НИУ ВШЭ

Кафедра

Корпоративных информационных систем

Командный проект
по дисциплине
«Программирование»

StateExamVariantsGenerator

Выполнили студенты группы ББИ 155:

Борцова Полина

Мишакин Николай

Соболева Александра

Руководитель:

Ефремов Сергей Геннадьевич

Москва, 2016 год

Содержание

1. Краткая аннотация проекта

1.1. Название проекта

StateExamVariantsGenerator

1.2. Основные характеристики проекта

Данное приложение позволяет составлять варианты ЕГЭ по математике (как профильные, так и базовые), случайным образом комбинируя задания из следующих открытых банков:

- http://mathege.ru/
- http://alexlarin.net/ege/
- http://base.mathege.ru/
- https://mathb-ege.sdamgia.ru/

Школьники с помощью данного приложения смогут готовиться к ЕГЭ по математике, составляя полные экзаменационные варианты или выбирая свой индивидуальный набор задач (например, тех, которые у них хуже всего получаются). Приложение также оснащено функцией записи выбранных заданий в PDF файл, что позволит учителям и репетиторам быстро формировать раздаточный материал для проведения занятий.

1.3. Основной функционал приложения

1. Составление полного экзаменационного варианта, случайным образом комбинируя задания из открытых банков
2. Выбор уровня сложности заданий (базовый или профильный)
3. Составление варианта вручную (возможность выбора определенных заданий и их количества)
4. Запись составленного варианта в PDF файл

1.4. Адрес удаленного репозитория на https://github.com/

https://github.com/Polina2711/VariantsGenerator

2. Участники команды и их роли

Николай Мишакин

1. Создание функционала для считывания и записи id всех заданий из открытых банков в текстовый файл (решение **GetTasksIdFromSite**)
2. Установка во всех окнах, используемых приложением, сетки grid
3. Разработка методов, позволяющих по id задания или прямой ссылке на него получить текст задания (класс **ParserFromMathege.cs**)
4. Работа с исключениями
5. Тестирование приложения

Полина Борцова

1. Дизайн WPF
2. Написание кода для кнопок, связанных с составлением полного базового варианта и базовой части ручной сборки варианта
3. Написание кода для кнопок, осуществляющих запись в PDF файл
4. Разработка интерфейсов **IVariantGenerator.cs** и **IOutputFormer.cs** и вспомогательного класса **Factory.cs.** Перестройка работы классов **VariantGenerator.cs, BaseVariantGenerator.cs, OutputFormer.cs** и **BaseOutputFormer.cs** с учетом появившихся интерфейсов

Александра Соболева

1. Разработка классов **BaseOutputFormer.cs, OutputFormer.cs**, **BasePDFWriter.cs, PDFWriter.cs, BaseVariantGenerator.cs, VariantGenerator.cs** и интерфейса **IPDFWriter.cs**
2. Написание кода для кнопок, связанных с составлением полного профильного варианта и профильной части ручной сборки варианта
3. Предотвращение некоторых возможных исключений с помощью блокировки кнопок
4. Работа с делегатами

3. Перечень классов с кратким описанием

Проект приложения состоит из трех решений:

1. *GetTaskIdFromSite*

GetNumber.cs

Основная задача класса – получение и запись id всех заданий (базового уровня – метод **GetNumberOfBaseTask**, профильного уровня – **GetNumberOfTasks**) из открытых банков в текстовый файл

Program.cs

Содержит в себе основной метод Main.

2. StateExam.UI

Данное решение включает в себя окна WPF, каждое из которых посредством соответствующих классов связано с логикой программы (**StateExamVariants.Data**)

StartWindow.xaml.cs (code behind file)

Класс, отвечающий за стартовое окно программы.

Данный класс обеспечивает:

- составление базового/профильного полного экзаменационного варианта;
- составление варианта вручную.

Связывает пользовательский интерфейс со следующими классами логики: **ParserFromMathege.cs**, через интерфейс **IVariantGenerator.cs** с классами **VariantGenerator.cs** и **BaseVariantGenerator.cs** (в зависимости от уровня сложности составляемого варианта), вызывает окно **MainWindow.xaml**

ALVariant.xaml.cs (code behind file)

Класс, отвечающий за окно со сгенерированным вариантом профильного уровня (как вручную, так и нет).

