МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Національний аерокосмічний університет ім. М.Є. Жуковського

«Харківський авіаційний інститут»

Факультет радіотехнічних систем літальних апаратів

Кафедра комп’ютерних систем та мереж

**Курсова робота**

|  |  |
| --- | --- |
| з | *Об’єктно-орієнтованого програмування* |
|  | (назва дисципліни) |
| на тему | *«Генератор документiв про вищу освiту*» |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Виконав: студент | | | *2* | курсу групи № | | | | *525a* |
| напряму підготовки (спеціальності) | | | | | | | | |
| *150102 – комп’ютерна інженерія* | | | | | | | | |
| (шифр і назва напряму підготовки (спеціальності)) | | | | | | | | |
| *Маркевич В.* | | | | | | | | |
| (прізвище й ініціали студента) | | | | | | | | |
| Керівник: | *старший викладач* | | | | | | | |
| *Бабешко Євген Васильович* | | | | | | | | |
| (посада, науковий ступінь, прізвище й ініціали) | | | | | | | | |
| Національна шкала: | | | | |  | | | |
| Кількість балів: | | | | |  | | | |
| Оцінка ECTS: | | | | |  | | | |
| Члени комісії: | |  | | | |  |  | |
|  | | (підпис) | | | |  | (прізвище й ініціали) | |
|  | |  | | | |  |  | |
|  | | (підпис) | | | |  | (прізвище й ініціали) | |
|  | |  | | | |  |  | |
|  | | (підпис) | | | |  | (прізвище й ініціали) | |

Харків – 2017

Міністерство освіти і науки України

Національний аерокосмічний університет ім. М. Є. Жуковського

«Харківський авіаційний інститут»

|  |  |  |
| --- | --- | --- |
| Факультет | *радіотехнічних систем літальних апаратів* | |
|  | (повне найменування) | |
| Кафедра | *комп’ютерних систем та мереж* | |
|  | (повне найменування) | |
| Рівень вищої освіти | | *перший (бакалаврський)* |
|  | | (назва) |
| Напрям підготовки | | *150102 – комп’ютерна інженерія* |
|  | | (код та найменування) |

**ЗАВДАННЯ**

**НА КУРСОВИЙ ПРОЕКТ (РОБОТУ) СТУДЕНТУ**

|  |  |  |
| --- | --- | --- |
| *Маркевич* | | |
| (прізвище, ім’я, по батькові) | | |
| 1. Тема проекту (роботи) | *«Генератор документiв про вищу освiту»* | |
|  | | |
|  | | |
| Керівник проекту (роботи) | | *Бабешко Євген Васильович старший викладач* |
|  | | ( прізвище, ім’я, по батькові, науковий ступінь, вчене звання) |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 2. Строк подання студентом проекту (роботи) | | | *25.05.2017* | |
| 3. Вихідні дані до проекту (роботи) | | *Фамілія, ім’я та по-батькові студенту,* | | |
| *дані про спеціальність, спеціалізацію, отриману кваліфікацію, пройдену учбову* | | | | |
| *програму.* | | | | |
|  | | | | |
|  | | | | |
| 4. Зміст пояснювальної записки (перелік завдань, які потрібно розв’язати) | | | |  |
| *Реферат, Постановка задачі, Проектування, Розробка, Верифікація та* | | | | |
| *тестування, Висновок, Перелік посилань, Додаток А. Технічне завдання,* | | | | |
| *Додаток Б. Вихідні тексти програми, Додаток В. Керівництво Оператора,* | | | | |
| *Додаток Г. Презентація* | | | | |
|  | | | | |
|  | | | | |
| 5. Перелік графічного матеріалу | *діаграма варіантів використання,* | | | |
| *діаграма послідовності, діаграма пакетів, діаграма класів* | | | | |
|  | | | | |
| *.* | | | | |
|  | | | | |

6. Консультанти розділів проекту (роботи)

|  |  |  |  |
| --- | --- | --- | --- |
| Розділ | Прізвище, ініціали та посада  консультанта | Підпис, дата | |
| завдання видав | завдання  прийняв |
| *Основний* | *Бабешко Є. В. старший викладач* | *24.01.2017* | *24.01.2017* |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Нормоконтроль |  | *Бабешко Є. В.* | *«* | *25* | *»* | *05* | *20* | *17* | *р* |
|  | (підпис) | (ініціали та прізвище) |  |  |  |  |  |  |  |

7. Дата видачі завдання « 24 » 01 2017 р.

**КАЛЕНДАРНИЙ ПЛАН**

|  |  |  |  |
| --- | --- | --- | --- |
| №  з/п | Назва етапів курсового  проекту (роботи) | Строк виконання етапів проекту  ( роботи ) | Примітка |
| *1* | *Реферат* | *24.01.2017* |  |
| *2* | *Постанова задачі* | *07.02.2017* |  |
| *3* | *Розробка ТЗ* | *21.02.2017* |  |
| *4* | *Проектування* | *07.03.2017* |  |
| *5* | *Розробка* | *21.03.2017* |  |
| *6* | *Тестування та верифікація* | *04.04.2017* |  |
| *7* | *Висновки* | *18.04.2017* |  |
| *8* | *Розробка керівництва оператора* | *02.04.2017* |  |
| *9* | *Оформлення пояснювальної записки* | *16.04.2017* |  |
| *10* | *Розробка презентації* | *16.04.2017* |  |

|  |  |  |
| --- | --- | --- |
| **Студент** |  | *Маркевич В.* |
|  | (підпис) | (прізвище та ініціали) |
| **Керівник проекту (роботи)** |  | *Бабешко Є.В.* |
|  | (підпис) | (прізвище та ініціали) |

РЕФЕРАТ

Пояснительная записка на курсовой проект: 67 с., 65 ил., 12 табл., 4 приложения, 5 источников.

Данная работа посвящена задаче по созданию генератора документов о высшем образовании.

Целью данной работы является разработка приложения, которое может быть использовано для генерации документов о высшем образовании..

Приложение было реализовано с помощью языка программирования С#. За графический интерфейс отвечал Windows Forms. Для доступа к объектам файла Word использовалась библиотека Microsoft.Interop.Word.

