****

**КОЛЛЕДЖ КОСМИЧЕСКОГО МАШИНОСТРОЕНИЯ И ТЕХНОЛОГИЙ**

|  |
| --- |
|  |

«Допущен к защите»

Председатель цикловой комиссии \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ А.А. Эшанов

«\_\_\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_ 2020 г.

**Информационно-аналитическая система мониторинга движения контингента**

**Пояснительная записка**

**ДП.09.02.03.21.21ПЗ**

**Дипломный проект**

**Королев, 2021**

Изм.

Лист

№ докум.

Подпись

Дата

Лист

2

ДП.09.02.03.21.21ПЗ

Разработал

Юшаков Н. Р.

Консультант

Бобкова Н.Ю.

Рецензент

*Бобкова Н. Ю.*

Конс.по эк.ч.

Капралов А. О.

Норм.контоль.

Бобкова Н.Ю.

Мониторинг движения контингента

Лит.

Листов

133

Группа П1-17

**Содержание**

[**Введение** 3](#_Toc74306289)

[**Теоретическая часть** 4](#_Toc74306290)

[**Описание предметной области** 4](#_Toc74306291)

[**Постановка задачи** 6](#_Toc74306292)

[**Аналитическая часть** 7](#_Toc74306293)

[**Анализ имеющихся программных решений** 7](#_Toc74306294)

[**Описание бизнес-процессов сбора и обработки данных о контингенте AS-IS** 10](#_Toc74306295)

[**Проектная часть** 12](#_Toc74306296)

[**Описание бизнес-процессов сбора и обработки данных о контингенте TO-BE** 12](#_Toc74306297)

[**Описание ER-модели** 14](#_Toc74306298)

[**Описание алгоритмов решения** 18](#_Toc74306299)

[**Обоснование выбора инструментов, сред, языков** 19](#_Toc74306300)

[**Описание структуры программного комплекса** 21](#_Toc74306301)

[**Описание процедур и функций** 22](#_Toc74306302)

[**Организационно-экономическая часть** 30](#_Toc74306303)

[**Эксплуатационный раздел** 30](#_Toc74306304)

[**Руководство оператора** 30](#_Toc74306305)

[**Руководство системного программиста** 38](#_Toc74306306)

[**Раздел техники безопасности** 45](#_Toc74306307)

[**Расчет экономической эффективности применения программного комплекса** 49](#_Toc74306308)

[**Заключение** 59](#_Toc74306309)

[**Список использованных источников информации** 60](#_Toc74306310)

[**Приложения** 61](#_Toc74306311)

# **Введение**

В наше время сухих цифр и статистик люди хотят знать о малейших изменениях этих самых цифр и статистик. Это касается многих сфер нашей жизни и образовательная сфера – одна из них. В этой сфере собирается огромное количество статистических данных, но в рамках данного дипломного проекта будет затронут только мониторинг движения контингента студентов средних и высших учебных заведений.

К сожалению, в наше время не существует разработок которые узконаправленно специализировались на мониторинге движения контингента, существующие на данный момент разработки входят в целые комплексы по управлению учебными заведениями и не представляется возможности получить или купить только один модуль.

Поэтому целью данного дипломного проекта является написание отдельного приложения для мониторинга движения контингента для возможности использовать данный продукт обособленно и без приобретения целого комплекса для управления учебными заведениями

# **Теоретическая часть**

## **Описание предметной области**

Учреждения среднего профессионального образования и высшие учебные заведения по своему режиму, формам и методам обучения существенно отличаются от школы.

В учреждения среднего профессионального образования практикуются такие формы обучения, как лекции, практические и лабораторные занятия

В вузах практикуются такие формы обучения, как лекции, практические и лабораторные занятия, семинары, коллоквиумы.

Традиционная система обучения в вузе и учреждениях СПО, состоящая из разных форм занятия, требует от студента самостоятельности и самодисциплины. В отличие от школы ежедневный контроль за качеством усвоения учебного материала отсутствует. Проверка знаний студентов осуществляется на практических занятиях, контрольных работах и т. п., а также в конце семестра, на зачетах и экзаменах.

Некоторые студенты могут не справиться с такими нагрузками или свалившейся на них самостоятельности и могут быть отчислены. Некоторые студены могут перевестись на другие специальности если поймут, что им не нравится выбранная ими до этого специальность. А некоторые могут и вовсе поменять учебное заведение. Именно поэтому и ведется мониторинг движения контингента студентов, чтобы легко можно было прослеживать все описанные выше ситуации.

Статистические данные о движении контингента собираются ежемесячно. В этих данных отражены такие перемещения студентов как:

1. Зачисление студентов на определенную специальность
2. Зачисление студентов на определенный курс
3. Сколько из них лиц с ОВЗ, инвалидов, дети-инвалидов
4. Сколько из них на места в рамках квоты целевого приема
5. Сколько из низ иностранных граждан
6. Отчисление студентов с их разделением на пункты 3, 4, 5
7. Сколько ушло в академический отпуск
8. Сколько вернулось из академического отпуска

И еще некоторые позиции.

Эти статистические данные могут понадобиться в абсолютно разных сферах работы учебного заведения. Например, для отчетности перед руководством, или для статистического сравнения движения контингента в зависимости от специализация или месяца.

## **Постановка задачи**

Исходя из всего что было описано ранее была сформирована задача на дипломный проект – разработать приложение для просмотра и анализа данных о мониторинге движения контингента студентов и БД для хранения данных о движении контингента.

Функционал программы:

1. Возможность добавления данных о движении контингента студентов
2. Возможность сравнения данных о движении контингента студентов
3. Возможность экспортировать данные о движении контингента студентов в файлы для дальнейшего их использования
4. Возможность просматривать данные о движении контингента студентов в программе

Для вышеперечисленного функционала необходимо:

1. Создать БД с таблицами и запросами
2. Создать приложение с интерфейсом
3. Создать файл-форму для заполнения данными о движении контингента с дальнейшей загрузкой этих данных в БД и программу

# **Аналитическая часть**

## **Анализ имеющихся программных решений**

В данном разделе представлено две программы которые упрощают работу учебных заведений, к сожалению, в данный момент не так много хороших и популярных программ для мониторинга движения контингента и поэтому первая программа имеет такой функционал, а вторая программа лишена данной возможности, но она важна для демонстрации того что большинство разработчиков не делают акцент на отдельных областях работы учебного заведения, а хотят, чтобы их продукт использовался цельно во всем учебном заведении.

Конечно, когда разрабатываются целые комплексы для автоматизации учебных заведений это хорошо, но каждое учебное заведение уже использует ПО для автоматизации отдельных сфер работы и поэтому таким учебным заведениям не целесообразно переходить на новую систему, ведь это большие экономические затраты и большие затраты на обучения персонала для работы с новым ПО.

1**С:Университет ПРОФ**

Программа 1С:Университет ПРОФ одна из самых распространенных программ для управления учебными заведениями. Данная программа позволяет автоматизировать практически все сферы деятельности учебного заведения, полный список представлен на рисунке 1.

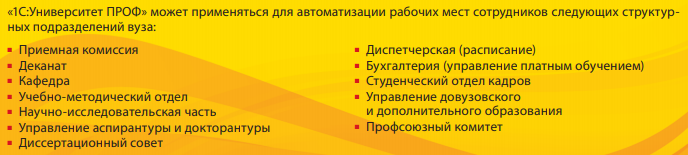


Рисунок 1. Список структурных подразделений, которые могут быть автоматизированы.

Данная программа позволяет выполнять большой объем работы с контингентом:

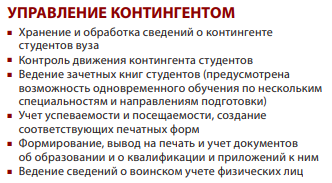


Рисунок 2. Возможности по работе с контингентом (начало)

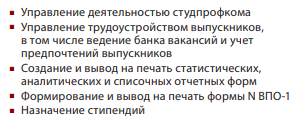


Рисунок 3. Возможности по работе с контингентом (конец)

Данная программа отлично подойдет для учебных заведений которые еще не используют другие программные средства для автоматизации отдельных сфер деятельности учебного заведения.

**Универсальная система учета**

Данная программа позволяет практически полностью автоматизировать работу любых учебных заведений.

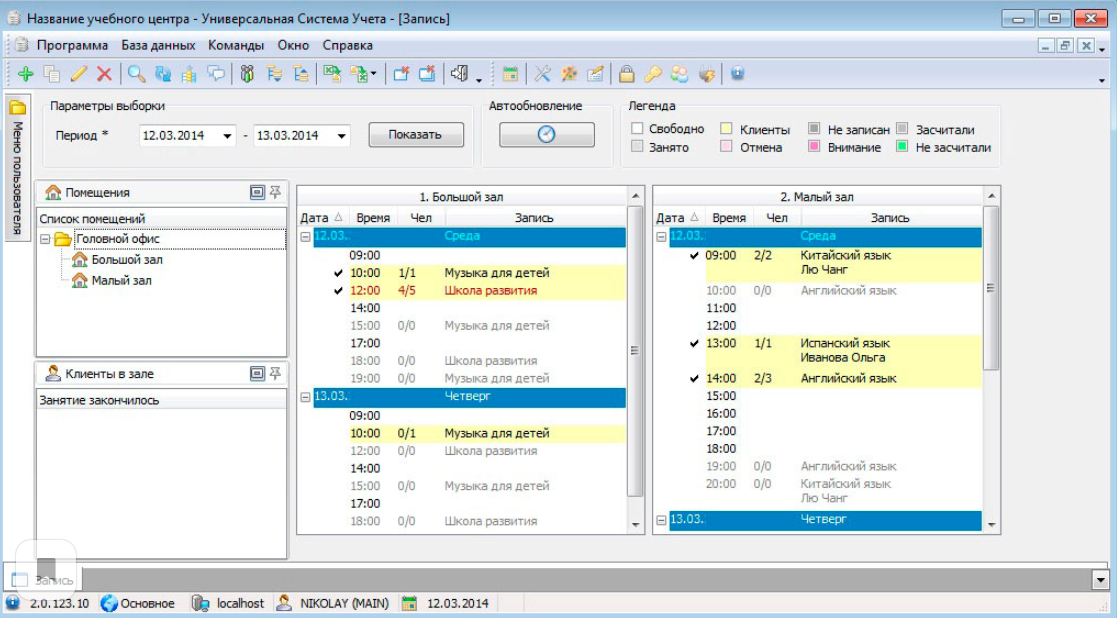


Рисунок 4. Создание расписания в Универсальной системе учета

Данная программа также отлично подходит для тех учебных заведений которые еще не используют другие программы для автоматизации отдельных сфер деятельности учебного заведения.

Но несмотря на все возможности данных программ, у них есть минусы, которые не позволяют данным продуктам быть универсальными в каждом учебном заведении, ниже представлена таблица сравнения данных программ, а также возможности программы, которая будет разработана в этом дипломном проекте.

Таблица 1. Сравнение существующих решений с дипломным проектом

|  |  |  |  |
| --- | --- | --- | --- |
| Критерий оценки | Программное решение | | |
| 1С:Университет ПРОФ | Универсальная система учета | StudInStudOut |
| Бесплатная | - | - | + |
| Возможность установки отдельного модуля | - | - | + |
| Отсутствие избыточного функционала | - | - | + |

## **Описание бизнес-процессов сбора и обработки данных о контингенте AS-IS**

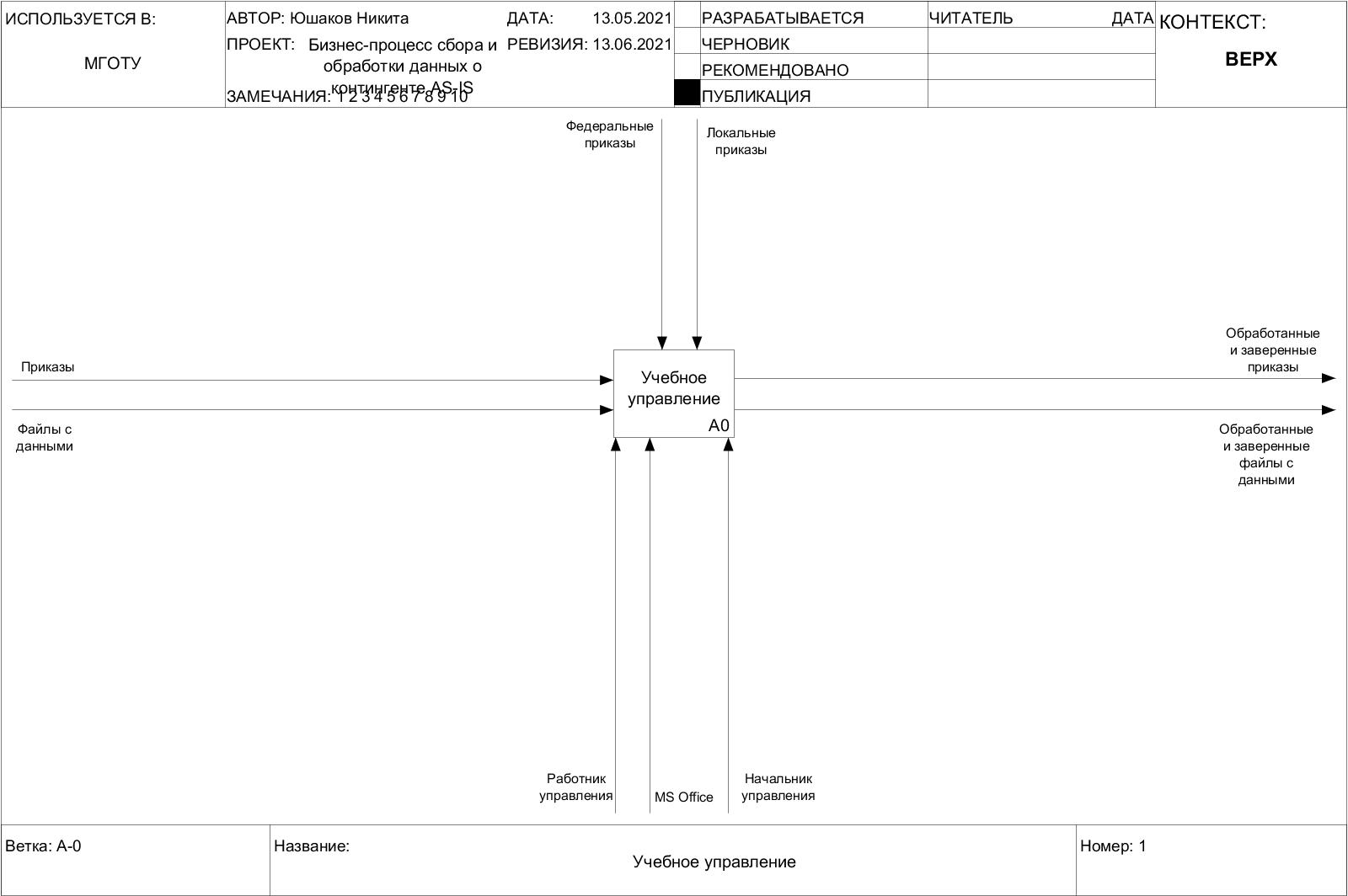


Рисунок 5. Принцип обработки данных в учебном управлении

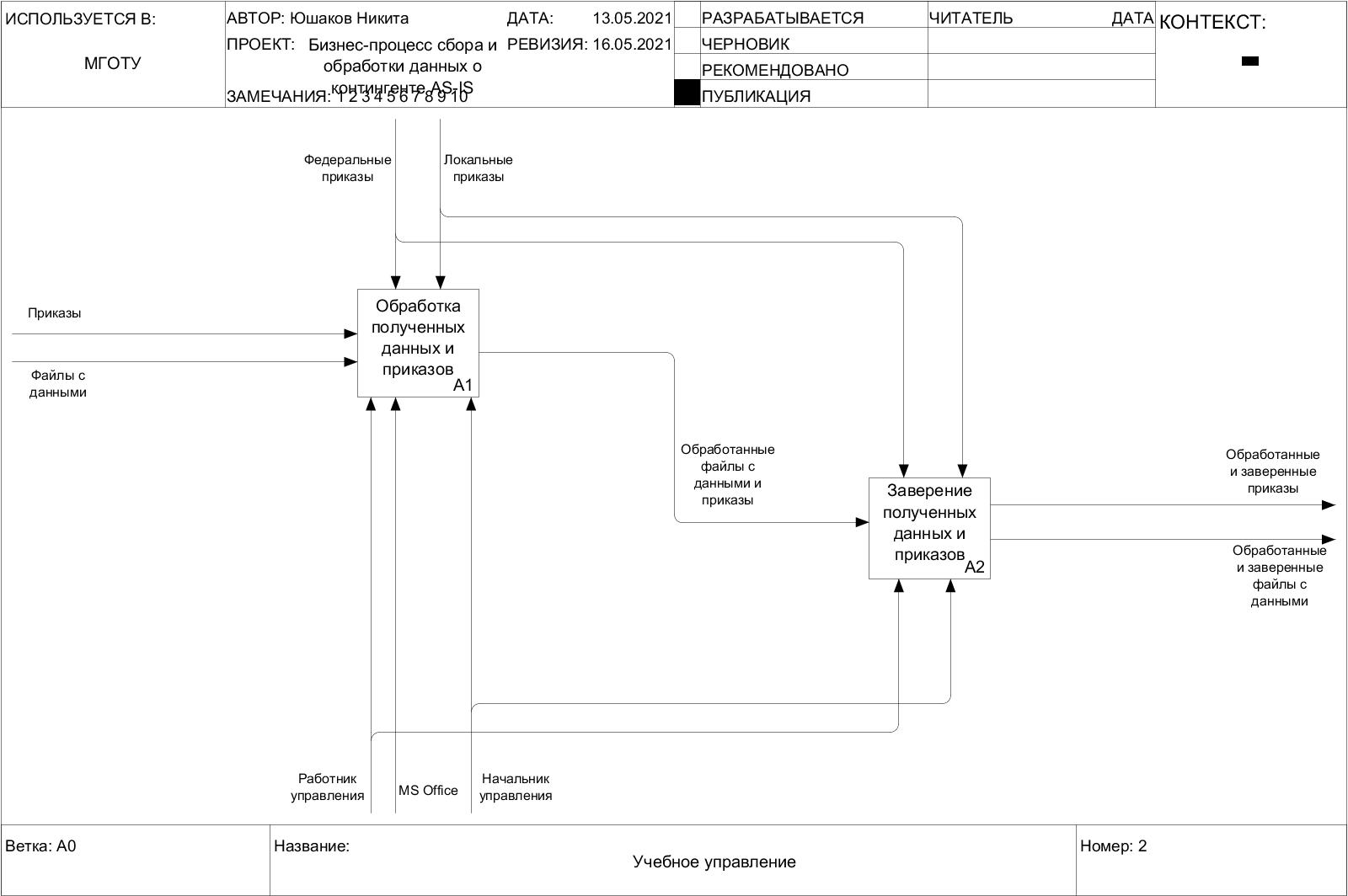


Рисунок 6. Технология обработки данных в учебном управлении

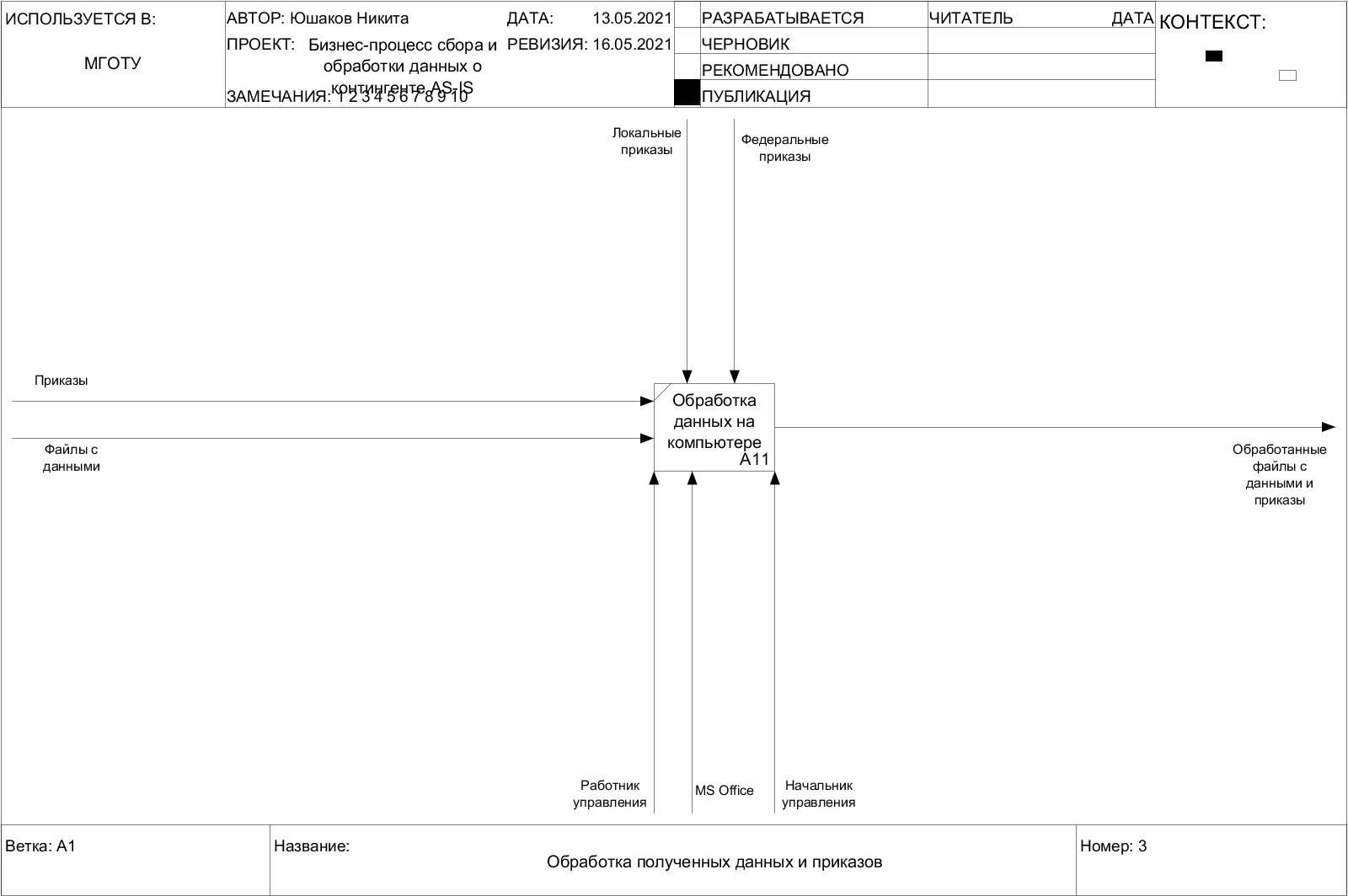


Рисунок 7. Технология обработки полученных данных и приказов

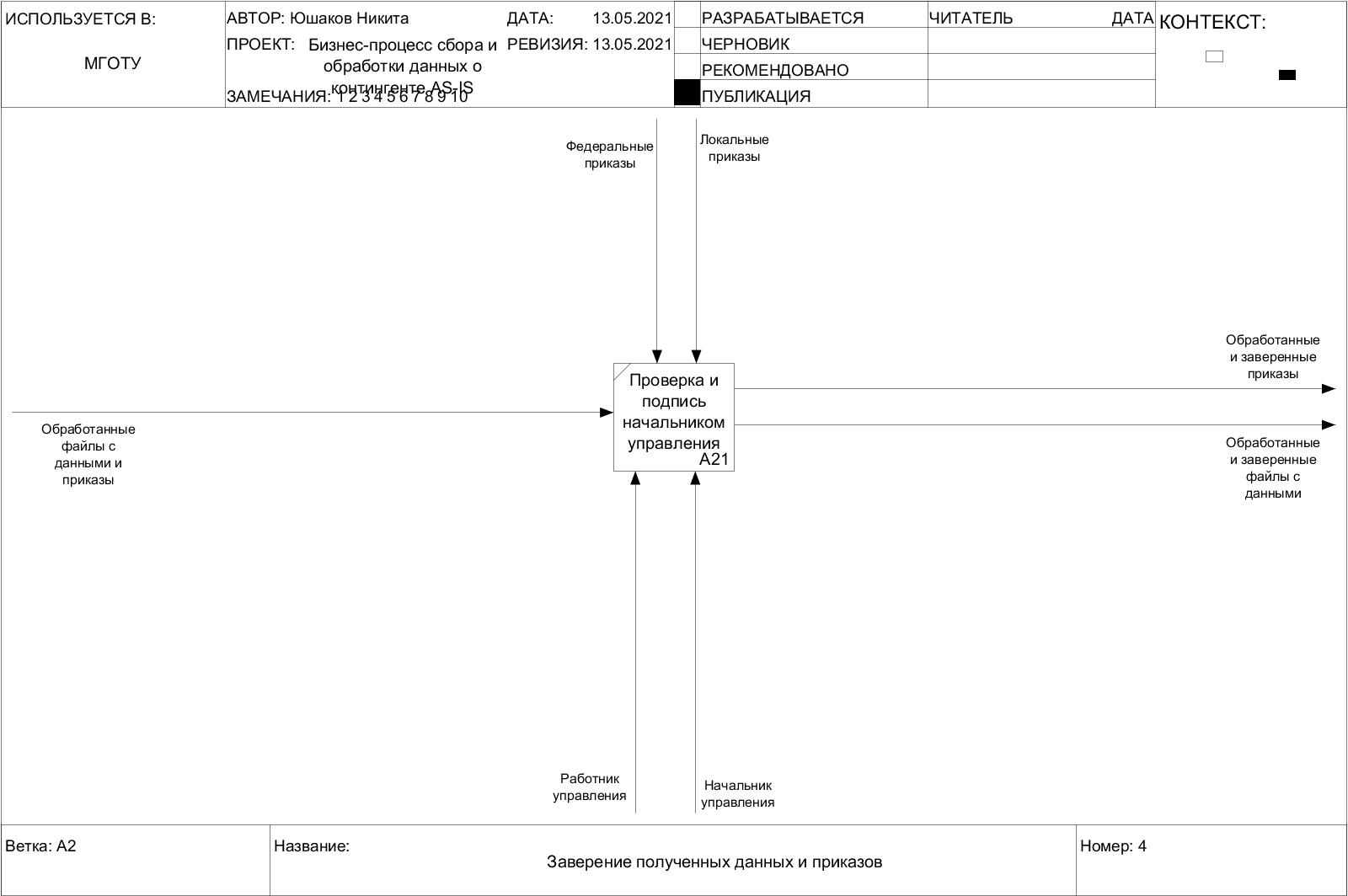


Рисунок 8. Технология заверения полученных данных и приказов

# **Проектная часть**

## **Описание бизнес-процессов сбора и обработки данных о контингенте TO-BE**

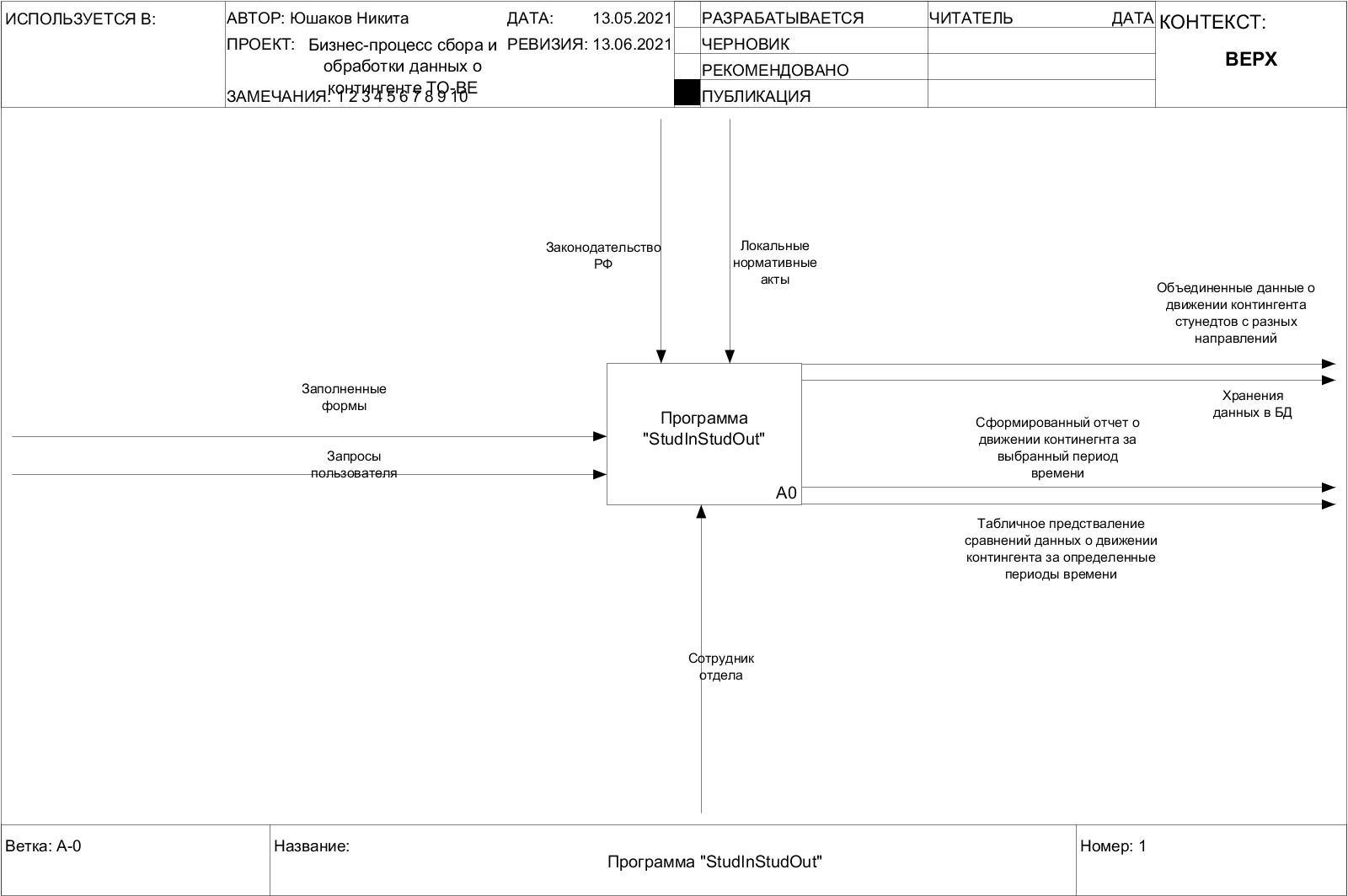


Рисунок 9. Программа "StudInStudOut"