Работа с кнопками (=номерами заданий), при нажатии которых происходит вызов окна **Exercise.xaml**, содержащего условие соответствующей задачи.

Работа с кнопкой, добавляющей составленный вариант в PDF файл.

Использует метод **Outputer** для минимизации повторяющихся строк кода

Связывает пользовательский интерфейс со следующими классами логики: через интерфейс **IOutputFormer.cs** с классом **OutputFormer.cs**, через интерфейс **IVariantGenerator.cs** с классом **VariantGenerator.cs,** осуществляется подписка на событие **GetValueFromDict**.

BLVariant.xaml.cs (code behind file)

Класс, отвечающий за окно со сгенерированным вариантом базового уровня (как вручную, так и нет).

Работа с кнопками (=номерами заданий), при нажатии которых происходит вызов окна **Exercise.xaml***,* содержащего условие соответствующей задачи.

Работа с кнопкой, добавляющей составленный вариант в PDF файл.

Использует метод **Outputer** для минимизации повторяющихся строк кода

Связывает пользовательский интерфейс со следующими классами логики: через интерфейс **IOutputFormer.cs** с классом **BaseOutputFormer.cs**, через интерфейс **IVariantGenerator.cs** с классом **BaseVariantGenerator.cs,** осуществляется подписка на событие **GetValueFromDict**.

MainWindow.xaml.cs (code behind fie)

Класс, отвечающий за окно, содержащее перечень всех заданий (1-20), возможность выбора их количества в формируемом варианте и возможность выбора уровня сложности (базовый или профильный).

Формирует вариант, на основе выбранных учащимся заданий, их количества и уровня сложности. Предотвращает возможное исключение благодаря блокировке кнопки “Generate” до того, пока не будут выбраны все, необходимые для дальнейшей работы программы функции. Вызывает одно и окон: **BLVariant.xaml** или **ALVariant.xaml** (в зависимости от выбранного уровня сложности). Блокирует в вызванном окне кнопки невыбранных заданий.

Связывает пользовательский интерфейс со следующими классами логики: **ParserFromMathege.cs**, через интерфейс **IVariantGenerator.cs** с классами **VariantGenerator.cs** и **BaseVariantGenerator.cs** (в зависимости от уровня сложности составляемого варианта), осуществляется подписка на событие **GetTaskProblem**

Exercise.xaml.cs (code behind file)

Не выполняет никаких дополнительных функций, кроме автоматически созданных при создании окна.

3. StateExamVariants.Data

IVariantGenerator.cs

Интерфейс, наследующий классы **VariantGenerator.cs** и **BaseVariantGenerator.cs**

VariantGenerator.cs

Основные элементы данного класса:

- событие **GetTaskProblem**,
- словарь **GetDict**, хранящий номер задания (1-20) и условия задачи,
- объект класса Random,
- метод **Generator**, осуществляющий случайный выбор id задания из заранее сформированных текстовых файлов и записывающий номер задания и соответствующее условие(-я) в словарь,
- метод **GetDict**, возвращающий словарь,
- метод **GetValueByKeyFromDict**, возвращающий условие(-я) задачи по принятому номеру задания,
- метод **ClearDict**, очищающий словарь.

BaseVariantGenerator.cs

Аналогичный предыдущему класс для составления базовых вариантов.

IOutputFormer.cs

Интерфейс, наследующий классы **BaseOutputFormer.cs** и **OutptutFormer.cs**

OutputFormer.cs

Основные элементы данного класса:

- событие **GetValueFromDict**,
- метод **FormOutputText**, принимающий номер задания и возвращающий соответствующий список, содержащий условие заданий для последующего вывода

BaseOutputFormer.cs

Аналогичный класс для формирования выходной информации по задачам базового уровня

Factory.cs

Класс, связывающий объекты интерфейса с объектами соответствующих классов

IPDFWriter.cs

Интерфейс, наследующий классы **BasePDFWriter.cs** и **PDFWtiter.cs**

PDFWtiter.cs

Основные элементы данного класса:

- событие **GetAllDict**
- метод **AddToPDFFile**, принимающий путь к файлу, записывающий составленный вариант в PDF файл

BasePDFWriter.cs

Аналогичный класс для записи в PDF файл вариантов базового уровня.