Содержание

[1 ПОСТАНОВКА ЗАДАЧИ 5](#_Toc484445558)

[1.1 Анализ предметной области 5](#_Toc484445559)

[1.2 Анализ существующих решений 9](#_Toc484445560)

[1.3 Постановка задачи 9](#_Toc484445561)

[2 ПРОЕКТИРОВАНИЕ 10](#_Toc484445562)

[2.1 Выбор и обоснование архитектуры 10](#_Toc484445563)

[2.2 Диаграмма прецедентов 11](#_Toc484445564)

[2.3 Диаграмма последовательности 13](#_Toc484445565)

[2.4 Проектирование графического интерфейса 14](#_Toc484445566)

[3 РАЗРАБОТКА 19](#_Toc484445567)

[3.1 Диаграмма пакетов 19](#_Toc484445568)

[3.2 Диаграмма классов 20](#_Toc484445569)

[3.3 Разработка и описание ключевых методов классов 22](#_Toc484445570)

[4 ВЕРИФИКАЦИЯ И ТЕСТИРОВАНИЕ ПРОГРАММЫ 29](#_Toc484445571)

[4.1 Верификация 29](#_Toc484445572)

[4.2 Тестирование программы 33](#_Toc484445573)

[ЗАКЛЮЧЕНИЕ 43](#_Toc484445574)

[ПЕРЕЧЕНЬ ССЫЛОК 44](#_Toc484445575)

[ПРИЛОЖЕНИЕ А. ТЕХНИЧЕСКОЕ ЗАДАНИЕ 45](#_Toc484445576)

[ПРИЛОЖЕНИЕ Б. ИСХОДНЫЕ КОДЫ ПРОГРАММЫ 48](#_Toc484445577)

[ПРИЛОЖЕНИЕ В. РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ 66](#_Toc484445578)

[ПРИЛОЖЕНИЕ Г. ПРЕЗЕНТАЦИЯ 74](#_Toc484445579)

# ПОСТАНОВКА ЗАДАЧИ

## 1.1 Анализ предметной области

Текстовый файл — компьютерный [файл](https://ru.wikipedia.org/wiki/%D0%A4%D0%B0%D0%B9%D0%BB), содержащий [текстовые данные](https://ru.wikipedia.org/wiki/%D0%A2%D0%B5%D0%BA%D1%81%D1%82%D0%BE%D0%B2%D1%8B%D0%B5_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D0%B5). Текстовым файлам противопоставляются [двоичные (бинарные) файлы](https://ru.wikipedia.org/wiki/%D0%94%D0%B2%D0%BE%D0%B8%D1%87%D0%BD%D1%8B%D0%B9_%D1%84%D0%B0%D0%B9%D0%BB), в которых содержатся данные, не рассчитанные на интерпретацию в качестве текстовых (например, файлы, хранящие текст в закодированном или сжатом виде, или хранящие не текст, а звук, изображение или иные данные).

В отличие от термина «[текстовые данные](https://ru.wikipedia.org/wiki/%D0%A2%D0%B5%D0%BA%D1%81%D1%82%D0%BE%D0%B2%D1%8B%D0%B5_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D0%B5)» («текстовый формат данных»), характеризующего содержимое данных, термин «текстовый файл» относится к файлу и характеризует его как контейнер, хранящий такие данные.

Текстовый файл содержит последовательность символов (в основном печатных знаков, принадлежащих тому или иному набору символов). Эти символы обычно сгруппированы в строки. В современных системах строки разделяются разделителями строк, в прошлом же применялось хранение строк в виде записей постоянной или переменной длины. Иногда конец текстового файла (особенно если в файловой системе не хранится информация о размере файла) также отмечается одним или более специальными знаками, известными как маркеры конца файла. [1].

Microsoft Word (часто — MS Word, WinWord или просто Word) — [текстовый процессор](https://ru.wikipedia.org/wiki/%D0%A2%D0%B5%D0%BA%D1%81%D1%82%D0%BE%D0%B2%D1%8B%D0%B9_%D0%BF%D1%80%D0%BE%D1%86%D0%B5%D1%81%D1%81%D0%BE%D1%80), предназначенный для создания, просмотра и редактирования [текстовых документов](https://ru.wikipedia.org/wiki/%D0%A2%D0%B5%D0%BA%D1%81%D1%82%D0%BE%D0%B2%D1%8B%D0%B9_%D1%84%D0%B0%D0%B9%D0%BB), с локальным применением простейших форм [таблично](https://ru.wikipedia.org/wiki/%D0%A2%D0%B0%D0%B1%D0%BB%D0%B8%D1%86%D0%B0" \o "Таблица)-[матричных](https://ru.wikipedia.org/wiki/%D0%9C%D0%B0%D1%82%D1%80%D0%B8%D1%86%D0%B0_(%D0%BC%D0%B0%D1%82%D0%B5%D0%BC%D0%B0%D1%82%D0%B8%D0%BA%D0%B0))алгоритмов. Выпускается [корпорацией Microsoft](https://ru.wikipedia.org/wiki/Microsoft) в составе [пакета](https://ru.wikipedia.org/wiki/%D0%9E%D1%84%D0%B8%D1%81%D0%BD%D1%8B%D0%B9_%D0%BF%D0%B0%D0%BA%D0%B5%D1%82) [Microsoft Office](https://ru.wikipedia.org/wiki/Microsoft_Office" \o "Microsoft Office). Первая версия была написана [Ричардом Броди](https://ru.wikipedia.org/w/index.php?title=%D0%91%D1%80%D0%BE%D0%B4%D0%B8,_%D0%A0%D0%B8%D1%87%D0%B0%D1%80%D0%B4&action=edit&redlink=1) ([Richard Brodie](https://en.wikipedia.org/wiki/Richard_Brodie" \o "en:Richard Brodie)) для [IBM PC](https://ru.wikipedia.org/wiki/IBM_PC), использующих [DOS](https://ru.wikipedia.org/wiki/DOS), в 1983 году. Позднее выпускались версии для [Apple Macintosh](https://ru.wikipedia.org/wiki/Macintosh" \o "Macintosh)(1984), [SCO UNIX](https://ru.wikipedia.org/wiki/SCO_UNIX) и [Microsoft Windows](https://ru.wikipedia.org/wiki/Microsoft_Windows" \o "Microsoft Windows) (1989). Текущей версией является Microsoft Office Word 2016 для [Windows](https://ru.wikipedia.org/wiki/Windows" \o "Windows) и [macOS](https://ru.wikipedia.org/wiki/MacOS" \o "MacOS).

Microsoft Word является наиболее популярным из используемых в данный момент текстовых процессоров, что сделало его бинарный формат документа стандартом де-факто, и многие конкурирующие программы имеют поддержку совместимости с данным форматом. Расширение «.doc» на платформе IBM PC стало синонимом двоичного формата Word 97—2000. .doc или .DOC (от англ. document) — расширение имени файла, используемое для файлов, представляющих текст, с разметкой или без. Расширение «.doc» часто использовалось для обозначения текстовых файлов, содержащих текст без форматирования, позже стало использоваться для двоичных форматов с разметкой. Фильтры экспорта и импорта в данный формат присутствуют в большинстве текстовых процессоров. Формат документа разных версий Word меняется, различия бывают довольно тонкими [2].

При разработке решений Word в Visual Studio выполняется взаимодействие с объектной моделью Word. Эта объектная модель состоит из классов и интерфейсов, которые предоставляются в основной сборке взаимодействия для Word и определены в пространстве имен Microsoft.Office.Interop.Word.