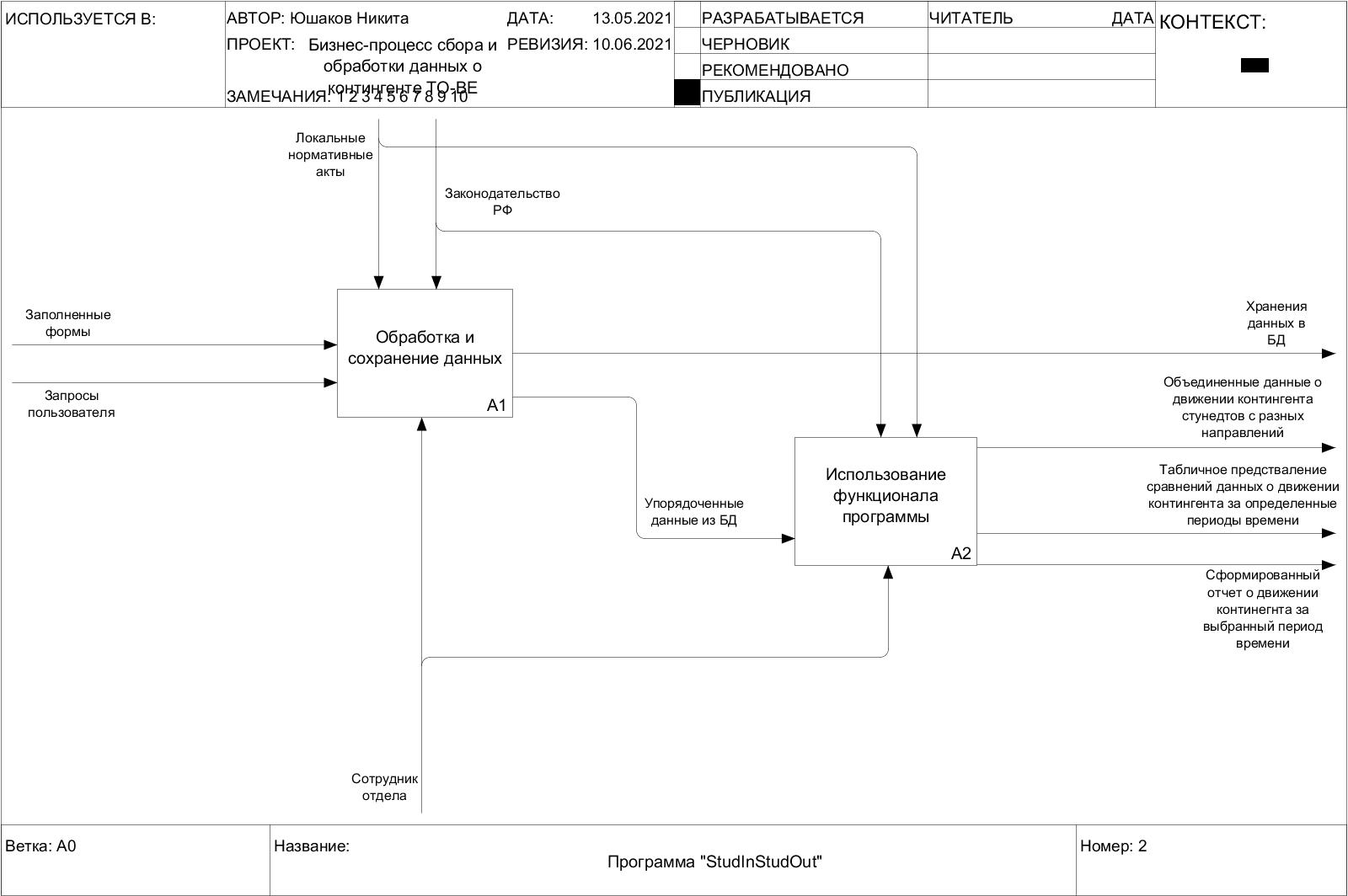


Рисунок 10. Упрощенный процесс работы

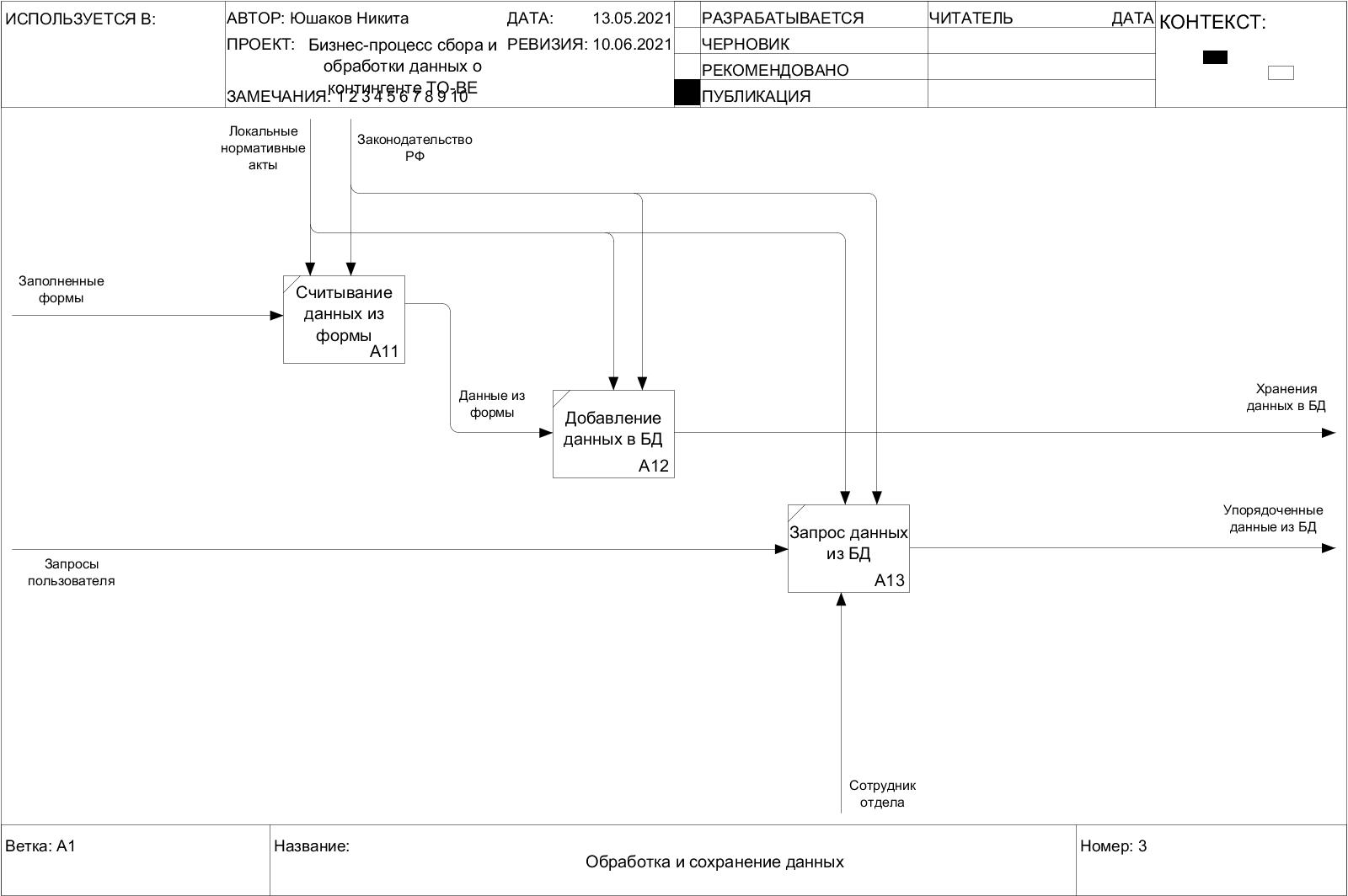


Рисунок 11. Функционал программы

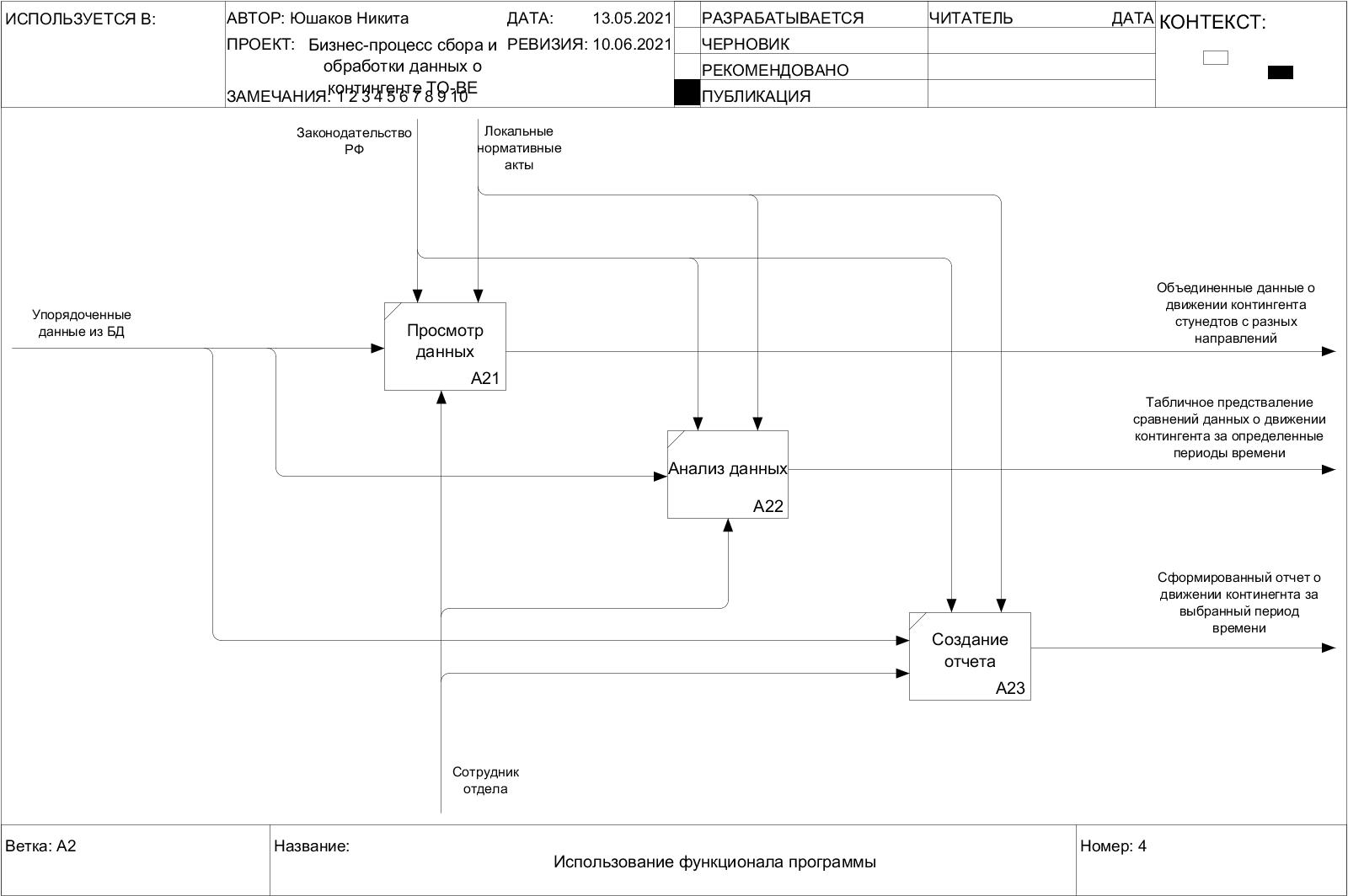


Рисунок 12 Обработка и сохранение данных в программе

## **Описание ER-модели**

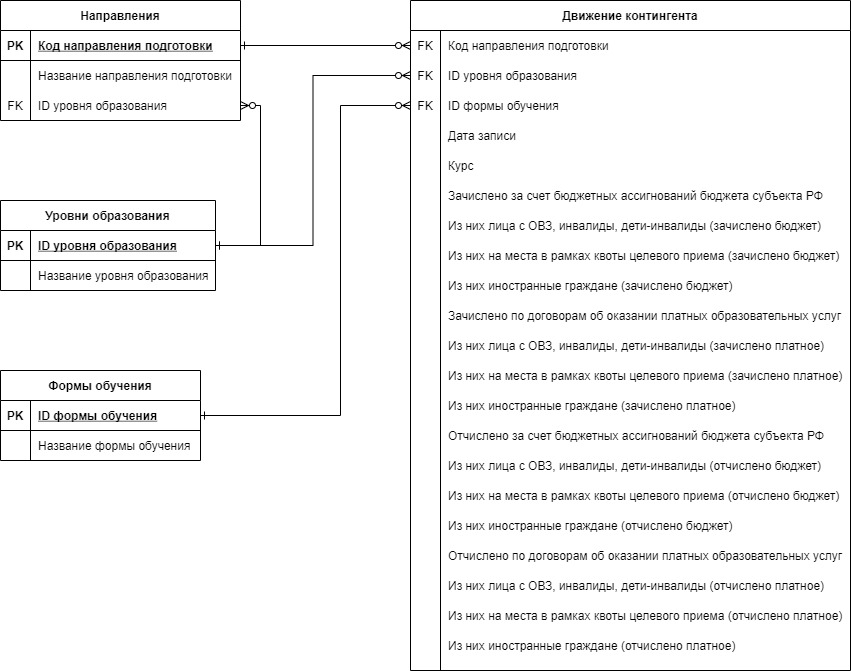


Рисунок 13. ER-модель базы данных

В базе данных присутствуют следующие таблицы:

1. Направления, хранящая информацию о направлениях образования и содержащая следующие поля:
   1. Код направления подготовки – ключевое поле данной таблицы, хранит код направления подготовки.
   2. Название направления подготовки – хранит название направления подготовки
   3. ID уровня образования – внешний ключ, хранит ID уровня образования
2. Уровни образования, хранящая информацию об уровнях образования и содержащая следующие поля:
   1. ID уровня образования – ключевое поле данной таблицы, хранит ID уровня образования
   2. Название уровня образования – хранит название уровня образования
3. Формы обучения, хранящая информацию о формах обучения и содержащая следующие поля:
   1. ID формы обучения – ключевое поле данной таблицы, хранит ID формы обучения
   2. Название формы обучения – хранит название формы обучения
4. Движение контингента, хранящая информацию о движении контингента и содержащая следующие поля:
   1. Код направления подготовки – внешний ключ, хранит код направления подготовки
   2. ID уровня образования – внешний ключ, хранит ID уровня образования
   3. ID формы обучения – внешний ключ, хранит ID формы обучения
   4. Дата записи – хранит дату записи
   5. Курс – хранит курс
   6. Зачислено за счет бюджетных ассигнований бюджета субъекта РФ – хранит информацию о зачисленных за счет бюджетных ассигнований бюджета субъекта РФ студентах
   7. Из них лица с ОВЗ, инвалиды, дети-инвалиды (зачислено бюджет) – хранит информацию о лицах с ОВЗ, инвалидах, детях-инвалидах, зачисленных на бюджет
   8. Из них на места в рамках квоты целевого приема (зачислено бюджет) – хранит информацию о лицах на места в рамках квоты целевого приема, зачисленных на бюджет
   9. Из них иностранные граждане (зачислено бюджет) – хранит информацию о иностранных гражданах, зачисленных на бюджет
   10. Зачислено по договорам об оказании платных образовательных услуг – хранит информацию о зачисленных по договорам об оказании платных образовательных услуг студентах
   11. Из них лица с ОВЗ, инвалиды, дети-инвалиды (зачислено платное) – хранит информацию о лицах с ОВЗ, инвалидах, детях-инвалидах, зачисленных на платной основе
   12. Из них на места в рамках квоты целевого приема (зачислено платное) – хранит информацию о лицах на места в рамках квоты целевого приема, зачисленных на платной основе
   13. Из них иностранные граждане (зачислено платное) – хранит информацию о иностранных гражданах, зачисленных на платной основе
   14. Отчислено за счет бюджетных ассигнований бюджета субъекта РФ – хранит информацию об отчисленных за счет бюджетных ассигнований бюджета субъекта РФ студентов
   15. Из них лица с ОВЗ, инвалиды, дети-инвалиды (отчислено бюджет) – хранит информацию о лицах с ОВЗ, инвалидах, детях-инвалидах, отчисленных с бюджета
   16. Из них на места в рамках квоты целевого приема (отчислено бюджет) – хранит информацию о лицах на места в рамках квоты целевого приема, отчисленных с бюджета
   17. Из них иностранные граждане (отчислено бюджет) – хранит информацию о иностранных гражданах, отчисленных с бюджета
   18. Отчислено по договорам об оказании платных образовательных услуг – хранит информацию об отчисленных по договорам об оказании платных образовательных услуг студентах
   19. Из них лица с ОВЗ, инвалиды, дети-инвалиды (отчислено платное) – хранит информацию о лицах с ОВЗ, инвалидах, детях-инвалидах, отчисленных с обучения на платной основе
   20. Из них на места в рамках квоты целевого приема (отчислено платное) – хранит информацию о лицах на места в рамках квоты целевого приема, отчисленных с обучения на платной основе
   21. Из них иностранные граждане (отчислено платное) – хранит информацию о иностранных гражданах, отчисленных с обучения на платной основе

## **Описание алгоритмов решения**



Рисунок 14. Алгоритм решения

После начала работы программы у пользователя будет пять возможных дальнейших действий:

1. Сравнение данных за определенный период времени – данное действие позволяет увидеть сравнение данных о движении контингента за выбранный период времени с аналогичными данными за прошлый месяц и за прошлый год.
2. Формирование отчета – данное действие позволяет пользователю экспортировать выбранные данные в Excel в виде отчета.
3. Добавление информации – данное действие позволяет пользователю добавить новые данные о движении контингента.
4. Настройки – данное действие позволяет пользователю перейти в окно настроек и настроить некоторые элементы программы под себя.
5. Выход – данное действие позволяет пользователю выйти из программы.

## **Обоснование выбора инструментов, сред, языков**

Для разработки клиент-серверного приложения были произведены сравнения различных инструментов по некоторым критериям и был выбран лучший инструмент в каждой категории.

Таблица 2. Выбор языка программирования

|  |  |  |  |
| --- | --- | --- | --- |
| Критерий\ Язык программирования | Python | C++ | C# |
| Высокая скорость разработки | + | - | - |
| Простота разработки | + | - | - |
| Простота работы с библиотеками | + | + | + |
| Опыт работы с ЯП | + | + | + |

Из-за перечисленных в таблице критериев языком программирования для дипломного проекта был выбран Python.

Таблица 3. Выбор сервера

|  |  |  |  |
| --- | --- | --- | --- |
| Критерий \ Сервер | MSSQL | MySQL | SQLite |
| Опыт работы с сервером | + | - | - |
| Наличие официальной документации на русском языке | + | + | + |

Наличие официальной документации, опыт работы, а также удобство при работе с сервером MSSQL в языке программирования Python делает данный сервер самым удобным для данного проекта.

Таблица 4. Выбор графической среды для работы с БД

|  |  |  |  |
| --- | --- | --- | --- |
| Критерий \ Графическая среда | SSMS | HeidiSQL | Red Gate SQL Prompt |
| Опыт работы с сервером | + | - | - |
| Наличие официальной документации на русском языке | + | - | - |

Наличие официальной документации на русском языке – важнейший фактор в выборе графической среды для работы с БД.

Таблица 5. Выбор IDE для Python 3.9.5

|  |  |  |
| --- | --- | --- |
| Критерий \ IDE | PyCharm | IDLE Python |
| Наличие проверки на PEP 8 | + | - |
| Удобство при работе с несколькими модулями | + | - |
| Возможность просмотреть каталог библиотек | + | - |
| Удобство при работе с библиотеками | + | - |

Все вышеперечисленные критерии крайне важны при работе над большим проектом и IDE PyCharm включает в себя все эти удобства, что делает данную IDE лучшим выбором для работы над данным проектом

## **Описание структуры программного комплекса**

Таблица 6. Модули программного комплекса

|  |  |
| --- | --- |
| Программный модуль | Выполняемые функции |
| main.py | Главный модуль, создающий главную экранную форму |
| ui\_control.py | Управляющий модуль инициализирующий и выполняющий все функции программы |
| dialog\_ui.py | Экранная форма для пользовательского отчета, собирающая необходимую информацию и передающая ее в excel\_control.py |
| main\_ui.py | Главная экранная форма |
| dialog\_simple\_ui.py | Экранная форма для встроенного отчета, собирающая необходимую информацию и передающая ее в excel\_control.py |
| excel\_control.py | Модуль для работы с Excel |
| db\_control.py | Модуль для работы с БД |

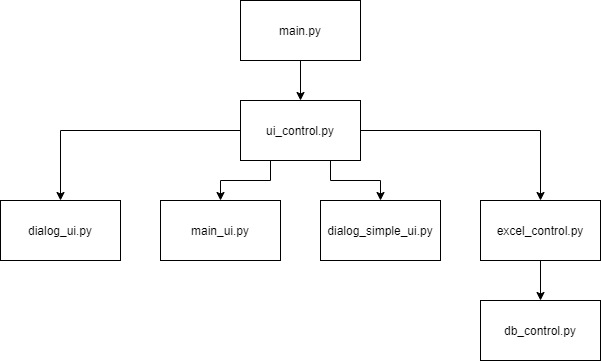


Рисунок 15. Структура программного комплекса

## **Описание процедур и функций**

В данном проекте процедуры и функции присутствуют в шести из семи модулях, описанных выше: ui\_control.py, dialog\_ui.py, main\_ui.py, dialog\_simple\_ui.py, excel\_control.py, db\_control.py.

В модуле ui\_control.py присутствуют следующие процедуры и функции:

1. \_\_init\_\_(self, \_window) – конструктор класса UiControl, инициализирующий все атрибуты данного класса. Принимает один аргумент - \_window – экземпляр класса QMainWindow() библиотеки PyQt5.
2. set\_menu\_and\_tool\_bars (self) – процедура, инициализирующая действия в меню и панели инструментов, а также устанавливающая триггеры для них.
3. load\_table(self) – процедура, загружающая данные, полученные из БД в таблицу в программе.
4. data\_analysis(self) – процедура, анализирующая данные о движении контингента за определенные промежутки времени.
5. set\_style\_last\_month(self, data\_new, data\_old) – процедура, устанавливающая стиль и данные для таблицы с данными сравнения движения контингента за выбранный месяц и за месяц перед выбранным. Принимает два аргумента – data\_new – данные о движении контингента за выбранный месяц и data\_old – данные о движении контингента за месяц перед выбранным.
6. set\_style\_last\_year(self, data\_new, data\_old) – процедура, устанавливающая стиль и данные для таблицы с данными сравнения движения контингента за выбранный месяц выбранного года и за тот же месяц годом ранее. Принимает два аргумента – data\_new – данные о движении контингента за выбранный месяц выбранного года и data\_old – данные о движении контингента за тот же месяц годом ранее.
7. get\_data\_old\_month(self, row\_index) – функция, возвращающая данные о движении контингента за месяц перед выбранным. Принимает один аргумент – row\_index – индекс строки выбранной пользователем.
8. get\_data\_old\_year(self, row\_index) – функция, возвращающая данные о движении контингента за год до выбранного. Принимает один аргумент – row\_index – индекс строки выбранной пользователем.
9. get\_data\_new(self, row\_index) – функция, возвращающая данные о движении контингента выбранный период времени. Принимает один аргумент – row\_index – индекс строки выбранной пользователем.
10. open\_page\_1(self) – процедура открывающая первый лист главной экранной формы.
11. open\_page\_2(self) – процедура открывающая второй лист главной экранной формы.
12. update\_all(self) – процедура обновляющая данные в БД и в программе.
13. update\_combo\_boxes(self) – процедура, обновляющая информацию в combo box-ах в программе.
14. user\_settings(self) – процедура для выбора и установки пользовательских настроек.
15. open\_dialog(self) – процедура, вызывающая экранную форму пользовательского отчета.
16. open\_dialog\_simple(self, trigger) – процедура, вызывающая экранную форму встроенного отчета. Принимает один аргумент – trigger – название выбранного встроенного отчета.

В модуле dialog\_ui.py присутствуют следующие процедуры и функции:

1. \_\_init\_\_(self, dialog) – конструктор класса UiDialog, инициализирующая все атрибуты данного класса. Принимает один аргумент – dialog – экземпляр класса QDialog() библиотеки PyQt5.
2. retranslate\_ui(self, dialog) – процедура, инициализирующая текст всех элементов интерфейса. Принимает один аргумент – dialog – экземпляр класса QDialog() библиотеки PyQt5.
3. close\_window(self) – процедура, закрывающая экранную форму пользовательского отчета.
4. set\_combo\_boxes(self) – процедура, устанавливающая значения combo box-ов экранной формы пользовательского отчета.
5. get\_condition(self) – процедура для получения данных необходимых для экспорта.

В модуле main\_ui.py присутствуют следующие процедуры и функции:

1. \_\_init\_\_(self, window) – конструктор класса UiWindow, инициализирующая все атрибуты данного класса. Принимает один аргумент – window – экземпляр класса QMainWindow() библиотеки PyQt5.
2. retranslate\_ui(self, window) – процедура, инициализирующая текст всех элементов интерфейса. Принимает один аргумент – window – экземпляр класса QMainWindow() библиотеки PyQt5.

В модуле dialog\_simple\_ui.py присутствуют следующие процедуры и функции:

1. \_\_init\_\_(self, dialog, trigger) – конструктор класса UiDialogSimple, инициализирующая все атрибуты данного класса. Принимает два аргумента – dialog – экземпляр класса QDialog() библиотеки PyQt5 и trigger - название выбранного встроенного отчета.
2. retranslate\_ui(self, dialog) – процедура, инициализирующая текст всех элементов интерфейса. Принимает один аргумент – dialog – экземпляр класса QDialog() библиотеки PyQt5.
3. close\_window(self) – процедура, закрывающая экранную форму пользовательского отчета.
4. set\_combo\_boxes(self) – процедура, устанавливающая значения combo box-ов экранной формы пользовательского отчета.
5. get\_condition(self) – процедура для получения данных необходимых для экспорта.

В модуле excel\_control.py присутствуют следующие процедуры и функции:

1. \_\_init\_\_(self) - конструктор класса ExcelControl, инициализирующая все атрибуты данного класса.
2. get\_excel\_data(self) – процедура, получающая данные из файлов, выбранных пользователем.
3. get\_file\_names(self) – функция, возвращающая названия и пути к файлам которые выбрал пользователь.
4. set\_file\_name(self) – функция, возвращающая название файла которое ввел пользователь.
5. export\_data(self, sheet\_1, for, sheet\_2=None, sheet\_3=None) – процедура, экспортирующая данные в Excel. Принимает четыре аргумента – sheet\_1 – данные для первого листа Excel, foe – название формы обучения, sheet\_2 – данные для второго листа Excel и sheet\_3 – данные для третьего листа Excel.

В модуле db\_control.py присутствуют следующие процедуры и функции:

1. \_\_init\_\_(self) - конструктор класса DbControl, инициализирующая все атрибуты данного класса.
2. insert\_data(self, course, info, statistic) – процедура, добавляющая данные из Excel в БД. Принимает 3 аргумента – course – номер курса, info – список данных для идентификации значений, statistic – список, хранящий в себе данные о движении контингента.
3. get\_data(self, loe, foe, direction, year, month, course=0) – функция, возвращающая данные из БД по выбранным параметрам. Принимает шесть аргументов – loe – уровень образования, foe – форма образования, direction – направление образования, year – год, month – месяц, course – курс.
4. get\_disabled(self, loe, foe, direction, year, month, course=0) – функция, возвращающая данные о лицах с ОВЗ, инвалидах, детях-инвалидах из БД по выбранным параметрам. Принимает шесть аргументов – loe – уровень образования, foe – форма образования, direction – направление образования, year – год, month – месяц, course – курс.
5. get\_target(self, loe, foe, direction, year, month, course=0) – функция, возвращающая данные о целевом наборе из БД по выбранным параметрам. Принимает шесть аргументов – loe – уровень образования, foe – форма образования, direction – направление образования, year – год, month – месяц, course – курс.
6. get\_foreigners(self, loe, foe, direction, year, month, course=0) – функция, возвращающая данные об иностранцах из БД по выбранным параметрам. Принимает шесть аргументов – loe – уровень образования, foe – форма образования, direction – направление образования, year – год, month – месяц, course – курс.
7. get\_directions(self, loe) – функция, возвращающая список направлений по определенному уровню обучения. Принимает один аргумент – loe – уровень образования.
8. get\_years(self, loe) – функция, возвращающая все года, данные за которые есть в БД на определенном уровне обучения. Принимает один аргумент – loe – уровень образования.
9. get\_months(self, loe) – функция, возвращающая все месяца, данные за которые есть в БД на определенном уровне обучения. Принимает один аргумент – loe – уровень образования.

Также процедуры присутствуют и в БД:

1. Добавление данных – данная процедура добавляет данные о движении контингента в БД.

Принимает следующие аргументы:

* 1. dir\_id – ID направления подготовки
  2. dir\_name – название направления подготовки
  3. foe\_name – название формы обучения
  4. loe\_name – название уровня обучения
  5. date – дата записи
  6. course – курс
  7. free\_in – сколько зачислено за счет бюджетных ассигнований бюджета субъекта РФ
  8. disabled\_in – сколько из них лица с ОВЗ, инвалиды, дети-инвалиды
  9. target\_in – сколько из них на места в рамках квоты целевого приема
  10. foreigners\_in – сколько из них иностранные граждане
  11. pay\_in – сколько зачислено по договорам об оказании платных образовательных услуг
  12. disabled\_pi – сколько из них лица с ОВЗ, инвалиды, дети-инвалиды
  13. target\_pi – сколько из них на места в рамках квоты целевого приема
  14. foreigners\_pi – сколько из них иностранные граждане
  15. free\_out – сколько отчислено за счет бюджетных ассигнований бюджета субъекта РФ
  16. disabled\_fo – сколько из них лица с ОВЗ, инвалиды, дети-инвалиды
  17. target\_fo – сколько из них на места в рамках квоты целевого приема
  18. foreigners\_fo – сколько из них иностранные граждане
  19. pay\_out – сколько отчислено по договорам об оказании платных образовательных услуг
  20. disabled\_po – сколько из них лица с ОВЗ, инвалиды, дети-инвалиды
  21. target\_po – сколько из них на места в рамках квоты целевого приема
  22. foreigners\_po – сколько из них иностранные граждане

1. Получение данных – данная процедура возвращает данные о движении контингента, которые соответствуют заданным требованиям

Принимает следующие аргументы:

* 1. loe\_name – название уровня обучения
  2. foe\_name – название формы обучения
  3. dir\_name – название направления подготовки
  4. year – год записи
  5. month – месяц записи
  6. course – курс

1. Получение данных о целевиках – данная процедура возвращает данные о движении контингента лиц на места в рамках квоты целевого приема, которые соответствуют заданным требованиям

Принимает следующие аргументы:

* 1. loe\_name – название уровня обучения
  2. foe\_name – название формы обучения
  3. dir\_name – название направления подготовки
  4. year – год записи
  5. month – месяц записи
  6. course - курс

1. Получение данных об инвалидах – данная процедура возвращает данные о движении контингента лиц с ОВЗ, инвалидов, инвалидов-детей, которые соответствуют заданным требованиям

Принимает следующие аргументы:

* 1. loe\_name – название уровня обучения
  2. foe\_name – название формы обучения
  3. dir\_name – название направления подготовки
  4. year – год записи
  5. month – месяц записи
  6. course - курс

1. Получение данных об иностранцах – данная процедура возвращает данные о движении контингента иностранных граждан, которые соответствуют заданным требованиям

Принимает следующие аргументы:

* 1. loe\_name – название уровня обучения
  2. foe\_name – название формы обучения
  3. dir\_name – название направления подготовки
  4. year – год записи
  5. month – месяц записи
  6. course - курс

1. Получение лет – данная процедура возвращает список лет, за которые есть данные о движении контингента

Принимает следующие аргументы:

* 1. loe\_name – название уровня обучения

1. Получение месяцев – данная процедура возвращает список месяцев, за которые есть данные о движении контингента

Принимает следящие аргументы:

* 1. loe\_name – название уровня обучения

1. Получение направлений – данная процедура возвращает список месяцев, за которые есть данные о движении контингента

Принимает следящие аргументы:

* 1. loe\_name – название уровня обучения

# **Организационно-экономическая часть**

## **Эксплуатационный раздел**

### **Руководство оператора**

**АННОТАЦИЯ**

В данном программном документе приведено руководство оператора по применению и эксплуатации программы «StudInStudOut», предназначенной для облегчения мониторинга движения контингента студентов.

В данном программном документе, в разделе «Назначение программы» указаны сведения о назначении программы и информация, достаточная для понимания функций программы и ее эксплуатации.

В разделе «Условия выполнения программы» указаны условия, необходимые для выполнения программы (минимальный состав аппаратных и программных средств и т.п.).

В данном программном документе, в разделе «Выполнение программы» указана последовательность действий оператора, обеспечивающих загрузку, запуск, выполнение и завершение программы, приведено описание функций, формата и возможных вариантов команд, с помощью которых оператор осуществляет загрузку и управляет выполнением программы, а также ответы программы на эти команды.

В разделе «Сообщения оператору» приведены тексты сообщений, выдаваемых в ходе выполнения программы, описание их содержания и соответствующие действия оператора

Оформление программного документа «Руководство оператора» произведено по требованиям ЕСПД (ГОСТ 19.101-77[1], ГОСТ 19.103-77[2], ГОСТ 19.104-78\*[3], ГОСТ 19.105-78\*[4], ГОСТ 19.106-78\*[5], ГОСТ 19.505-79\*[6], ГОСТ 19.604-78\*[7]).

1. **Назначение программы**
   1. **Функциональное назначение программы**

Специальное программное обеспечение «StudInStudOut» используется для хранения и просмотра данных о движении контингента студентов, а также для анализа этих данных и их экспорта в виде отчета.

* 1. **Эксплуатационное назначение программы**

Специальное программное обеспечение «StudInStudOut» может эксплуатироваться на объектах любого масштаба в сфере гостиничного бизнеса для облегчения работы персонала.

* 1. **Состав функций**
     1. **Функция импорта данных**

Данная функция позволяет импортировать данные из заполненной формы в БД и в программу.

* + 1. **Функция смены уровня образования**

Данная функция позволяет менять уровни образования и просматривать данные о движении контингента за выбранный уровень образования.

* + 1. **Функция выборки данных**

Данная функция позволяет производить выборку данных по определенным параметрам.

* + 1. **Функция сортировки данных**

Данная функция позволяет сортировать данные в таблице по выбранному столбцу.

* + 1. **Функция сравнения данных**

Данная функция позволяет сравнивать данные о движении контингента за выбранный период времени и данными за месяц перед выбранным и за прошлый год выбранного месяца.

* + 1. **Функция смены шрифта и размера шрифта таблиц**

Данная функция позволяет изменять шрифт и его размер в таблицах программы на заданный пользователем.

* + 1. **Функция общего экспорта данных**

Данная функция позволяет экспортировать все данные по выбранным параметрам в Excel файл.

* + 1. **Функция экспорта данных о** **лицах с ОВЗ, инвалидах, детях-инвалидах**

Данная функция позволяет экспортировать данные о лицах с ОВЗ, инвалидах, детях-инвалидах по выбранным параметрам в Excel файл.

* + 1. **Функция экспорта данных о лицах на места в рамках квоты целевого приема**

Данная функция позволяет экспортировать данные о лицах на места в рамках квоты целевого приема по выбранным параметрам в Excel файл.

* + 1. **Функция экспорта данных об иностранных гражданах**

Данная функция позволяет экспортировать данные об иностранных гражданах по выбранным параметрам в Excel файл.

* + 1. **Функция пользовательского экспорта данных**

Данная функция позволяет экспортировать данные выбранные пользователем по выбранным параметрам в Excel файл.

1. **Условия выполнения программы**
   1. **Минимальный состав аппаратных средств**

ОС: Windows 10

Процессор: Как минимум 1 ГГц или SoC.

ОЗУ: 1 ГБ (для 32-разрядных систем) или 2 ГБ (для 64-разрядных систем).

Место на жестком диске: 16 ГБ (для 32-разрядных систем) или 20 ГБ (для 64-разрядных систем).

Видеоадаптер: DirectX версии не ниже 9 с драйвером WDDM 1.0.

Дисплей: 800 x 600.

* 1. **Минимальный состав программных средств**

Для работы программного средства требуется:

1. MSSQL Server 2019 Express или MSSQL Server 2019

2. SQL Server Management Studio 18.9.1

* 1. **Требование к персоналу (пользователю)**

Конечный пользователь программы должен обладать практическими навыками работы с графическим пользовательским интерфейсом операционной системы.

1. **Выполнение программы**
   1. **Загрузка и запуск программы**
   2. **Выполнение программы**
      1. **Выполнение функции импорта данных**

После нажатия пользователем кнопки «Загрузить данные» на панели задач или в меню, откроется окно где пользователь может выбрать файлы с данными. После этого данные из файлов, если они заполнены верно, будут загружены в БД и в программу для дальнейшей работы.

* + 1. **Выполнение функции смены системы образования**

Пользователь может нажать на вкладыш с названием уровня образования и получить данные о движении контингента по выбранному уровню образования

* + 1. **Выполнение функции выборки данных**

Пользователю доступны четыре критерия для выборки данных:

1. Форма обучения
2. Направление
3. Год
4. Месяц

Выбирая определенные значения этих критериев в таблицу будут подгружаться данные соответствующие описанным выше условиям.

* + 1. **Выполнение функции сортировки данных**

Пользователь может нажать ЛКМ на название колонки в любой таблице и данные будут отсортированы по возрастанию данных в этой колонке. Второе нажатие ЛКМ на колонку отсортирует таблицу по убыванию данных.

* + 1. **Выполнение функции сравнения данных**

Двойное нажатие ЛКМ на любую ячейку в таблице позволит пользователю увидеть сравнение данных за месяц, идущий до выбранного и за прошлый год выбранного месяца.

* + 1. **Выполнение функции смены шрифта и размера шрифта таблиц**

После нажатия пользователем кнопки «Шрифт» на панели задач или в меню откроется окно для выбора размера шрифта и самого шрифта. После выбора шрифта и его размера и нажатия кнопки «ОК» выбранный шрифт и размер шрифта будут применены ко всем таблицам в программе.