ParserFromMathege.cs

Основные элементы данного класса:

- метод **GetTask**, возвращающий условие задания по принимаемому id задания в открытом банке профильных заданий
- метод **GetBaseTask**, возвращающий условие задания по принимаемому id задания в открытом банке базовых заданий

4. Пользовательский интерфейс программы и работа приложения

Приложение ***StateExamVariantsGenerator*** имеет несколько основных функций, которые изложены в следующем кратком описании:

Когда пользователь запускает программу, перед ним появляется окно (рис.1) с тремя активными кнопками для:

- “Generate base variant” - составления варианта базового уровня
- “Generate advanced variant” - составления варианта профильного уровня
- “Generate variant manually” - составления варианта вручную.

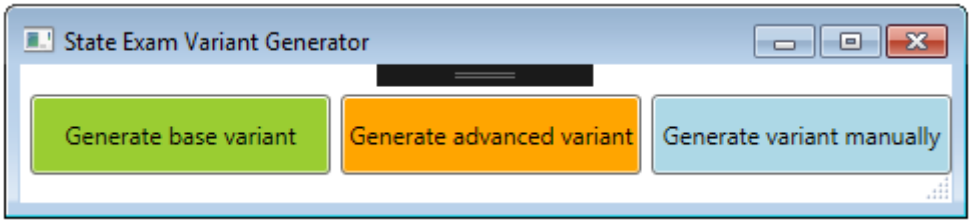

Рис.1

При нажатии кнопки “Generate base variant” или “Generate advanced variant” открывается соответственно окно Base Level Variant (рис. 2.1) или Advanced Level Variant (рис 2.2)

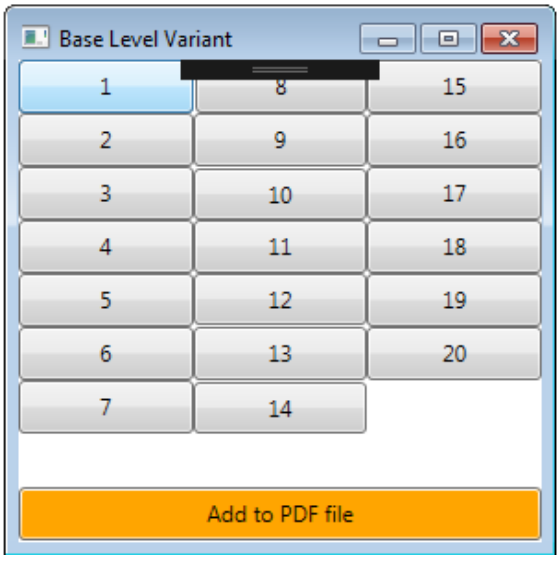

Рис.2.1

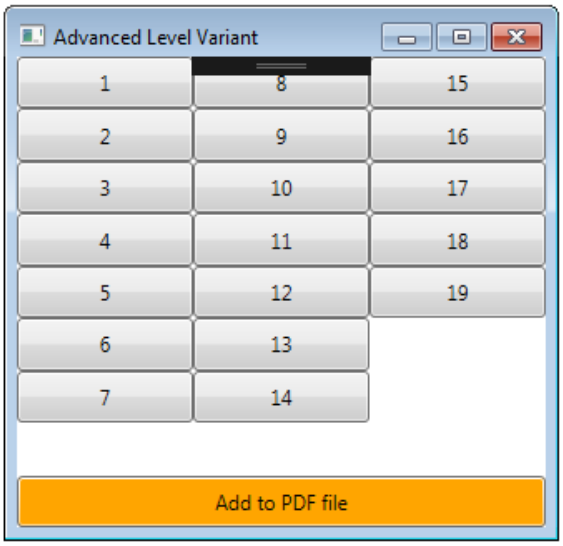

Рис.2.2

При нажатии на кнопку с номером задания на любом из этих окон, открывается окно Tasks, содержащее номер и условие задания (рис 2.3)