## Анализ существующих решений

Поскольку данная задача является строго прикладной, то аналогов у неё не существует, т.к. разрабатывается она под специальное техническое задание.

## Постановка задачи

Используя динамическую библиотеку, следует разработать программу, которая будет генерировать по государственному образцу .doc-файл-форму диплома о получении степени бакалавра, сохраняя уникальное значение номера сертификата.

# ПРОЕКТИРОВАНИЕ

## 2.1 Выбор и обоснование архитектуры

Качественная архитектура является основой разработки любого проекта. Таковой можно считать только выгодную архитектуру, которая упрощает процесс создания программы и делает его более понятным. Таким образом можно сказать, что чем проще становится разработчику понять его задачу, тем лучше организована программа. Также результатом хорошо продуманной архитектуры является простота отладки программы, возможность вносить поправки и расширять возможности, не ломая при этом того, что было создано ранее. Архитектура программного обеспечения начинается с набора требований. Они могут быть выражены в форме диаграмм, блок-схем процесса, моделей или документированных списков задач эксплуатации, которые должно выполнять программное обеспечение. Обычно клиент или партнер также выражает менее точные требования, такие как внешний вид или способ работы определенных пользовательских интерфейсов для часто встречающихся задач. Требования также должны включать в себя информацию о существующем программном обеспечении, системах, оборудовании и сетях, с которыми будет взаимодействовать новое программное обеспечение; и другие факторы, такие как план развертывания и обслуживания, и, конечно же, доступный бюджет проекта[8].

В данном проекте будет использоваться трехуровневая архитектура. Разрабатываемая система разделается на три уровня, уровень представления или же интерфейс, уровень логики, которая включает в себя реализацию алгоритмов и уровень файловой системы.



Рисунок 2.1 – Архитектура данного проекта

## Диаграмма прецедентов

Функции, которые реализованные в проекте, можно визуализировать с помощью диаграммы прецедентов.



Рисунок 2.2 – Диаграмма прецедентов

В таблицах 2.1 – 2.7 описаны прецеденты представленные на рисунке 2.2

Таблица 2.1 – Прецедент ввода данных с клавиатуры

|  |  |
| --- | --- |
| Название прецедента | «Ввод данных с клавиатуры» |
| Описание | Данный прецедент позволяет вводить данные с клавиатуры. |

Таблица 2.2 – Прецедент проверки

|  |  |
| --- | --- |
| Название прецедента | «Проверка на корректность» |
| Описание | Прецедент проверяет входные данные на корректность. |

Таблица 2.3 – Прецедент копирования

|  |  |
| --- | --- |
| Название прецедента | «Копирование документа-образца» |
| Описание | Прецедент осуществляет копирование документа с сохраненными стилями в указанное место для последующей редакции. |

Таблица 2.4 – Прецедент корректирования данных

|  |  |
| --- | --- |
| Название прецедента | «Корректирование данных» |
| Описание | Прецедент реализует замену кодовых слов в документе на данные о студенте. |

Таблица 2.5 – Прецедент сохранения

|  |  |
| --- | --- |
| Название прецедента | «Сохранение документа» |
| Описание | Прецедент сохраняет .doc файл с изменениями |

## Диаграмма последовательности



Рисунок 2.3 – Диаграмма последовательности

## Проектирование графического интерфейса

В ходе разработки пользовательского интерфейса было реализовано оконное приложение с использованием WindowForms [9], которое соответствует всем описанным в техническом задании требованиям.

Приложение состоит из главной формы:

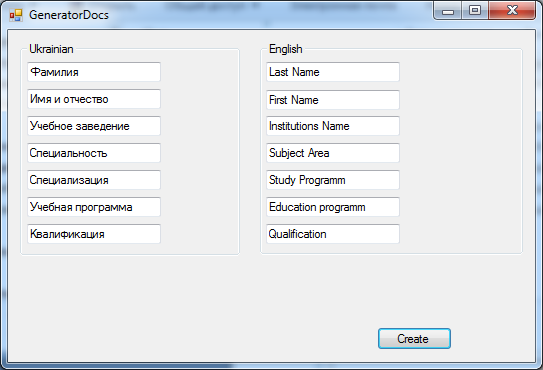


Рисунок 2.4 – Внешний вид главного окна приложения

Рабочая область содержит такие кнопки:

1. «Сreate» – создание документа.

Рабочая область содержит следующие TextBox:

1. «Фамилия»
2. «Имя и Отчество»
3. «Учебное заведение»
4. «Специальность»
5. «Специализация»
6. «Учебная программа»
7. «Квалификация»
8. «Last Name»
9. «First Name»
10. «Institutions Name»
11. «Subject Area»
12. «Study Programm»
13. «Education Programm»
14. «Qualification»

Пример работы:

1. Ввод осуществляется при помощи клавиш

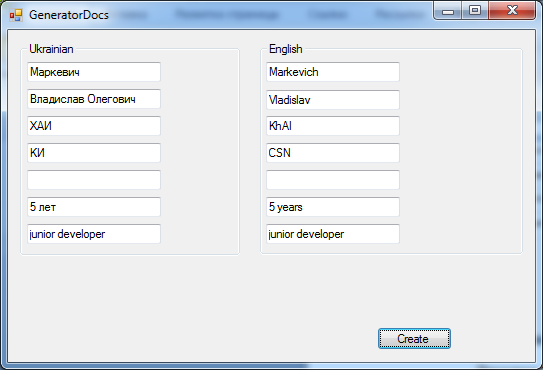


Рисунок 2.5 – Введение данных

1. Выбор папки для сохранения документа

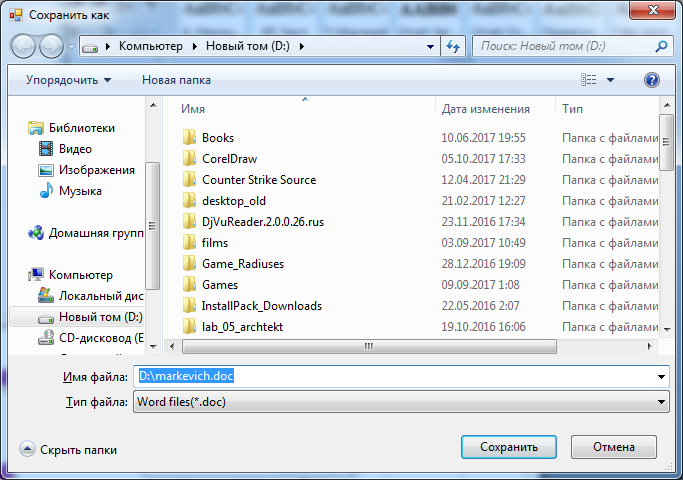


Рисунок 2.6 – Выбор папки для сохранения документа

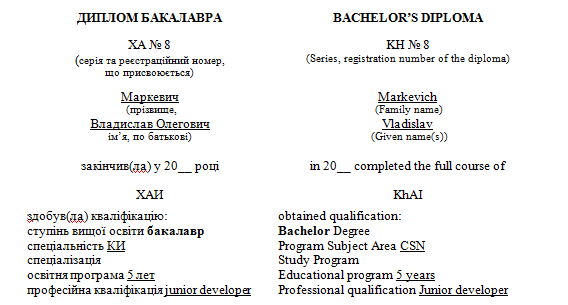


Рисунок 2.7 – Результат проверки сгенерированного документа

# РАЗРАБОТКА

## Диаграмма пакетов



Рисунок 3.1 – Диаграмма пакетов

На диаграмме пакетов, изображеной на рисунке 3.1 показана двухурованевая организация архитектуры. Класс MainWin относится к уровню графического интерфейса. Класс Editting реализует копирование файла, замену и генерацию уникального номера сертификата.

## 3.2 Диаграмма классов

В ходе выполнения данной работы для решения поставленной задачи были разработаны классы DataModifiing, GraphBuilder, GraphReadingAndWriting, IReadingAndWritingGraph, MainForm, ModelingOfGraph которые представлены на рисунке 3.2.

Описание:

1. MainWin– класс для графического отображения ввода данных;
2. Editting– класс для редактирования файла .doc;

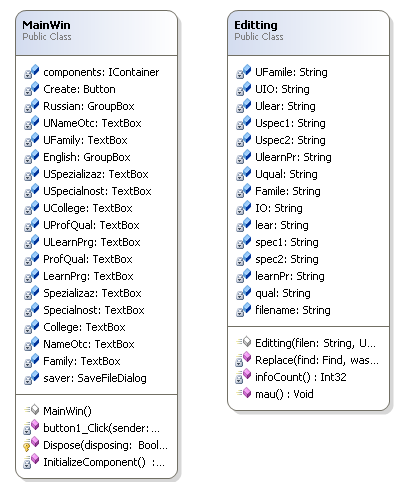


Рисунок 3.2 – Диаграмма классов

## Разработка и описание ключевых методов классов



Рисунок 3.3 – Общий алгоритм работы программы

В таблице 3.1 представлены поля и методы класса MainWin

Таблица 3.1 – Таблица со списком полей и методов класса MainWin.

|  |  |  |
| --- | --- | --- |
|  | *Поля* | *Назначение* |
| 1. | Russian (GroupBox) | Группирование данных на русском языке |
| 2. | English (GroupBox) | Группирование данных на английском языке |
| 3. | Create(Button) | Создание документа |
| 4. | saver(SaveFileDialog) | Выбор директории для сохранения |
| 5. | all-others-data | Данные для вставки |
|  | *Методы* |  |
| 6. | button1\_Click | Обработчик нажатия на клавишу Create |

В таблице 3.2 представлены поля и методы класса Editting

Таблица 3.2 – Таблица со списком полей и методов класса Editting

|  |  |  |
| --- | --- | --- |
|  | *Поля* | *Назначение* |
| 1. | filename | Имя файла, над которым будут проводить операции |
| 2. | all-others-data | Данные для вставки |
|  | *Методы* | *Назначение* |
| 1. | Replace | Метод, в котором производится замещение ключевых слов данными |
| 2. | infoCount | Метод, генерирующий уникальный номер для каждого бакалавра |
| 3. | mau | Собирает все функции класса в определенном порядке |

# ВЕРИФИКАЦИЯ И ТЕСТИРОВАНИЕ ПРОГРАММЫ

## 4.1 Верификация

Верификацией называется проверка соответствия результатов отдельных этапов разработки программной системы требованиям и ограничениям, сформулированным для них на предыдущих этапах. [10]

Согласно техническому заданию для реализации поставленной задачи был разработан проект, который позволяет:

1. Осуществлять ввод данных с клавиатуры
2. Задавать имя для документа
3. Выполнять генерацию уникального номера
4. Генерировать конечный документ.

Задачу 1 выполняет прецеденты: «Ввод данных с клавиатуры», результат представлен на рисунке 4.1

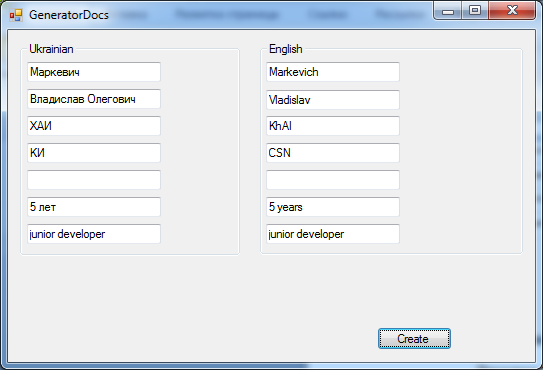


Рисунок 4.1 – Иллюстрация выполнения задачи 1

Результат выполнения задачи №2 представлен на рисунке 4.2

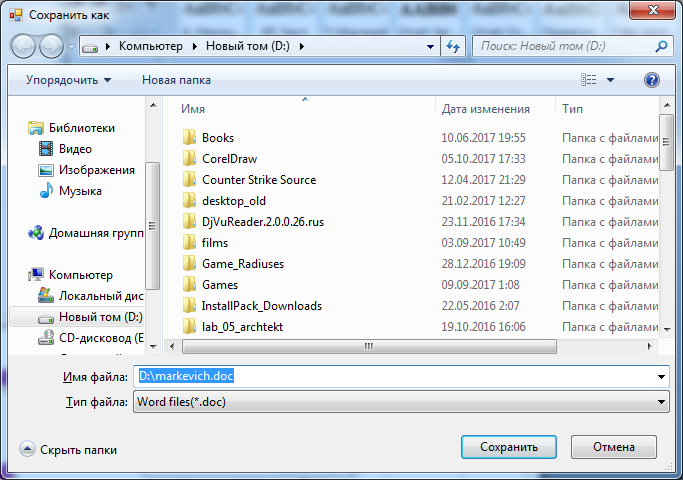


Рисунок 4.2 – Иллюстрация выполнения задачи 2

Результат выполнения задачи №3 представлен на рисунке 4.3



Рисунок 4.3 – Иллюстрация выполнения задачи 3

Результат выполнения задачи №4 представлен на рисунке 4.4

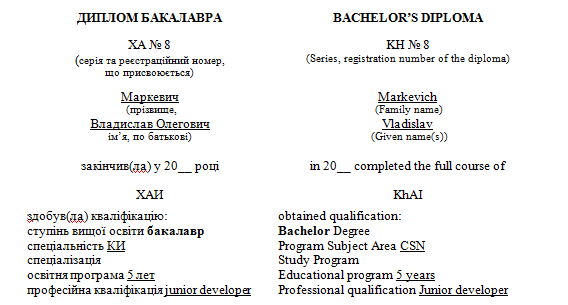


Рисунок 4.4 – Иллюстрация выполнения задачи 4

## 4.2 Тестирование программы

Тестирование программного обеспечения (Software Testing) - проверка соответствия между реальным и ожидаемым поведением программы, осуществляемая на конечном наборе тестов, выбранном определенным образом.