* + 1. **Выполнение функции общего экспорта**

После нажатия пользователем кнопки «Полный отчет» на панели задач или в меню откроется окно для выбора параметров для экспорта данных. После нажатия кнопки «ОК» откроется окно где пользователь выбирает название файла и его расположения. После нажатия кнопки «Сохранить» будет создан файл с данными соответствующими критериям которые выбрал пользователь. В данном файле будут храниться все данный соответствующие выбранным критериям.

* + 1. **Выполнение функции экспорта данных о лицах с ОВЗ, инвалидах, детях-инвалидах**

После нажатия пользователем кнопки «Лица с ОВЗ, инвалиды, дети-инвалиды» на панели задач или в меню откроется окно для выбора параметров для экспорта данных. После нажатия кнопки «ОК» откроется окно где пользователь выбирает название файла и его расположения. После нажатия кнопки «Сохранить» будет создан файл с данными соответствующими критериям которые выбрал пользователь. В данном файле будут храниться все данный соответствующие выбранным критериям о лицах с ОВЗ, инвалидах, детях-инвалидах.

* + 1. **Выполнение функции экспорта данных о лицах на места в рамках квоты целевого приема**

После нажатия пользователем кнопки «Места в рамках квоты целевого приема» на панели задач или в меню откроется окно для выбора параметров для экспорта данных. После нажатия кнопки «ОК» откроется окно где пользователь выбирает название файла и его расположения. После нажатия кнопки «Сохранить» будет создан файл с данными соответствующими критериям которые выбрал пользователь. В данном файле будут храниться все данный соответствующие выбранным критериям о лицах на места в рамках квоты целевого приема.

* + 1. **Выполнение функции экспорта данных об иностранных гражданах**

После нажатия пользователем кнопки «Иностранные граждане» на панели задач или в меню откроется окно для выбора параметров для экспорта данных. После нажатия кнопки «ОК» откроется окно где пользователь выбирает название файла и его расположения. После нажатия кнопки «Сохранить» будет создан файл с данными соответствующими критериям которые выбрал пользователь. В данном файле будут храниться все данный соответствующие выбранным критериям об иностранных гражданах.

* + 1. **Выполнение функции пользовательского экспорта данных**

После нажатия пользователем кнопки «Настраиваемый отчет» на панели задач или в меню откроется окно для выбора параметров для экспорта данных. После нажатия кнопки «ОК» откроется окно где пользователь выбирает название файла и его расположения. После нажатия кнопки «Сохранить» будет создан файл с данными соответствующими критериям которые выбрал пользователь. В данном файле будут храниться все данный соответствующие выбранным критериям и выбранным пунктам для экспорта.

* 1. **Завершение работы программы**

Чтобы завершить работу программы нажмите на крестик в верхнем правом углу окна программы.

1. **Сообщения оператору**
   1. **Сообщение об успешном импорте данных**

При успешном импорте данных появится сообщение об успешном импорте данных

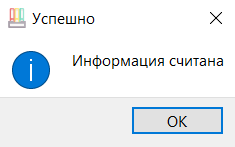


Рисунок 16. Сообщение об успешном импорте данных

* 1. **Сообщение о подаче не корректного файла**

При попытке импортировать данные из не корректно заполненного файла-формы появится сообщение о неправильно заполненном файле с его названием

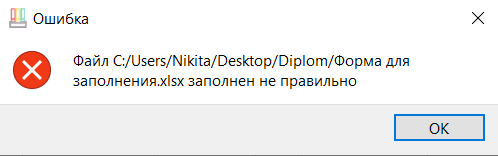


Рисунок 17. Сообщение о подаче не корректного файла

* 1. **Сообщение об успешном экспорте данных**

При успешном экспорте данных появится сообщение об успешном экспорте данных

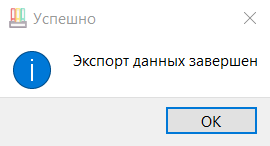


Рисунок 18. Сообщение об успешном экспорте данных.

### **Руководство системного программиста**

**АННОТАЦИЯ**

В данном программном документе приведено руководство системного программиста по настройке и использованию программы «StudInStudOut», предназначенной для облегчения мониторинга движения контингента студентов.

В данном программном документе, в разделе «Общие сведения о программе» указаны назначение и функции программы и сведения о технических и программных средствах, обеспечивающих выполнение данной программы, а также требования к персоналу.

В разделе «Структура программы» приведены сведения о структуре программы, ее составных частях, о связях между составными частями и о связях с другими программами.

В данном программном документе, в разделе «Настройка программы» приведено описание действий по настройке программы на условия конкретного применения (настройка на состав технических и программных средств, выбор функций и др.).

В разделе «Проверка программы» приведено описание способов проверки, позволяющих дать общее заключение о работоспособности программы (контрольные примеры, методы прогона, результаты).

В данном программном документе, в разделе «Сообщения системному программисту» указаны тексты сообщений, выдаваемых в ходе выполнения настройки, проверки программы, а также в ходе выполнения программы, описание их содержания и действий, которые необходимо предпринять по этим сообщениям.

Оформление программного документа «Руководство системного программиста» произведено по требованиям ЕСПД (ГОСТ 19.101-77[1], ГОСТ 19.103-77[2], ГОСТ 19.104-78\*[3], ГОСТ 19.105-78\*[4], ГОСТ 19.106-78\*[5], ГОСТ 19.503-79\*[6], ГОСТ 19.604-78\*[7]).

1. **Общие сведения о программе**
   1. **Назначение программы**

Специальное программное обеспечение «StudInStudOut» используется для хранения и просмотра данных о движении контингента студентов, а также для анализа этих данных и их экспорта в виде отчета.

Специальное программное обеспечение «StudInStudOut» может эксплуатироваться на объектах любого масштаба в сфере гостиничного бизнеса для облегчения работы персонала.

* 1. **Состав функций**
     1. **Функция импорта данных**

Данная функция позволяет импортировать данные из заполненной формы в БД и в программу.

* + 1. **Функция смены уровня образования**

Данная функция позволяет менять уровни образования и просматривать данные о движении контингента за выбранный уровень образования.

* + 1. **Функция выборки данных**

Данная функция позволяет производить выборку данных по определенным параметрам.

* + 1. **Функция сортировки данных**

Данная функция позволяет сортировать данные в таблице по выбранному столбцу.

* + 1. **Функция сравнения данных**

Данная функция позволяет сравнивать данные о движении контингента за выбранный период времени и данными за месяц перед выбранным и за прошлый год выбранного месяца.

* + 1. **Функция смены шрифта и размера шрифта таблиц**

Данная функция позволяет изменять шрифт и его размер в таблицах программы на заданный пользователем.

* + 1. **Функция общего экспорта данных**

Данная функция позволяет экспортировать все данные по выбранным параметрам в Excel файл.

* + 1. **Функция экспорта данных о лицах с ОВЗ, инвалидах, детях-инвалидах**

Данная функция позволяет экспортировать данные о лицах с ОВЗ, инвалидах, детях-инвалидах по выбранным параметрам в Excel файл.

* + 1. **Функция экспорта данных о лицах на места в рамках квоты целевого приема**

Данная функция позволяет экспортировать данные о лицах на места в рамках квоты целевого приема по выбранным параметрам в Excel файл.

* + 1. **Функция экспорта данных об иностранных гражданах**

Данная функция позволяет экспортировать данные об иностранных гражданах по выбранным параметрам в Excel файл.

* + 1. **Функция пользовательского экспорта данных**

Данная функция позволяет экспортировать данные выбранные пользователем по выбранным параметрам в Excel файл.

* 1. **Минимальный состав технических средств**

ОС: Windows 10

Процессор: Как минимум 1 ГГц или SoC.

ОЗУ: 1 ГБ (для 32-разрядных систем) или 2 ГБ (для 64-разрядных систем).

Место на жестком диске: 16 ГБ (для 32-разрядных систем) или 20 ГБ (для 64-разрядных систем).

Видеоадаптер: DirectX версии не ниже 9 с драйвером WDDM 1.0.

Дисплей: 800 x 600.

* 1. **Минимальный состав программных средств**

Для работы программного средства требуется:

1. MSSQL Server 2019 Express или MSSQL Server 2019

2. SQL Server Management Studio 18.9.1

* 1. **Требование к персоналу (системному программисту)**

Системный программист должен иметь минимум среднее техническое образование.

В перечень задач, выполняемых системным программистом, должны входить:

1. Задача поддержания работоспособности технических средств;
2. Задача установки (инсталляции) и поддержания работоспособности системных программных средств – операционной системы;
3. Задача установки (инсталляции) и поддержания работоспособности программы «StudInStudOut»
4. **Структура программы**
   1. **Сведения о структуре программы**

Структурно программа разделена на 2 подсистемы:

1. Подсистема «Публичная часть»
   1. Модуль главной экранной формы
   2. Модуль экранной формы пользовательского экспорта данных
   3. Модуль экранной формы встроенного экспорта данных
2. Подсистема «Закрытая часть»
   1. Модуль для работы с БД
   2. Модуль для работы с Excel
   3. Модуль для контроля интерфейса
   4. Модуль запуска программы
   5. **Сведения о составных частях программы**
3. Модуль главной экранной формы – содержит инициализацию главной формы
4. Модуль экранной формы пользовательского экспорта данных – содержит инициализацию формы пользовательского экспорта данных
5. Модуль экранной формы встроенного экспорта данных – содержит инициализацию формы встроенного экспорта данных
6. Модуль для работы с БД – контролирует и обеспечивает связь программы с БД
7. Модуль для работы с Excel – контролирует и обеспечивает возможность работы с файлами Excel для импорта и экспорта данных
8. Модуль для контроля интерфейса – контролирует все действия в программе и реагирует на все взаимодействия с ней
9. Модуль запуска программы – запускает программу
   1. **Сведения о связях между составными частями программы,**

Связи между составными частями программы показаны на рисунке ниже

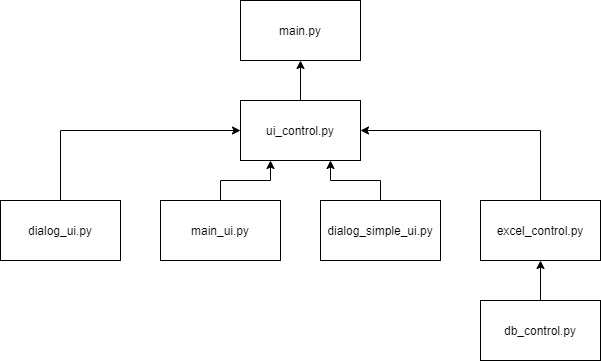


Рисунок 19. Связь между составными частями программы

* 1. **Сведения о связях с другими программами**

Программа во время своей работы обращается к БД для иморта и экспорта данных

1. **Настройка программы**
   1. **Настройка на состав технических средств**

Программа «StudInStudOut» не требует настройки на состав технических средств

* 1. **Настройка на состав программных средств**

Для установки и настройки программы требуется выполнить представленные ниже пункты:

1. Поместить StudInStudOut.exe в один каталог с приложенным каталогом «Изображения» и приложенным файлом «Config.txt»
2. Запустить SSMS установленный ранее и скопировать содержание строки «Server name» и вставить ее в файл «Config.txt»

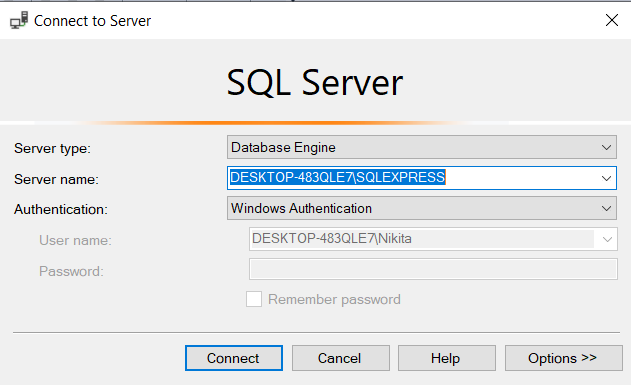


Рисунок 20. Строка "Server name"

1. Нажать «Connect» в SSMS, далее «Open file» выбрать приложенный скрипт «DB\_script» и после его открытия нажать «Execute»
2. **Проверка программы**
   1. **Проверка на соединение с БД**

Для проверки на соединение с БД необходимо запустить программу и нажать кнопку на панели инструментов «Полный отчет» и далее в открытом окне нажать продолжить, ввести любое название для файла и нажать сохранить. Если после нажатия появляется окно которое говорит об успешном экспорте, то соединение с БД присутствует.

Дальнейших тестов не требуется.

1. **Сообщения системному программисту**

Специальных сообщений системному программисту не предусмотрено, все сообщения предназначены для оператора и ознакомиться с ними можно в руководстве оператора.

## **Раздел техники безопасности**

При эксплуатации ПК на работника могут оказывать действия, следующие опасные и вредные производственные факторы:

* Повышенный уровень электромагнитных излучений;
* Повышенный уровень статического электричества;
* Повышенная ионизация воздуха;
* Статические физические перегрузки;
* Перенапряжение зрительных анализаторов;

Работник обязан:

* Выполнять только ту работу, которая определена его должностной инструкцией.
* Содержать в чистоте рабочее место.
* Соблюдать режим труда и отдыха в зависимости от продолжительности, вида и категории трудовой деятельности.
* Соблюдать меры пожарной безопасности.

К работе с ПК допускаются лица: имеющие персональные навыки, изучившие руководство по эксплуатации персонального компьютера и знающие порядок включения и отключения электронных устройств, а также программ; прошедшие вводный инструктаж, а также инструктаж по безопасности труда непосредственно на рабочем месте. Профессиональные пользователи должны проходить обязательные предварительные (при поступлении на работу) и периодические медосмотры. К непосредственной работе с персональным компьютером допускаются лица, не имеющие медицинских противопоказаний.

Также следует иметь в виду, что:

* Расстояние между рабочими столами с компьютерами в направлении тыла поверхности одного видеомонитора и экрана другого видеомонитора.
* Площадь на одно рабочее место ПК должна составлять не менее 6,0 кв. м, а объем - не менее 20,0 куб. м.
* Верхний край экрана следует располагать на уровне глаз или чуть ниже.
* Оптимальное расстояние от глаз до экрана 600-700 мм, но не ближе 500 мм.
* Высота клавиатуры должна быть отрегулирована так, чтобы кисти рук держались прямо (возможно применение подставок для кистей рук).
* С целью снижения статического напряжения мышц шейно-плечевой области и спины для предупреждения развития утомления необходимо, чтобы рабочий стул кресло) позволял изменять позу, был подъемно - поворотным и регулируемым по высоте и углам наклона сидения и спинки от переднего сидения.
* В производственных помещениях, в которых работа на ПК является основной должны обеспечиваться оптимальные параметры микроклимата. Для нормализации аэроионного фактора помещений с компьютерами необходимо использовать устройства автоматического регулирования ионного режима воздушной среды.
* Рабочие места с ПК при выполнении творческой работы, требующей значительного умственного напряжения или высокой концентрации внимания, следует изолировать друг от друга перегородками высотой 1,5-2,0 м.
* В помещениях с ПК ежедневно должна проводится влажная уборка.
* Помещения с ПК должны быть оснащены аптечкой первой помощи и углекислотными огнетушителями.
* Кабели компьютера должны располагаться так, чтобы их нельзя было повредить неосторожным движением.
* Для предотвращения образовании и защиты от статического электричества, в помещении, где установлен компьютер, необходимо применять нейтрализаторы и увлажнители. Должна быть эффективная вентиляция и поддерживаться относительная влажность воздуха на уровне 40-60 % (можно разместить вблизи компьютера цветы или аквариум).

Для обеспечения оптимальной работоспособности и сохранности здоровья пользователей, на протяжении рабочей смены должны устанавливаться регламентированные перерывы. Продолжительность непрерывной работы с персональным компьютером без регламентированного перерыва не должна превышать 2 часов.

Невыполнение требований является нарушением производственной дисциплины. Виновные в этом несут ответственность, в порядке, установленном действующим законодательством.

При начале работы с ПК, работник обязан:

* Осмотреть и привести в порядок рабочее место.
* Отрегулировать освещенность на рабочем месте, убедиться в достаточности освещенности, отсутствии бликов на экране.
* Проверить правильность подключения оборудования в электросеть.
* Проверить исправность проводящих проводов и отсутствие оголённых участков проводов.
* Убедиться в наличии защитного заземления.
* Протереть салфеткой поверхность экрана.
* Убедиться в отсутствии дисков в дисководах ПК.
* Проверить правильность установки стола, стула, положения оборудования, угла наклона экрана, положение клавиатуры, положение мыши на специальном коврике, а также расположение элементов компьютера в соответствии с требованиями эргономики и в целях исключения длительных напряжений тела.

Работнику запрещается приступать к работе с ПК при:

* Отсутствии специальной вилки с подключением заземления.
* Обнаружении неисправности оборудования.
* При размещении ПК в ряд на расстоянии менее 1,2 м, при расположении рабочих мест с компьютерами в колонку на расстоянии менее 2,0 м, при рядном расположении дисплеев экранами друг к другу.
* Работнику запрещается протирать влажной или мокрой салфеткой электрооборудование, которое находится под напряжением. Влажную или любую другую уборку производить только при отключенном оборудовании.
* Работник обязан сообщить руководителю об обнаруженной неисправности оборудования. Не пользоваться неисправным оборудованием. Приступить к работе только после устранения нарушений в работе или неисправностей оборудования.

Работник не должен разрешать работать лицам, не имеющим допуска к работе с опасным оборудованием или с ПК.

## **Расчет экономической эффективности применения программного комплекса**

**Расчет стоимости одного часа работы ПК**

Стоимость одного часа работы ПК производится по формуле:

,

где:

– затраты на содержание и обслуживание ПК;

– полезный фонд времени работы на ПК;

,

где:

– зарплата обслуживающего персонала;

– страховые взносы, зачисляемые в государственные внебюджетные фонды;

– амортизационные отчисления;

– затраты на ремонт оборудования;

– затраты на электроэнергию;

– затраты на материалы.

**Расчет зарплаты обслуживающего персонала**

Таблица 1. Обслуживающий персонал

|  |  |  |  |
| --- | --- | --- | --- |
| Обслуживающий персонал | Заработная плата в месяц (руб.) | Количество работников (чел.) | Фонд оплаты труда (руб.) |
| Диспетчер | 30000 | 1 | 30000 |
| ИТОГО | | | **30000** |

,

,

где:

– заработная плата работника в месяц,

– количество обслуживаемых работником машин,

– количество машин, на которых установлена программа.

= 1 (шт.),

= 1 (шт.),

= (30000 / 1) \* 1 = 30000 (руб.),

= 30000 (руб.)

**Расчет страховых взносов, зачисляемых в государственные внебюджетные фонды**

,

где:

* – взносы на социальное страхование и обеспечение (%).

Таблица 2. Ставки страховых взносов, зачисляемых в государственные внебюджетные фонды

|  |  |
| --- | --- |
| Государственные внебюджетные фонды | Ставка налога (%) |
| Пенсионный фонд РФ | 22 |
| Фонд социального страхования РФ | 2,9 |
| Фонд обязательного медицинского страхования РФ | 5,1 |
| Страховые взносы на социальное страхование от несчастных случаев на производстве и профзаболеваний | 0,2 |
| **ИТОГО** | **30,2** |

= (30000 \* 30,2) / 100% = 9060 (руб.)

Расчет амортизационных отчислений в месяц**:**

,

где:

* – первоначальная стоимость оборудования,
* – ставка годовой нормы амортизации (%), = 20%.

Таблица 3. Состав оборудования

|  |  |  |  |
| --- | --- | --- | --- |
| Наименование оборудования | Цена за единицу (руб.) | Количество единиц оборудования | Стоимость оборудования (руб.) |
| Ноутбук Acer EX215-51K (NX.EFPER.008) | 30000 | 1 | 30000 |
| Считывающее устройство RFID 522 | 1000 | 1 | 1000 |
| ИТОГО | | | **31000** |

= (31000 \* 20) / (100 \* 12) = 517 (руб.)

Расчет на ремонт оборудования

,

где:

* – ставка годовых затрат на планово-предупредительный ремонт оборудования (%).

= 10%,

= (31000 \* 10) / (100 \* 12) = 258 (руб.)

**Расчет затрат на электроэнергию**

,

где:

* – общая мощность оборудования,
* – полезный фонд времени работы ПК,
* – цена 1 Квт/ч (руб.).

,

где:

* – режимный фонд работы оборудования (час),
* – время на профилактику и ремонт оборудования (час).

,

где:

* – количество календарных дней за отчетный период,
* – количество выходных дней за отчетный период,
* – количество праздничных дней за отчетный период,
* – количество предпраздничных дней за отчетный период,
* – длительность смены (час),
* –количество смен,
* – количество часов, на которые сокращается рабочий день в предпраздничные дни.

,

где:

* – ставка плановых потерь на профилактику и ремонт оборудования (%).

Таблица 4. Общая мощность оборудования

|  |  |  |  |
| --- | --- | --- | --- |
| Наименование оборудования | Мощность единицы оборудования (Квт/час) | Количество оборудования | Суммарная мощность (Квт/час) |
| Ноутбук Acer EX215-51K (NX.EFPER.008) | 0,3 | 1 | 0,3 |
| ИТОГО | | | **0,3** |

Расчет ведется за май,

= 4,34,

= 10,

= 8 (часов),

= 1,

= 1 (час),

= (21 – 8 – 3) \* 8 \* 1 – 0 \* 1 = 100 (часов),

= (2 \* 10) / 100 = 20 (часов),

= 100 – 20 = 80 (часа),

= 0,3 \* 80 \* 4,34 = 104.16(руб.)

**Расчет затрат на материалы**

По данным бухгалтерии расходы на материалы составили 284,5 руб.

= 284,5.

**Расчет стоимости часа работы на ПК**

= 30000 + 9060 + 517+ 258 + 104,16 + 284,5= 40223,66 (руб.),

= 40223,66 / (144 \* 1) = 279,33 (руб.)

Таблица 5. Структура стоимости работы на ПК

| № | Элемент затрат | Сумма (руб.) | Удельный вес  (%) |
| --- | --- | --- | --- |
| 1 | Заработная плата обслуживающего персонала | 30000 | 74,58 |
| 2 | Страховые взносы, зачисляемые в государственные внебюджетные фонды | 9060 | 22,52 |
| 3 | Амортизационные отчисления | 517 | 1,29 |
| 4 | Затраты на ремонт оборудования | 258 | 0,64 |
| 5 | Затраты на электроэнергию | 104,16 | 0,26 |
| 6 | Затраты на материалы | 284,5 | 0,71 |
| ИТОГО | | **40223,66** | **100,00** |

**Расчет затрат на составление и отладку программы**

**Расчет стоимости решения задачи на ПК**

Стоимость решения задачи на ПК рассчитывается по формуле:

,

где:

* – заработная плата руководителя,
* – заработная плата студента,
* – страховые взносы, зачисляемые в государственные внебюджетные фонды,
* – накладные расходы,
* – затраты на материалы, используемые при составлении и отладке программы,
* – затраты времени на отладку программы и решения ее на ПК.

Расчет заработной платы руководителя

,

где:

* – время, отводимое на консультации по дипломному проектированию,
* – часовая тарифная ставка руководителя,
* – количество студентов у руководителя.

= 13 (часов),

= 300 (руб.),

= 4 (чел.),

= (13 \* 300) / 4 = 975 (руб.)

**Расчет заработной платы студента**

,

где:

* – стипендия студента,
* – количество учебных часов в месяце,

– затраты времени на отладку программы и решение задачи на ПК.

,

где:

– средняя занятость в течение дня,

– количество дней для отладки программы.

,

где:

Др – количество рабочих дней в месяце.

= С 1 января 2021 года в Московской области действует минимальная заработная плата в размере 15000 рублей (Соглашение о минимальной заработной плате в Московской области от 01.03.18 № 41). Именно такую минимальную зарплату применяют все работодатели области, которые в январе 2019 года не перешли на применение федерального МРОТ.

Согласно ст.133 ТК РФ, оклад сотрудников не должен быть ниже установленного регионального МРОТ. Устанавливается он путем регионального соглашения о минимальном размере заработной платы. Соглашение относительно минимальной зарплаты в Московской обл. правительством, областным объединением организаций профсоюзов и работодателей Московской обл. от 01декабря 2016г.).

= 6 (часа),

= 5 (дн.),

= 24 \* 6 = 80 (часов),

= 6 \* 5 = 30 (часов),

= (15000 \* 30) / 144 = 3125 (руб.)

**Расчет страховых взносов, зачисляемых в государственные внебюджетные фонды**

,

где:

– взносы на социальное страхование и обеспечение (%).

= 30,2;

= (975 \* 30,2) / 100 = 294,45 (руб.)

**Затраты на материалы, используемые при составлении и отладке программы**

Таблица 6. Затраты на материалы, используемые при составлении и отладке программы

|  |  |  |  |
| --- | --- | --- | --- |
| Наименование материалов | Цена за единицу (руб.) | Количество | Стоимость (руб.) |
| Бумага | 1,50 | 95 | 142,5 |
| Файлы (А4) | 142 (1 упаковка) | 1 | 142 |
| ИТОГО | | | **284,5** |

= 284,5 (руб.),

= 975 + 3125 + 294,45 + 284,5 + 30 \* 231,91 = 11636,25 (руб.)

**Расчет стоимости решения задачи вручную**

Стоимость решения задачи вручную рассчитывается по формуле:

,

где:

* – затраты времени на решение задачи вручную,
* – часовая ставка студента, решающего задачу (руб.).

,

= 13750 / 144 = 95,49 (руб.),

= 6 (часа),

= 6 \* 95,49 \* (1 + (30,2 / 100) = 745,97 (руб.)

**Расчет показателей эффективности применения ПК**

Расчет экономии труда за счет применения ПК

,

где:

– предполагаемое число обращений к программе,

– затраты времени на составление и отладку программы на ПК,

– время решения задачи на ПК.

,

где:

– количество дней на составление и отладку программы на ПК.

,

где:

– время ввода исходной информации,

– машинное время,

– время вывода информации на печать.

= 0,02 (часа),

= 0,0013 (часа),

, = 0,00026 (часа),

= 0,02 + 0,0013 + 0,00026 = 0,02156(часа),

= 4 \* 2 = 8 (часов),

,

= (375 х 1) + (375 / 2) х 1 = 563,

= 4 \* 563 - (8 + 563 \* 0,02156) = 2231,86 (руб.)

**Расчет денежной экономической эффективности применения ПК**

,

где:

– затраты на работу программы.

,

= 0,02156 \* 279,33 = 6,02 (руб.),

= 745,97 \* 563 – (11636,25 + 563 \* 6,02) = 404955,6 (руб.)

**Расчет роста производительности труда от применения ПК**

,

= (2231,86 / (6 \* 563)) \* 100% = 66,07 %.

**Вывод**

Были рассчитаны следующие экономические показатели: стоимость одного часа работы ПК 279,33 (руб.), затраты на составление и отладку программы 11636,25 (руб.), рост производительности труда составляет 66,07 %.

Учитывая вышеприведенный показатели, можно сделать вывод, что проект экономически эффективен и принесет прибыль после своего внедрения.

# **Заключение**

В данном дипломном проекте была разработана программа для мониторинга движения контингента студентов «StudInStudOut» для облечгения работы людей, работающих с этими данными.

Были продемонстрированы существующие разработки с их недостатками.

Были написаны руководство оператора и руководство системного программиста для тех, кто будет работать с данной программой и настраивать ее.

Данная программа обладает большим потенциалом для улучшения и разширения функционала чтобы упростить работы с даннми о движении контингента еще сильнее.

# **Список использованных источников информации**

1. ГОСТ 19.101-77 ЕСПД. Виды программ и программных документов
2. ГОСТ 19.103-77 ЕСПД. Обозначение программ и программных документов
3. ГОСТ 19.104-78\* ЕСПД. Основные надписи
4. ГОСТ 19.105-78\* ЕСПД. Общие требования к программным документам
5. ГОСТ 19.106-78\* ЕСПД. Общие требования к программным документам, выполненным печатным способом
6. ГОСТ 19.505-79\* ЕСПД. Руководство оператора. Требования к содержанию и оформлению
7. ГОСТ 19.604-78\* ЕСПД. Правила внесения изменений в программные документы, выполненные печатным способом
8. Основы алгоритмизации и программирования на Python - <https://znanium.com/catalog/document?id=368498>
9. Язык программирования высокого уровня Python. Функции, структуры данных, дополнительные модули - <https://znanium.com/catalog/document?id=339835>

# **Приложения**

**Приложение 1. Код программы**

**Листинг 1. main.py**

import sys

from ui\_control import UiControl

from PyQt5 import QtWidgets

if \_\_name\_\_ == "\_\_main\_\_":

app = QtWidgets.QApplication(sys.argv)

window = QtWidgets.QMainWindow()

interface = UiControl(window)

window.showMaximized()

sys.exit(app.exec\_())

**Листинг 2. main\_ui.py**

from PyQt5 import QtCore, QtGui, QtWidgets

class UiWindow(object):

def \_\_init\_\_(self, window):

"""Конструктор класса UiWindow"""

super(UiWindow, self).\_\_init\_\_()

window.setObjectName("MainWindow")

window.resize(743, 367)

icon = QtGui.QIcon()

icon.addPixmap(QtGui.QPixmap("Изображения/free-icon-binder-4624080.png"), QtGui.QIcon.Normal, QtGui.QIcon.Off)

window.setWindowIcon(icon)

window.setIconSize(QtCore.QSize(30, 30))

self.centralwidget = QtWidgets.QWidget(window)

self.centralwidget.setObjectName("centralwidget")

self.gridLayout = QtWidgets.QGridLayout(self.centralwidget)

self.gridLayout.setObjectName("gridLayout")

self.stackedWidget = QtWidgets.QStackedWidget(self.centralwidget)

font = QtGui.QFont()

font.setPointSize(14)

self.stackedWidget.setFont(font)

self.stackedWidget.setStyleSheet("")

self.stackedWidget.setObjectName("stackedWidget")

self.page = QtWidgets.QWidget()

self.page.setObjectName("page")

self.gridLayout\_4 = QtWidgets.QGridLayout(self.page)

self.gridLayout\_4.setObjectName("gridLayout\_4")

self.tabWidget = QtWidgets.QTabWidget(self.page)

font = QtGui.QFont()

font.setPointSize(14)

self.tabWidget.setFont(font)

self.tabWidget.setStyleSheet("background-color: rgb(211, 211, 211);")

self.tabWidget.setObjectName("tabWidget")

self.bachelor\_course = QtWidgets.QWidget()

self.bachelor\_course.setStyleSheet("")

self.bachelor\_course.setObjectName("bachelor\_course")

self.gridLayout\_3 = QtWidgets.QGridLayout(self.bachelor\_course)

self.gridLayout\_3.setObjectName("gridLayout\_3")

self.grid\_layout\_b = QtWidgets.QGridLayout()

self.grid\_layout\_b.setObjectName("grid\_layout\_b")

self.foe\_label\_b = QtWidgets.QLabel(self.bachelor\_course)

self.foe\_label\_b.setMaximumSize(QtCore.QSize(16777215, 28))

font = QtGui.QFont()

font.setPointSize(14)

self.foe\_label\_b.setFont(font)

self.foe\_label\_b.setFocusPolicy(QtCore.Qt.NoFocus)

self.foe\_label\_b.setStyleSheet("background-color: rgb(255, 255, 255);")

self.foe\_label\_b.setAlignment(QtCore.Qt.AlignCenter)

self.foe\_label\_b.setObjectName("foe\_label\_b")

self.grid\_layout\_b.addWidget(self.foe\_label\_b, 0, 0, 1, 1)

self.foe\_combo\_box\_b = QtWidgets.QComboBox(self.bachelor\_course)

font = QtGui.QFont()

font.setPointSize(14)

self.foe\_combo\_box\_b.setFont(font)

self.foe\_combo\_box\_b.setStyleSheet("background-color: rgb(255, 255, 255);")

self.foe\_combo\_box\_b.setObjectName("foe\_combo\_box\_b")

self.foe\_combo\_box\_b.addItem("")

self.foe\_combo\_box\_b.addItem("")

self.foe\_combo\_box\_b.addItem("")

self.foe\_combo\_box\_b.addItem("")

self.grid\_layout\_b.addWidget(self.foe\_combo\_box\_b, 1, 0, 1, 1)

self.direction\_label\_b = QtWidgets.QLabel(self.bachelor\_course)

self.direction\_label\_b.setMaximumSize(QtCore.QSize(16777215, 28))

font = QtGui.QFont()

font.setPointSize(14)

self.direction\_label\_b.setFont(font)

self.direction\_label\_b.setFocusPolicy(QtCore.Qt.NoFocus)

self.direction\_label\_b.setStyleSheet("background-color: rgb(255, 255, 255);")

self.direction\_label\_b.setAlignment(QtCore.Qt.AlignCenter)

self.direction\_label\_b.setObjectName("direction\_label\_b")

self.grid\_layout\_b.addWidget(self.direction\_label\_b, 0, 1, 1, 1)

self.year\_combo\_box\_b = QtWidgets.QComboBox(self.bachelor\_course)

font = QtGui.QFont()

font.setPointSize(14)

self.year\_combo\_box\_b.setFont(font)

self.year\_combo\_box\_b.setStyleSheet("background-color: rgb(255, 255, 255);")

self.year\_combo\_box\_b.setObjectName("year\_combo\_box\_b")

self.year\_combo\_box\_b.addItem("")

self.grid\_layout\_b.addWidget(self.year\_combo\_box\_b, 1, 2, 1, 1)

self.year\_label\_b = QtWidgets.QLabel(self.bachelor\_course)

self.year\_label\_b.setMaximumSize(QtCore.QSize(16777215, 28))

font = QtGui.QFont()

font.setPointSize(14)

self.year\_label\_b.setFont(font)

self.year\_label\_b.setFocusPolicy(QtCore.Qt.NoFocus)

self.year\_label\_b.setStyleSheet("background-color: rgb(255, 255, 255);")

self.year\_label\_b.setFrameShape(QtWidgets.QFrame.NoFrame)

self.year\_label\_b.setAlignment(QtCore.Qt.AlignCenter)

self.year\_label\_b.setObjectName("year\_label\_b")

self.grid\_layout\_b.addWidget(self.year\_label\_b, 0, 2, 1, 1)

self.month\_combo\_box\_b = QtWidgets.QComboBox(self.bachelor\_course)

font = QtGui.QFont()

font.setPointSize(14)

self.month\_combo\_box\_b.setFont(font)