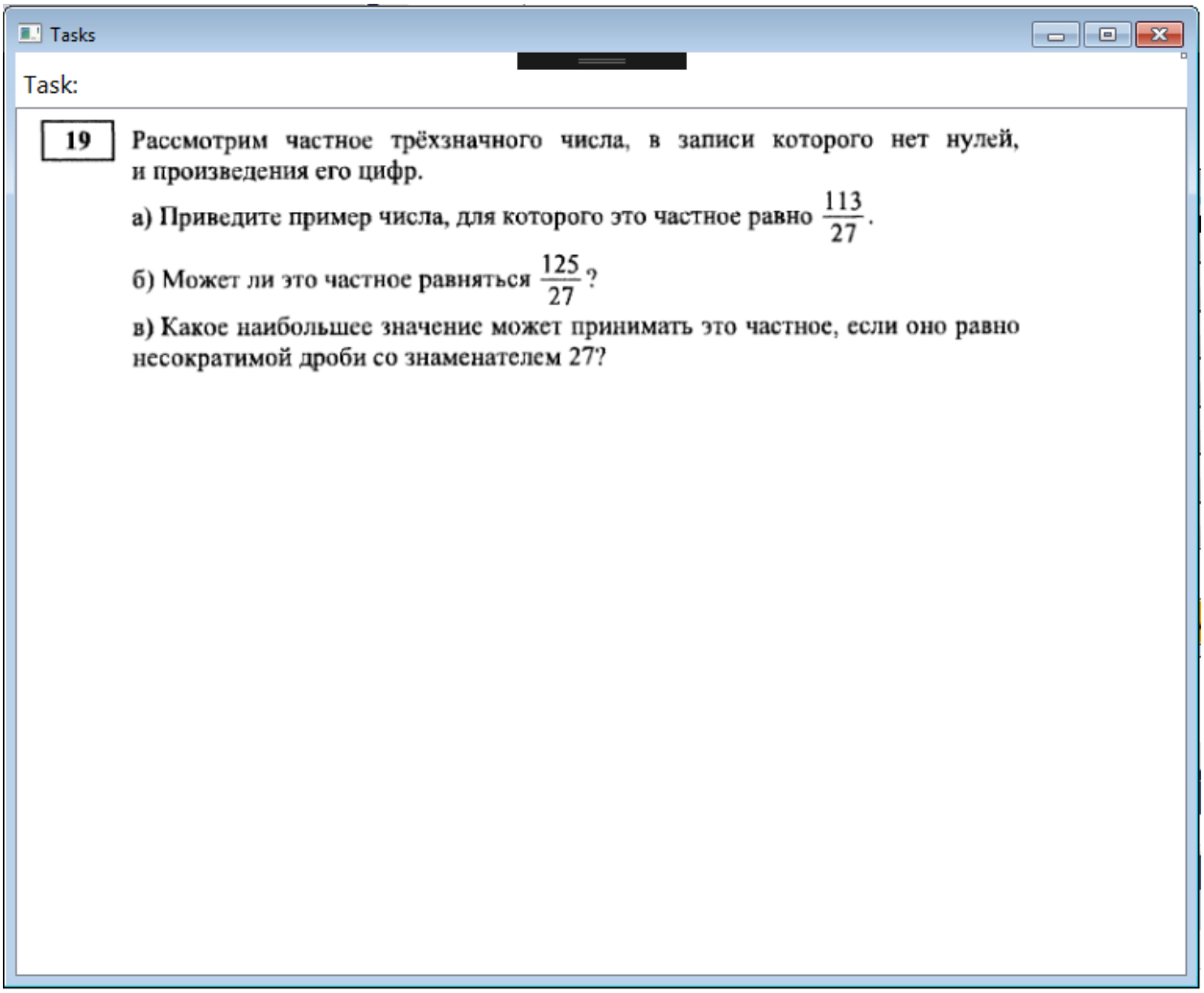

Рис.2.3

При нажатии на кнопку “Generate variant manually”, открывается окно Manual generator (рис 3.1)