Поскольку основные аспекты тестов пройдены, следует пройтись по «запрещенным» конструкциям ветвлений, то есть тем, которые могут вызвать исключение. Например, создание уже существующего файла в одной директории – рис. 4.5 и 4.6.

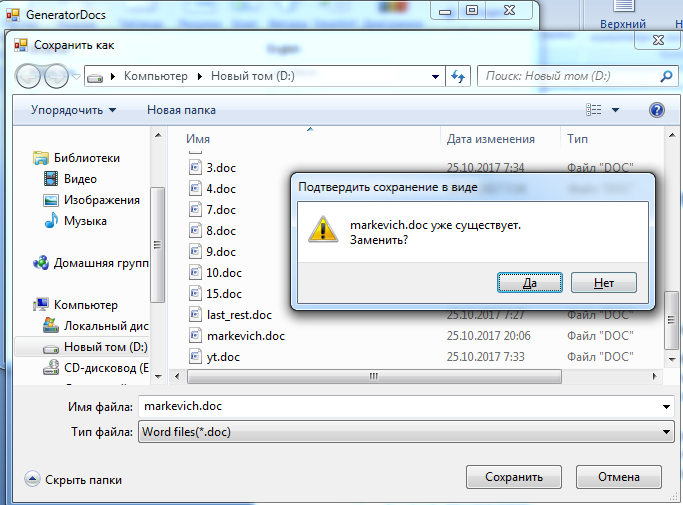


Рисунок 4.5 – Пересоздание файла

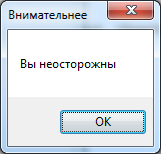


Рисунок 4.6 – Выдача предупреждения

ЗАКЛЮЧЕНИЕ

В процессе выполнения курсового проекта были изучены основные принципы работы с библиотекой Microsoft.Office.Interop.Word. Была проведена работа по созданию приложения для генерации документов. Программа верифицирована и протестирована. Тесты пройдены успешно.

В дальнейшем возможно усовершенствование программы:

1. Улучшение интерфейса;
2. Расширение списка генерируемых документов.
3. Улучшение и оптимизация алгоритма по замене данных.

# ПЕРЕЧЕНЬ ССЫЛОК

1. Текстовый файл [Электронный ресурс] / Режим доступа https://ru.wikipedia.org/wiki/Текстовый\_файл
2. Microsoft Word [Электронный ресурс] / Режим доступа https://ru.wikipedia.org/wiki/Microsoft\_Word
3. Общие сведения об объектной модели Word [Электронный ресурс] / Режим доступа https://msdn.microsoft.com/ru-ru/library/kw65a0we.aspx
4. .DOC [Электронный ресурс] / Режим доступа https://ru.wikipedia.org/wiki/.doc
5. SaveFileDialog [Электронный ресурс] / Режим https://metanit.com/sharp/windowsforms/4.20.php

# ПРИЛОЖЕНИЕ А. ТЕХНИЧЕСКОЕ ЗАДАНИЕ

**1 Введение**

**1.1 Наименование программы**

Наименование программы – Генерация документов о получении высшего образования.

**1.2 Краткая характеристика области применения**

Данная программа предназначена для генерации документов о получении степени бакалавра согласно приказу №525 от 12.05.2015 Министерства Образования и Науки Украины.

**2 Основание для разработки**

**2.1 Основание для проведения разработки**

Основания для разработки – задание на курсовой проект по дисциплине «Объектно-ориентированное программирование» на тему «Генерация документов о высшем образовании», выдано 24.01.2017.

**2.2 Наименование и условное обозначение темы разработки**

Наименование темы разработки – «Разработка программы для генерации документов о высшем образовании».

**3 Назначение разработки**

**3.1 Функциональное назначение программы**

Данная программа предоставляет возможность генерации документов об образовании для бакалавра и генерации уникального номера сертификата.

**4** **Эксплуатационное назначение**

Программный продукт был разработан средствами Windows Forms в среде разработки Visual Studio 2017. Программа предназначена для работы на ОС Windows и не является кроссплатформенной, изменения в графическом отображении возможны.  
Программа не нуждается в облуживании. Программа распространяется как откомпилированный файл \*.exe. Сбой или отказ работы программы может привести к зависанию самой программы и ОС.

**5 Требования к программе или программному изделию**

**5.1 Требования к функциональным характеристикам**

**5.1.1 Требования к составу выполняемых функций**

Программа должна выполнять следующие функции:

а) осуществлять ввод данных с клавиатуры

б) задавать имя для документа;

в) выполнять генерацию уникального номера;

г) генерировать конечный документ.

**5.1.2 Требования к организации входных данных**

**5.1.3 Требования к организации выходных данных**

В качестве выходных данных должен создаваться файл формата .doc.

**5.2 Требования к составу и параметрам технических средств**

В состав технических средств должен входить x86-совместимый компьютер, используемый в качестве рабочей станции и включающий в себя:

а) процессор с тактовой частотой, МГц - 400, не менее;

б) материнскую плату с FSB, МГц - 330, не менее;

в) оперативную память объемом, Мб -1, не менее;

г) свободное дисковое пространство для хранения программы, Мб – 1, не менее;

е) дисплей и видеоадаптер любого типа, позволяющий отображать информацию в текстовом режиме;

ж) клавиатуру;

з) мышь.

**5.3 Требования к информационной и программной совместимости**

**5.3.1 Требования к информационным структурам и методам решения**

Требования к информационным структурам на входе и выходе, а также к методам решения не предъявляются.

**5.3.2 Требования к исходным кодам и языкам программирования**

Для разработки программы в качестве языка программирования нужно использовать язык C# поддерживающий спецификацию компании производителя Microsoft, а за графический интерфейс отвечает Windows Forms. Для проектирования необходимо использовать среду разработки Microsoft Visual Studio версии 2010 и выше.

Исходные тексты программ должны содержать достаточное количество комментариев и быть доступными для любого читающего. Необходимо использовать осмысленные и корректные названия, объясняющие свое назначение.

**5.3.3 Требования к программным средствам, используемым программой**

Для функционирования программы на компьютере должно быть установлено следующее ПО:

* ОС Windows 8.1 и выше;
* платформа .NET Framework 4.5 и выше;
* Microsoft Visual Studio версии 2015 и выше.