self.month\_combo\_box\_b.setStyleSheet("background-color: rgb(255, 255, 255);")

self.month\_combo\_box\_b.setObjectName("month\_combo\_box\_b")

self.month\_combo\_box\_b.addItem("")

self.month\_combo\_box\_b.addItem("")

self.month\_combo\_box\_b.addItem("")

self.month\_combo\_box\_b.addItem("")

self.month\_combo\_box\_b.addItem("")

self.month\_combo\_box\_b.addItem("")

self.month\_combo\_box\_b.addItem("")

self.month\_combo\_box\_b.addItem("")

self.month\_combo\_box\_b.addItem("")

self.month\_combo\_box\_b.addItem("")

self.month\_combo\_box\_b.addItem("")

self.month\_combo\_box\_b.addItem("")

self.month\_combo\_box\_b.addItem("")

self.grid\_layout\_b.addWidget(self.month\_combo\_box\_b, 1, 3, 1, 1)

self.direction\_combo\_box\_b = QtWidgets.QComboBox(self.bachelor\_course)

font = QtGui.QFont()

font.setPointSize(14)

self.direction\_combo\_box\_b.setFont(font)

self.direction\_combo\_box\_b.setStyleSheet("background-color: rgb(255, 255, 255);")

self.direction\_combo\_box\_b.setObjectName("direction\_combo\_box\_b")

self.direction\_combo\_box\_b.addItem("")

self.grid\_layout\_b.addWidget(self.direction\_combo\_box\_b, 1, 1, 1, 1)

self.month\_label\_b = QtWidgets.QLabel(self.bachelor\_course)

self.month\_label\_b.setMaximumSize(QtCore.QSize(16777215, 28))

font = QtGui.QFont()

font.setPointSize(14)

self.month\_label\_b.setFont(font)

self.month\_label\_b.setFocusPolicy(QtCore.Qt.NoFocus)

self.month\_label\_b.setStyleSheet("background-color: rgb(255, 255, 255);")

self.month\_label\_b.setAlignment(QtCore.Qt.AlignCenter)

self.month\_label\_b.setObjectName("month\_label\_b")

self.grid\_layout\_b.addWidget(self.month\_label\_b, 0, 3, 1, 1)

self.gridLayout\_3.addLayout(self.grid\_layout\_b, 0, 0, 1, 1)

self.table\_widget\_b = QtWidgets.QTableWidget(self.bachelor\_course)

font = QtGui.QFont()

font.setPointSize(10)

self.table\_widget\_b.setFont(font)

self.table\_widget\_b.setStyleSheet("background-color: rgb(255, 255, 255);")

self.table\_widget\_b.setObjectName("table\_widget\_b")

self.table\_widget\_b.setColumnCount(0)

self.table\_widget\_b.setRowCount(0)

self.gridLayout\_3.addWidget(self.table\_widget\_b, 1, 0, 1, 1)

self.tabWidget.addTab(self.bachelor\_course, "")

self.master\_course = QtWidgets.QWidget()

self.master\_course.setObjectName("master\_course")

self.gridLayout\_6 = QtWidgets.QGridLayout(self.master\_course)

self.gridLayout\_6.setObjectName("gridLayout\_6")

self.grid\_layout\_m = QtWidgets.QGridLayout()

self.grid\_layout\_m.setObjectName("grid\_layout\_m")

self.foe\_label\_m = QtWidgets.QLabel(self.master\_course)

self.foe\_label\_m.setMaximumSize(QtCore.QSize(16777215, 28))

font = QtGui.QFont()

font.setPointSize(14)

self.foe\_label\_m.setFont(font)

self.foe\_label\_m.setFocusPolicy(QtCore.Qt.NoFocus)

self.foe\_label\_m.setStyleSheet("background-color: rgb(255, 255, 255);")

self.foe\_label\_m.setAlignment(QtCore.Qt.AlignCenter)

self.foe\_label\_m.setObjectName("foe\_label\_m")

self.grid\_layout\_m.addWidget(self.foe\_label\_m, 0, 0, 1, 1)

self.foe\_combo\_box\_m = QtWidgets.QComboBox(self.master\_course)

font = QtGui.QFont()

font.setPointSize(14)

self.foe\_combo\_box\_m.setFont(font)

self.foe\_combo\_box\_m.setStyleSheet("background-color: rgb(255, 255, 255);")

self.foe\_combo\_box\_m.setObjectName("foe\_combo\_box\_m")

self.foe\_combo\_box\_m.addItem("")

self.foe\_combo\_box\_m.addItem("")

self.foe\_combo\_box\_m.addItem("")

self.foe\_combo\_box\_m.addItem("")

self.grid\_layout\_m.addWidget(self.foe\_combo\_box\_m, 1, 0, 1, 1)

self.direction\_label\_m = QtWidgets.QLabel(self.master\_course)

self.direction\_label\_m.setMaximumSize(QtCore.QSize(16777215, 28))

font = QtGui.QFont()

font.setPointSize(14)

self.direction\_label\_m.setFont(font)

self.direction\_label\_m.setFocusPolicy(QtCore.Qt.NoFocus)

self.direction\_label\_m.setStyleSheet("background-color: rgb(255, 255, 255);")

self.direction\_label\_m.setAlignment(QtCore.Qt.AlignCenter)

self.direction\_label\_m.setObjectName("direction\_label\_m")

self.grid\_layout\_m.addWidget(self.direction\_label\_m, 0, 1, 1, 1)

self.year\_combo\_box\_m = QtWidgets.QComboBox(self.master\_course)

font = QtGui.QFont()

font.setPointSize(14)

self.year\_combo\_box\_m.setFont(font)

self.year\_combo\_box\_m.setStyleSheet("background-color: rgb(255, 255, 255);")

self.year\_combo\_box\_m.setObjectName("year\_combo\_box\_m")

self.year\_combo\_box\_m.addItem("")

self.grid\_layout\_m.addWidget(self.year\_combo\_box\_m, 1, 2, 1, 1)

self.year\_label\_m = QtWidgets.QLabel(self.master\_course)

self.year\_label\_m.setMaximumSize(QtCore.QSize(16777215, 28))

font = QtGui.QFont()

font.setPointSize(14)

self.year\_label\_m.setFont(font)

self.year\_label\_m.setFocusPolicy(QtCore.Qt.NoFocus)

self.year\_label\_m.setStyleSheet("background-color: rgb(255, 255, 255);")

self.year\_label\_m.setFrameShape(QtWidgets.QFrame.NoFrame)

self.year\_label\_m.setAlignment(QtCore.Qt.AlignCenter)

self.year\_label\_m.setObjectName("year\_label\_m")

self.grid\_layout\_m.addWidget(self.year\_label\_m, 0, 2, 1, 1)

self.month\_combo\_box\_m = QtWidgets.QComboBox(self.master\_course)

font = QtGui.QFont()

font.setPointSize(14)

self.month\_combo\_box\_m.setFont(font)

self.month\_combo\_box\_m.setStyleSheet("background-color: rgb(255, 255, 255);")

self.month\_combo\_box\_m.setObjectName("month\_combo\_box\_m")

self.month\_combo\_box\_m.addItem("")

self.month\_combo\_box\_m.addItem("")

self.month\_combo\_box\_m.addItem("")

self.month\_combo\_box\_m.addItem("")

self.month\_combo\_box\_m.addItem("")

self.month\_combo\_box\_m.addItem("")

self.month\_combo\_box\_m.addItem("")

self.month\_combo\_box\_m.addItem("")

self.month\_combo\_box\_m.addItem("")

self.month\_combo\_box\_m.addItem("")

self.month\_combo\_box\_m.addItem("")

self.month\_combo\_box\_m.addItem("")

self.month\_combo\_box\_m.addItem("")

self.grid\_layout\_m.addWidget(self.month\_combo\_box\_m, 1, 3, 1, 1)

self.direction\_combo\_box\_m = QtWidgets.QComboBox(self.master\_course)

font = QtGui.QFont()

font.setPointSize(14)

self.direction\_combo\_box\_m.setFont(font)

self.direction\_combo\_box\_m.setStyleSheet("background-color: rgb(255, 255, 255);")

self.direction\_combo\_box\_m.setObjectName("direction\_combo\_box\_m")

self.direction\_combo\_box\_m.addItem("")

self.grid\_layout\_m.addWidget(self.direction\_combo\_box\_m, 1, 1, 1, 1)

self.month\_label\_m = QtWidgets.QLabel(self.master\_course)

self.month\_label\_m.setMaximumSize(QtCore.QSize(16777215, 28))

font = QtGui.QFont()

font.setPointSize(14)

self.month\_label\_m.setFont(font)

self.month\_label\_m.setFocusPolicy(QtCore.Qt.NoFocus)

self.month\_label\_m.setStyleSheet("background-color: rgb(255, 255, 255);")

self.month\_label\_m.setAlignment(QtCore.Qt.AlignCenter)

self.month\_label\_m.setObjectName("month\_label\_m")

self.grid\_layout\_m.addWidget(self.month\_label\_m, 0, 3, 1, 1)

self.gridLayout\_6.addLayout(self.grid\_layout\_m, 0, 0, 1, 1)

self.table\_widget\_m = QtWidgets.QTableWidget(self.master\_course)

font = QtGui.QFont()

font.setPointSize(10)

self.table\_widget\_m.setFont(font)

self.table\_widget\_m.setStyleSheet("background-color: rgb(255, 255, 255);")

self.table\_widget\_m.setObjectName("table\_widget\_m")

self.table\_widget\_m.setColumnCount(0)

self.table\_widget\_m.setRowCount(0)

self.gridLayout\_6.addWidget(self.table\_widget\_m, 1, 0, 1, 1)

self.tabWidget.addTab(self.master\_course, "")

self.specialist\_degree = QtWidgets.QWidget()

self.specialist\_degree.setObjectName("specialist\_degree")

self.gridLayout\_8 = QtWidgets.QGridLayout(self.specialist\_degree)

self.gridLayout\_8.setObjectName("gridLayout\_8")

self.grid\_layout\_s = QtWidgets.QGridLayout()

self.grid\_layout\_s.setObjectName("grid\_layout\_s")

self.foe\_label\_s = QtWidgets.QLabel(self.specialist\_degree)

self.foe\_label\_s.setMaximumSize(QtCore.QSize(16777215, 28))

font = QtGui.QFont()

font.setPointSize(14)

self.foe\_label\_s.setFont(font)

self.foe\_label\_s.setFocusPolicy(QtCore.Qt.NoFocus)

self.foe\_label\_s.setStyleSheet("background-color: rgb(255, 255, 255);")

self.foe\_label\_s.setAlignment(QtCore.Qt.AlignCenter)

self.foe\_label\_s.setObjectName("foe\_label\_s")

self.grid\_layout\_s.addWidget(self.foe\_label\_s, 0, 0, 1, 1)

self.foe\_combo\_box\_s = QtWidgets.QComboBox(self.specialist\_degree)

font = QtGui.QFont()

font.setPointSize(14)

self.foe\_combo\_box\_s.setFont(font)

self.foe\_combo\_box\_s.setStyleSheet("background-color: rgb(255, 255, 255);")

self.foe\_combo\_box\_s.setObjectName("foe\_combo\_box\_s")

self.foe\_combo\_box\_s.addItem("")

self.foe\_combo\_box\_s.addItem("")

self.foe\_combo\_box\_s.addItem("")

self.foe\_combo\_box\_s.addItem("")

self.grid\_layout\_s.addWidget(self.foe\_combo\_box\_s, 1, 0, 1, 1)

self.direction\_label\_s = QtWidgets.QLabel(self.specialist\_degree)

self.direction\_label\_s.setMaximumSize(QtCore.QSize(16777215, 28))

font = QtGui.QFont()

font.setPointSize(14)

self.direction\_label\_s.setFont(font)

self.direction\_label\_s.setFocusPolicy(QtCore.Qt.NoFocus)

self.direction\_label\_s.setStyleSheet("background-color: rgb(255, 255, 255);")

self.direction\_label\_s.setAlignment(QtCore.Qt.AlignCenter)

self.direction\_label\_s.setObjectName("direction\_label\_s")

self.grid\_layout\_s.addWidget(self.direction\_label\_s, 0, 1, 1, 1)

self.year\_combo\_box\_s = QtWidgets.QComboBox(self.specialist\_degree)

font = QtGui.QFont()

font.setPointSize(14)

self.year\_combo\_box\_s.setFont(font)

self.year\_combo\_box\_s.setStyleSheet("background-color: rgb(255, 255, 255);")

self.year\_combo\_box\_s.setObjectName("year\_combo\_box\_s")

self.year\_combo\_box\_s.addItem("")

self.grid\_layout\_s.addWidget(self.year\_combo\_box\_s, 1, 2, 1, 1)

self.year\_label\_s = QtWidgets.QLabel(self.specialist\_degree)

self.year\_label\_s.setMaximumSize(QtCore.QSize(16777215, 28))

font = QtGui.QFont()

font.setPointSize(14)

self.year\_label\_s.setFont(font)

self.year\_label\_s.setFocusPolicy(QtCore.Qt.NoFocus)

self.year\_label\_s.setStyleSheet("background-color: rgb(255, 255, 255);")

self.year\_label\_s.setFrameShape(QtWidgets.QFrame.NoFrame)

self.year\_label\_s.setAlignment(QtCore.Qt.AlignCenter)

self.year\_label\_s.setObjectName("year\_label\_s")

self.grid\_layout\_s.addWidget(self.year\_label\_s, 0, 2, 1, 1)

self.month\_combo\_box\_s = QtWidgets.QComboBox(self.specialist\_degree)

font = QtGui.QFont()

font.setPointSize(14)

self.month\_combo\_box\_s.setFont(font)

self.month\_combo\_box\_s.setStyleSheet("background-color: rgb(255, 255, 255);")

self.month\_combo\_box\_s.setObjectName("month\_combo\_box\_s")

self.month\_combo\_box\_s.addItem("")

self.month\_combo\_box\_s.addItem("")

self.month\_combo\_box\_s.addItem("")

self.month\_combo\_box\_s.addItem("")

self.month\_combo\_box\_s.addItem("")

self.month\_combo\_box\_s.addItem("")

self.month\_combo\_box\_s.addItem("")

self.month\_combo\_box\_s.addItem("")

self.month\_combo\_box\_s.addItem("")

self.month\_combo\_box\_s.addItem("")

self.month\_combo\_box\_s.addItem("")

self.month\_combo\_box\_s.addItem("")

self.month\_combo\_box\_s.addItem("")

self.grid\_layout\_s.addWidget(self.month\_combo\_box\_s, 1, 3, 1, 1)

self.direction\_combo\_box\_s = QtWidgets.QComboBox(self.specialist\_degree)

font = QtGui.QFont()

font.setPointSize(14)

self.direction\_combo\_box\_s.setFont(font)

self.direction\_combo\_box\_s.setStyleSheet("background-color: rgb(255, 255, 255);")

self.direction\_combo\_box\_s.setObjectName("direction\_combo\_box\_s")

self.direction\_combo\_box\_s.addItem("")

self.grid\_layout\_s.addWidget(self.direction\_combo\_box\_s, 1, 1, 1, 1)

self.month\_label\_s = QtWidgets.QLabel(self.specialist\_degree)

self.month\_label\_s.setMaximumSize(QtCore.QSize(16777215, 28))

font = QtGui.QFont()

font.setPointSize(14)

self.month\_label\_s.setFont(font)

self.month\_label\_s.setFocusPolicy(QtCore.Qt.NoFocus)

self.month\_label\_s.setStyleSheet("background-color: rgb(255, 255, 255);")

self.month\_label\_s.setAlignment(QtCore.Qt.AlignCenter)

self.month\_label\_s.setObjectName("month\_label\_s")

self.grid\_layout\_s.addWidget(self.month\_label\_s, 0, 3, 1, 1)

self.gridLayout\_8.addLayout(self.grid\_layout\_s, 0, 0, 1, 1)

self.table\_widget\_s = QtWidgets.QTableWidget(self.specialist\_degree)

font = QtGui.QFont()

font.setPointSize(10)

self.table\_widget\_s.setFont(font)

self.table\_widget\_s.setStyleSheet("background-color: rgb(255, 255, 255);")

self.table\_widget\_s.setObjectName("table\_widget\_s")

self.table\_widget\_s.setColumnCount(0)

self.table\_widget\_s.setRowCount(0)

self.gridLayout\_8.addWidget(self.table\_widget\_s, 1, 0, 1, 1)

self.tabWidget.addTab(self.specialist\_degree, "")

self.CSET = QtWidgets.QWidget()

self.CSET.setObjectName("CSET")

self.gridLayout\_10 = QtWidgets.QGridLayout(self.CSET)

self.gridLayout\_10.setObjectName("gridLayout\_10")

self.grid\_layout\_CSET = QtWidgets.QGridLayout()

self.grid\_layout\_CSET.setObjectName("grid\_layout\_CSET")

self.foe\_label\_CSET = QtWidgets.QLabel(self.CSET)

self.foe\_label\_CSET.setMaximumSize(QtCore.QSize(16777215, 28))

font = QtGui.QFont()

font.setPointSize(14)

self.foe\_label\_CSET.setFont(font)

self.foe\_label\_CSET.setFocusPolicy(QtCore.Qt.NoFocus)

self.foe\_label\_CSET.setStyleSheet("background-color: rgb(255, 255, 255);")

self.foe\_label\_CSET.setAlignment(QtCore.Qt.AlignCenter)

self.foe\_label\_CSET.setObjectName("foe\_label\_CSET")

self.grid\_layout\_CSET.addWidget(self.foe\_label\_CSET, 0, 0, 1, 1)

self.foe\_combo\_box\_CSET = QtWidgets.QComboBox(self.CSET)

font = QtGui.QFont()

font.setPointSize(14)

self.foe\_combo\_box\_CSET.setFont(font)

self.foe\_combo\_box\_CSET.setStyleSheet("background-color: rgb(255, 255, 255);")

self.foe\_combo\_box\_CSET.setObjectName("foe\_combo\_box\_CSET")

self.foe\_combo\_box\_CSET.addItem("")

self.foe\_combo\_box\_CSET.addItem("")

self.foe\_combo\_box\_CSET.addItem("")

self.foe\_combo\_box\_CSET.addItem("")

self.grid\_layout\_CSET.addWidget(self.foe\_combo\_box\_CSET, 1, 0, 1, 1)

self.direction\_label\_CSET = QtWidgets.QLabel(self.CSET)

self.direction\_label\_CSET.setMaximumSize(QtCore.QSize(16777215, 28))

font = QtGui.QFont()

font.setPointSize(14)

self.direction\_label\_CSET.setFont(font)

self.direction\_label\_CSET.setFocusPolicy(QtCore.Qt.NoFocus)

self.direction\_label\_CSET.setStyleSheet("background-color: rgb(255, 255, 255);")

self.direction\_label\_CSET.setAlignment(QtCore.Qt.AlignCenter)

self.direction\_label\_CSET.setObjectName("direction\_label\_CSET")

self.grid\_layout\_CSET.addWidget(self.direction\_label\_CSET, 0, 1, 1, 1)

self.year\_combo\_box\_CSET = QtWidgets.QComboBox(self.CSET)

font = QtGui.QFont()

font.setPointSize(14)

self.year\_combo\_box\_CSET.setFont(font)

self.year\_combo\_box\_CSET.setStyleSheet("background-color: rgb(255, 255, 255);")

self.year\_combo\_box\_CSET.setObjectName("year\_combo\_box\_CSET")

self.year\_combo\_box\_CSET.addItem("")

self.grid\_layout\_CSET.addWidget(self.year\_combo\_box\_CSET, 1, 2, 1, 1)

self.year\_label\_CSET = QtWidgets.QLabel(self.CSET)

self.year\_label\_CSET.setMaximumSize(QtCore.QSize(16777215, 28))

font = QtGui.QFont()

font.setPointSize(14)

self.year\_label\_CSET.setFont(font)

self.year\_label\_CSET.setFocusPolicy(QtCore.Qt.NoFocus)

self.year\_label\_CSET.setStyleSheet("background-color: rgb(255, 255, 255);")

self.year\_label\_CSET.setFrameShape(QtWidgets.QFrame.NoFrame)

self.year\_label\_CSET.setAlignment(QtCore.Qt.AlignCenter)

self.year\_label\_CSET.setObjectName("year\_label\_CSET")

self.grid\_layout\_CSET.addWidget(self.year\_label\_CSET, 0, 2, 1, 1)

self.month\_combo\_box\_CSET = QtWidgets.QComboBox(self.CSET)

font = QtGui.QFont()

font.setPointSize(14)

self.month\_combo\_box\_CSET.setFont(font)

self.month\_combo\_box\_CSET.setStyleSheet("background-color: rgb(255, 255, 255);")

self.month\_combo\_box\_CSET.setObjectName("month\_combo\_box\_CSET")

self.month\_combo\_box\_CSET.addItem("")

self.month\_combo\_box\_CSET.addItem("")

self.month\_combo\_box\_CSET.addItem("")

self.month\_combo\_box\_CSET.addItem("")

self.month\_combo\_box\_CSET.addItem("")

self.month\_combo\_box\_CSET.addItem("")

self.month\_combo\_box\_CSET.addItem("")

self.month\_combo\_box\_CSET.addItem("")

self.month\_combo\_box\_CSET.addItem("")

self.month\_combo\_box\_CSET.addItem("")

self.month\_combo\_box\_CSET.addItem("")

self.month\_combo\_box\_CSET.addItem("")

self.month\_combo\_box\_CSET.addItem("")

self.grid\_layout\_CSET.addWidget(self.month\_combo\_box\_CSET, 1, 3, 1, 1)

self.direction\_combo\_box\_CSET = QtWidgets.QComboBox(self.CSET)

font = QtGui.QFont()

font.setPointSize(14)

self.direction\_combo\_box\_CSET.setFont(font)

self.direction\_combo\_box\_CSET.setStyleSheet("background-color: rgb(255, 255, 255);")

self.direction\_combo\_box\_CSET.setObjectName("direction\_combo\_box\_CSET")

self.direction\_combo\_box\_CSET.addItem("")

self.grid\_layout\_CSET.addWidget(self.direction\_combo\_box\_CSET, 1, 1, 1, 1)

self.month\_label\_CSET = QtWidgets.QLabel(self.CSET)

self.month\_label\_CSET.setMaximumSize(QtCore.QSize(16777215, 28))

font = QtGui.QFont()

font.setPointSize(14)

self.month\_label\_CSET.setFont(font)

self.month\_label\_CSET.setFocusPolicy(QtCore.Qt.NoFocus)

self.month\_label\_CSET.setStyleSheet("background-color: rgb(255, 255, 255);")

self.month\_label\_CSET.setAlignment(QtCore.Qt.AlignCenter)

self.month\_label\_CSET.setObjectName("month\_label\_CSET")

self.grid\_layout\_CSET.addWidget(self.month\_label\_CSET, 0, 3, 1, 1)

self.gridLayout\_10.addLayout(self.grid\_layout\_CSET, 0, 0, 1, 1)

self.table\_widget\_CSET = QtWidgets.QTableWidget(self.CSET)

font = QtGui.QFont()

font.setPointSize(10)

self.table\_widget\_CSET.setFont(font)

self.table\_widget\_CSET.setStyleSheet("background-color: rgb(255, 255, 255);")

self.table\_widget\_CSET.setObjectName("table\_widget\_CSET")

self.table\_widget\_CSET.setColumnCount(0)

self.table\_widget\_CSET.setRowCount(0)

self.gridLayout\_10.addWidget(self.table\_widget\_CSET, 1, 0, 1, 1)

self.tabWidget.addTab(self.CSET, "")

self.TSTD = QtWidgets.QWidget()

self.TSTD.setObjectName("TSTD")

self.gridLayout\_12 = QtWidgets.QGridLayout(self.TSTD)

self.gridLayout\_12.setObjectName("gridLayout\_12")

self.grid\_layout\_TSTD = QtWidgets.QGridLayout()

self.grid\_layout\_TSTD.setObjectName("grid\_layout\_TSTD")

self.foe\_label\_TSTD = QtWidgets.QLabel(self.TSTD)

self.foe\_label\_TSTD.setMaximumSize(QtCore.QSize(16777215, 28))

font = QtGui.QFont()

font.setPointSize(14)

self.foe\_label\_TSTD.setFont(font)

self.foe\_label\_TSTD.setFocusPolicy(QtCore.Qt.NoFocus)

self.foe\_label\_TSTD.setStyleSheet("background-color: rgb(255, 255, 255);")

self.foe\_label\_TSTD.setAlignment(QtCore.Qt.AlignCenter)

self.foe\_label\_TSTD.setObjectName("foe\_label\_TSTD")

self.grid\_layout\_TSTD.addWidget(self.foe\_label\_TSTD, 0, 0, 1, 1)

self.foe\_combo\_box\_TSTD = QtWidgets.QComboBox(self.TSTD)

font = QtGui.QFont()

font.setPointSize(14)

self.foe\_combo\_box\_TSTD.setFont(font)

self.foe\_combo\_box\_TSTD.setStyleSheet("background-color: rgb(255, 255, 255);")

self.foe\_combo\_box\_TSTD.setObjectName("foe\_combo\_box\_TSTD")

self.foe\_combo\_box\_TSTD.addItem("")

self.foe\_combo\_box\_TSTD.addItem("")

self.foe\_combo\_box\_TSTD.addItem("")

self.foe\_combo\_box\_TSTD.addItem("")

self.grid\_layout\_TSTD.addWidget(self.foe\_combo\_box\_TSTD, 1, 0, 1, 1)

self.direction\_label\_TSTD = QtWidgets.QLabel(self.TSTD)

self.direction\_label\_TSTD.setMaximumSize(QtCore.QSize(16777215, 28))

font = QtGui.QFont()

font.setPointSize(14)

self.direction\_label\_TSTD.setFont(font)

self.direction\_label\_TSTD.setFocusPolicy(QtCore.Qt.NoFocus)

self.direction\_label\_TSTD.setStyleSheet("background-color: rgb(255, 255, 255);")

self.direction\_label\_TSTD.setAlignment(QtCore.Qt.AlignCenter)

self.direction\_label\_TSTD.setObjectName("direction\_label\_TSTD")

self.grid\_layout\_TSTD.addWidget(self.direction\_label\_TSTD, 0, 1, 1, 1)

self.year\_combo\_box\_TSTD = QtWidgets.QComboBox(self.TSTD)

font = QtGui.QFont()

font.setPointSize(14)

self.year\_combo\_box\_TSTD.setFont(font)

self.year\_combo\_box\_TSTD.setStyleSheet("background-color: rgb(255, 255, 255);")

self.year\_combo\_box\_TSTD.setObjectName("year\_combo\_box\_TSTD")

self.year\_combo\_box\_TSTD.addItem("")

self.grid\_layout\_TSTD.addWidget(self.year\_combo\_box\_TSTD, 1, 2, 1, 1)

self.year\_label\_TSTD = QtWidgets.QLabel(self.TSTD)

self.year\_label\_TSTD.setMaximumSize(QtCore.QSize(16777215, 28))

font = QtGui.QFont()

font.setPointSize(14)

self.year\_label\_TSTD.setFont(font)

self.year\_label\_TSTD.setFocusPolicy(QtCore.Qt.NoFocus)

self.year\_label\_TSTD.setStyleSheet("background-color: rgb(255, 255, 255);")

self.year\_label\_TSTD.setFrameShape(QtWidgets.QFrame.NoFrame)

self.year\_label\_TSTD.setAlignment(QtCore.Qt.AlignCenter)

self.year\_label\_TSTD.setObjectName("year\_label\_TSTD")

self.grid\_layout\_TSTD.addWidget(self.year\_label\_TSTD, 0, 2, 1, 1)

self.month\_combo\_box\_TSTD = QtWidgets.QComboBox(self.TSTD)

font = QtGui.QFont()

font.setPointSize(14)

self.month\_combo\_box\_TSTD.setFont(font)

self.month\_combo\_box\_TSTD.setStyleSheet("background-color: rgb(255, 255, 255);")

self.month\_combo\_box\_TSTD.setObjectName("month\_combo\_box\_TSTD")

self.month\_combo\_box\_TSTD.addItem("")

self.month\_combo\_box\_TSTD.addItem("")

self.month\_combo\_box\_TSTD.addItem("")

self.month\_combo\_box\_TSTD.addItem("")

self.month\_combo\_box\_TSTD.addItem("")

self.month\_combo\_box\_TSTD.addItem("")

self.month\_combo\_box\_TSTD.addItem("")

self.month\_combo\_box\_TSTD.addItem("")

self.month\_combo\_box\_TSTD.addItem("")

self.month\_combo\_box\_TSTD.addItem("")

self.month\_combo\_box\_TSTD.addItem("")

self.month\_combo\_box\_TSTD.addItem("")

self.month\_combo\_box\_TSTD.addItem("")

self.grid\_layout\_TSTD.addWidget(self.month\_combo\_box\_TSTD, 1, 3, 1, 1)

self.direction\_combo\_box\_TSTD = QtWidgets.QComboBox(self.TSTD)

font = QtGui.QFont()

font.setPointSize(14)

self.direction\_combo\_box\_TSTD.setFont(font)

self.direction\_combo\_box\_TSTD.setStyleSheet("background-color: rgb(255, 255, 255);")

self.direction\_combo\_box\_TSTD.setObjectName("direction\_combo\_box\_TSTD")

self.direction\_combo\_box\_TSTD.addItem("")

self.grid\_layout\_TSTD.addWidget(self.direction\_combo\_box\_TSTD, 1, 1, 1, 1)

self.month\_label\_TSTD = QtWidgets.QLabel(self.TSTD)

self.month\_label\_TSTD.setMaximumSize(QtCore.QSize(16777215, 28))

font = QtGui.QFont()

font.setPointSize(14)

self.month\_label\_TSTD.setFont(font)

self.month\_label\_TSTD.setFocusPolicy(QtCore.Qt.NoFocus)

self.month\_label\_TSTD.setStyleSheet("background-color: rgb(255, 255, 255);")

self.month\_label\_TSTD.setAlignment(QtCore.Qt.AlignCenter)

self.month\_label\_TSTD.setObjectName("month\_label\_TSTD")

self.grid\_layout\_TSTD.addWidget(self.month\_label\_TSTD, 0, 3, 1, 1)

self.gridLayout\_12.addLayout(self.grid\_layout\_TSTD, 0, 0, 1, 1)

self.table\_widget\_TSTD = QtWidgets.QTableWidget(self.TSTD)

font = QtGui.QFont()

font.setPointSize(10)

self.table\_widget\_TSTD.setFont(font)

self.table\_widget\_TSTD.setStyleSheet("background-color: rgb(255, 255, 255);")

self.table\_widget\_TSTD.setObjectName("table\_widget\_TSTD")

self.table\_widget\_TSTD.setColumnCount(0)

self.table\_widget\_TSTD.setRowCount(0)

self.gridLayout\_12.addWidget(self.table\_widget\_TSTD, 1, 0, 1, 1)

self.tabWidget.addTab(self.TSTD, "")

self.gridLayout\_4.addWidget(self.tabWidget, 0, 0, 1, 1)

self.stackedWidget.addWidget(self.page)

self.page\_2 = QtWidgets.QWidget()

self.page\_2.setObjectName("page\_2")

self.gridLayout\_2 = QtWidgets.QGridLayout(self.page\_2)

self.gridLayout\_2.setObjectName("gridLayout\_2")

self.last\_month\_label = QtWidgets.QLabel(self.page\_2)

self.last\_month\_label.setObjectName("last\_month\_label")

self.gridLayout\_2.addWidget(self.last\_month\_label, 0, 0, 1, 1)

self.table\_widget\_last\_month = QtWidgets.QTableWidget(self.page\_2)

font = QtGui.QFont()

font.setPointSize(10)

self.table\_widget\_last\_month.setFont(font)

self.table\_widget\_last\_month.setObjectName("table\_widget\_last\_month")

self.table\_widget\_last\_month.setColumnCount(0)

self.table\_widget\_last\_month.setRowCount(0)

self.table\_widget\_last\_month.horizontalHeader().setMinimumSectionSize(47)

self.table\_widget\_last\_month.verticalHeader().setDefaultSectionSize(31)

self.table\_widget\_last\_month.verticalHeader().setMinimumSectionSize(28)

self.gridLayout\_2.addWidget(self.table\_widget\_last\_month, 1, 0, 1, 1)

self.last\_year\_label = QtWidgets.QLabel(self.page\_2)

self.last\_year\_label.setObjectName("last\_year\_label")

self.gridLayout\_2.addWidget(self.last\_year\_label, 2, 0, 1, 1)

self.table\_widget\_last\_year = QtWidgets.QTableWidget(self.page\_2)

font = QtGui.QFont()

font.setPointSize(10)

self.table\_widget\_last\_year.setFont(font)

self.table\_widget\_last\_year.setObjectName("table\_widget\_last\_year")

self.table\_widget\_last\_year.setColumnCount(0)

self.table\_widget\_last\_year.setRowCount(0)

self.table\_widget\_last\_year.horizontalHeader().setMinimumSectionSize(47)

self.table\_widget\_last\_year.verticalHeader().setDefaultSectionSize(31)

self.table\_widget\_last\_year.verticalHeader().setMinimumSectionSize(28)

self.gridLayout\_2.addWidget(self.table\_widget\_last\_year, 3, 0, 1, 1)

self.back\_button = QtWidgets.QPushButton(self.page\_2)

self.back\_button.setObjectName("back\_button")

self.gridLayout\_2.addWidget(self.back\_button, 4, 0, 1, 1)

self.stackedWidget.addWidget(self.page\_2)

self.gridLayout.addWidget(self.stackedWidget, 0, 0, 1, 1)

window.setCentralWidget(self.centralwidget)

self.menubar = QtWidgets.QMenuBar(window)

self.menubar.setGeometry(QtCore.QRect(0, 0, 743, 26))

self.menubar.setDefaultUp(False)

self.menubar.setNativeMenuBar(True)

self.menubar.setObjectName("menubar")

window.setMenuBar(self.menubar)

self.toolBar = QtWidgets.QToolBar(window)

self.toolBar.setObjectName("toolBar")

window.addToolBar(QtCore.Qt.TopToolBarArea, self.toolBar)

self.retranslate\_ui(window)

self.stackedWidget.setCurrentIndex(0)

self.tabWidget.setCurrentIndex(0)

QtCore.QMetaObject.connectSlotsByName(window)

def retranslate\_ui(self, window):

"""Установка всех встроенных названий"""