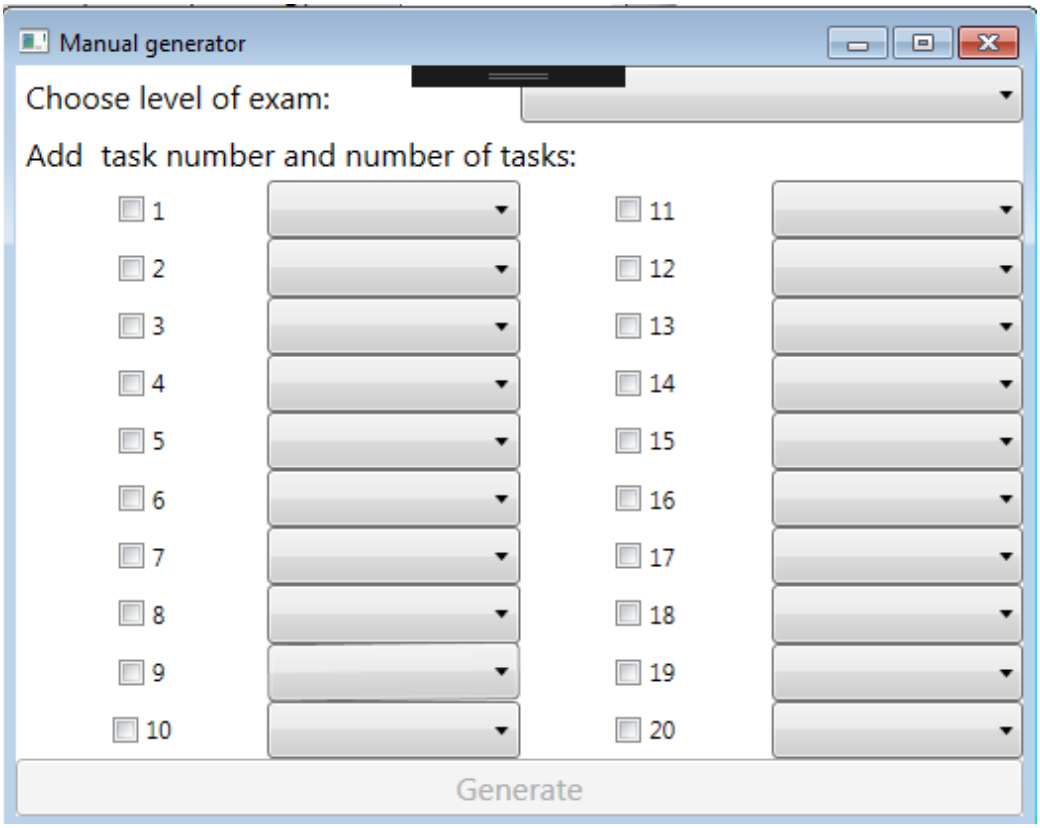

Рис. 3.1

После выбора пользователем уровня сложности варианта, хотя бы одного задания и количества выбранных задания, становится активной кнопка “Generate” (рис 3.2)