**5.3.4 Требования к защите информации и программ**

Программа не имеет доступ к конфиденциальной информации, поэтому никакие требования к защите информации и программы не предъявляются.

**6 Требования к программной документации**

**6.1 Предварительный состав программной документации**

В результате разработки программы должна быть представлена следующая программная документация:

а) техническое задание;

б) схемы алгоритмов и данных;

в) текст программы;

г) результаты тестирования;

д) пояснительная записка;

е) руководство пользователя.

Кроме программного обеспечения на диске обязательно должна находиться пояснительная записка в электронном виде, включающая в себя весь комплект документации, предусмотренный в п. 5.1 настоящего технического задания, а также слайды презентации.

**7 Стадии и этапы разработки**

**7.1 Стадии разработки**

Разработка должны быть проведена в 4 этапа:

1. Проектирование.

2. Разработка пользовательского интерфейса.

3. Разработка проекта в целом.

4. Разработка руководства пользователя.

**7.2 Этапы разработки**

На этапе проектирования необходимо определить тему работы и разработать техническое задание. На этапе разработки пользовательского интерфейса необходимо разработать интерфейс программы. На этапе разработки проекта в целом необходимо разработать проект программы и провести ее тестирование. В заключительном этапе необходимо разработать руководство пользователя и подготовить программу к использованию.

# ПРИЛОЖЕНИЕ Б. ЛИСТИНГ ПРОГРАММЫ

MainWin.cs

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.IO;

namespace GeneratorDocs

{

public partial class MainWin : Form

{

public MainWin()

{

InitializeComponent();

}

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

{

saver.Filter = "Word files(\*.doc)|\*.doc";

if (saver.ShowDialog() == DialogResult.Cancel)

return;

string filename = saver.FileName;

string pathParent = "Parent.doc";

try

{

File.Copy(pathParent, filename);

Editting g = new Editting(filename, UFamily.Text, UNameOtc.Text, UCollege.Text, USpecialnost.Text, USpezializaz.Text, ULearnPrg.Text, UProfQual.Text, Family.Text, NameOtc.Text, College.Text, Specialnost.Text, Spezializaz.Text, LearnPrg.Text, ProfQual.Text);

g.mau();

}

catch (Exception г)

{

MessageBox.Show("Вы неосторожны", "Внимательнее");

}

}

}

}

Editting.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.IO;

using Word = Microsoft.Office.Interop.Word;

namespace GeneratorDocs

{

public class Editting

{

string UFamile;

string UIO;

string Ulear;

string Uspec1;

string Uspec2;

string UlearnPr;

string Uqual;

string Famile;

string IO;

string lear;

string spec1;

string spec2;

string learnPr;

string qual;

string filename;

public Editting(string filen, string U1, string U2, string U3, string U4, string U5, string U6, string U7, string E1, string E2, string E3, string E4, string E5, string E6, string E7)

{

UFamile = U1;

UIO = U2;

Ulear = U3;

Uspec1 = U4;

Uspec2 = U5;

UlearnPr = U6;

Uqual = U7;

Famile = E1;

IO = E2;

lear = E3;

spec1 = E4;

spec2 = E5;

learnPr = E6;

qual = E7;

filename = filen;

}

/// <summary>

/// Функция для замены

/// </summary>

/// <param name="was">То, что было</param>

/// <param name="will">Что должно быть</param>

private void Replace(Word.Find find, string was, string will)

{

find.ClearFormatting();

find.Replacement.ClearFormatting();

find.Text = was;

find.Replacement.Text = will;

object missing = Type.Missing;

object replaceAll = Microsoft.Office.Interop.Word.WdReplace.wdReplaceAll;

find.Execute(ref missing, ref missing, ref missing, ref missing, ref missing,

ref missing, ref missing, ref missing, ref missing, ref missing,

ref replaceAll, ref missing, ref missing, ref missing, ref missing);

}

private int infoCount()

{

int chys;

StreamReader read = new StreamReader("info.count");

chys = Convert.ToInt32(read.ReadLine());

read.Close();

chys++;

StreamWriter write = new StreamWriter("info.count", false);

write.WriteLine(chys);

write.Close();

return chys;

}

public void mau()

{

Microsoft.Office.Interop.Word.Application app = new Microsoft.Office.Interop.Word.Application();

app.Visible = false;

Object filenam = filename;

Object confirmConversions = Type.Missing;

Object readOnly = Type.Missing;

Object addToRecentFiles = Type.Missing;

Object passwordDocument = Type.Missing;

Object passwordTemplate = Type.Missing;

Object revert = Type.Missing;

Object writePasswordDocument = Type.Missing;

Object writePasswordTemplate = Type.Missing;

Object format = Type.Missing;

Object encoding = Type.Missing;

Object visible = Type.Missing;

Object openConflictDocument = Type.Missing;

Object openAndRepair = Type.Missing;

Object documentDirection = Type.Missing;

Object noEncodingDialog = Type.Missing;

app.Documents.Open(ref filenam, ref confirmConversions, ref readOnly, ref addToRecentFiles,

ref passwordDocument, ref passwordTemplate, ref revert, ref writePasswordDocument, ref writePasswordTemplate,

ref format, ref encoding, ref visible, ref openConflictDocument, ref openAndRepair, ref documentDirection, ref noEncodingDialog);

Microsoft.Office.Interop.Word.Find find = app.Selection.Find;

Replace(find, "РегН", "ХА");

Replace(find, "RegN", "KH");

Replace(find, "Хвамилие", UFamile);

Replace(find, "Имярек", UIO);

Replace(find, "УчебнЗав", Ulear);

Replace(find, "спецсть", Uspec1);

Replace(find, "спеця", Uspec2);

Replace(find, "обучпрг", UlearnPr);

Replace(find, "профквал", Uqual);

Replace(find, "LastName", Famile);

Replace(find, "FirstName", IO);

Replace(find, "NameCollege", lear);

Replace(find, "SubArea", spec1);

Replace(find, "StPrg", spec2);

Replace(find, "EdPrg", learnPr);

Replace(find, "PrQual", qual);

int T = infoCount();

Replace(find, "000000", Convert.ToString(T));

object saveOptionsObject = Word.WdSaveOptions.wdSaveChanges;

app.Quit(saveOptionsObject, Type.Missing, Type.Missing);

}

}

# ПРИЛОЖЕНИЕ В. РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ

1. Назначение программы

Программа представляет собой удобный и интуитивно понятный пользовательский интерфейс для создания документа по шаблону.

1. Системные требования

В состав технических средств должен входить x86-совместимый компьютер, используемый в качестве рабочей станции и включающий в себя:

а) процессор с тактовой частотой, МГц - 400, не менее;

б) материнскую плату с FSB, МГц - 330, не менее;

в) оперативную память объемом, Мб -1, не менее;

г) свободное дисковое пространство для хранения программы, Мб – 1, не менее;

е) дисплей и видеоадаптер любого типа, позволяющий отображать информацию в текстовом режиме;

ж) клавиатуру;

з) мышь.