\_translate = QtCore.QCoreApplication.translate

window.setWindowTitle(\_translate("MainWindow", "StudInStudOut"))

self.foe\_label\_b.setText(\_translate("MainWindow", "Форма обучения"))

self.foe\_combo\_box\_b.setItemText(0, \_translate("MainWindow", "Все"))

self.foe\_combo\_box\_b.setItemText(1, \_translate("MainWindow", "Очная"))

self.foe\_combo\_box\_b.setItemText(2, \_translate("MainWindow", "Очно-заочная"))

self.foe\_combo\_box\_b.setItemText(3, \_translate("MainWindow", "Заочная"))

self.direction\_label\_b.setText(\_translate("MainWindow", "Направление"))

self.year\_combo\_box\_b.setItemText(0, \_translate("MainWindow", "Все"))

self.year\_label\_b.setText(\_translate("MainWindow", "Год"))

self.month\_combo\_box\_b.setItemText(0, \_translate("MainWindow", "Все"))

self.month\_combo\_box\_b.setItemText(1, \_translate("MainWindow", "Январь"))

self.month\_combo\_box\_b.setItemText(2, \_translate("MainWindow", "Февраль"))

self.month\_combo\_box\_b.setItemText(3, \_translate("MainWindow", "Март"))

self.month\_combo\_box\_b.setItemText(4, \_translate("MainWindow", "Апрель"))

self.month\_combo\_box\_b.setItemText(5, \_translate("MainWindow", "Май"))

self.month\_combo\_box\_b.setItemText(6, \_translate("MainWindow", "Июнь"))

self.month\_combo\_box\_b.setItemText(7, \_translate("MainWindow", "Июль"))

self.month\_combo\_box\_b.setItemText(8, \_translate("MainWindow", "Август"))

self.month\_combo\_box\_b.setItemText(9, \_translate("MainWindow", "Сентябрь"))

self.month\_combo\_box\_b.setItemText(10, \_translate("MainWindow", "Октябрь"))

self.month\_combo\_box\_b.setItemText(11, \_translate("MainWindow", "Ноябрь"))

self.month\_combo\_box\_b.setItemText(12, \_translate("MainWindow", "Декабрь"))

self.direction\_combo\_box\_b.setItemText(0, \_translate("MainWindow", "Все"))

self.month\_label\_b.setText(\_translate("MainWindow", "Месяц"))

self.tabWidget.setTabText(self.tabWidget.indexOf(self.bachelor\_course), \_translate("MainWindow", "Бакалавриат"))

self.foe\_label\_m.setText(\_translate("MainWindow", "Форма обучения"))

self.foe\_combo\_box\_m.setItemText(0, \_translate("MainWindow", "Все"))

self.foe\_combo\_box\_m.setItemText(1, \_translate("MainWindow", "Очная"))

self.foe\_combo\_box\_m.setItemText(2, \_translate("MainWindow", "Очно-заочная"))

self.foe\_combo\_box\_m.setItemText(3, \_translate("MainWindow", "Заочная"))

self.direction\_label\_m.setText(\_translate("MainWindow", "Направление"))

self.year\_combo\_box\_m.setItemText(0, \_translate("MainWindow", "Все"))

self.year\_label\_m.setText(\_translate("MainWindow", "Год"))

self.month\_combo\_box\_m.setItemText(0, \_translate("MainWindow", "Все"))

self.month\_combo\_box\_m.setItemText(1, \_translate("MainWindow", "Январь"))

self.month\_combo\_box\_m.setItemText(2, \_translate("MainWindow", "Февраль"))

self.month\_combo\_box\_m.setItemText(3, \_translate("MainWindow", "Март"))

self.month\_combo\_box\_m.setItemText(4, \_translate("MainWindow", "Апрель"))

self.month\_combo\_box\_m.setItemText(5, \_translate("MainWindow", "Май"))

self.month\_combo\_box\_m.setItemText(6, \_translate("MainWindow", "Июнь"))

self.month\_combo\_box\_m.setItemText(7, \_translate("MainWindow", "Июль"))

self.month\_combo\_box\_m.setItemText(8, \_translate("MainWindow", "Август"))

self.month\_combo\_box\_m.setItemText(9, \_translate("MainWindow", "Сентябрь"))

self.month\_combo\_box\_m.setItemText(10, \_translate("MainWindow", "Октябрь"))

self.month\_combo\_box\_m.setItemText(11, \_translate("MainWindow", "Ноябрь"))

self.month\_combo\_box\_m.setItemText(12, \_translate("MainWindow", "Декабрь"))

self.direction\_combo\_box\_m.setItemText(0, \_translate("MainWindow", "Все"))

self.month\_label\_m.setText(\_translate("MainWindow", "Месяц"))

self.tabWidget.setTabText(self.tabWidget.indexOf(self.master\_course), \_translate("MainWindow", "Магистратура"))

self.foe\_label\_s.setText(\_translate("MainWindow", "Форма обучения"))

self.foe\_combo\_box\_s.setItemText(0, \_translate("MainWindow", "Все"))

self.foe\_combo\_box\_s.setItemText(1, \_translate("MainWindow", "Очная"))

self.foe\_combo\_box\_s.setItemText(2, \_translate("MainWindow", "Очно-заочная"))

self.foe\_combo\_box\_s.setItemText(3, \_translate("MainWindow", "Заочная"))

self.direction\_label\_s.setText(\_translate("MainWindow", "Направление"))

self.year\_combo\_box\_s.setItemText(0, \_translate("MainWindow", "Все"))

self.year\_label\_s.setText(\_translate("MainWindow", "Год"))

self.month\_combo\_box\_s.setItemText(0, \_translate("MainWindow", "Все"))

self.month\_combo\_box\_s.setItemText(1, \_translate("MainWindow", "Январь"))

self.month\_combo\_box\_s.setItemText(2, \_translate("MainWindow", "Февраль"))

self.month\_combo\_box\_s.setItemText(3, \_translate("MainWindow", "Март"))

self.month\_combo\_box\_s.setItemText(4, \_translate("MainWindow", "Апрель"))

self.month\_combo\_box\_s.setItemText(5, \_translate("MainWindow", "Май"))

self.month\_combo\_box\_s.setItemText(6, \_translate("MainWindow", "Июнь"))

self.month\_combo\_box\_s.setItemText(7, \_translate("MainWindow", "Июль"))

self.month\_combo\_box\_s.setItemText(8, \_translate("MainWindow", "Август"))

self.month\_combo\_box\_s.setItemText(9, \_translate("MainWindow", "Сентябрь"))

self.month\_combo\_box\_s.setItemText(10, \_translate("MainWindow", "Октябрь"))

self.month\_combo\_box\_s.setItemText(11, \_translate("MainWindow", "Ноябрь"))

self.month\_combo\_box\_s.setItemText(12, \_translate("MainWindow", "Декабрь"))

self.direction\_combo\_box\_s.setItemText(0, \_translate("MainWindow", "Все"))

self.month\_label\_s.setText(\_translate("MainWindow", "Месяц"))

self.tabWidget.setTabText(self.tabWidget.indexOf(self.specialist\_degree), \_translate("MainWindow",

"Специалитет"))

self.foe\_label\_CSET.setText(\_translate("MainWindow", "Форма обучения"))

self.foe\_combo\_box\_CSET.setItemText(0, \_translate("MainWindow", "Все"))

self.foe\_combo\_box\_CSET.setItemText(1, \_translate("MainWindow", "Очная"))

self.foe\_combo\_box\_CSET.setItemText(2, \_translate("MainWindow", "Очно-заочная"))

self.foe\_combo\_box\_CSET.setItemText(3, \_translate("MainWindow", "Заочная"))

self.direction\_label\_CSET.setText(\_translate("MainWindow", "Направление"))

self.year\_combo\_box\_CSET.setItemText(0, \_translate("MainWindow", "Все"))

self.year\_label\_CSET.setText(\_translate("MainWindow", "Год"))

self.month\_combo\_box\_CSET.setItemText(0, \_translate("MainWindow", "Все"))

self.month\_combo\_box\_CSET.setItemText(1, \_translate("MainWindow", "Январь"))

self.month\_combo\_box\_CSET.setItemText(2, \_translate("MainWindow", "Февраль"))

self.month\_combo\_box\_CSET.setItemText(3, \_translate("MainWindow", "Март"))

self.month\_combo\_box\_CSET.setItemText(4, \_translate("MainWindow", "Апрель"))

self.month\_combo\_box\_CSET.setItemText(5, \_translate("MainWindow", "Май"))

self.month\_combo\_box\_CSET.setItemText(6, \_translate("MainWindow", "Июнь"))

self.month\_combo\_box\_CSET.setItemText(7, \_translate("MainWindow", "Июль"))

self.month\_combo\_box\_CSET.setItemText(8, \_translate("MainWindow", "Август"))

self.month\_combo\_box\_CSET.setItemText(9, \_translate("MainWindow", "Сентябрь"))

self.month\_combo\_box\_CSET.setItemText(10, \_translate("MainWindow", "Октябрь"))

self.month\_combo\_box\_CSET.setItemText(11, \_translate("MainWindow", "Ноябрь"))

self.month\_combo\_box\_CSET.setItemText(12, \_translate("MainWindow", "Декабрь"))

self.direction\_combo\_box\_CSET.setItemText(0, \_translate("MainWindow", "Все"))

self.month\_label\_CSET.setText(\_translate("MainWindow", "Месяц"))

self.tabWidget.setTabText(self.tabWidget.indexOf(self.CSET), \_translate("MainWindow", "ККМТ"))

self.foe\_label\_TSTD.setText(\_translate("MainWindow", "Форма обучения"))

self.foe\_combo\_box\_TSTD.setItemText(0, \_translate("MainWindow", "Все"))

self.foe\_combo\_box\_TSTD.setItemText(1, \_translate("MainWindow", "Очная"))

self.foe\_combo\_box\_TSTD.setItemText(2, \_translate("MainWindow", "Очно-заочная"))

self.foe\_combo\_box\_TSTD.setItemText(3, \_translate("MainWindow", "Заочная"))

self.direction\_label\_TSTD.setText(\_translate("MainWindow", "Направление"))

self.year\_combo\_box\_TSTD.setItemText(0, \_translate("MainWindow", "Все"))

self.year\_label\_TSTD.setText(\_translate("MainWindow", "Год"))

self.month\_combo\_box\_TSTD.setItemText(0, \_translate("MainWindow", "Все"))

self.month\_combo\_box\_TSTD.setItemText(1, \_translate("MainWindow", "Январь"))

self.month\_combo\_box\_TSTD.setItemText(2, \_translate("MainWindow", "Февраль"))

self.month\_combo\_box\_TSTD.setItemText(3, \_translate("MainWindow", "Март"))

self.month\_combo\_box\_TSTD.setItemText(4, \_translate("MainWindow", "Апрель"))

self.month\_combo\_box\_TSTD.setItemText(5, \_translate("MainWindow", "Май"))

self.month\_combo\_box\_TSTD.setItemText(6, \_translate("MainWindow", "Июнь"))

self.month\_combo\_box\_TSTD.setItemText(7, \_translate("MainWindow", "Июль"))

self.month\_combo\_box\_TSTD.setItemText(8, \_translate("MainWindow", "Август"))

self.month\_combo\_box\_TSTD.setItemText(9, \_translate("MainWindow", "Сентябрь"))

self.month\_combo\_box\_TSTD.setItemText(10, \_translate("MainWindow", "Октябрь"))

self.month\_combo\_box\_TSTD.setItemText(11, \_translate("MainWindow", "Ноябрь"))

self.month\_combo\_box\_TSTD.setItemText(12, \_translate("MainWindow", "Декабрь"))

self.direction\_combo\_box\_TSTD.setItemText(0, \_translate("MainWindow", "Все"))

self.month\_label\_TSTD.setText(\_translate("MainWindow", "Месяц"))

self.tabWidget.setTabText(self.tabWidget.indexOf(self.TSTD), \_translate("MainWindow", "ТТД"))

self.last\_month\_label.setText(\_translate("MainWindow", "По сравнению с предыдущим месяцем:"))

self.last\_year\_label.setText(\_translate("MainWindow", "По сравнению с этим месяцем прошлого года:"))

self.back\_button.setText(\_translate("MainWindow", "Назад"))

self.toolBar.setWindowTitle(\_translate("MainWindow", "toolBar"))

**Листинг 3. dialog\_ui.py**

import excel\_control

from PyQt5 import QtCore, QtGui, QtWidgets

class UiDialog(excel\_control.ExcelControl):

def \_\_init\_\_(self, dialog):

"""Конструктор класса UiDialog"""

super(UiDialog, self).\_\_init\_\_()

self.dialog = dialog

dialog.setObjectName("Dialog")

dialog.resize(1411, 565)

size\_policy = QtWidgets.QSizePolicy(QtWidgets.QSizePolicy.Preferred, QtWidgets.QSizePolicy.Preferred)

size\_policy.setHorizontalStretch(141)

size\_policy.setVerticalStretch(56)

size\_policy.setHeightForWidth(dialog.sizePolicy().hasHeightForWidth())

dialog.setSizePolicy(size\_policy)

dialog.setMinimumSize(QtCore.QSize(1411, 565))

dialog.setMaximumSize(QtCore.QSize(1411, 565))

icon = QtGui.QIcon()

icon.addPixmap(QtGui.QPixmap("Изображения/free-icon-binder-4624080.png"), QtGui.QIcon.Normal, QtGui.QIcon.Off)

dialog.setWindowIcon(icon)

dialog.setSizeGripEnabled(False)

dialog.setModal(False)

self.gridLayout\_4 = QtWidgets.QGridLayout(dialog)

self.gridLayout\_4.setObjectName("gridLayout\_4")

self.gridLayout = QtWidgets.QGridLayout()

self.gridLayout.setObjectName("gridLayout")

self.course\_label\_d = QtWidgets.QLabel(dialog)

font = QtGui.QFont()

font.setPointSize(12)

self.course\_label\_d.setFont(font)

self.course\_label\_d.setAlignment(QtCore.Qt.AlignCenter)

self.course\_label\_d.setObjectName("course\_label\_d")

self.gridLayout.addWidget(self.course\_label\_d, 0, 3, 1, 1)

self.loe\_combo\_box\_d = QtWidgets.QComboBox(dialog)

font = QtGui.QFont()

font.setPointSize(12)

self.loe\_combo\_box\_d.setFont(font)

self.loe\_combo\_box\_d.setObjectName("loe\_combo\_box\_b")

self.loe\_combo\_box\_d.addItem("")

self.loe\_combo\_box\_d.addItem("")

self.loe\_combo\_box\_d.addItem("")

self.loe\_combo\_box\_d.addItem("")

self.loe\_combo\_box\_d.addItem("")

self.loe\_combo\_box\_d.addItem("")

self.gridLayout.addWidget(self.loe\_combo\_box\_d, 1, 0, 1, 1)

self.direction\_combo\_box\_d = QtWidgets.QComboBox(dialog)

font = QtGui.QFont()

font.setPointSize(12)

self.direction\_combo\_box\_d.setFont(font)

self.direction\_combo\_box\_d.setObjectName("direction\_combo\_box\_d")

self.direction\_combo\_box\_d.addItem("")

self.gridLayout.addWidget(self.direction\_combo\_box\_d, 1, 1, 1, 1)

self.direction\_label\_d = QtWidgets.QLabel(dialog)

font = QtGui.QFont()

font.setPointSize(12)

self.direction\_label\_d.setFont(font)

self.direction\_label\_d.setAlignment(QtCore.Qt.AlignCenter)

self.direction\_label\_d.setObjectName("direction\_label\_d")

self.gridLayout.addWidget(self.direction\_label\_d, 0, 1, 1, 1)

self.foe\_combo\_box\_d = QtWidgets.QComboBox(dialog)

font = QtGui.QFont()

font.setPointSize(12)

self.foe\_combo\_box\_d.setFont(font)

self.foe\_combo\_box\_d.setObjectName("foe\_combo\_box\_d")

self.foe\_combo\_box\_d.addItem("")

self.foe\_combo\_box\_d.addItem("")

self.foe\_combo\_box\_d.addItem("")

self.foe\_combo\_box\_d.addItem("")

self.gridLayout.addWidget(self.foe\_combo\_box\_d, 1, 2, 1, 1)

self.course\_combo\_box\_d = QtWidgets.QComboBox(dialog)

font = QtGui.QFont()

font.setPointSize(12)

self.course\_combo\_box\_d.setFont(font)

self.course\_combo\_box\_d.setObjectName("course\_combo\_box\_d")

self.course\_combo\_box\_d.addItem("")

self.course\_combo\_box\_d.addItem("")

self.course\_combo\_box\_d.addItem("")

self.course\_combo\_box\_d.addItem("")

self.course\_combo\_box\_d.addItem("")

self.course\_combo\_box\_d.addItem("")

self.course\_combo\_box\_d.addItem("")

self.gridLayout.addWidget(self.course\_combo\_box\_d, 1, 3, 1, 1)

self.loe\_label\_d = QtWidgets.QLabel(dialog)

font = QtGui.QFont()

font.setPointSize(12)

self.loe\_label\_d.setFont(font)

self.loe\_label\_d.setAlignment(QtCore.Qt.AlignCenter)

self.loe\_label\_d.setObjectName("loe\_label\_d")

self.gridLayout.addWidget(self.loe\_label\_d, 0, 0, 1, 1)

self.for\_label\_d = QtWidgets.QLabel(dialog)

font = QtGui.QFont()

font.setPointSize(12)

self.for\_label\_d.setFont(font)

self.for\_label\_d.setAlignment(QtCore.Qt.AlignCenter)

self.for\_label\_d.setObjectName("for\_label\_d")

self.gridLayout.addWidget(self.for\_label\_d, 0, 2, 1, 1)

self.year\_label\_d = QtWidgets.QLabel(dialog)

font = QtGui.QFont()

font.setPointSize(12)

self.year\_label\_d.setFont(font)

self.year\_label\_d.setAlignment(QtCore.Qt.AlignCenter)

self.year\_label\_d.setObjectName("year\_label\_d")

self.gridLayout.addWidget(self.year\_label\_d, 0, 4, 1, 1)

self.month\_label\_d = QtWidgets.QLabel(dialog)

font = QtGui.QFont()

font.setPointSize(12)

self.month\_label\_d.setFont(font)

self.month\_label\_d.setAlignment(QtCore.Qt.AlignCenter)

self.month\_label\_d.setObjectName("month\_label\_d")

self.gridLayout.addWidget(self.month\_label\_d, 0, 5, 1, 1)

self.year\_combo\_box\_d = QtWidgets.QComboBox(dialog)

font = QtGui.QFont()

font.setPointSize(12)

self.year\_combo\_box\_d.setFont(font)

self.year\_combo\_box\_d.setObjectName("year\_combo\_box\_d")

self.year\_combo\_box\_d.addItem("")

self.gridLayout.addWidget(self.year\_combo\_box\_d, 1, 4, 1, 1)

self.month\_combo\_box\_d = QtWidgets.QComboBox(dialog)

font = QtGui.QFont()

font.setPointSize(12)

self.month\_combo\_box\_d.setFont(font)

self.month\_combo\_box\_d.setObjectName("month\_combo\_box\_d")

self.month\_combo\_box\_d.addItem("")

self.month\_combo\_box\_d.addItem("")

self.month\_combo\_box\_d.addItem("")

self.month\_combo\_box\_d.addItem("")

self.month\_combo\_box\_d.addItem("")

self.month\_combo\_box\_d.addItem("")

self.month\_combo\_box\_d.addItem("")

self.month\_combo\_box\_d.addItem("")

self.month\_combo\_box\_d.addItem("")

self.month\_combo\_box\_d.addItem("")

self.month\_combo\_box\_d.addItem("")

self.month\_combo\_box\_d.addItem("")

self.month\_combo\_box\_d.addItem("")

self.gridLayout.addWidget(self.month\_combo\_box\_d, 1, 5, 1, 1)

self.gridLayout\_4.addLayout(self.gridLayout, 0, 0, 1, 4)

self.line\_2 = QtWidgets.QFrame(dialog)

font = QtGui.QFont()

font.setPointSize(12)

self.line\_2.setFont(font)

self.line\_2.setFrameShadow(QtWidgets.QFrame.Raised)

self.line\_2.setLineWidth(1)

self.line\_2.setMidLineWidth(0)

self.line\_2.setFrameShape(QtWidgets.QFrame.HLine)

self.line\_2.setObjectName("line\_2")

self.gridLayout\_4.addWidget(self.line\_2, 1, 3, 1, 1)

self.line\_4 = QtWidgets.QFrame(dialog)

font = QtGui.QFont()

font.setPointSize(12)

self.line\_4.setFont(font)

self.line\_4.setFrameShadow(QtWidgets.QFrame.Raised)

self.line\_4.setLineWidth(1)

self.line\_4.setMidLineWidth(0)

self.line\_4.setFrameShape(QtWidgets.QFrame.HLine)

self.line\_4.setObjectName("line\_4")

self.gridLayout\_4.addWidget(self.line\_4, 3, 3, 1, 1)

self.line = QtWidgets.QFrame(dialog)

font = QtGui.QFont()

font.setPointSize(12)

self.line.setFont(font)

self.line.setFrameShadow(QtWidgets.QFrame.Raised)

self.line.setLineWidth(1)

self.line.setMidLineWidth(0)

self.line.setFrameShape(QtWidgets.QFrame.HLine)

self.line.setObjectName("line")

self.gridLayout\_4.addWidget(self.line, 1, 1, 1, 1)

self.gridLayout\_2 = QtWidgets.QGridLayout()

self.gridLayout\_2.setObjectName("gridLayout\_2")

self.line\_8 = QtWidgets.QFrame(dialog)

font = QtGui.QFont()

font.setPointSize(12)

self.line\_8.setFont(font)

self.line\_8.setFrameShadow(QtWidgets.QFrame.Raised)

self.line\_8.setLineWidth(1)

self.line\_8.setMidLineWidth(0)

self.line\_8.setFrameShape(QtWidgets.QFrame.HLine)

self.line\_8.setObjectName("line\_8")

self.gridLayout\_2.addWidget(self.line\_8, 11, 0, 1, 1)

self.enrollment\_all\_check\_box\_d = QtWidgets.QCheckBox(dialog)

font = QtGui.QFont()

font.setPointSize(12)

self.enrollment\_all\_check\_box\_d.setFont(font)

self.enrollment\_all\_check\_box\_d.setTristate(False)

self.enrollment\_all\_check\_box\_d.setObjectName("enrollment\_all\_check\_box\_d")

self.gridLayout\_2.addWidget(self.enrollment\_all\_check\_box\_d, 0, 0, 1, 1)

self.foreigners\_eb\_check\_box\_d = QtWidgets.QCheckBox(dialog)

font = QtGui.QFont()

font.setPointSize(12)

self.foreigners\_eb\_check\_box\_d.setFont(font)

self.foreigners\_eb\_check\_box\_d.setObjectName("foreigners\_eb\_check\_box\_d")

self.gridLayout\_2.addWidget(self.foreigners\_eb\_check\_box\_d, 10, 0, 1, 1)

self.foreigners\_ea\_check\_box\_d = QtWidgets.QCheckBox(dialog)

font = QtGui.QFont()

font.setPointSize(12)

self.foreigners\_ea\_check\_box\_d.setFont(font)

self.foreigners\_ea\_check\_box\_d.setObjectName("foreigners\_ea\_check\_box\_d")

self.gridLayout\_2.addWidget(self.foreigners\_ea\_check\_box\_d, 3, 0, 1, 1)

self.enrollment\_paid\_check\_box\_d = QtWidgets.QCheckBox(dialog)

font = QtGui.QFont()

font.setPointSize(12)

self.enrollment\_paid\_check\_box\_d.setFont(font)

self.enrollment\_paid\_check\_box\_d.setObjectName("enrollment\_paid\_check\_box\_d")

self.gridLayout\_2.addWidget(self.enrollment\_paid\_check\_box\_d, 12, 0, 1, 1)

self.target\_ep\_check\_box\_d = QtWidgets.QCheckBox(dialog)

font = QtGui.QFont()

font.setPointSize(12)

self.target\_ep\_check\_box\_d.setFont(font)

self.target\_ep\_check\_box\_d.setObjectName("target\_ep\_check\_box\_d")

self.gridLayout\_2.addWidget(self.target\_ep\_check\_box\_d, 14, 0, 1, 1)

self.line\_7 = QtWidgets.QFrame(dialog)

font = QtGui.QFont()

font.setPointSize(12)

self.line\_7.setFont(font)

self.line\_7.setFrameShadow(QtWidgets.QFrame.Raised)

self.line\_7.setLineWidth(1)

self.line\_7.setMidLineWidth(0)

self.line\_7.setFrameShape(QtWidgets.QFrame.HLine)

self.line\_7.setObjectName("line\_7")

self.gridLayout\_2.addWidget(self.line\_7, 4, 0, 1, 1)

self.disabled\_ea\_check\_box\_d = QtWidgets.QCheckBox(dialog)

font = QtGui.QFont()

font.setPointSize(12)

self.disabled\_ea\_check\_box\_d.setFont(font)

self.disabled\_ea\_check\_box\_d.setObjectName("disabled\_ea\_check\_box\_d")

self.gridLayout\_2.addWidget(self.disabled\_ea\_check\_box\_d, 1, 0, 1, 1)

self.foreigners\_ep\_check\_box\_d = QtWidgets.QCheckBox(dialog)

font = QtGui.QFont()

font.setPointSize(12)

self.foreigners\_ep\_check\_box\_d.setFont(font)

self.foreigners\_ep\_check\_box\_d.setObjectName("foreigners\_ep\_check\_box\_d")

self.gridLayout\_2.addWidget(self.foreigners\_ep\_check\_box\_d, 15, 0, 1, 1)

self.target\_ea\_check\_box\_d = QtWidgets.QCheckBox(dialog)

font = QtGui.QFont()

font.setPointSize(12)

self.target\_ea\_check\_box\_d.setFont(font)

self.target\_ea\_check\_box\_d.setObjectName("target\_ea\_check\_box\_d")

self.gridLayout\_2.addWidget(self.target\_ea\_check\_box\_d, 2, 0, 1, 1)

self.target\_eb\_check\_box\_d = QtWidgets.QCheckBox(dialog)

font = QtGui.QFont()

font.setPointSize(12)

self.target\_eb\_check\_box\_d.setFont(font)

self.target\_eb\_check\_box\_d.setObjectName("target\_eb\_check\_box\_d")

self.gridLayout\_2.addWidget(self.target\_eb\_check\_box\_d, 7, 0, 1, 1)

self.disabled\_ep\_check\_box\_d = QtWidgets.QCheckBox(dialog)

font = QtGui.QFont()

font.setPointSize(12)

self.disabled\_ep\_check\_box\_d.setFont(font)

self.disabled\_ep\_check\_box\_d.setObjectName("disabled\_ep\_check\_box\_d")

self.gridLayout\_2.addWidget(self.disabled\_ep\_check\_box\_d, 13, 0, 1, 1)

self.enrollment\_budget\_check\_box\_d = QtWidgets.QCheckBox(dialog)

font = QtGui.QFont()

font.setPointSize(12)

self.enrollment\_budget\_check\_box\_d.setFont(font)

self.enrollment\_budget\_check\_box\_d.setObjectName("enrollment\_budget\_check\_box\_d")

self.gridLayout\_2.addWidget(self.enrollment\_budget\_check\_box\_d, 5, 0, 1, 1)

self.disabled\_eb\_check\_box\_d = QtWidgets.QCheckBox(dialog)

font = QtGui.QFont()

font.setPointSize(12)

self.disabled\_eb\_check\_box\_d.setFont(font)

self.disabled\_eb\_check\_box\_d.setObjectName("disabled\_eb\_check\_box\_d")

self.gridLayout\_2.addWidget(self.disabled\_eb\_check\_box\_d, 6, 0, 1, 1)

self.gridLayout\_4.addLayout(self.gridLayout\_2, 2, 1, 1, 1)

self.continue\_button\_d = QtWidgets.QPushButton(dialog)

font = QtGui.QFont()

font.setPointSize(12)

self.continue\_button\_d.setFont(font)

self.continue\_button\_d.setObjectName("continue\_button\_d")

self.gridLayout\_4.addWidget(self.continue\_button\_d, 4, 1, 1, 1)

self.gridLayout\_3 = QtWidgets.QGridLayout()

self.gridLayout\_3.setObjectName("gridLayout\_3")

self.foreigners\_da\_check\_box\_d = QtWidgets.QCheckBox(dialog)

font = QtGui.QFont()

font.setPointSize(12)

self.foreigners\_da\_check\_box\_d.setFont(font)

self.foreigners\_da\_check\_box\_d.setObjectName("foreigners\_da\_check\_box\_d")

self.gridLayout\_3.addWidget(self.foreigners\_da\_check\_box\_d, 3, 0, 1, 1)

self.deduct\_budget\_check\_box\_d = QtWidgets.QCheckBox(dialog)

font = QtGui.QFont()

font.setPointSize(12)

self.deduct\_budget\_check\_box\_d.setFont(font)

self.deduct\_budget\_check\_box\_d.setObjectName("deduct\_budget\_check\_box\_d")

self.gridLayout\_3.addWidget(self.deduct\_budget\_check\_box\_d, 5, 0, 1, 1)

self.line\_9 = QtWidgets.QFrame(dialog)

font = QtGui.QFont()

font.setPointSize(12)

self.line\_9.setFont(font)

self.line\_9.setFrameShadow(QtWidgets.QFrame.Raised)

self.line\_9.setLineWidth(1)

self.line\_9.setMidLineWidth(0)

self.line\_9.setFrameShape(QtWidgets.QFrame.HLine)

self.line\_9.setObjectName("line\_9")

self.gridLayout\_3.addWidget(self.line\_9, 9, 0, 1, 1)

self.disabled\_db\_check\_box\_d = QtWidgets.QCheckBox(dialog)

font = QtGui.QFont()

font.setPointSize(12)

self.disabled\_db\_check\_box\_d.setFont(font)

self.disabled\_db\_check\_box\_d.setObjectName("disabled\_db\_check\_box\_d")

self.gridLayout\_3.addWidget(self.disabled\_db\_check\_box\_d, 6, 0, 1, 1)

self.target\_db\_check\_box\_d = QtWidgets.QCheckBox(dialog)

font = QtGui.QFont()

font.setPointSize(12)

self.target\_db\_check\_box\_d.setFont(font)

self.target\_db\_check\_box\_d.setObjectName("target\_db\_check\_box\_d")

self.gridLayout\_3.addWidget(self.target\_db\_check\_box\_d, 7, 0, 1, 1)

self.target\_dp\_check\_box\_d = QtWidgets.QCheckBox(dialog)

font = QtGui.QFont()

font.setPointSize(12)

self.target\_dp\_check\_box\_d.setFont(font)

self.target\_dp\_check\_box\_d.setObjectName("target\_dp\_check\_box\_d")

self.gridLayout\_3.addWidget(self.target\_dp\_check\_box\_d, 12, 0, 1, 1)

self.foreigners\_dp\_check\_box\_d = QtWidgets.QCheckBox(dialog)

font = QtGui.QFont()

font.setPointSize(12)

self.foreigners\_dp\_check\_box\_d.setFont(font)

self.foreigners\_dp\_check\_box\_d.setObjectName("foreigners\_dp\_check\_box\_d")

self.gridLayout\_3.addWidget(self.foreigners\_dp\_check\_box\_d, 13, 0, 1, 1)

self.target\_da\_check\_box\_d = QtWidgets.QCheckBox(dialog)

font = QtGui.QFont()

font.setPointSize(12)

self.target\_da\_check\_box\_d.setFont(font)

self.target\_da\_check\_box\_d.setObjectName("target\_da\_check\_box\_d")

self.gridLayout\_3.addWidget(self.target\_da\_check\_box\_d, 2, 0, 1, 1)

self.disabled\_dp\_check\_box\_d = QtWidgets.QCheckBox(dialog)

font = QtGui.QFont()

font.setPointSize(12)

self.disabled\_dp\_check\_box\_d.setFont(font)

self.disabled\_dp\_check\_box\_d.setObjectName("disabled\_dp\_check\_box\_d")

self.gridLayout\_3.addWidget(self.disabled\_dp\_check\_box\_d, 11, 0, 1, 1)

self.disabled\_da\_check\_box\_d = QtWidgets.QCheckBox(dialog)

font = QtGui.QFont()

font.setPointSize(12)

self.disabled\_da\_check\_box\_d.setFont(font)

self.disabled\_da\_check\_box\_d.setObjectName("disabled\_da\_check\_box\_d")

self.gridLayout\_3.addWidget(self.disabled\_da\_check\_box\_d, 1, 0, 1, 1)

self.deduct\_paid\_check\_box = QtWidgets.QCheckBox(dialog)

font = QtGui.QFont()

font.setPointSize(12)

self.deduct\_paid\_check\_box.setFont(font)

self.deduct\_paid\_check\_box.setObjectName("deduct\_paid\_check\_box")

self.gridLayout\_3.addWidget(self.deduct\_paid\_check\_box, 10, 0, 1, 1)

self.deduct\_all\_check\_box\_d = QtWidgets.QCheckBox(dialog)

font = QtGui.QFont()

font.setPointSize(12)

self.deduct\_all\_check\_box\_d.setFont(font)

self.deduct\_all\_check\_box\_d.setObjectName("deduct\_all\_check\_box\_d")

self.gridLayout\_3.addWidget(self.deduct\_all\_check\_box\_d, 0, 0, 1, 1)

self.foreigners\_db\_check\_box\_d = QtWidgets.QCheckBox(dialog)

font = QtGui.QFont()

font.setPointSize(12)

self.foreigners\_db\_check\_box\_d.setFont(font)

self.foreigners\_db\_check\_box\_d.setObjectName("foreigners\_db\_check\_box\_d")

self.gridLayout\_3.addWidget(self.foreigners\_db\_check\_box\_d, 8, 0, 1, 1)

self.line\_10 = QtWidgets.QFrame(dialog)

font = QtGui.QFont()

font.setPointSize(12)

self.line\_10.setFont(font)

self.line\_10.setFrameShadow(QtWidgets.QFrame.Raised)

self.line\_10.setLineWidth(1)

self.line\_10.setMidLineWidth(0)

self.line\_10.setFrameShape(QtWidgets.QFrame.HLine)

self.line\_10.setObjectName("line\_10")

self.gridLayout\_3.addWidget(self.line\_10, 4, 0, 1, 1)

self.gridLayout\_4.addLayout(self.gridLayout\_3, 2, 3, 1, 1)

self.back\_button\_d = QtWidgets.QPushButton(dialog)

font = QtGui.QFont()

font.setPointSize(12)

self.back\_button\_d.setFont(font)

self.back\_button\_d.setObjectName("back\_button\_d")

self.gridLayout\_4.addWidget(self.back\_button\_d, 4, 3, 1, 1)

self.line\_3 = QtWidgets.QFrame(dialog)

font = QtGui.QFont()

font.setPointSize(12)

self.line\_3.setFont(font)

self.line\_3.setFrameShadow(QtWidgets.QFrame.Raised)

self.line\_3.setLineWidth(1)

self.line\_3.setMidLineWidth(0)

self.line\_3.setFrameShape(QtWidgets.QFrame.HLine)

self.line\_3.setObjectName("line\_3")

self.gridLayout\_4.addWidget(self.line\_3, 3, 1, 1, 1)

self.line\_5 = QtWidgets.QFrame(dialog)

font = QtGui.QFont()

font.setPointSize(12)

self.line\_5.setFont(font)

self.line\_5.setFrameShadow(QtWidgets.QFrame.Raised)

self.line\_5.setLineWidth(1)

self.line\_5.setMidLineWidth(0)

self.line\_5.setFrameShape(QtWidgets.QFrame.VLine)

self.line\_5.setObjectName("line\_5")

self.gridLayout\_4.addWidget(self.line\_5, 4, 2, 1, 1)

self.line\_6 = QtWidgets.QFrame(dialog)

font = QtGui.QFont()

font.setPointSize(12)

self.line\_6.setFont(font)

self.line\_6.setFrameShadow(QtWidgets.QFrame.Raised)

self.line\_6.setLineWidth(1)

self.line\_6.setMidLineWidth(0)

self.line\_6.setFrameShape(QtWidgets.QFrame.VLine)

self.line\_6.setObjectName("line\_6")

self.gridLayout\_4.addWidget(self.line\_6, 2, 2, 1, 1)

self.retranslate\_ui(dialog)