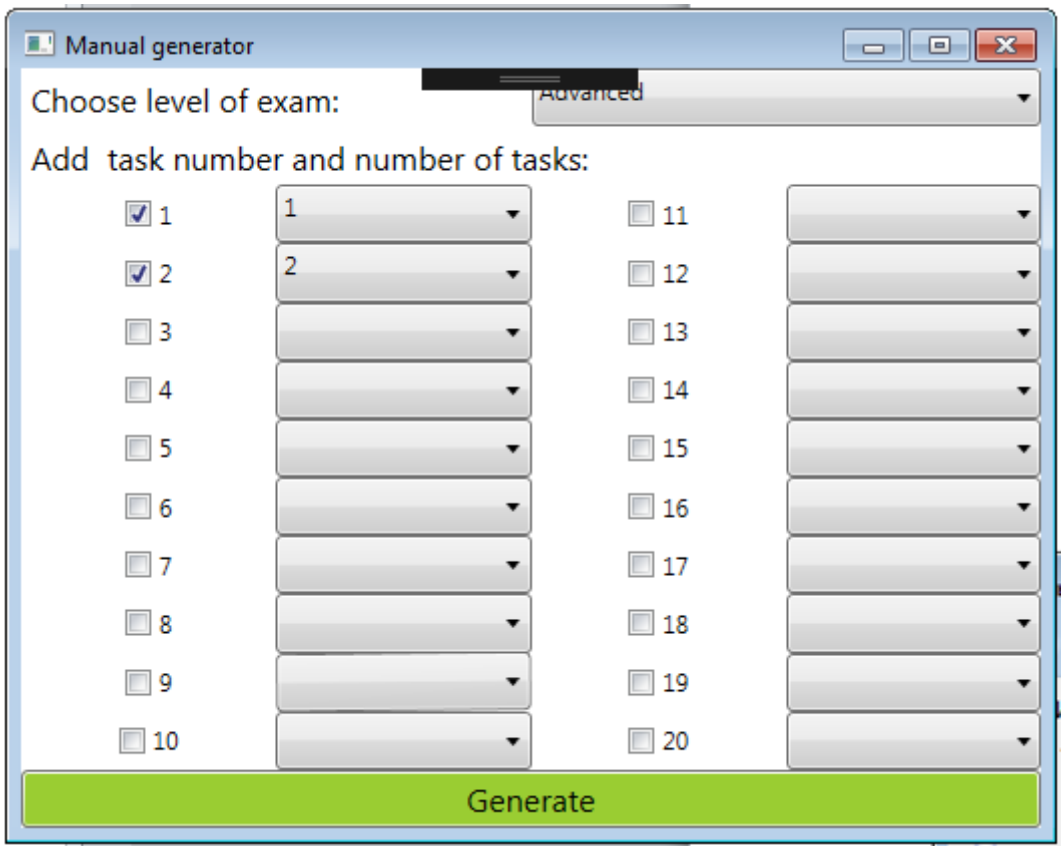

Рис. 3.2

После нажатия на кнопку “Generate” в зависимости от выбранного уровня сложности открывается одно из окон Base Level Variant (рис. 2.1) или Advanced Level Variant (рис 2.2), в которых активными являются только кнопки выбранных пользователем заданий (рис 4.1)

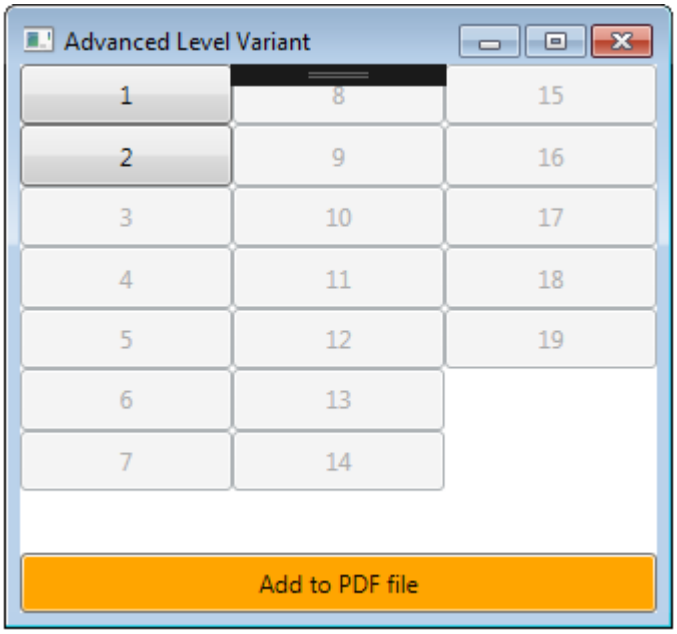

Рис.4.1

Если при выборе заданий было выбрано больше одного, то при нажатии на кнопку с заданием окне Tasks задания с соответствующим номером будут расположены друг за другом (рис. 4.2)