1. Выполнение программы
2. Запустите приложение «GeneratorDocs.exe».

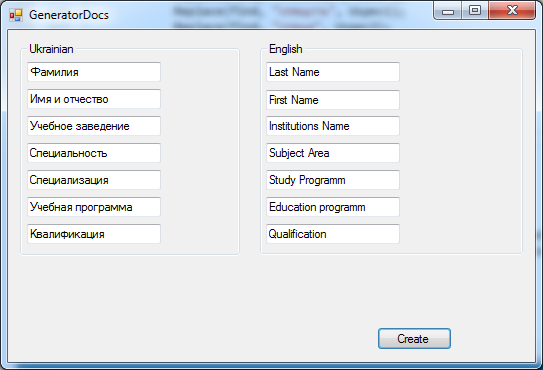


Рисунок Р.П.1 – Внешний вид приложения.

1. Введите данные в соответствующие поля с клавиатуры
2. При нажатии кнопки «Create» на экран будет выведено окно Проводника с предложением ввести имя документа и сохранить его в одной из директорий.
3. Закрыть программу можно нажав крестик в правом верхнем углу.

# ПРИЛОЖЕНИЕ Г. ПРЕЗЕНТАЦИЯ

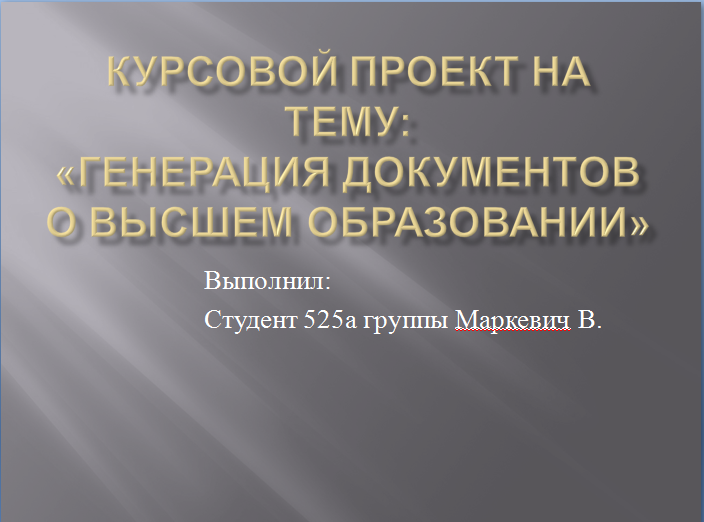


Рисунок П.1 – Слайд 1

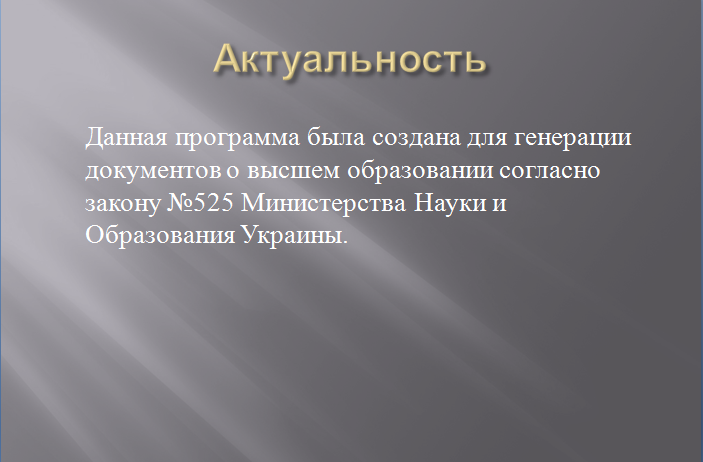


Рисунок П.2 – Слайд 2

