Groups[1].Value[0];

// Процент встречаемости в позиции num

decimal freq = decimal.Parse(capt\_m.Groups[2].Value);

\_dict\_for\_num.Add(letter, freq);

}

frq.Data.Add(num, \_dict\_for\_num);

}

}

return frq;

}

}

}

Файл Homework3.Model/ISettings.cs

//

// Дисциплина: Программирование

// Студент: Мачнев Алексей

// Работа: Контрольное Домашнее задание за 3й модуль

// Дата написаниия: 24 февраля 2016 г.

// Файл содержит описание интерфейса ISettings

//

using System.Collections.Generic;

using System.Threading.Tasks;

namespace Homework3.Model

{

/// <summary>

/// Интерфейс описывает настройки программы

/// </summary>

public interface ISettings

{

/// <summary>

/// Должен возвращать количество позиций для отображения в списке

/// </summary>

uint CountOfPositions { get; set; }

/// <summary>

/// Должен возвращать список выбранных в тексте номеров позиций

/// </summary>

IEnumerable<uint> CheckedPositions { get; }

/// <summary>

/// Сообщает о том, что в списке отмечена позиция pos

/// </summary>

/// <param name="pos">Значение отмеченнной в списке позиции</param>

void Check(uint pos);

/// <summary>

/// Сообщает о том, что в списке снята отметка с позиции pos

/// </summary>

/// <param name="pos">Значение позиции, с которой снята отметка</param>

void Uncheck(uint pos);

/// <summary>

/// Сообщает о том, что в списке сняты отметки со всех позиций

/// </summary>

void UncheckAll();

/// <summary>

/// Возвращает количество знаков после запятой при выводе значений

/// </summary>

uint Precision { get; set; }

/// <summary>

/// Выполняет асинхронное сохранение настроек в файл по пути path

/// </summary>

/// <param name="path">Путь к файлу, в котором нужно сохранить настройки</param>

/// <returns></returns>

Task Save(string path);

}

}

Файл Homework3.Model/Settings.cs

//

// Дисциплина: Программирование

// Студент: Мачнев Алексей

// Работа: Контрольное Домашнее задание за 3й модуль

// Дата написаниия: 24 февраля 2016 г.

// Файл содержит описание класса Settings

//

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.IO;

using System.Text.RegularExpressions;

using System;

namespace Homework3.Model

{

/// <summary>

/// Класс является реализацией интерфейса ISettings

/// </summary>

public class Settings : ISettings

{

// Шаблон записи, описывающий все настройки

const string pattern = @"\s\*Точность:\s\*(\d+)\s\*[\r\n]+\s\*Позиций:\s\*(\d+)\s\*[\r\n]+\s\*Отмеченные\s\*позиции:(\s+\d+)\*";

// Шаблон записи об одной выбранной в списке позиции

const string checked\_pattern = @"\s+(\d+)";

// Кодировка файла

static readonly Encoding file\_encoding = Encoding.GetEncoding("windows-1251");

// Строка форматирования для сохранения настроек

const string to\_format = "Точность: {0}\nПозиций: {1}\n Отмеченные позиции: {2}";

/// <summary>

/// Выполняет асинхронную загрузку настроек из файла

/// </summary>

/// <param name="path">Путь к файлу</param>

/// <returns>Объект, представляющий настройки</returns>

public static async Task<Settings> Load(string path)

{

Settings settings = new Settings(1, 0);

using (StreamReader reader = new StreamReader(path, file\_encoding))

{

string content = await reader.ReadToEndAsync();

if (!(await Task.Run(() => Regex.IsMatch(content, pattern, RegexOptions.IgnoreCase))))

{

throw new LoadException("Неправильный формат файла настроек", path);

}

foreach(Match m in Regex.Matches(content, pattern, RegexOptions.IgnoreCase))

{

settings.Precision = uint.Parse(m.Groups[1].Value);

settings.CountOfPositions = uint.Parse(m.Groups[2].Value);

foreach(Capture capt in m.Groups[3].Captures)

{

Match capt\_match = Regex.Match(capt.Value, checked\_pattern);

settings.Check(uint.Parse(capt\_match.Groups[1].Value));

}

}

}

return settings;

}

private List<uint> checked\_pos;

/// <summary>

/// Создаёт объект этого класса с пустым списком отмеченных в списке позиций

/// </summary>

/// <param name="count\_of\_positions">Количество позиций для отображения в списке</param>

/// <param name="precision">Количество знаков после запятой при выводе значений</param>

public Settings(uint count\_of\_positions, uint precision)

{

this.CountOfPositions = count\_of\_positions;

this.Precision = precision;

CheckedPositions = new List<uint>();

}

/// <summary>

/// Возвращает или задаёт список отмеченных в списке позиций

/// </summary>

public IEnumerable<uint> CheckedPositions

{

get { return checked\_pos.OrderBy(p => p); }

set { this.checked\_pos = new List<uint>(value); }

}

/// <summary>

/// Возвращает или задаёт количество позиций для отображения в списке

/// </summary>

public uint CountOfPositions { get; set; }

/// <summary>

/// Возвращает или задаёт количество знаков после запятой при выводе значений

/// </summary>

public uint Precision { get; set; }

/// <summary>

/// Сообщает о том, что в списке выбрана позиция num

/// </summary>

/// <param name="pos">Номер позиции, которая выбрана</param>

public void Check(uint pos)