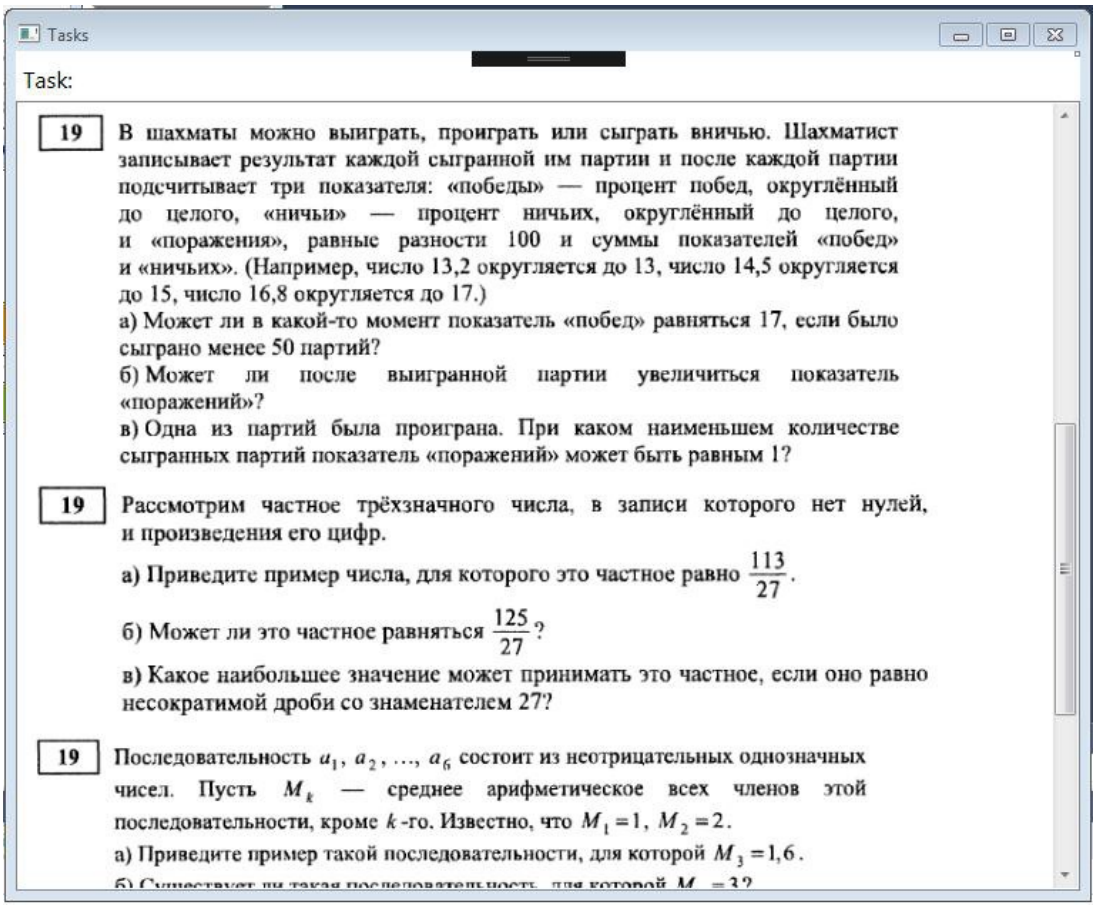

Рис.4.2

При нажатии на кнопку “Add to PDF file” в любом из окон Base Level Variant (рис. 2.1) или Advanced Level Variant (рис 2.2), откроется окно «Сохранить как», позволяющее пользователю выбрать место хранения и название файла (рис. 5.1)