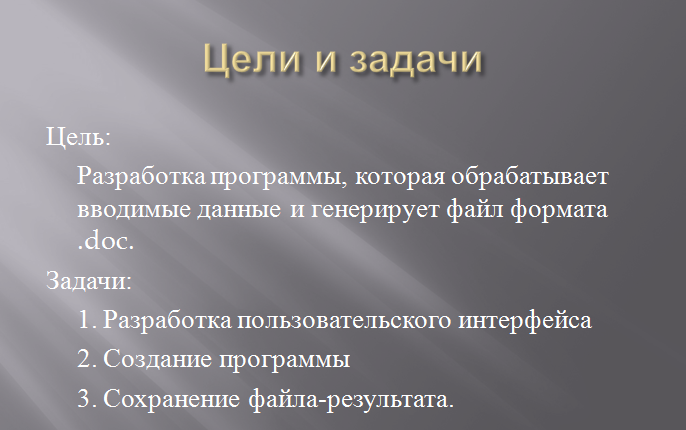


Рисунок П.3 – Слайд 3

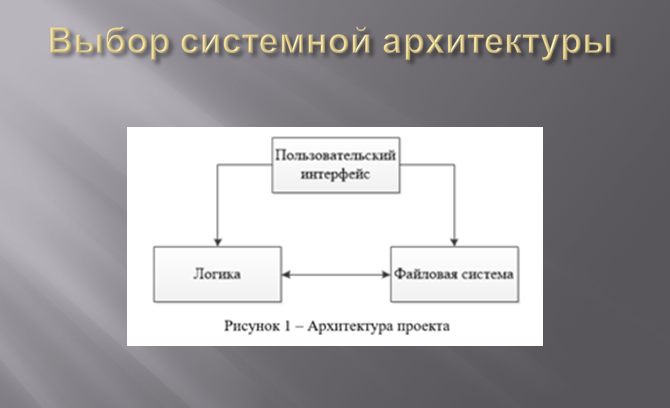


Рисунок П.4 – Слайд 4



Рисунок П.5 – Слайд 5

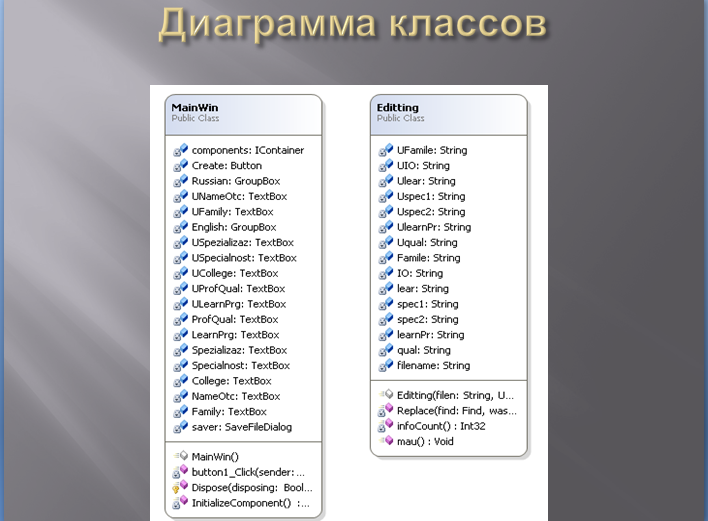


Рисунок П.6 – Слайд 6

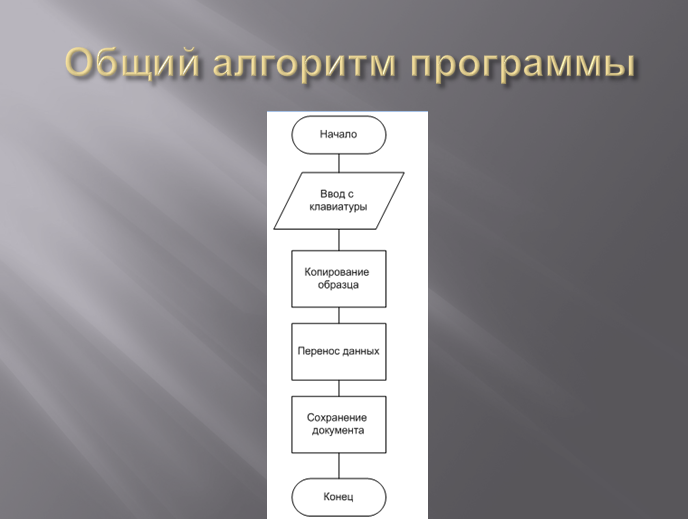


Рисунок П.7 – Слайд 7

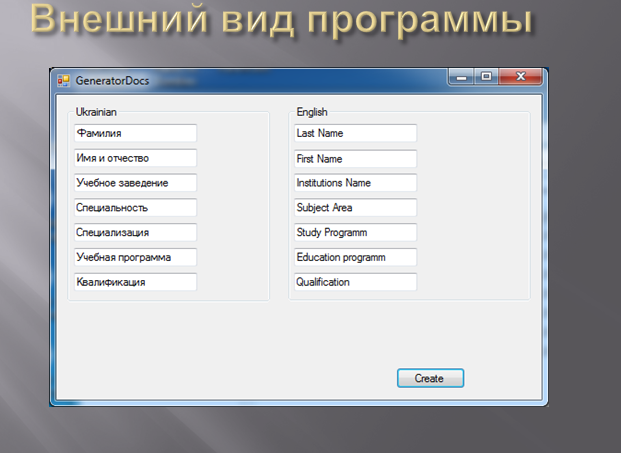


Рисунок П.8 – Слайд 8

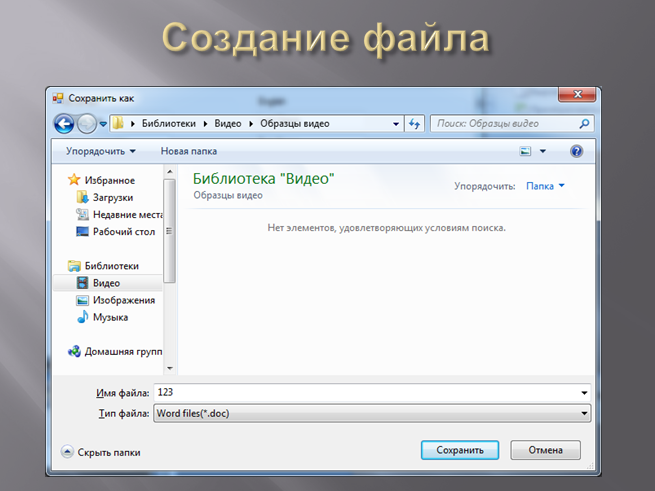


Рисунок П.9 – Слайд 9

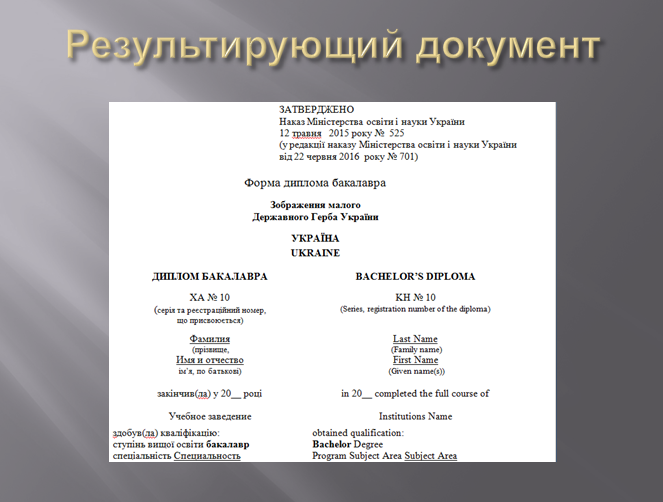


Рисунок П.10 – Слайд 10

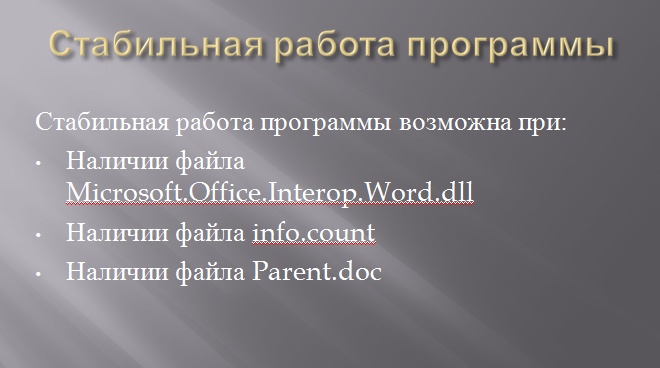


Рисунок П.11 – Слайд 11

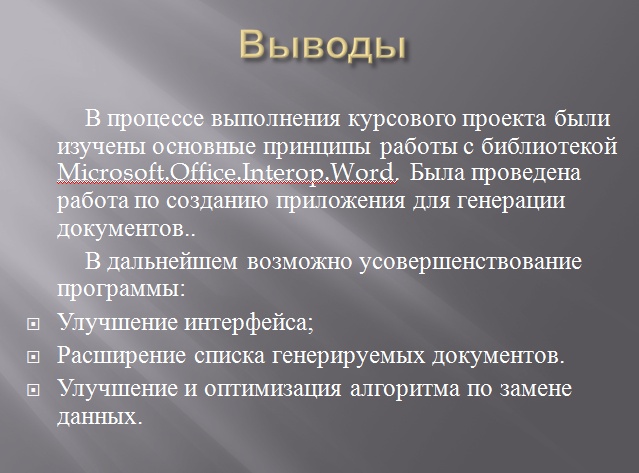


Рисунок П.12 – Слайд 12