QtCore.QMetaObject.connectSlotsByName(dialog)

dialog.setTabOrder(self.loe\_combo\_box\_d, self.direction\_combo\_box\_d)

dialog.setTabOrder(self.direction\_combo\_box\_d, self.foe\_combo\_box\_d)

dialog.setTabOrder(self.foe\_combo\_box\_d, self.course\_combo\_box\_d)

dialog.setTabOrder(self.course\_combo\_box\_d, self.year\_combo\_box\_d)

dialog.setTabOrder(self.year\_combo\_box\_d, self.month\_combo\_box\_d)

dialog.setTabOrder(self.month\_combo\_box\_d, self.enrollment\_all\_check\_box\_d)

dialog.setTabOrder(self.enrollment\_all\_check\_box\_d, self.disabled\_ea\_check\_box\_d)

dialog.setTabOrder(self.disabled\_ea\_check\_box\_d, self.target\_ea\_check\_box\_d)

dialog.setTabOrder(self.target\_ea\_check\_box\_d, self.foreigners\_ea\_check\_box\_d)

dialog.setTabOrder(self.foreigners\_ea\_check\_box\_d, self.enrollment\_budget\_check\_box\_d)

dialog.setTabOrder(self.enrollment\_budget\_check\_box\_d, self.disabled\_eb\_check\_box\_d)

dialog.setTabOrder(self.disabled\_eb\_check\_box\_d, self.target\_eb\_check\_box\_d)

dialog.setTabOrder(self.target\_eb\_check\_box\_d, self.foreigners\_eb\_check\_box\_d)

dialog.setTabOrder(self.foreigners\_eb\_check\_box\_d, self.enrollment\_paid\_check\_box\_d)

dialog.setTabOrder(self.enrollment\_paid\_check\_box\_d, self.disabled\_ep\_check\_box\_d)

dialog.setTabOrder(self.disabled\_ep\_check\_box\_d, self.target\_ep\_check\_box\_d)

dialog.setTabOrder(self.target\_ep\_check\_box\_d, self.foreigners\_ep\_check\_box\_d)

dialog.setTabOrder(self.foreigners\_ep\_check\_box\_d, self.deduct\_all\_check\_box\_d)

dialog.setTabOrder(self.deduct\_all\_check\_box\_d, self.disabled\_da\_check\_box\_d)

dialog.setTabOrder(self.disabled\_da\_check\_box\_d, self.target\_da\_check\_box\_d)

dialog.setTabOrder(self.target\_da\_check\_box\_d, self.foreigners\_da\_check\_box\_d)

dialog.setTabOrder(self.foreigners\_da\_check\_box\_d, self.deduct\_budget\_check\_box\_d)

dialog.setTabOrder(self.deduct\_budget\_check\_box\_d, self.disabled\_db\_check\_box\_d)

dialog.setTabOrder(self.disabled\_db\_check\_box\_d, self.target\_db\_check\_box\_d)

dialog.setTabOrder(self.target\_db\_check\_box\_d, self.foreigners\_db\_check\_box\_d)

dialog.setTabOrder(self.foreigners\_db\_check\_box\_d, self.deduct\_paid\_check\_box)

dialog.setTabOrder(self.deduct\_paid\_check\_box, self.disabled\_dp\_check\_box\_d)

dialog.setTabOrder(self.disabled\_dp\_check\_box\_d, self.target\_dp\_check\_box\_d)

dialog.setTabOrder(self.target\_dp\_check\_box\_d, self.foreigners\_dp\_check\_box\_d)

dialog.setTabOrder(self.foreigners\_dp\_check\_box\_d, self.continue\_button\_d)

dialog.setTabOrder(self.continue\_button\_d, self.back\_button\_d)

# Триггеры для пользовательского отчета

self.loe\_combo\_box\_d.currentIndexChanged.connect(self.set\_combo\_boxes)

self.back\_button\_d.clicked.connect(self.close\_window)

self.continue\_button\_d.clicked.connect(self.get\_condition)

# Предварительная донастройка

self.set\_combo\_boxes()

def retranslate\_ui(self, dialog):

"""Установка всех встроенных названий"""

\_translate = QtCore.QCoreApplication.translate

dialog.setWindowTitle(\_translate("Dialog", "StudInStudOut"))

self.course\_label\_d.setText(\_translate("Dialog", "Курс"))

self.loe\_combo\_box\_d.setItemText(0, \_translate("Dialog", "Все"))

self.loe\_combo\_box\_d.setItemText(1, \_translate("Dialog", "Бакалавриат"))

self.loe\_combo\_box\_d.setItemText(2, \_translate("Dialog", "Магистратура"))

self.loe\_combo\_box\_d.setItemText(3, \_translate("Dialog", "Специалитет"))

self.loe\_combo\_box\_d.setItemText(4, \_translate("Dialog", "ККМТ"))

self.loe\_combo\_box\_d.setItemText(5, \_translate("Dialog", "ТТД"))

self.direction\_combo\_box\_d.setItemText(0, \_translate("Dialog", "Все"))

self.direction\_label\_d.setText(\_translate("Dialog", "Название направления подготовки"))

self.foe\_combo\_box\_d.setItemText(0, \_translate("Dialog", "Все"))

self.foe\_combo\_box\_d.setItemText(1, \_translate("Dialog", "Очная"))

self.foe\_combo\_box\_d.setItemText(2, \_translate("Dialog", "Очно-заочная"))

self.foe\_combo\_box\_d.setItemText(3, \_translate("Dialog", "Заочная"))

self.course\_combo\_box\_d.setItemText(0, \_translate("Dialog", "Все"))

self.course\_combo\_box\_d.setItemText(1, \_translate("Dialog", "1"))

self.course\_combo\_box\_d.setItemText(2, \_translate("Dialog", "2"))

self.course\_combo\_box\_d.setItemText(3, \_translate("Dialog", "3"))

self.course\_combo\_box\_d.setItemText(4, \_translate("Dialog", "4"))

self.course\_combo\_box\_d.setItemText(5, \_translate("Dialog", "5"))

self.course\_combo\_box\_d.setItemText(6, \_translate("Dialog", "6"))

self.loe\_label\_d.setText(\_translate("Dialog", "Уровень обучения"))

self.for\_label\_d.setText(\_translate("Dialog", "Форма обучения"))

self.year\_label\_d.setText(\_translate("Dialog", "Год"))

self.month\_label\_d.setText(\_translate("Dialog", "Месяц"))

self.year\_combo\_box\_d.setItemText(0, \_translate("Dialog", "Все"))

self.month\_combo\_box\_d.setItemText(0, \_translate("Dialog", "Все"))

self.month\_combo\_box\_d.setItemText(1, \_translate("Dialog", "Январь"))

self.month\_combo\_box\_d.setItemText(2, \_translate("Dialog", "Февраль"))

self.month\_combo\_box\_d.setItemText(3, \_translate("Dialog", "Март"))

self.month\_combo\_box\_d.setItemText(4, \_translate("Dialog", "Апрель"))

self.month\_combo\_box\_d.setItemText(5, \_translate("Dialog", "Май"))

self.month\_combo\_box\_d.setItemText(6, \_translate("Dialog", "Июнь"))

self.month\_combo\_box\_d.setItemText(7, \_translate("Dialog", "Июль"))

self.month\_combo\_box\_d.setItemText(8, \_translate("Dialog", "Август"))

self.month\_combo\_box\_d.setItemText(9, \_translate("Dialog", "Сентябрь"))

self.month\_combo\_box\_d.setItemText(10, \_translate("Dialog", "Октябрь"))

self.month\_combo\_box\_d.setItemText(11, \_translate("Dialog", "Ноябрь"))

self.month\_combo\_box\_d.setItemText(12, \_translate("Dialog", "Декабрь"))

self.enrollment\_all\_check\_box\_d.setText(\_translate("Dialog", "Всего зачислено"))

self.foreigners\_eb\_check\_box\_d.setText(\_translate("Dialog", "Из них иностранные граждане (зачислено бюджет)"))

self.foreigners\_ea\_check\_box\_d.setText(\_translate("Dialog", "Из них иностранные граждане (всего зачислено)"))

self.enrollment\_paid\_check\_box\_d.setText(\_translate("Dialog",

"Зачислено по договорам об оказании платных "

"образовательных услуг"))

self.target\_ep\_check\_box\_d.setText(\_translate("Dialog",

"Из них на места в рамках квоты целевого приема "

"(зачислено платное)"))

self.disabled\_ea\_check\_box\_d.setText(\_translate("Dialog",

"Из них лица с ОВЗ, инвалиды, дети-инвалиды "

"(всего зачислено)"))

self.foreigners\_ep\_check\_box\_d.setText(\_translate("Dialog", "Из них иностранные граждане (зачислено платное)"))

self.target\_ea\_check\_box\_d.setText(\_translate("Dialog",

"Из них на места в рамках квоты целевого приема "

"(всего зачислено)"))

self.target\_eb\_check\_box\_d.setText(\_translate("Dialog",

"Из них на места в рамках квоты целевого приема "

"(зачислено бюджет)"))

self.disabled\_ep\_check\_box\_d.setText(\_translate("Dialog",

"Из них лица с ОВЗ, инвалиды, дети-инвалиды "

"(зачислено платное)"))

self.enrollment\_budget\_check\_box\_d.setText(\_translate("Dialog",

"Зачислено за счет бюджетных ассигнований бюджета "

"субъекта РФ"))

self.disabled\_eb\_check\_box\_d.setText(\_translate("Dialog",

"Из них лица с ОВЗ, инвалиды, дети-инвалиды "

"(зачислено бюджет)"))

self.continue\_button\_d.setText(\_translate("Dialog", "Продолжить"))

self.foreigners\_da\_check\_box\_d.setText(\_translate("Dialog", "Из них иностранные граждане (всего отчислено)"))

self.deduct\_budget\_check\_box\_d.setText(\_translate("Dialog",

"Отчислено за счет бюджетных ассигнований бюджета "

"субъекта РФ"))

self.disabled\_db\_check\_box\_d.setText(\_translate("Dialog",

"Из них лица с ОВЗ, инвалиды, дети-инвалиды "

"(отчислено бюджет)"))

self.target\_db\_check\_box\_d.setText(\_translate("Dialog",

"Из них на места в рамках квоты целевого приема "

"(отчислено бюджет)"))

self.target\_dp\_check\_box\_d.setText(\_translate("Dialog",

"Из них на места в рамках квоты целевого приема "

"(отчислено платное)"))

self.foreigners\_dp\_check\_box\_d.setText(\_translate("Dialog", "Из них иностранные граждане (отчислено платное)"))

self.target\_da\_check\_box\_d.setText(\_translate("Dialog",

"Из них на места в рамках квоты целевого приема "

"(всего отчислено)"))

self.disabled\_dp\_check\_box\_d.setText(\_translate("Dialog",

"Из них лица с ОВЗ, инвалиды, дети-инвалиды "

"(отчислено платное)"))

self.disabled\_da\_check\_box\_d.setText(\_translate("Dialog",

"Из них лица с ОВЗ, инвалиды, дети-инвалиды (всего отчислено)"))

self.deduct\_paid\_check\_box.setText(\_translate("Dialog",

"Отчислено по договорам об оказании платных "

"образовательных услуг"))

self.deduct\_all\_check\_box\_d.setText(\_translate("Dialog", "Всего отчислено"))

self.foreigners\_db\_check\_box\_d.setText(\_translate("Dialog", "Из них иностранные граждане (отчислено бюджет)"))

self.back\_button\_d.setText(\_translate("Dialog", "Назад"))

def close\_window(self):

"""Закрывает окно диалога"""

self.dialog.close()

def set\_combo\_boxes(self):

"""Устанавливает встроенные значения combo box"""

self.direction\_combo\_box\_d.blockSignals(True)

self.direction\_combo\_box\_d.clear()

self.direction\_combo\_box\_d.addItems(["Все", \*self.get\_directions(self.loe\_combo\_box\_d.currentText())])

self.direction\_combo\_box\_d.blockSignals(False)

self.year\_combo\_box\_d.blockSignals(True)

self.year\_combo\_box\_d.clear()

self.year\_combo\_box\_d.addItems(["Все", \*self.get\_years(self.loe\_combo\_box\_d.currentText())])

self.year\_combo\_box\_d.blockSignals(False)

self.month\_combo\_box\_d.blockSignals(True)

self.month\_combo\_box\_d.clear()

self.month\_combo\_box\_d.addItems(["Все", \*self.get\_months(self.loe\_combo\_box\_d.currentText())])

self.month\_combo\_box\_d.blockSignals(False)

def get\_condition(self):

"""Получение данных необходимых для экспорта"""

book\_1 = {"Названия направления подготовки": [],

"Уровень обучения": [], "Форма обучения": [], "Дата записи": [], "Курс": [], "Всего зачислено": [],

"Из них лица с ОВЗ, инвалиды, дети-инвалиды (всего зачислено)": [],

"Из них на места в рамках квоты целевого приема (всего зачислено)": [],

"Из них иностранные граждане (всего зачислено)": [], "Всего отчислено": [],

"Из них лица с ОВЗ, инвалиды, дети-инвалиды (всего отчислено)": [],

"Из них на места в рамках квоты целевого приема (всего отчислено)": [],

"Из них иностранные граждане (всего отчислено)": [],

"Зачислено за счет бюджетных ассигнований бюджета субъекта РФ": [],

"Из них лица с ОВЗ, инвалиды, дети-инвалиды (зачислено бюджет)": [],

"Из них на места в рамках квоты целевого приема (зачислено бюджет)": [],

"Из них иностранные граждане (зачислено бюджет)": [],

"Зачислено по договорам об оказании платных образовательных услуг": [],

"Из них лица с ОВЗ, инвалиды, дети-инвалиды (зачислено платное)": [],

"Из них на места в рамках квоты целевого приема (зачислено платное)": [],

"Из них иностранные граждане (зачислено платное)": [],

"Отчислено за счет бюджетных ассигнований бюджета субъекта РФ": [],

"Из них лица с ОВЗ, инвалиды, дети-инвалиды (отчислено бюджет)": [],

"Из них на места в рамках квоты целевого приема (отчислено бюджет)": [],

"Из них иностранные граждане (отчислено бюджет)": [],

"Отчислено по договорам об оказании платных образовательных услуг": [],

"Из них лица с ОВЗ, инвалиды, дети-инвалиды (отчислено платное)": [],

"Из них на места в рамках квоты целевого приема (отчислено платное)": [],

"Из них иностранные граждане (отчислено платное)": []}

book\_2 = {"Названия направления подготовки": [],

"Уровень обучения": [], "Форма обучения": [], "Дата записи": [], "Курс": [], "Всего зачислено": [],

"Из них лица с ОВЗ, инвалиды, дети-инвалиды (всего зачислено)": [],

"Из них на места в рамках квоты целевого приема (всего зачислено)": [],

"Из них иностранные граждане (всего зачислено)": [], "Всего отчислено": [],

"Из них лица с ОВЗ, инвалиды, дети-инвалиды (всего отчислено)": [],

"Из них на места в рамках квоты целевого приема (всего отчислено)": [],

"Из них иностранные граждане (всего отчислено)": [],

"Зачислено за счет бюджетных ассигнований бюджета субъекта РФ": [],

"Из них лица с ОВЗ, инвалиды, дети-инвалиды (зачислено бюджет)": [],

"Из них на места в рамках квоты целевого приема (зачислено бюджет)": [],

"Из них иностранные граждане (зачислено бюджет)": [],

"Зачислено по договорам об оказании платных образовательных услуг": [],

"Из них лица с ОВЗ, инвалиды, дети-инвалиды (зачислено платное)": [],

"Из них на места в рамках квоты целевого приема (зачислено платное)": [],

"Из них иностранные граждане (зачислено платное)": [],

"Отчислено за счет бюджетных ассигнований бюджета субъекта РФ": [],

"Из них лица с ОВЗ, инвалиды, дети-инвалиды (отчислено бюджет)": [],

"Из них на места в рамках квоты целевого приема (отчислено бюджет)": [],

"Из них иностранные граждане (отчислено бюджет)": [],

"Отчислено по договорам об оказании платных образовательных услуг": [],

"Из них лица с ОВЗ, инвалиды, дети-инвалиды (отчислено платное)": [],

"Из них на места в рамках квоты целевого приема (отчислено платное)": [],

"Из них иностранные граждане (отчислено платное)": []}

book\_3 = {"Названия направления подготовки": [],

"Уровень обучения": [], "Форма обучения": [], "Дата записи": [], "Курс": [], "Всего зачислено": [],

"Из них лица с ОВЗ, инвалиды, дети-инвалиды (всего зачислено)": [],

"Из них на места в рамках квоты целевого приема (всего зачислено)": [],

"Из них иностранные граждане (всего зачислено)": [], "Всего отчислено": [],

"Из них лица с ОВЗ, инвалиды, дети-инвалиды (всего отчислено)": [],

"Из них на места в рамках квоты целевого приема (всего отчислено)": [],

"Из них иностранные граждане (всего отчислено)": [],

"Зачислено за счет бюджетных ассигнований бюджета субъекта РФ": [],

"Из них лица с ОВЗ, инвалиды, дети-инвалиды (зачислено бюджет)": [],

"Из них на места в рамках квоты целевого приема (зачислено бюджет)": [],

"Из них иностранные граждане (зачислено бюджет)": [],

"Зачислено по договорам об оказании платных образовательных услуг": [],

"Из них лица с ОВЗ, инвалиды, дети-инвалиды (зачислено платное)": [],

"Из них на места в рамках квоты целевого приема (зачислено платное)": [],

"Из них иностранные граждане (зачислено платное)": [],

"Отчислено за счет бюджетных ассигнований бюджета субъекта РФ": [],

"Из них лица с ОВЗ, инвалиды, дети-инвалиды (отчислено бюджет)": [],

"Из них на места в рамках квоты целевого приема (отчислено бюджет)": [],

"Из них иностранные граждане (отчислено бюджет)": [],

"Отчислено по договорам об оказании платных образовательных услуг": [],

"Из них лица с ОВЗ, инвалиды, дети-инвалиды (отчислено платное)": [],

"Из них на места в рамках квоты целевого приема (отчислено платное)": [],

"Из них иностранные граждане (отчислено платное)": []}

is\_needed = [True, True, True, True, True, self.enrollment\_all\_check\_box\_d.isChecked(),

self.disabled\_ea\_check\_box\_d.isChecked(), self.target\_ea\_check\_box\_d.isChecked(),

self.foreigners\_ea\_check\_box\_d.isChecked(), self.deduct\_all\_check\_box\_d.isChecked(),

self.disabled\_da\_check\_box\_d.isChecked(), self.target\_da\_check\_box\_d.isChecked(),

self.foreigners\_da\_check\_box\_d.isChecked(), self.enrollment\_budget\_check\_box\_d.isChecked(),

self.disabled\_eb\_check\_box\_d.isChecked(), self.target\_eb\_check\_box\_d.isChecked(),

self.foreigners\_eb\_check\_box\_d.isChecked(), self.enrollment\_paid\_check\_box\_d.isChecked(),

self.disabled\_ep\_check\_box\_d.isChecked(), self.target\_ep\_check\_box\_d.isChecked(),

self.foreigners\_ep\_check\_box\_d.isChecked(), self.deduct\_budget\_check\_box\_d.isChecked(),

self.disabled\_db\_check\_box\_d.isChecked(), self.target\_db\_check\_box\_d.isChecked(),

self.foreigners\_db\_check\_box\_d.isChecked(), self.deduct\_paid\_check\_box.isChecked(),

self.disabled\_dp\_check\_box\_d.isChecked(), self.target\_dp\_check\_box\_d.isChecked(),

self.foreigners\_dp\_check\_box\_d.isChecked()]

if self.foe\_combo\_box\_d.currentText() == "Все":

data = self.get\_data(self.loe\_combo\_box\_d.currentText(), "Очная",

self.direction\_combo\_box\_d.currentText(), self.year\_combo\_box\_d.currentText(),

self.month\_combo\_box\_d.currentText(), self.course\_combo\_box\_d.currentText())

for k, chapter in enumerate(book\_1, 1):

for recording in data:

if is\_needed[k - 1]:

book\_1[chapter].append(recording[k])

book\_1 = {key: item for key, item in book\_1.items() if item}

data = self.get\_data(self.loe\_combo\_box\_d.currentText(), "Очно-заочная",

self.direction\_combo\_box\_d.currentText(), self.year\_combo\_box\_d.currentText(),

self.month\_combo\_box\_d.currentText(), self.course\_combo\_box\_d.currentText())

for k, chapter in enumerate(book\_2, 1):

for recording in data:

if is\_needed[k - 1]:

book\_2[chapter].append(recording[k])

book\_2 = {key: item for key, item in book\_2.items() if item}

data = self.get\_data(self.loe\_combo\_box\_d.currentText(), "Заочная",

self.direction\_combo\_box\_d.currentText(), self.year\_combo\_box\_d.currentText(),

self.month\_combo\_box\_d.currentText(), self.course\_combo\_box\_d.currentText())

for k, chapter in enumerate(book\_3, 1):

for recording in data:

if is\_needed[k - 1]:

book\_3[chapter].append(recording[k])

book\_3 = {key: item for key, item in book\_3.items() if item}

self.export\_data(book\_1, self.foe\_combo\_box\_d.currentText(), book\_2, book\_3)

elif self.foe\_combo\_box\_d.currentText() == "Очная":

data = self.get\_data(self.loe\_combo\_box\_d.currentText(), "Очная",

self.direction\_combo\_box\_d.currentText(), self.year\_combo\_box\_d.currentText(),

self.month\_combo\_box\_d.currentText(), self.course\_combo\_box\_d.currentText())

for k, chapter in enumerate(book\_1, 1):

for recording in data:

if is\_needed[k - 1]:

book\_1[chapter].append(recording[k])

book\_1 = {key: item for key, item in book\_1.items() if item}

self.export\_data(book\_1, "Очная")

elif self.foe\_combo\_box\_d.currentText() == "Очно-заочная":

data = self.get\_data(self.loe\_combo\_box\_d.currentText(), "Очно-заочная",

self.direction\_combo\_box\_d.currentText(), self.year\_combo\_box\_d.currentText(),

self.month\_combo\_box\_d.currentText(), self.course\_combo\_box\_d.currentText())

for k, chapter in enumerate(book\_1, 1):

for recording in data:

if is\_needed[k - 1]:

book\_1[chapter].append(recording[k])

book\_1 = {key: item for key, item in book\_1.items() if item}

self.export\_data(book\_1, "Очно-заочная")

elif self.foe\_combo\_box\_d.currentText() == "Заочная":

data = self.get\_data(self.loe\_combo\_box\_d.currentText(), "Заочная",

self.direction\_combo\_box\_d.currentText(), self.year\_combo\_box\_d.currentText(),

self.month\_combo\_box\_d.currentText(), self.course\_combo\_box\_d.currentText())

for k, chapter in enumerate(book\_1, 1):

for recording in data:

if is\_needed[k - 1]:

book\_1[chapter].append(recording[k])

book\_1 = {key: item for key, item in book\_1.items() if item}

self.export\_data(book\_1, "Заочная")

**Листинг 4. dialog\_simple\_ui.py**

import excel\_control

from PyQt5 import QtCore, QtGui, QtWidgets

class UiDialogSimple(excel\_control.ExcelControl):

def \_\_init\_\_(self, dialog, trigger):

"""Конструктор класса UiDialog\_simple"""

super(UiDialogSimple, self).\_\_init\_\_()

self.dialog = dialog

self.trigger = trigger

dialog.setObjectName("Dialog")

dialog.resize(360, 542)

dialog.setMinimumSize(QtCore.QSize(360, 542))

dialog.setMaximumSize(QtCore.QSize(360, 542))

icon = QtGui.QIcon()

icon.addPixmap(QtGui.QPixmap("Изображения/free-icon-binder-4624080.png"), QtGui.QIcon.Normal, QtGui.QIcon.Off)

dialog.setWindowIcon(icon)

self.gridLayout\_2 = QtWidgets.QGridLayout(dialog)

self.gridLayout\_2.setObjectName("gridLayout\_2")

self.gridLayout = QtWidgets.QGridLayout()

self.gridLayout.setObjectName("gridLayout")

self.month\_label\_md = QtWidgets.QLabel(dialog)

font = QtGui.QFont()

font.setPointSize(12)

self.month\_label\_md.setFont(font)

self.month\_label\_md.setAlignment(QtCore.Qt.AlignCenter)

self.month\_label\_md.setObjectName("month\_label\_md")

self.gridLayout.addWidget(self.month\_label\_md, 16, 1, 1, 1)

self.loe\_label\_md = QtWidgets.QLabel(dialog)

font = QtGui.QFont()

font.setPointSize(12)

self.loe\_label\_md.setFont(font)

self.loe\_label\_md.setAlignment(QtCore.Qt.AlignCenter)

self.loe\_label\_md.setObjectName("loe\_label\_md")

self.gridLayout.addWidget(self.loe\_label\_md, 1, 1, 1, 1)

self.course\_combo\_box\_md = QtWidgets.QComboBox(dialog)

font = QtGui.QFont()

font.setPointSize(12)

self.course\_combo\_box\_md.setFont(font)

self.course\_combo\_box\_md.setObjectName("course\_combo\_box\_md")

self.course\_combo\_box\_md.addItem("")

self.course\_combo\_box\_md.addItem("")

self.course\_combo\_box\_md.addItem("")

self.course\_combo\_box\_md.addItem("")

self.course\_combo\_box\_md.addItem("")

self.course\_combo\_box\_md.addItem("")

self.course\_combo\_box\_md.addItem("")

self.gridLayout.addWidget(self.course\_combo\_box\_md, 11, 1, 1, 1)

self.line\_2 = QtWidgets.QFrame(dialog)

self.line\_2.setFrameShadow(QtWidgets.QFrame.Raised)

self.line\_2.setMidLineWidth(0)

self.line\_2.setFrameShape(QtWidgets.QFrame.HLine)

self.line\_2.setObjectName("line\_2")

self.gridLayout.addWidget(self.line\_2, 15, 1, 1, 1)

self.line\_3 = QtWidgets.QFrame(dialog)

self.line\_3.setFrameShadow(QtWidgets.QFrame.Raised)

self.line\_3.setMidLineWidth(0)

self.line\_3.setFrameShape(QtWidgets.QFrame.HLine)

self.line\_3.setObjectName("line\_3")

self.gridLayout.addWidget(self.line\_3, 12, 1, 1, 1)

self.loe\_combo\_box\_md = QtWidgets.QComboBox(dialog)

font = QtGui.QFont()

font.setPointSize(12)

self.loe\_combo\_box\_md.setFont(font)

self.loe\_combo\_box\_md.setObjectName("loe\_combo\_box\_md")

self.loe\_combo\_box\_md.addItem("")

self.loe\_combo\_box\_md.addItem("")

self.loe\_combo\_box\_md.addItem("")

self.loe\_combo\_box\_md.addItem("")

self.loe\_combo\_box\_md.addItem("")

self.loe\_combo\_box\_md.addItem("")

self.gridLayout.addWidget(self.loe\_combo\_box\_md, 2, 1, 1, 1)

self.foe\_combo\_box\_md = QtWidgets.QComboBox(dialog)

font = QtGui.QFont()

font.setPointSize(12)

self.foe\_combo\_box\_md.setFont(font)

self.foe\_combo\_box\_md.setObjectName("foe\_combo\_box\_md")

self.foe\_combo\_box\_md.addItem("")

self.foe\_combo\_box\_md.addItem("")

self.foe\_combo\_box\_md.addItem("")

self.foe\_combo\_box\_md.addItem("")

self.gridLayout.addWidget(self.foe\_combo\_box\_md, 8, 1, 1, 1)

self.month\_combo\_box\_md = QtWidgets.QComboBox(dialog)

font = QtGui.QFont()

font.setPointSize(12)

self.month\_combo\_box\_md.setFont(font)

self.month\_combo\_box\_md.setObjectName("month\_combo\_box\_md")

self.month\_combo\_box\_md.addItem("")

self.gridLayout.addWidget(self.month\_combo\_box\_md, 17, 1, 1, 1)

self.course\_label\_md = QtWidgets.QLabel(dialog)

font = QtGui.QFont()

font.setPointSize(12)

self.course\_label\_md.setFont(font)

self.course\_label\_md.setAlignment(QtCore.Qt.AlignCenter)

self.course\_label\_md.setObjectName("course\_label\_md")

self.gridLayout.addWidget(self.course\_label\_md, 10, 1, 1, 1)

self.year\_combo\_box\_md = QtWidgets.QComboBox(dialog)

font = QtGui.QFont()

font.setPointSize(12)

self.year\_combo\_box\_md.setFont(font)

self.year\_combo\_box\_md.setObjectName("year\_combo\_box\_md")

self.year\_combo\_box\_md.addItem("")

self.gridLayout.addWidget(self.year\_combo\_box\_md, 14, 1, 1, 1)

self.year\_label\_md = QtWidgets.QLabel(dialog)

font = QtGui.QFont()

font.setPointSize(12)

self.year\_label\_md.setFont(font)

self.year\_label\_md.setAlignment(QtCore.Qt.AlignCenter)

self.year\_label\_md.setObjectName("year\_label\_md")

self.gridLayout.addWidget(self.year\_label\_md, 13, 1, 1, 1)

self.direction\_combo\_box\_md = QtWidgets.QComboBox(dialog)

font = QtGui.QFont()

font.setPointSize(12)

self.direction\_combo\_box\_md.setFont(font)

self.direction\_combo\_box\_md.setObjectName("direction\_combo\_box\_md")

self.direction\_combo\_box\_md.addItem("")

self.gridLayout.addWidget(self.direction\_combo\_box\_md, 5, 1, 1, 1)

self.line\_5 = QtWidgets.QFrame(dialog)

self.line\_5.setFrameShadow(QtWidgets.QFrame.Raised)

self.line\_5.setMidLineWidth(0)

self.line\_5.setFrameShape(QtWidgets.QFrame.HLine)

self.line\_5.setObjectName("line\_5")

self.gridLayout.addWidget(self.line\_5, 6, 1, 1, 1)

self.line\_4 = QtWidgets.QFrame(dialog)

self.line\_4.setFrameShadow(QtWidgets.QFrame.Raised)

self.line\_4.setMidLineWidth(0)

self.line\_4.setFrameShape(QtWidgets.QFrame.HLine)

self.line\_4.setObjectName("line\_4")

self.gridLayout.addWidget(self.line\_4, 9, 1, 1, 1)

self.back\_button\_md = QtWidgets.QPushButton(dialog)

font = QtGui.QFont()

font.setPointSize(12)

self.back\_button\_md.setFont(font)

self.back\_button\_md.setObjectName("back\_button\_md")

self.gridLayout.addWidget(self.back\_button\_md, 20, 1, 1, 1)

self.line = QtWidgets.QFrame(dialog)

self.line.setFrameShadow(QtWidgets.QFrame.Raised)

self.line.setMidLineWidth(0)

self.line.setFrameShape(QtWidgets.QFrame.HLine)

self.line.setObjectName("line")

self.gridLayout.addWidget(self.line, 18, 1, 1, 1)

self.foe\_label\_md = QtWidgets.QLabel(dialog)

font = QtGui.QFont()

font.setPointSize(12)

self.foe\_label\_md.setFont(font)

self.foe\_label\_md.setAlignment(QtCore.Qt.AlignCenter)

self.foe\_label\_md.setObjectName("foe\_label\_md")

self.gridLayout.addWidget(self.foe\_label\_md, 7, 1, 1, 1)

self.direction\_label\_md = QtWidgets.QLabel(dialog)

font = QtGui.QFont()

font.setPointSize(12)

self.direction\_label\_md.setFont(font)

self.direction\_label\_md.setAlignment(QtCore.Qt.AlignCenter)

self.direction\_label\_md.setObjectName("direction\_label\_md")

self.gridLayout.addWidget(self.direction\_label\_md, 4, 1, 1, 1)

self.line\_6 = QtWidgets.QFrame(dialog)

self.line\_6.setFrameShadow(QtWidgets.QFrame.Raised)

self.line\_6.setMidLineWidth(0)

self.line\_6.setFrameShape(QtWidgets.QFrame.HLine)

self.line\_6.setObjectName("line\_6")

self.gridLayout.addWidget(self.line\_6, 3, 1, 1, 1)

self.continue\_button\_md = QtWidgets.QPushButton(dialog)

font = QtGui.QFont()

font.setPointSize(12)

self.continue\_button\_md.setFont(font)

self.continue\_button\_md.setObjectName("continue\_button\_md")

self.gridLayout.addWidget(self.continue\_button\_md, 19, 1, 1, 1)

self.gridLayout\_2.addLayout(self.gridLayout, 0, 0, 1, 1)

self.retranslate\_ui(dialog)