{

checked\_pos.Add(pos);

}

/// <summary>

/// Выполняет асинхронное сохранение настроек в файл по пути path

/// </summary>

/// <param name="path">Путь к файлу</param>

/// <returns></returns>

public async Task Save(string path)

{

string pos\_arr = "";

foreach(var pos in CheckedPositions)

{

pos\_arr = string.Format("{0} {1}", pos\_arr, pos);

}

string to\_save = string.Format(to\_format, this.Precision, this.CountOfPositions, pos\_arr);

using (StreamWriter writer = new StreamWriter(File.Open(path, FileMode.Create, FileAccess.Write), file\_encoding))

{

await writer.WriteAsync(to\_save);

}

}

/// <summary>

/// Сообщает о том, что в списке снята отметка с позиции pos

/// </summary>

/// <param name="pos">Значение позиции, с которой снята отметка</param>

public void Uncheck(uint pos)

{

checked\_pos.Remove(pos);

}

/// <summary>

/// Сообщает о том, что в списке сняты отметки со всех позиций

/// </summary>

public void UncheckAll()

{

this.checked\_pos.Clear();

}

}

}

Файл Homework3.Model/LoadException.cs

//

// Дисциплина: Программирование

// Студент: Мачнев Алексей

// Работа: Контрольное Домашнее задание за 3й модуль

// Дата написаниия: 24 февраля 2016 г.

// Файл содержит описание класса ошибки FormatException

//

using System;

namespace Homework3.Model

{

/// <summary>

/// Возникает при попытке чтения из файла, данные в котором хранятся в некорректном формате

/// </summary>

public class LoadException : FormatException

{

/// <summary>

/// Создает это исключение

/// </summary>

/// <param name="message">Сообщение</param>

/// <param name="file\_name">

/// Путь к файлу, попытка чтения которого была осуществлена

/// </param>

public LoadException(string message, string file\_name) : base(message)

{

this.FileName = file\_name;

}

/// <summary>

/// Возвращает путь к файлу, попытка чтения которого была осуществлена

/// </summary>

public String FileName { get; private set; }

}

}

Файл Homework3/ViewModels/FrqViewModel.cs

//

// Дисциплина: Программирование

// Студент: Мачнев Алексей

// Работа: Контрольное Домашнее задание за 3й модуль

// Дата написаниия: 24 февраля 2016 г.

// Файл содержит описание класса FrqViewModel

//

using System.Collections.Generic;

using System.Linq;

using Homework3.Model;

namespace Homework3.ViewModels

{

// Является моделью представления для IFrqData

public class FrqViewModel

{

public FrqViewModel(IFrqData data)

{

this.Data = data;

}

// Возвращает список букв

public IEnumerable<char> Letters

{

get

{

var q = from a in Data.Data.Values

from p in a.Keys

select p;

return q.Distinct().OrderBy(p => char.ToUpperInvariant(p));

}

}

// Возвращает список описанных позиций в слове

public IEnumerable<uint> Positions

{

get

{

return Data.Data.Keys.OrderBy(p => p);

}

}

public IFrqData Data

{

get; set;

}

// Возвращает массив строк, каждая из которых

// является строковым представлением частоты буквы

// для заданной позиции в слове

// с заданной точностью

public string[] RowAt(uint index, uint precision)

{

var q = from p in Data.Data[index].Values

select decimal.Round(p, (int)precision).ToString();

return q.ToArray();

}

}

}

Файл Homework3/MainForm\_Controller.cs

//

// Дисциплина: Программирование

// Студент: Мачнев Алексей

// Работа: Контрольное Домашнее задание за 3й модуль

// Дата написаниия: 24 февраля 2016 г.

// Файл содержит описание контроллера формы MainForm

//

<using Сокращены>

namespace Homework3

{

public partial class MainForm

{

const string exception\_caption = "Ошибка";

private string save\_settings\_path = null;

// Обновляет содержимое списка позиций

private void updateListBox()

{

<Код сокращён>

}

// Обновляет диаграмму

private void updateChart()

{

<Код сокращён>

}

// Обновляет содержимое таблицы

private void updateGrid()

{

<Код сокращён>

}

// Открывает данные из файла

private async Task open\_data\_file()

{

<Код сокращён>

}

// Сохраняет настройки в файл без открытия диалогового окна выбора файла,

// если настройки ранее уже сохранялись или были открыты из файла

private async Task save\_settings()

{

<Код сокращён>

}

// Сохраняет настройки в файл, полученный из диалогового окна

private async Task save\_settings\_as()

{

<Код сокращён>

}

// Сохраняет настройки в файл по пути path

private async Task save\_settungs\_to(string path)

{

<Код сокращён>

}

// Открывает настройки из файла

private async Task open\_setting\_from\_file()

{

<Код сокращён>

}

// Открывает диалоговое окно с настройками

private void set\_settings()

{

<Код сокращён>

}

}

}