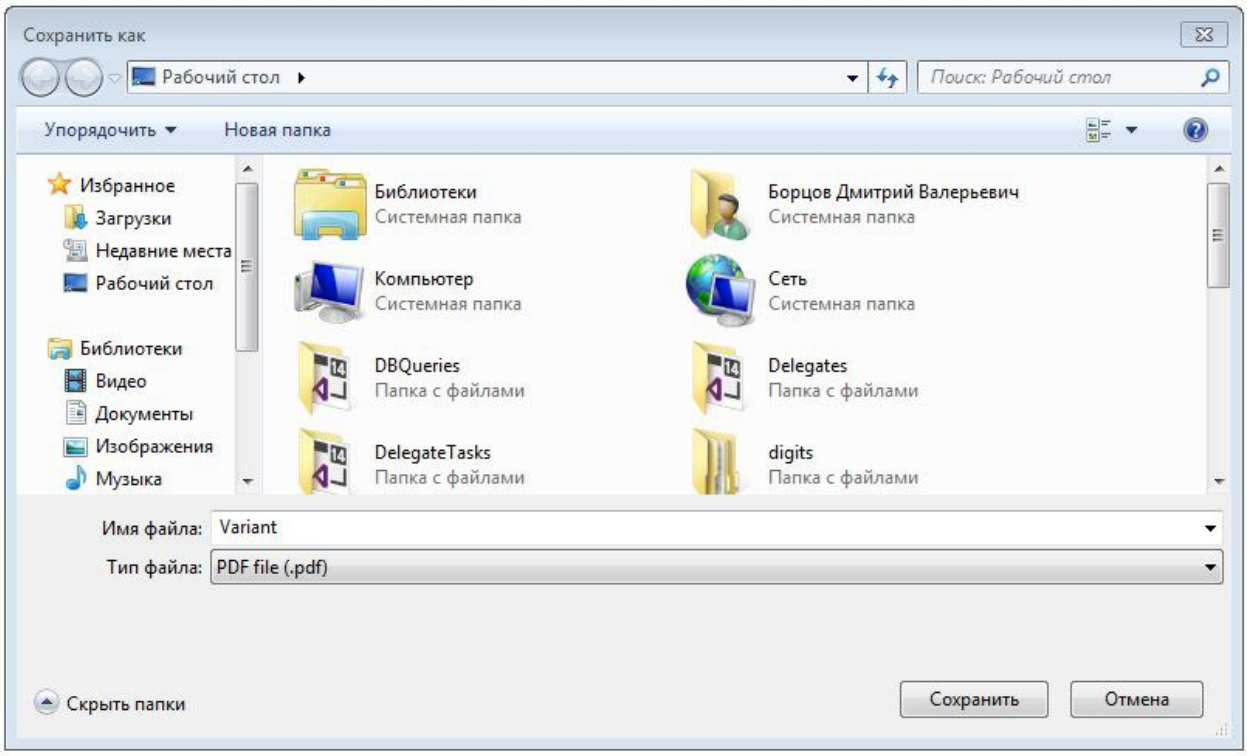

Рис.5.1

После нажатия кнопки «Сохранить», PDF файл будет сохранен в выбранном месте под выбранным названием (рис. 5.2)

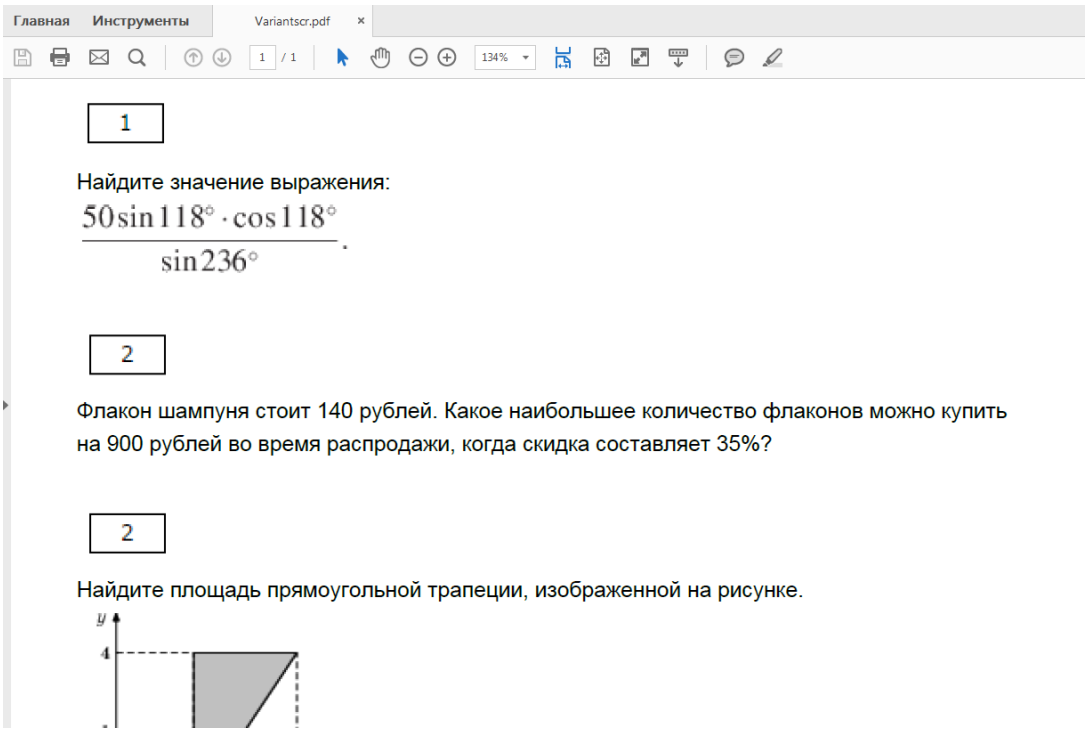

Рис.5.2

5. Примечания

В процессе написания программы был использован код из внешнего источника, а именно:

Решение **StateExam.UI**

Класс **BLVariant.xaml.cs** Класс **ALVariant.xaml.cs**

Строки **152-164** Строки **145-157**

Источник https://msdn.microsoft.com/ru-ru/library/aa969773(v=vs.110).aspx