QtCore.QMetaObject.connectSlotsByName(dialog)

dialog.setTabOrder(self.loe\_combo\_box\_md, self.direction\_combo\_box\_md)

dialog.setTabOrder(self.direction\_combo\_box\_md, self.foe\_combo\_box\_md)

dialog.setTabOrder(self.foe\_combo\_box\_md, self.course\_combo\_box\_md)

dialog.setTabOrder(self.course\_combo\_box\_md, self.year\_combo\_box\_md)

dialog.setTabOrder(self.year\_combo\_box\_md, self.month\_combo\_box\_md)

dialog.setTabOrder(self.month\_combo\_box\_md, self.back\_button\_md)

# Триггеры для встроенных отчетов

self.loe\_combo\_box\_md.currentIndexChanged.connect(self.set\_combo\_boxes)

self.back\_button\_md.clicked.connect(self.close\_window)

self.continue\_button\_md.clicked.connect(self.get\_condition)

# Предварительная донастройка

self.set\_combo\_boxes()

def retranslate\_ui(self, dialog):

"""Установка всех встроенных названий"""

\_translate = QtCore.QCoreApplication.translate

dialog.setWindowTitle(\_translate("Dialog", "StudInStudOut"))

self.month\_label\_md.setText(\_translate("Dialog", "Месяц"))

self.loe\_label\_md.setText(\_translate("Dialog", "Уровень образования"))

self.course\_combo\_box\_md.setItemText(0, \_translate("Dialog", "Все"))

self.course\_combo\_box\_md.setItemText(1, \_translate("Dialog", "1"))

self.course\_combo\_box\_md.setItemText(2, \_translate("Dialog", "2"))

self.course\_combo\_box\_md.setItemText(3, \_translate("Dialog", "3"))

self.course\_combo\_box\_md.setItemText(4, \_translate("Dialog", "4"))

self.course\_combo\_box\_md.setItemText(5, \_translate("Dialog", "5"))

self.course\_combo\_box\_md.setItemText(6, \_translate("Dialog", "6"))

self.loe\_combo\_box\_md.setItemText(0, \_translate("Dialog", "Все"))

self.loe\_combo\_box\_md.setItemText(1, \_translate("Dialog", "Бакалавриат"))

self.loe\_combo\_box\_md.setItemText(2, \_translate("Dialog", "Магистратура"))

self.loe\_combo\_box\_md.setItemText(3, \_translate("Dialog", "Специалитет"))

self.loe\_combo\_box\_md.setItemText(4, \_translate("Dialog", "ККМТ"))

self.loe\_combo\_box\_md.setItemText(5, \_translate("Dialog", "ТТД"))

self.foe\_combo\_box\_md.setItemText(0, \_translate("Dialog", "Все"))

self.foe\_combo\_box\_md.setItemText(1, \_translate("Dialog", "Очная"))

self.foe\_combo\_box\_md.setItemText(2, \_translate("Dialog", "Очно-заочная"))

self.foe\_combo\_box\_md.setItemText(3, \_translate("Dialog", "Заочная"))

self.month\_combo\_box\_md.setItemText(0, \_translate("Dialog", "Все"))

self.course\_label\_md.setText(\_translate("Dialog", "Курс"))

self.year\_combo\_box\_md.setItemText(0, \_translate("Dialog", "Все"))

self.year\_label\_md.setText(\_translate("Dialog", "Год"))

self.direction\_combo\_box\_md.setItemText(0, \_translate("Dialog", "Все"))

self.back\_button\_md.setText(\_translate("Dialog", "Назад"))

self.foe\_label\_md.setText(\_translate("Dialog", "Форма обучения"))

self.direction\_label\_md.setText(\_translate("Dialog", "Название направления подготовки"))

self.continue\_button\_md.setText(\_translate("Dialog", "Продолжить"))

def close\_window(self):

"""Закрывает окно диалога"""

self.dialog.close()

def set\_combo\_boxes(self):

"""Устанавливает встроенные значения combo box"""

self.direction\_combo\_box\_md.blockSignals(True)

self.direction\_combo\_box\_md.clear()

self.direction\_combo\_box\_md.addItems(["Все", \*self.get\_directions(self.loe\_combo\_box\_md.currentText())])

self.direction\_combo\_box\_md.blockSignals(False)

self.year\_combo\_box\_md.blockSignals(True)

self.year\_combo\_box\_md.clear()

self.year\_combo\_box\_md.addItems(["Все", \*self.get\_years(self.loe\_combo\_box\_md.currentText())])

self.year\_combo\_box\_md.blockSignals(False)

self.month\_combo\_box\_md.blockSignals(True)

self.month\_combo\_box\_md.clear()

self.month\_combo\_box\_md.addItems(["Все", \*self.get\_months(self.loe\_combo\_box\_md.currentText())])

self.month\_combo\_box\_md.blockSignals(False)

def get\_condition(self):

"""Получение данных необходимых для экспорта"""

if self.trigger == "Все":

book\_1 = {"Названия направления подготовки": [],

"Уровень обучения": [], "Форма обучения": [], "Дата записи": [], "Курс": [],

"Всего зачислено": [],

"Из них лица с ОВЗ, инвалиды, дети-инвалиды (всего зачислено)": [],

"Из них на места в рамках квоты целевого приема (всего зачислено)": [],

"Из них иностранные граждане (всего зачислено)": [], "Всего отчислено": [],

"Из них лица с ОВЗ, инвалиды, дети-инвалиды (всего отчислено)": [],

"Из них на места в рамках квоты целевого приема (всего отчислено)": [],

"Из них иностранные граждане (всего отчислено)": [],

"Зачислено за счет бюджетных ассигнований бюджета субъекта РФ": [],

"Из них лица с ОВЗ, инвалиды, дети-инвалиды (зачислено бюджет)": [],

"Из них на места в рамках квоты целевого приема (зачислено бюджет)": [],

"Из них иностранные граждане (зачислено бюджет)": [],

"Зачислено по договорам об оказании платных образовательных услуг": [],

"Из них лица с ОВЗ, инвалиды, дети-инвалиды (зачислено платное)": [],

"Из них на места в рамках квоты целевого приема (зачислено платное)": [],

"Из них иностранные граждане (зачислено платное)": [],

"Отчислено за счет бюджетных ассигнований бюджета субъекта РФ": [],

"Из них лица с ОВЗ, инвалиды, дети-инвалиды (отчислено бюджет)": [],

"Из них на места в рамках квоты целевого приема (отчислено бюджет)": [],

"Из них иностранные граждане (отчислено бюджет)": [],

"Отчислено по договорам об оказании платных образовательных услуг": [],

"Из них лица с ОВЗ, инвалиды, дети-инвалиды (отчислено платное)": [],

"Из них на места в рамках квоты целевого приема (отчислено платное)": [],

"Из них иностранные граждане (отчислено платное)": []}

book\_2 = {"Названия направления подготовки": [],

"Уровень обучения": [], "Форма обучения": [], "Дата записи": [], "Курс": [],

"Всего зачислено": [],

"Из них лица с ОВЗ, инвалиды, дети-инвалиды (всего зачислено)": [],

"Из них на места в рамках квоты целевого приема (всего зачислено)": [],

"Из них иностранные граждане (всего зачислено)": [], "Всего отчислено": [],

"Из них лица с ОВЗ, инвалиды, дети-инвалиды (всего отчислено)": [],

"Из них на места в рамках квоты целевого приема (всего отчислено)": [],

"Из них иностранные граждане (всего отчислено)": [],

"Зачислено за счет бюджетных ассигнований бюджета субъекта РФ": [],

"Из них лица с ОВЗ, инвалиды, дети-инвалиды (зачислено бюджет)": [],

"Из них на места в рамках квоты целевого приема (зачислено бюджет)": [],

"Из них иностранные граждане (зачислено бюджет)": [],

"Зачислено по договорам об оказании платных образовательных услуг": [],

"Из них лица с ОВЗ, инвалиды, дети-инвалиды (зачислено платное)": [],

"Из них на места в рамках квоты целевого приема (зачислено платное)": [],

"Из них иностранные граждане (зачислено платное)": [],

"Отчислено за счет бюджетных ассигнований бюджета субъекта РФ": [],

"Из них лица с ОВЗ, инвалиды, дети-инвалиды (отчислено бюджет)": [],

"Из них на места в рамках квоты целевого приема (отчислено бюджет)": [],

"Из них иностранные граждане (отчислено бюджет)": [],

"Отчислено по договорам об оказании платных образовательных услуг": [],

"Из них лица с ОВЗ, инвалиды, дети-инвалиды (отчислено платное)": [],

"Из них на места в рамках квоты целевого приема (отчислено платное)": [],

"Из них иностранные граждане (отчислено платное)": []}

book\_3 = {"Названия направления подготовки": [],

"Уровень обучения": [], "Форма обучения": [], "Дата записи": [], "Курс": [],

"Всего зачислено": [],

"Из них лица с ОВЗ, инвалиды, дети-инвалиды (всего зачислено)": [],

"Из них на места в рамках квоты целевого приема (всего зачислено)": [],

"Из них иностранные граждане (всего зачислено)": [], "Всего отчислено": [],

"Из них лица с ОВЗ, инвалиды, дети-инвалиды (всего отчислено)": [],

"Из них на места в рамках квоты целевого приема (всего отчислено)": [],

"Из них иностранные граждане (всего отчислено)": [],

"Зачислено за счет бюджетных ассигнований бюджета субъекта РФ": [],

"Из них лица с ОВЗ, инвалиды, дети-инвалиды (зачислено бюджет)": [],

"Из них на места в рамках квоты целевого приема (зачислено бюджет)": [],

"Из них иностранные граждане (зачислено бюджет)": [],

"Зачислено по договорам об оказании платных образовательных услуг": [],

"Из них лица с ОВЗ, инвалиды, дети-инвалиды (зачислено платное)": [],

"Из них на места в рамках квоты целевого приема (зачислено платное)": [],

"Из них иностранные граждане (зачислено платное)": [],

"Отчислено за счет бюджетных ассигнований бюджета субъекта РФ": [],

"Из них лица с ОВЗ, инвалиды, дети-инвалиды (отчислено бюджет)": [],

"Из них на места в рамках квоты целевого приема (отчислено бюджет)": [],

"Из них иностранные граждане (отчислено бюджет)": [],

"Отчислено по договорам об оказании платных образовательных услуг": [],

"Из них лица с ОВЗ, инвалиды, дети-инвалиды (отчислено платное)": [],

"Из них на места в рамках квоты целевого приема (отчислено платное)": [],

"Из них иностранные граждане (отчислено платное)": []}

if self.foe\_combo\_box\_md.currentText() == "Все":

data = self.get\_data(self.loe\_combo\_box\_md.currentText(), "Очная",

self.direction\_combo\_box\_md.currentText(), self.year\_combo\_box\_md.currentText(),

self.month\_combo\_box\_md.currentText(), self.course\_combo\_box\_md.currentText())

for k, chapter in enumerate(book\_1, 1):

for recording in data:

book\_1[chapter].append(recording[k])

book\_1 = {key: item for key, item in book\_1.items() if item}

data = self.get\_data(self.loe\_combo\_box\_md.currentText(), "Очно-заочная",

self.direction\_combo\_box\_md.currentText(), self.year\_combo\_box\_md.currentText(),

self.month\_combo\_box\_md.currentText(), self.course\_combo\_box\_md.currentText())

for k, chapter in enumerate(book\_2, 1):

for recording in data:

book\_2[chapter].append(recording[k])

book\_2 = {key: item for key, item in book\_2.items() if item}

data = self.get\_data(self.loe\_combo\_box\_md.currentText(), "Заочная",

self.direction\_combo\_box\_md.currentText(), self.year\_combo\_box\_md.currentText(),

self.month\_combo\_box\_md.currentText(), self.course\_combo\_box\_md.currentText())

for k, chapter in enumerate(book\_3, 1):

for recording in data:

book\_3[chapter].append(recording[k])

book\_3 = {key: item for key, item in book\_3.items() if item}

self.export\_data(book\_1, "Все", book\_2, book\_3)

elif self.foe\_combo\_box\_md.currentText() == "Очная":

data = self.get\_data(self.loe\_combo\_box\_md.currentText(), "Очная",

self.direction\_combo\_box\_md.currentText(), self.year\_combo\_box\_md.currentText(),

self.month\_combo\_box\_md.currentText(), self.course\_combo\_box\_md.currentText())

for k, chapter in enumerate(book\_1, 1):

for recording in data:

book\_1[chapter].append(recording[k])

book\_1 = {key: item for key, item in book\_1.items() if item}

self.export\_data(book\_1, "Очная")

elif self.foe\_combo\_box\_md.currentText() == "Очно-заочная":

data = self.get\_data(self.loe\_combo\_box\_md.currentText(), "Очно-заочная",

self.direction\_combo\_box\_md.currentText(), self.year\_combo\_box\_md.currentText(),

self.month\_combo\_box\_md.currentText(), self.course\_combo\_box\_md.currentText())

for k, chapter in enumerate(book\_1, 1):

for recording in data:

book\_1[chapter].append(recording[k])

book\_1 = {key: item for key, item in book\_1.items() if item}

self.export\_data(book\_1, "Очно-заочная")

elif self.foe\_combo\_box\_md.currentText() == "Заочная":

data = self.get\_data(self.loe\_combo\_box\_md.currentText(), "Заочная",

self.direction\_combo\_box\_md.currentText(), self.year\_combo\_box\_md.currentText(),

self.month\_combo\_box\_md.currentText(), self.course\_combo\_box\_md.currentText())

for k, chapter in enumerate(book\_1, 1):

for recording in data:

book\_1[chapter].append(recording[k])

book\_1 = {key: item for key, item in book\_1.items() if item}

self.export\_data(book\_1, "Заочная")

elif self.trigger == "Инвалиды":

book\_1 = {"Названия направления подготовки": [],

"Уровень обучения": [], "Форма обучения": [], "Дата записи": [], "Курс": [],

"Лица с ОВЗ, инвалиды, дети-инвалиды (всего зачислено)": [],

"Лица с ОВЗ, инвалиды, дети-инвалиды (всего отчислено)": [],

"Лица с ОВЗ, инвалиды, дети-инвалиды (зачислено бюджет)": [],

"Лица с ОВЗ, инвалиды, дети-инвалиды (зачислено платное)": [],

"Лица с ОВЗ, инвалиды, дети-инвалиды (отчислено бюджет)": [],

"Лица с ОВЗ, инвалиды, дети-инвалиды (отчислено платное)": []}

book\_2 = {"Названия направления подготовки": [],

"Уровень обучения": [], "Форма обучения": [], "Дата записи": [], "Курс": [],

"Лица с ОВЗ, инвалиды, дети-инвалиды (всего зачислено)": [],

"Лица с ОВЗ, инвалиды, дети-инвалиды (всего отчислено)": [],

"Лица с ОВЗ, инвалиды, дети-инвалиды (зачислено бюджет)": [],

"Лица с ОВЗ, инвалиды, дети-инвалиды (зачислено платное)": [],

"Лица с ОВЗ, инвалиды, дети-инвалиды (отчислено бюджет)": [],

"Лица с ОВЗ, инвалиды, дети-инвалиды (отчислено платное)": []}

book\_3 = {"Названия направления подготовки": [],

"Уровень обучения": [], "Форма обучения": [], "Дата записи": [], "Курс": [],

"Лица с ОВЗ, инвалиды, дети-инвалиды (всего зачислено)": [],

"Лица с ОВЗ, инвалиды, дети-инвалиды (всего отчислено)": [],

"Лица с ОВЗ, инвалиды, дети-инвалиды (зачислено бюджет)": [],

"Лица с ОВЗ, инвалиды, дети-инвалиды (зачислено платное)": [],

"Лица с ОВЗ, инвалиды, дети-инвалиды (отчислено бюджет)": [],

"Лица с ОВЗ, инвалиды, дети-инвалиды (отчислено платное)": []}

if self.foe\_combo\_box\_md.currentText() == "Все":

data = self.get\_disabled(self.loe\_combo\_box\_md.currentText(), "Очная",

self.direction\_combo\_box\_md.currentText(),

self.year\_combo\_box\_md.currentText(), self.month\_combo\_box\_md.currentText(),

self.course\_combo\_box\_md.currentText())

print(data)

for k, chapter in enumerate(book\_1):

for recording in data:

book\_1[chapter].append(recording[k])

book\_1 = {key: item for key, item in book\_1.items() if item}

data = self.get\_disabled(self.loe\_combo\_box\_md.currentText(), "Очно-заочная",

self.direction\_combo\_box\_md.currentText(),

self.year\_combo\_box\_md.currentText(), self.month\_combo\_box\_md.currentText(),

self.course\_combo\_box\_md.currentText())

for k, chapter in enumerate(book\_2):

for recording in data:

book\_2[chapter].append(recording[k])

book\_2 = {key: item for key, item in book\_2.items() if item}

data = self.get\_disabled(self.loe\_combo\_box\_md.currentText(), "Заочная",

self.direction\_combo\_box\_md.currentText(),

self.year\_combo\_box\_md.currentText(), self.month\_combo\_box\_md.currentText(),

self.course\_combo\_box\_md.currentText())

for k, chapter in enumerate(book\_3):

for recording in data:

book\_3[chapter].append(recording[k])

book\_3 = {key: item for key, item in book\_3.items() if item}

self.export\_data(book\_1, "Все", book\_2, book\_3)

elif self.foe\_combo\_box\_md.currentText() == "Очная":

data = self.get\_disabled(self.loe\_combo\_box\_md.currentText(), "Очная",

self.direction\_combo\_box\_md.currentText(),

self.year\_combo\_box\_md.currentText(), self.month\_combo\_box\_md.currentText(),

self.course\_combo\_box\_md.currentText())

for k, chapter in enumerate(book\_1):

for recording in data:

book\_1[chapter].append(recording[k])

book\_1 = {key: item for key, item in book\_1.items() if item}

self.export\_data(book\_1, "Очная")

elif self.foe\_combo\_box\_md.currentText() == "Очно-заочная":

data = self.get\_disabled(self.loe\_combo\_box\_md.currentText(), "Очно-заочная",

self.direction\_combo\_box\_md.currentText(),

self.year\_combo\_box\_md.currentText(), self.month\_combo\_box\_md.currentText(),

self.course\_combo\_box\_md.currentText())

for k, chapter in enumerate(book\_1):

for recording in data:

book\_1[chapter].append(recording[k])

book\_1 = {key: item for key, item in book\_1.items() if item}

self.export\_data(book\_1, "Очно-заочная")

elif self.foe\_combo\_box\_md.currentText() == "Заочная":

data = self.get\_disabled(self.loe\_combo\_box\_md.currentText(), "Заочная",

self.direction\_combo\_box\_md.currentText(),

self.year\_combo\_box\_md.currentText(), self.month\_combo\_box\_md.currentText(),

self.course\_combo\_box\_md.currentText())

for k, chapter in enumerate(book\_1):

for recording in data:

book\_1[chapter].append(recording[k])

book\_1 = {key: item for key, item in book\_1.items() if item}

self.export\_data(book\_1, "Заочная")

elif self.trigger == "Целевеки":

book\_1 = {"Названия направления подготовки": [],

"Уровень обучения": [], "Форма обучения": [], "Дата записи": [], "Курс": [],

"На места в рамках квоты целевого приема (всего зачислено)": [],

"На места в рамках квоты целевого приема (всего отчислено)": [],

"На места в рамках квоты целевого приема (зачислено бюджет)": [],

"На места в рамках квоты целевого приема (зачислено платное)": [],

"На места в рамках квоты целевого приема (отчислено бюджет)": [],

"На места в рамках квоты целевого приема (отчислено платное)": []}

book\_2 = {"Названия направления подготовки": [],

"Уровень обучения": [], "Форма обучения": [], "Дата записи": [], "Курс": [],

"На места в рамках квоты целевого приема (всего зачислено)": [],

"На места в рамках квоты целевого приема (всего отчислено)": [],

"На места в рамках квоты целевого приема (зачислено бюджет)": [],

"На места в рамках квоты целевого приема (зачислено платное)": [],

"На места в рамках квоты целевого приема (отчислено бюджет)": [],

"На места в рамках квоты целевого приема (отчислено платное)": []}

book\_3 = {"Названия направления подготовки": [],

"Уровень обучения": [], "Форма обучения": [], "Дата записи": [], "Курс": [],

"На места в рамках квоты целевого приема (всего зачислено)": [],

"На места в рамках квоты целевого приема (всего отчислено)": [],

"На места в рамках квоты целевого приема (зачислено бюджет)": [],

"На места в рамках квоты целевого приема (зачислено платное)": [],

"На места в рамках квоты целевого приема (отчислено бюджет)": [],

"На места в рамках квоты целевого приема (отчислено платное)": []}

if self.foe\_combo\_box\_md.currentText() == "Все":

data = self.get\_target(self.loe\_combo\_box\_md.currentText(), "Очная",

self.direction\_combo\_box\_md.currentText(),

self.year\_combo\_box\_md.currentText(), self.month\_combo\_box\_md.currentText(),

self.course\_combo\_box\_md.currentText())

print(data)

for k, chapter in enumerate(book\_1):

for recording in data:

book\_1[chapter].append(recording[k])

book\_1 = {key: item for key, item in book\_1.items() if item}

data = self.get\_target(self.loe\_combo\_box\_md.currentText(), "Очно-заочная",

self.direction\_combo\_box\_md.currentText(),

self.year\_combo\_box\_md.currentText(), self.month\_combo\_box\_md.currentText(),

self.course\_combo\_box\_md.currentText())

for k, chapter in enumerate(book\_2):

for recording in data:

book\_2[chapter].append(recording[k])

book\_2 = {key: item for key, item in book\_2.items() if item}

data = self.get\_target(self.loe\_combo\_box\_md.currentText(), "Заочная",

self.direction\_combo\_box\_md.currentText(),

self.year\_combo\_box\_md.currentText(), self.month\_combo\_box\_md.currentText(),

self.course\_combo\_box\_md.currentText())

for k, chapter in enumerate(book\_3):

for recording in data:

book\_3[chapter].append(recording[k])

book\_3 = {key: item for key, item in book\_3.items() if item}

self.export\_data(book\_1, "Все", book\_2, book\_3)

elif self.foe\_combo\_box\_md.currentText() == "Очная":

data = self.get\_target(self.loe\_combo\_box\_md.currentText(), "Очная",

self.direction\_combo\_box\_md.currentText(),

self.year\_combo\_box\_md.currentText(), self.month\_combo\_box\_md.currentText(),

self.course\_combo\_box\_md.currentText())

for k, chapter in enumerate(book\_1):

for recording in data:

book\_1[chapter].append(recording[k])

book\_1 = {key: item for key, item in book\_1.items() if item}

self.export\_data(book\_1, "Очная")

elif self.foe\_combo\_box\_md.currentText() == "Очно-заочная":

data = self.get\_target(self.loe\_combo\_box\_md.currentText(), "Очно-заочная",

self.direction\_combo\_box\_md.currentText(),

self.year\_combo\_box\_md.currentText(), self.month\_combo\_box\_md.currentText(),

self.course\_combo\_box\_md.currentText())

for k, chapter in enumerate(book\_1):

for recording in data:

book\_1[chapter].append(recording[k])

book\_1 = {key: item for key, item in book\_1.items() if item}

self.export\_data(book\_1, "Очно-заочная")

elif self.foe\_combo\_box\_md.currentText() == "Заочная":

data = self.get\_target(self.loe\_combo\_box\_md.currentText(), "Заочная",

self.direction\_combo\_box\_md.currentText(),

self.year\_combo\_box\_md.currentText(), self.month\_combo\_box\_md.currentText(),

self.course\_combo\_box\_md.currentText())

for k, chapter in enumerate(book\_1):

for recording in data:

book\_1[chapter].append(recording[k])

book\_1 = {key: item for key, item in book\_1.items() if item}

self.export\_data(book\_1, "Заочная")

elif self.trigger == "Иностранцы":

book\_1 = {"Названия направления подготовки": [],

"Уровень обучения": [], "Форма обучения": [], "Дата записи": [], "Курс": [],

"Иностранные граждане (всего зачислено)": [],

"Иностранные граждане (всего отчислено)": [],

"Иностранные граждане (зачислено бюджет)": [],

"Иностранные граждане (зачислено платное)": [],

"Иностранные граждане (отчислено бюджет)": [],

"Иностранные граждане (отчислено платное)": []}

book\_2 = {"Названия направления подготовки": [],

"Уровень обучения": [], "Форма обучения": [], "Дата записи": [], "Курс": [],

"Иностранные граждане (всего зачислено)": [],

"Иностранные граждане (всего отчислено)": [],

"Иностранные граждане (зачислено бюджет)": [],

"Иностранные граждане (зачислено платное)": [],

"Иностранные граждане (отчислено бюджет)": [],

"Иностранные граждане (отчислено платное)": []}

book\_3 = {"Названия направления подготовки": [],

"Уровень обучения": [], "Форма обучения": [], "Дата записи": [], "Курс": [],

"Иностранные граждане (всего зачислено)": [],

"Иностранные граждане (всего отчислено)": [],

"Иностранные граждане (зачислено бюджет)": [],

"Иностранные граждане (зачислено платное)": [],

"Иностранные граждане (отчислено бюджет)": [],

"Иностранные граждане (отчислено платное)": []}

if self.foe\_combo\_box\_md.currentText() == "Все":

data = self.get\_foreigners(self.loe\_combo\_box\_md.currentText(), "Очная",

self.direction\_combo\_box\_md.currentText(),

self.year\_combo\_box\_md.currentText(), self.month\_combo\_box\_md.currentText(),

self.course\_combo\_box\_md.currentText())

print(data)

for k, chapter in enumerate(book\_1):

for recording in data:

book\_1[chapter].append(recording[k])

book\_1 = {key: item for key, item in book\_1.items() if item}

data = self.get\_foreigners(self.loe\_combo\_box\_md.currentText(), "Очно-заочная",

self.direction\_combo\_box\_md.currentText(),

self.year\_combo\_box\_md.currentText(), self.month\_combo\_box\_md.currentText(),

self.course\_combo\_box\_md.currentText())

for k, chapter in enumerate(book\_2):

for recording in data:

book\_2[chapter].append(recording[k])

book\_2 = {key: item for key, item in book\_2.items() if item}

data = self.get\_foreigners(self.loe\_combo\_box\_md.currentText(), "Заочная",

self.direction\_combo\_box\_md.currentText(),

self.year\_combo\_box\_md.currentText(), self.month\_combo\_box\_md.currentText(),

self.course\_combo\_box\_md.currentText())

for k, chapter in enumerate(book\_3):

for recording in data:

book\_3[chapter].append(recording[k])

book\_3 = {key: item for key, item in book\_3.items() if item}

self.export\_data(book\_1, "Все", book\_2, book\_3)

elif self.foe\_combo\_box\_md.currentText() == "Очная":

data = self.get\_foreigners(self.loe\_combo\_box\_md.currentText(), "Очная",

self.direction\_combo\_box\_md.currentText(),

self.year\_combo\_box\_md.currentText(), self.month\_combo\_box\_md.currentText(),

self.course\_combo\_box\_md.currentText())

for k, chapter in enumerate(book\_1):

for recording in data:

book\_1[chapter].append(recording[k])

book\_1 = {key: item for key, item in book\_1.items() if item}

self.export\_data(book\_1, "Очная")

elif self.foe\_combo\_box\_md.currentText() == "Очно-заочная":

data = self.get\_foreigners(self.loe\_combo\_box\_md.currentText(), "Очно-заочная",

self.direction\_combo\_box\_md.currentText(),

self.year\_combo\_box\_md.currentText(), self.month\_combo\_box\_md.currentText(),

self.course\_combo\_box\_md.currentText())

for k, chapter in enumerate(book\_1):

for recording in data:

book\_1[chapter].append(recording[k])

book\_1 = {key: item for key, item in book\_1.items() if item}

self.export\_data(book\_1, "Очно-заочная")

elif self.foe\_combo\_box\_md.currentText() == "Заочная":

data = self.get\_foreigners(self.loe\_combo\_box\_md.currentText(), "Заочная",

self.direction\_combo\_box\_md.currentText(),

self.year\_combo\_box\_md.currentText(), self.month\_combo\_box\_md.currentText(),

self.course\_combo\_box\_md.currentText())

for k, chapter in enumerate(book\_1):

for recording in data:

book\_1[chapter].append(recording[k])

book\_1 = {key: item for key, item in book\_1.items() if item}

self.export\_data(book\_1, "Заочная")

**Листинг 5. ui\_control.py**

import PyQt5.Qt

import dialog\_ui

import dialog\_simple\_ui

import main\_ui

import excel\_control

from PyQt5 import QtWidgets, QtGui, QtCore

class UiControl(main\_ui.UiWindow, excel\_control.ExcelControl):

def \_\_init\_\_(self, \_window):

"""Конструктор класса UiControl"""

super(UiControl, self).\_\_init\_\_(window=\_window)

# Создание меню бара и установка иконок

self.data\_menu = self.menubar.addMenu("&Данные")

self.report = self.menubar.addMenu("&Отчет")

self.settings = self.menubar.addMenu("&Настройки")

icon\_file\_bar = QtGui.QIcon()

icon\_file\_bar.addPixmap(QtGui.QPixmap("Изображения/free-icon-import-724839.png"), QtGui.QIcon.Normal,

QtGui.QIcon.Off)

icon\_all\_export = QtGui.QIcon()

icon\_all\_export.addPixmap(QtGui.QPixmap("Изображения/free-icon-export-724836.png"), QtGui.QIcon.Normal,

QtGui.QIcon.Off)

icon\_disabled\_export = QtGui.QIcon()

icon\_disabled\_export.addPixmap(QtGui.QPixmap("Изображения/free-icon-disabled-person-2707234.png"),

QtGui.QIcon.Normal, QtGui.QIcon.Off)

icon\_target\_export = QtGui.QIcon()

icon\_target\_export.addPixmap(QtGui.QPixmap("Изображения/free-icon-target-audience-2282175.png"),

QtGui.QIcon.Normal, QtGui.QIcon.Off)

icon\_foreigners\_export = QtGui.QIcon()

icon\_foreigners\_export.addPixmap(QtGui.QPixmap("Изображения/free-icon-tourist-925642.png"), QtGui.QIcon.Normal,

QtGui.QIcon.Off)

icon\_user\_export = QtGui.QIcon()

icon\_user\_export.addPixmap(QtGui.QPixmap("Изображения/free-icon-check-4676065.png"), QtGui.QIcon.Normal,

QtGui.QIcon.Off)

icon\_settings = QtGui.QIcon()

icon\_settings.addPixmap(QtGui.QPixmap("Изображения/free-icon-font-size-1634828.png"), QtGui.QIcon.Normal,

QtGui.QIcon.Off)

self.open\_file\_action = QtWidgets.QAction(icon\_file\_bar, "&Загрузить данные")

self.customizable\_report\_action = QtWidgets.QAction(icon\_user\_export, "&Настраиваемый отчет")

self.settings\_action = QtWidgets.QAction(icon\_settings, "&Шрифт")

self.report\_all\_action = QtWidgets.QAction(icon\_all\_export, "&Полный отчет ")

self.report\_disabled\_action = QtWidgets.QAction(icon\_disabled\_export, "&Лица с ОВЗ, инвалиды, дети-инвалиды")

self.report\_target\_action = QtWidgets.QAction(icon\_target\_export, "&Места в рамках квоты целевого приема")

self.report\_foreigners\_action = QtWidgets.QAction(icon\_foreigners\_export, "&Иностранные граждане")

self.set\_menu\_and\_tool\_bars()

# Форматирование таблиц

self.stud\_table\_header = ["Код направления подготовки", "Названия направления подготовки", "Уровень обучения",

"Форма обучения", "Дата записи", "Курс", "Всего зачислено",

"Из них лица с ОВЗ, инвалиды, дети-инвалиды (всего зачислено)",

"Из них на места в рамках квоты целевого приема (всего зачислено)",

"Из них иностранные граждане (всего зачислено)", "Всего отчислено",

"Из них лица с ОВЗ, инвалиды, дети-инвалиды (всего отчислено)",

"Из них на места в рамках квоты целевого приема (всего отчислено)",

"Из них иностранные граждане (всего отчислено)",

"Зачислено за счет бюджетных ассигнований бюджета субъекта РФ",

"Из них лица с ОВЗ, инвалиды, дети-инвалиды (зачислено бюджет)",

"Из них на места в рамках квоты целевого приема (зачислено бюджет)",

"Из них иностранные граждане (зачислено бюджет)",

"Зачислено по договорам об оказании платных образовательных услуг",

"Из них лица с ОВЗ, инвалиды, дети-инвалиды (зачислено платное)",

"Из них на места в рамках квоты целевого приема (зачислено платное)",

"Из них иностранные граждане (зачислено платное)",

"Отчислено за счет бюджетных ассигнований бюджета субъекта РФ",

"Из них лица с ОВЗ, инвалиды, дети-инвалиды (отчислено бюджет)",

"Из них на места в рамках квоты целевого приема (отчислено бюджет)",

"Из них иностранные граждане (отчислено бюджет)",

"Отчислено по договорам об оказании платных образовательных услуг",

"Из них лица с ОВЗ, инвалиды, дети-инвалиды (отчислено платное)",

"Из них на места в рамках квоты целевого приема (отчислено платное)",

"Из них иностранные граждане (отчислено платное)"]

self.last\_table\_header = ["Код направления подготовки", "Названия направления подготовки", "Уровень обучения",

"Форма обучения", "Дата записи", "Курс", "Всего зачислено",

"Из них лица с ОВЗ, инвалиды, дети-инвалиды (всего зачислено)",

"Из них на места в рамках квоты целевого приема (всего зачислено)",

"Из них иностранные граждане (всего зачислено)", "Всего отчислено",

"Из них лица с ОВЗ, инвалиды, дети-инвалиды (всего отчислено)",

"Из них на места в рамках квоты целевого приема (всего отчислено)",

"Из них иностранные граждане (всего отчислено)"]

# Создание общих триггеров

self.tabWidget.currentChanged.connect(self.load\_table)

self.back\_button.clicked.connect(self.open\_page\_1)

self.open\_file\_action.triggered.connect(self.update\_all)

self.report\_all\_action.triggered.connect(lambda: self.open\_dialog\_simple("Все"))

self.report\_disabled\_action.triggered.connect(lambda: self.open\_dialog\_simple("Инвалиды"))

self.report\_target\_action.triggered.connect(lambda: self.open\_dialog\_simple("Целевеки"))

self.report\_foreigners\_action.triggered.connect(lambda: self.open\_dialog\_simple("Иностранцы"))

self.customizable\_report\_action.triggered.connect(self.open\_dialog)

self.settings\_action.triggered.connect(self.user\_settings)

self.left\_click = QtCore.pyqtSignal()

self.right\_click = QtCore.pyqtSignal()

# Триггеры для бакалавриата

self.foe\_combo\_box\_b.currentIndexChanged.connect(self.load\_table)

self.direction\_combo\_box\_b.currentIndexChanged.connect(self.load\_table)

self.year\_combo\_box\_b.currentIndexChanged.connect(self.load\_table)

self.month\_combo\_box\_b.currentIndexChanged.connect(self.load\_table)

self.table\_widget\_b.doubleClicked.connect(self.open\_page\_2)

# Триггеры для магистратуры

self.foe\_combo\_box\_m.currentIndexChanged.connect(self.load\_table)

self.direction\_combo\_box\_m.currentIndexChanged.connect(self.load\_table)

self.year\_combo\_box\_m.currentIndexChanged.connect(self.load\_table)

self.month\_combo\_box\_m.currentIndexChanged.connect(self.load\_table)

self.table\_widget\_m.doubleClicked.connect(self.open\_page\_2)

# Триггеры для специалитета

self.foe\_combo\_box\_s.currentIndexChanged.connect(self.load\_table)

self.direction\_combo\_box\_s.currentIndexChanged.connect(self.load\_table)

self.year\_combo\_box\_s.currentIndexChanged.connect(self.load\_table)

self.month\_combo\_box\_s.currentIndexChanged.connect(self.load\_table)

self.table\_widget\_s.doubleClicked.connect(self.open\_page\_2)

# Триггеры для ККМТ

self.foe\_combo\_box\_CSET.currentIndexChanged.connect(self.load\_table)

self.direction\_combo\_box\_CSET.currentIndexChanged.connect(self.load\_table)

self.year\_combo\_box\_CSET.currentIndexChanged.connect(self.load\_table)

self.month\_combo\_box\_CSET.currentIndexChanged.connect(self.load\_table)

self.table\_widget\_CSET.doubleClicked.connect(self.open\_page\_2)

# Триггеры для ТТД

self.foe\_combo\_box\_TSTD.currentIndexChanged.connect(self.load\_table)

self.direction\_combo\_box\_TSTD.currentIndexChanged.connect(self.load\_table)

self.year\_combo\_box\_TSTD.currentIndexChanged.connect(self.load\_table)

self.month\_combo\_box\_TSTD.currentIndexChanged.connect(self.load\_table)

self.table\_widget\_TSTD.doubleClicked.connect(self.open\_page\_2)

# Предварительные настройки

self.load\_table()

self.direction\_combo\_box\_b.addItems(self.get\_directions("Бакалавриат"))

self.direction\_combo\_box\_m.addItems(self.get\_directions("Магистратура"))

self.direction\_combo\_box\_s.addItems(self.get\_directions("Специалитет"))

self.direction\_combo\_box\_CSET.addItems(self.get\_directions("ККМТ"))

self.direction\_combo\_box\_TSTD.addItems(self.get\_directions("ТТД"))

self.year\_combo\_box\_b.addItems(self.get\_years("Бакалавриат"))

self.year\_combo\_box\_m.addItems(self.get\_years("Магистратура"))

self.year\_combo\_box\_s.addItems(self.get\_years("Специалитет"))

self.year\_combo\_box\_CSET.addItems(self.get\_years("ККМТ"))

self.year\_combo\_box\_TSTD.addItems(self.get\_years("ТТД"))

def set\_menu\_and\_tool\_bars(self):

"""Добавление действий в меню и панель инструментов, а также установка триггеров для них"""

self.data\_menu.addAction(self.open\_file\_action)

report = self.report.addMenu("&Встроенный отчет")

self.report.addAction(self.customizable\_report\_action)

report.addAction(self.report\_all\_action)

report.addAction(self.report\_disabled\_action)

report.addAction(self.report\_target\_action)

report.addAction(self.report\_foreigners\_action)

self.settings.addAction(self.settings\_action)

self.toolBar.addAction(self.open\_file\_action)

self.toolBar.addSeparator()

self.toolBar.addAction(self.report\_all\_action)

self.toolBar.addAction(self.report\_disabled\_action)

self.toolBar.addAction(self.report\_target\_action)

self.toolBar.addAction(self.report\_foreigners\_action)

self.toolBar.addAction(self.customizable\_report\_action)

self.toolBar.addSeparator()

self.toolBar.addAction(self.settings\_action)

self.toolBar.setMovable(False)

def load\_table(self):

"""Загрузка данных в таблицу"""

self.table\_widget\_last\_month.horizontalHeader().setSectionResizeMode(PyQt5.Qt.QHeaderView.ResizeToContents)

self.table\_widget\_last\_month.horizontalHeader().setMinimumSectionSize(0)

self.table\_widget\_last\_year.horizontalHeader().setSectionResizeMode(PyQt5.Qt.QHeaderView.ResizeToContents)

self.table\_widget\_last\_year.horizontalHeader().setMinimumSectionSize(0)

if self.tabWidget.currentIndex() == 0:

data = self.get\_data("Бакалавриат", self.foe\_combo\_box\_b.currentText(),

self.direction\_combo\_box\_b.currentText(), self.year\_combo\_box\_b.currentText(),

str(self.month\_combo\_box\_b.currentIndex()))

self.table\_widget\_b.setSortingEnabled(False)

self.table\_widget\_b.setRowCount(0)

self.table\_widget\_b.setColumnCount(len(self.stud\_table\_header))

self.table\_widget\_b.setHorizontalHeaderLabels(self.stud\_table\_header)

self.table\_widget\_b.horizontalHeader().setSectionResizeMode(PyQt5.Qt.QHeaderView.ResizeToContents)

self.table\_widget\_b.horizontalHeader().setMinimumSectionSize(0)

if data:

self.table\_widget\_b.setRowCount(len(data))

for row in range(len(data)):

for col in range(len(data[0])):

item = QtWidgets.QTableWidgetItem(str(data[row][col]))

item.setTextAlignment(QtCore.Qt.AlignCenter)

self.table\_widget\_b.setItem(row, col, item)

self.table\_widget\_b.setSortingEnabled(True)

elif self.tabWidget.currentIndex() == 1:

data = self.get\_data("Магистратура", self.foe\_combo\_box\_m.currentText(),

self.direction\_combo\_box\_m.currentText(), self.year\_combo\_box\_m.currentText(),

str(self.month\_combo\_box\_m.currentIndex()))

self.table\_widget\_m.setSortingEnabled(False)

self.table\_widget\_m.setRowCount(0)

self.table\_widget\_m.setColumnCount(len(self.stud\_table\_header))

self.table\_widget\_m.setHorizontalHeaderLabels(self.stud\_table\_header)

self.table\_widget\_m.horizontalHeader().setSectionResizeMode(PyQt5.Qt.QHeaderView.ResizeToContents)

self.table\_widget\_m.horizontalHeader().setMinimumSectionSize(0)

if data:

self.table\_widget\_m.setRowCount(len(data))

for row in range(len(data)):

for col in range(len(data[0])):

item = QtWidgets.QTableWidgetItem(str(data[row][col]))

item.setTextAlignment(QtCore.Qt.AlignCenter)

self.table\_widget\_m.setItem(row, col, item)

self.table\_widget\_m.setSortingEnabled(True)

elif self.tabWidget.currentIndex() == 2:

data = self.get\_data("Специалитет", self.foe\_combo\_box\_s.currentText(),

self.direction\_combo\_box\_s.currentText(), self.year\_combo\_box\_s.currentText(),

str(self.month\_combo\_box\_s.currentIndex()))

self.table\_widget\_s.setSortingEnabled(False)

self.table\_widget\_s.setRowCount(0)

self.table\_widget\_s.setColumnCount(len(self.stud\_table\_header))

self.table\_widget\_s.setHorizontalHeaderLabels(self.stud\_table\_header)

self.table\_widget\_s.horizontalHeader().setSectionResizeMode(PyQt5.Qt.QHeaderView.ResizeToContents)

self.table\_widget\_s.horizontalHeader().setMinimumSectionSize(0)

if data:

self.table\_widget\_s.setRowCount(len(data))

for row in range(len(data)):

for col in range(len(data[0])):

item = QtWidgets.QTableWidgetItem(str(data[row][col]))

item.setTextAlignment(QtCore.Qt.AlignCenter)

self.table\_widget\_s.setItem(row, col, item)

self.table\_widget\_s.setSortingEnabled(True)

elif self.tabWidget.currentIndex() == 3:

data = self.get\_data("ККМТ", self.foe\_combo\_box\_CSET.currentText(),

self.direction\_combo\_box\_CSET.currentText(), self.year\_combo\_box\_CSET.currentText(),

str(self.month\_combo\_box\_CSET.currentIndex()))

self.table\_widget\_CSET.setSortingEnabled(False)

self.table\_widget\_CSET.setRowCount(0)

self.table\_widget\_CSET.setColumnCount(len(self.stud\_table\_header))

self.table\_widget\_CSET.setHorizontalHeaderLabels(self.stud\_table\_header)

self.table\_widget\_CSET.horizontalHeader().setSectionResizeMode(PyQt5.Qt.QHeaderView.ResizeToContents)

self.table\_widget\_CSET.horizontalHeader().setMinimumSectionSize(0)

if data:

self.table\_widget\_CSET.setRowCount(len(data))

for row in range(len(data)):

for col in range(len(data[0])):

item = QtWidgets.QTableWidgetItem(str(data[row][col]))

item.setTextAlignment(QtCore.Qt.AlignCenter)

self.table\_widget\_CSET.setItem(row, col, item)

self.table\_widget\_CSET.setSortingEnabled(True)

elif self.tabWidget.currentIndex() == 4:

data = self.get\_data("ТТД", self.foe\_combo\_box\_TSTD.currentText(),

self.direction\_combo\_box\_TSTD.currentText(), self.year\_combo\_box\_TSTD.currentText(),

str(self.month\_combo\_box\_TSTD.currentIndex()))

self.table\_widget\_TSTD.setSortingEnabled(False)

self.table\_widget\_TSTD.setRowCount(0)

self.table\_widget\_TSTD.setColumnCount(len(self.stud\_table\_header))

self.table\_widget\_TSTD.setHorizontalHeaderLabels(self.stud\_table\_header)

self.table\_widget\_TSTD.horizontalHeader().setSectionResizeMode(PyQt5.Qt.QHeaderView.ResizeToContents)

self.table\_widget\_TSTD.horizontalHeader().setMinimumSectionSize(0)

if data:

self.table\_widget\_TSTD.setRowCount(len(data))

for row in range(len(data)):

for col in range(len(data[0])):

item = QtWidgets.QTableWidgetItem(str(data[row][col]))

item.setTextAlignment(QtCore.Qt.AlignCenter)

self.table\_widget\_TSTD.setItem(row, col, item)

self.table\_widget\_TSTD.setSortingEnabled(True)

def data\_analysis(self):

"""Анализирует информацию о движении контингента"""

if self.tabWidget.currentIndex() == 0:

for idx in self.table\_widget\_b.selectionModel().selectedIndexes():

row\_index = idx.row()

# Анализ за прошлый месяц

data\_old = self.get\_data\_old\_month(row\_index)

data\_new = self.get\_data\_new(row\_index)

self.table\_widget\_last\_month.setSortingEnabled(False)

self.table\_widget\_last\_month.setRowCount(0)

self.table\_widget\_last\_month.setColumnCount(len(self.last\_table\_header))

self.table\_widget\_last\_month.setHorizontalHeaderLabels(self.last\_table\_header)

self.table\_widget\_last\_month.horizontalHeader().setSectionResizeMode(

PyQt5.Qt.QHeaderView.ResizeToContents)

self.table\_widget\_last\_month.horizontalHeader().setMinimumSectionSize(0)

self.table\_widget\_last\_month.setRowCount(len(data\_new))

self.set\_style\_last\_month(data\_new, data\_old)

# Анализ за прошлый год

data\_old = self.get\_data\_old\_year(row\_index)

self.table\_widget\_last\_year.setSortingEnabled(False)

self.table\_widget\_last\_year.setRowCount(0)

self.table\_widget\_last\_year.setColumnCount(len(self.last\_table\_header))

self.table\_widget\_last\_year.setHorizontalHeaderLabels(self.last\_table\_header)

self.table\_widget\_last\_year.horizontalHeader().setSectionResizeMode(

PyQt5.Qt.QHeaderView.ResizeToContents)

self.table\_widget\_last\_year.horizontalHeader().setMinimumSectionSize(0)

self.table\_widget\_last\_year.setRowCount(len(data\_new))

self.set\_style\_last\_year(data\_new, data\_old)

elif self.tabWidget.currentIndex() == 1:

for idx in self.table\_widget\_m.selectionModel().selectedIndexes():

row\_index = idx.row()

# Анализ за прошлый месяц

data\_old = self.get\_data\_old\_month(row\_index)

data\_new = self.get\_data\_new(row\_index)

self.table\_widget\_last\_month.setSortingEnabled(False)

self.table\_widget\_last\_month.setRowCount(0)

self.table\_widget\_last\_month.setColumnCount(len(self.last\_table\_header))

self.table\_widget\_last\_month.setHorizontalHeaderLabels(self.last\_table\_header)

self.table\_widget\_last\_month.horizontalHeader().setSectionResizeMode(

PyQt5.Qt.QHeaderView.ResizeToContents)

self.table\_widget\_last\_month.horizontalHeader().setMinimumSectionSize(0)

self.table\_widget\_last\_month.setRowCount(len(data\_new))

self.set\_style\_last\_month(data\_new, data\_old)

# Анализ за прошлый год

data\_old = self.get\_data\_old\_year(row\_index)

self.table\_widget\_last\_year.setSortingEnabled(False)

self.table\_widget\_last\_year.setRowCount(0)

self.table\_widget\_last\_year.setColumnCount(len(self.last\_table\_header))

self.table\_widget\_last\_year.setHorizontalHeaderLabels(self.last\_table\_header)

self.table\_widget\_last\_year.horizontalHeader().setSectionResizeMode(

PyQt5.Qt.QHeaderView.ResizeToContents)

self.table\_widget\_last\_year.horizontalHeader().setMinimumSectionSize(0)

self.table\_widget\_last\_year.setRowCount(len(data\_new))

self.set\_style\_last\_year(data\_new, data\_old)

elif self.tabWidget.currentIndex() == 2:

for idx in self.table\_widget\_s.selectionModel().selectedIndexes():

row\_index = idx.row()

# Анализ за прошлый месяц

data\_old = self.get\_data\_old\_month(row\_index)

data\_new = self.get\_data\_new(row\_index)

self.table\_widget\_last\_month.setSortingEnabled(False)

self.table\_widget\_last\_month.setRowCount(0)

self.table\_widget\_last\_month.setColumnCount(len(self.last\_table\_header))

self.table\_widget\_last\_month.setHorizontalHeaderLabels(self.last\_table\_header)

self.table\_widget\_last\_month.horizontalHeader().setSectionResizeMode(

PyQt5.Qt.QHeaderView.ResizeToContents)

self.table\_widget\_last\_month.horizontalHeader().setMinimumSectionSize(0)

self.table\_widget\_last\_month.setRowCount(len(data\_new))

self.set\_style\_last\_month(data\_new, data\_old)

# Анализ за прошлый год

data\_old = self.get\_data\_old\_year(row\_index)

self.table\_widget\_last\_year.setSortingEnabled(False)

self.table\_widget\_last\_year.setRowCount(0)

self.table\_widget\_last\_year.setColumnCount(len(self.last\_table\_header))

self.table\_widget\_last\_year.setHorizontalHeaderLabels(self.last\_table\_header)

self.table\_widget\_last\_year.horizontalHeader().setSectionResizeMode(

PyQt5.Qt.QHeaderView.ResizeToContents)

self.table\_widget\_last\_year.horizontalHeader().setMinimumSectionSize(0)

self.table\_widget\_last\_year.setRowCount(len(data\_new))

self.set\_style\_last\_year(data\_new, data\_old)

elif self.tabWidget.currentIndex() == 3:

for idx in self.table\_widget\_CSET.selectionModel().selectedIndexes():

row\_index = idx.row()

# Анализ за прошлый месяц

data\_old = self.get\_data\_old\_month(row\_index)

data\_new = self.get\_data\_new(row\_index)

self.table\_widget\_last\_month.setSortingEnabled(False)

self.table\_widget\_last\_month.setRowCount(0)

self.table\_widget\_last\_month.setColumnCount(len(self.last\_table\_header))

self.table\_widget\_last\_month.setHorizontalHeaderLabels(self.last\_table\_header)

self.table\_widget\_last\_month.horizontalHeader().setSectionResizeMode(

PyQt5.Qt.QHeaderView.ResizeToContents)

self.table\_widget\_last\_month.horizontalHeader().setMinimumSectionSize(0)

self.table\_widget\_last\_month.setRowCount(len(data\_new))

self.set\_style\_last\_month(data\_new, data\_old)

# Анализ за прошлый год

data\_old = self.get\_data\_old\_year(row\_index)

self.table\_widget\_last\_year.setSortingEnabled(False)

self.table\_widget\_last\_year.setRowCount(0)

self.table\_widget\_last\_year.setColumnCount(len(self.last\_table\_header))

self.table\_widget\_last\_year.setHorizontalHeaderLabels(self.last\_table\_header)

self.table\_widget\_last\_year.horizontalHeader().setSectionResizeMode(

PyQt5.Qt.QHeaderView.ResizeToContents)

self.table\_widget\_last\_year.horizontalHeader().setMinimumSectionSize(0)

self.table\_widget\_last\_year.setRowCount(len(data\_new))

self.set\_style\_last\_year(data\_new, data\_old)

else:

for idx in self.table\_widget\_TSTD.selectionModel().selectedIndexes():

row\_index = idx.row()

# Анализ за прошлый месяц

data\_old = self.get\_data\_old\_month(row\_index)

data\_new = self.get\_data\_new(row\_index)

self.table\_widget\_last\_month.setSortingEnabled(False)

self.table\_widget\_last\_month.setRowCount(0)

self.table\_widget\_last\_month.setColumnCount(len(self.last\_table\_header))

self.table\_widget\_last\_month.setHorizontalHeaderLabels(self.last\_table\_header)

self.table\_widget\_last\_month.horizontalHeader().setSectionResizeMode(

PyQt5.Qt.QHeaderView.ResizeToContents)

self.table\_widget\_last\_month.horizontalHeader().setMinimumSectionSize(0)

self.table\_widget\_last\_month.setRowCount(len(data\_new))

self.set\_style\_last\_month(data\_new, data\_old)

# Анализ за прошлый год

data\_old = self.get\_data\_old\_year(row\_index)

self.table\_widget\_last\_year.setSortingEnabled(False)

self.table\_widget\_last\_year.setRowCount(0)

self.table\_widget\_last\_year.setColumnCount(len(self.last\_table\_header))

self.table\_widget\_last\_year.setHorizontalHeaderLabels(self.last\_table\_header)

self.table\_widget\_last\_year.horizontalHeader().setSectionResizeMode(

PyQt5.Qt.QHeaderView.ResizeToContents)

self.table\_widget\_last\_year.horizontalHeader().setMinimumSectionSize(0)

self.table\_widget\_last\_year.setRowCount(len(data\_new))

self.set\_style\_last\_year(data\_new, data\_old)

def set\_style\_last\_month(self, data\_new, data\_old):

"""Установка стиля для таблицы с анализом за предыдущий месяц"""

if data\_old:

for row in range(len(data\_new)):

for col in range(0, 6):

item = QtWidgets.QTableWidgetItem(str(data\_new[row][col]))

item.setTextAlignment(QtCore.Qt.AlignCenter)

self.table\_widget\_last\_month.setItem(row, col, item)

for col in range(6, 10):

if data\_new[row][col] - data\_old[row][col] < 0:

item = QtWidgets.QTableWidgetItem(str(data\_new[row][col] - data\_old[row][col]))

item.setTextAlignment(QtCore.Qt.AlignCenter)

item.setForeground(QtGui.QColor(255, 0, 0))

elif data\_new[row][col] - data\_old[row][col] > 0:

item = QtWidgets.QTableWidgetItem("+" + str(data\_new[row][col] - data\_old[row][col]))

item.setTextAlignment(QtCore.Qt.AlignCenter)

item.setForeground(QtGui.QColor(0, 128, 0))

else:

item = QtWidgets.QTableWidgetItem(str(data\_new[row][col] - data\_old[row][col]))

item.setTextAlignment(QtCore.Qt.AlignCenter)

self.table\_widget\_last\_month.setItem(row, col, item)

for col in range(10, 14):

if data\_new[row][col] - data\_old[row][col] > 0:

item = QtWidgets.QTableWidgetItem("+" + str(data\_new[row][col] - data\_old[row][col]))

item.setTextAlignment(QtCore.Qt.AlignCenter)

item.setForeground(QtGui.QColor(255, 0, 0))

elif data\_new[row][col] - data\_old[row][col] < 0:

item = QtWidgets.QTableWidgetItem(str(data\_new[row][col] - data\_old[row][col]))

item.setTextAlignment(QtCore.Qt.AlignCenter)

item.setForeground(QtGui.QColor(0, 128, 0))

else:

item = QtWidgets.QTableWidgetItem(str(data\_new[row][col] - data\_old[row][col]))

item.setTextAlignment(QtCore.Qt.AlignCenter)

self.table\_widget\_last\_month.setItem(row, col, item)

else:

for row in range(len(data\_new)):

for col in range(0, 6):

item = QtWidgets.QTableWidgetItem(str(data\_new[row][col]))

item.setTextAlignment(QtCore.Qt.AlignCenter)

self.table\_widget\_last\_month.setItem(row, col, item)

for col in range(6, 10):

if data\_new[row][col] - 0 < 0:

item = QtWidgets.QTableWidgetItem("-" + str(data\_new[row][col] - 0))

item.setTextAlignment(QtCore.Qt.AlignCenter)

item.setForeground(QtGui.QColor(255, 0, 0))

elif data\_new[row][col] - 0 > 0:

item = QtWidgets.QTableWidgetItem("+" + str(data\_new[row][col] - 0))

item.setTextAlignment(QtCore.Qt.AlignCenter)

item.setForeground(QtGui.QColor(0, 128, 0))

else:

item = QtWidgets.QTableWidgetItem(str(data\_new[row][col] - 0))

item.setTextAlignment(QtCore.Qt.AlignCenter)

self.table\_widget\_last\_month.setItem(row, col, item)

for col in range(10, 14):

if data\_new[row][col] - 0 > 0:

item = QtWidgets.QTableWidgetItem("+" + str(data\_new[row][col] - 0))

item.setTextAlignment(QtCore.Qt.AlignCenter)

item.setForeground(QtGui.QColor(255, 0, 0))

elif data\_new[row][col] - 0 < 0:

item = QtWidgets.QTableWidgetItem("-" + str(data\_new[row][col] - 0))

item.setTextAlignment(QtCore.Qt.AlignCenter)

item.setForeground(QtGui.QColor(0, 128, 0))

else:

item = QtWidgets.QTableWidgetItem(str(data\_new[row][col] - 0))

item.setTextAlignment(QtCore.Qt.AlignCenter)

self.table\_widget\_last\_month.setItem(row, col, item)

def set\_style\_last\_year(self, data\_new, data\_old):

"""Установка стиля для таблицы с анализом за предыдущий год"""

if data\_old:

for row in range(len(data\_new)):

for col in range(0, 6):

item = QtWidgets.QTableWidgetItem(str(data\_new[row][col]))

item.setTextAlignment(QtCore.Qt.AlignCenter)

self.table\_widget\_last\_year.setItem(row, col, item)

for col in range(6, 10):

if data\_new[row][col] - data\_old[row][col] < 0:

item = QtWidgets.QTableWidgetItem(str(data\_new[row][col] - data\_old[row][col]))

item.setTextAlignment(QtCore.Qt.AlignCenter)

item.setForeground(QtGui.QColor(255, 0, 0))

elif data\_new[row][col] - data\_old[row][col] > 0:

item = QtWidgets.QTableWidgetItem("+" + str(data\_new[row][col] - data\_old[row][col]))

item.setTextAlignment(QtCore.Qt.AlignCenter)

item.setForeground(QtGui.QColor(0, 128, 0))

else:

item = QtWidgets.QTableWidgetItem(str(data\_new[row][col] - data\_old[row][col]))

item.setTextAlignment(QtCore.Qt.AlignCenter)

self.table\_widget\_last\_year.setItem(row, col, item)

for col in range(10, 14):

if data\_new[row][col] - data\_old[row][col] > 0:

item = QtWidgets.QTableWidgetItem("+" + str(data\_new[row][col] - data\_old[row][col]))

item.setTextAlignment(QtCore.Qt.AlignCenter)

item.setForeground(QtGui.QColor(255, 0, 0))

elif data\_new[row][col] - data\_old[row][col] < 0:

item = QtWidgets.QTableWidgetItem(str(data\_new[row][col] - data\_old[row][col]))

item.setTextAlignment(QtCore.Qt.AlignCenter)

item.setForeground(QtGui.QColor(0, 128, 0))

else:

item = QtWidgets.QTableWidgetItem(str(data\_new[row][col] - data\_old[row][col]))

item.setTextAlignment(QtCore.Qt.AlignCenter)

self.table\_widget\_last\_year.setItem(row, col, item)

else:

for row in range(len(data\_new)):

for col in range(0, 6):

item = QtWidgets.QTableWidgetItem(str(data\_new[row][col]))

item.setTextAlignment(QtCore.Qt.AlignCenter)

self.table\_widget\_last\_year.setItem(row, col, item)

for col in range(6, 10):

if data\_new[row][col] - 0 < 0:

item = QtWidgets.QTableWidgetItem("-" + str(data\_new[row][col] - 0))

item.setTextAlignment(QtCore.Qt.AlignCenter)

item.setForeground(QtGui.QColor(255, 0, 0))

elif data\_new[row][col] - 0 > 0:

item = QtWidgets.QTableWidgetItem("+" + str(data\_new[row][col] - 0))

item.setTextAlignment(QtCore.Qt.AlignCenter)

item.setForeground(QtGui.QColor(0, 128, 0))

else:

item = QtWidgets.QTableWidgetItem(str(data\_new[row][col] - 0))

item.setTextAlignment(QtCore.Qt.AlignCenter)

self.table\_widget\_last\_year.setItem(row, col, item)

for col in range(10, 14):

if data\_new[row][col] - 0 > 0:

item = QtWidgets.QTableWidgetItem("+" + str(data\_new[row][col] - 0))

item.setTextAlignment(QtCore.Qt.AlignCenter)

item.setForeground(QtGui.QColor(255, 0, 0))

elif data\_new[row][col] - 0 < 0:

item = QtWidgets.QTableWidgetItem("-" + str(data\_new[row][col] - 0))

item.setTextAlignment(QtCore.Qt.AlignCenter)

item.setForeground(QtGui.QColor(0, 128, 0))

else:

item = QtWidgets.QTableWidgetItem(str(data\_new[row][col] - 0))

item.setTextAlignment(QtCore.Qt.AlignCenter)

self.table\_widget\_last\_year.setItem(row, col, item)

def get\_data\_old\_month(self, row\_index):

"""Возвращает данные за предыдущий месяц"""

if self.tabWidget.currentIndex() == 0:

data\_old = self.get\_data(self.table\_widget\_b.item(row\_index, 2).text(),

self.table\_widget\_b.item(row\_index, 3).text(),

self.table\_widget\_b.item(row\_index, 1).text(),

self.table\_widget\_b.item(row\_index, 4).text().split('-')[0],

str(int(self.table\_widget\_b.item(row\_index, 4).text().

split('-')[1]) - 1))

elif self.tabWidget.currentIndex() == 1:

data\_old = self.get\_data(self.table\_widget\_m.item(row\_index, 2).text(),

self.table\_widget\_m.item(row\_index, 3).text(),

self.table\_widget\_m.item(row\_index, 1).text(),

self.table\_widget\_m.item(row\_index, 4).text().split('-')[0],

str(int(self.table\_widget\_m.item(row\_index, 4).text().

split('-')[1]) - 1))

elif self.tabWidget.currentIndex() == 2:

data\_old = self.get\_data(self.table\_widget\_s.item(row\_index, 2).text(),

self.table\_widget\_s.item(row\_index, 3).text(),

self.table\_widget\_s.item(row\_index, 1).text(),

self.table\_widget\_s.item(row\_index, 4).text().split('-')[0],

str(int(self.table\_widget\_s.item(row\_index, 4).text().

split('-')[1]) - 1))

elif self.tabWidget.currentIndex() == 3:

data\_old = self.get\_data(self.table\_widget\_CSET.item(row\_index, 2).text(),

self.table\_widget\_CSET.item(row\_index, 3).text(),

self.table\_widget\_CSET.item(row\_index, 1).text(),

self.table\_widget\_CSET.item(row\_index, 4).text().split('-')[0],

str(int(self.table\_widget\_CSET.item(row\_index, 4).text().

split('-')[1]) - 1))

else:

data\_old = self.get\_data(self.table\_widget\_TSTD.item(row\_index, 2).text(),

self.table\_widget\_TSTD.item(row\_index, 3).text(),

self.table\_widget\_TSTD.item(row\_index, 1).text(),

self.table\_widget\_TSTD.item(row\_index, 4).text().split('-')[0],

str(int(self.table\_widget\_TSTD.item(row\_index, 4).text().

split('-')[1]) - 1))

return data\_old

def get\_data\_old\_year(self, row\_index):

"""Возвращает данные за прошлый год выбранного месяца"""

if self.tabWidget.currentIndex() == 0:

data\_old = self.get\_data(self.table\_widget\_b.item(row\_index, 2).text(),

self.table\_widget\_b.item(row\_index, 3).text(),

self.table\_widget\_b.item(row\_index, 1).text(),

str(int(self.table\_widget\_b.item(row\_index, 4).text().

split('-')[0]) - 1),

self.table\_widget\_b.item(row\_index, 4).text().split('-')[1])

elif self.tabWidget.currentIndex() == 1:

data\_old = self.get\_data(self.table\_widget\_m.item(row\_index, 2).text(),

self.table\_widget\_m.item(row\_index, 3).text(),

self.table\_widget\_m.item(row\_index, 1).text(),

str(int(self.table\_widget\_m.item(row\_index, 4).text().

split('-')[0]) - 1),

self.table\_widget\_m.item(row\_index, 4).text().split('-')[1])

elif self.tabWidget.currentIndex() == 2:

data\_old = self.get\_data(self.table\_widget\_s.item(row\_index, 2).text(),

self.table\_widget\_s.item(row\_index, 3).text(),

self.table\_widget\_s.item(row\_index, 1).text(),

str(int(self.table\_widget\_s.item(row\_index, 4).text().

split('-')[0]) - 1),

self.table\_widget\_s.item(row\_index, 4).text().split('-')[1])

elif self.tabWidget.currentIndex() == 3:

data\_old = self.get\_data(self.table\_widget\_CSET.item(row\_index, 2).text(),

self.table\_widget\_CSET.item(row\_index, 3).text(),

self.table\_widget\_CSET.item(row\_index, 1).text(),

str(int(self.table\_widget\_CSET.item(row\_index, 4).text().

split('-')[0]) - 1),

self.table\_widget\_CSET.item(row\_index, 4).text().split('-')[1])

else:

data\_old = self.get\_data(self.table\_widget\_TSTD.item(row\_index, 2).text(),

self.table\_widget\_TSTD.item(row\_index, 3).text(),

self.table\_widget\_TSTD.item(row\_index, 1).text(),

str(int(self.table\_widget\_TSTD.item(row\_index, 4).text().

split('-')[0]) - 1),

self.table\_widget\_TSTD.item(row\_index, 4).text().split('-')[1])

return data\_old

def get\_data\_new(self, row\_index):

"""Возварщает данные за выбранный период"""

if self.tabWidget.currentIndex() == 0:

data\_new = self.get\_data(self.table\_widget\_b.item(row\_index, 2).text(),

self.table\_widget\_b.item(row\_index, 3).text(),

self.table\_widget\_b.item(row\_index, 1).text(),

self.table\_widget\_b.item(row\_index, 4).text().split('-')[0],

self.table\_widget\_b.item(row\_index, 4).text().split('-')[1])

elif self.tabWidget.currentIndex() == 1:

data\_new = self.get\_data(self.table\_widget\_m.item(row\_index, 2).text(),

self.table\_widget\_m.item(row\_index, 3).text(),

self.table\_widget\_m.item(row\_index, 1).text(),

self.table\_widget\_m.item(row\_index, 4).text().split('-')[0],

self.table\_widget\_m.item(row\_index, 4).text().split('-')[1])

elif self.tabWidget.currentIndex() == 2:

data\_new = self.get\_data(self.table\_widget\_s.item(row\_index, 2).text(),

self.table\_widget\_s.item(row\_index, 3).text(),

self.table\_widget\_s.item(row\_index, 1).text(),

self.table\_widget\_s.item(row\_index, 4).text().split('-')[0],

self.table\_widget\_s.item(row\_index, 4).text().split('-')[1])

elif self.tabWidget.currentIndex() == 3:

data\_new = self.get\_data(self.table\_widget\_CSET.item(row\_index, 2).text(),

self.table\_widget\_CSET.item(row\_index, 3).text(),

self.table\_widget\_CSET.item(row\_index, 1).text(),

self.table\_widget\_CSET.item(row\_index, 4).text().split('-')[0],

self.table\_widget\_CSET.item(row\_index, 4).text().split('-')[1])

else:

data\_new = self.get\_data(self.table\_widget\_TSTD.item(row\_index, 2).text(),

self.table\_widget\_TSTD.item(row\_index, 3).text(),

self.table\_widget\_TSTD.item(row\_index, 1).text(),

self.table\_widget\_TSTD.item(row\_index, 4).text().split('-')[0],

self.table\_widget\_TSTD.item(row\_index, 4).text().split('-')[1])

return data\_new

def open\_page\_1(self):

"""Открывает 1 лист программы"""

self.stackedWidget.setCurrentIndex(0)

def open\_page\_2(self):

"""Открывает 2 лист программы"""

self.stackedWidget.setCurrentIndex(1)

self.data\_analysis()

def update\_all(self):

"""Добавляет информацию в БД и обновляет текущую таблицу"""

self.get\_excel\_data()

self.load\_table()

self.update\_combo\_boxes()

def update\_combo\_boxes(self):

"""Обновление информации в combo box"""

self.direction\_combo\_box\_b.blockSignals(True)

self.direction\_combo\_box\_b.clear()

self.direction\_combo\_box\_b.addItems(["Все", \*self.get\_directions("Бакалавриат")])

self.direction\_combo\_box\_b.blockSignals(False)

self.direction\_combo\_box\_m.blockSignals(True)

self.direction\_combo\_box\_m.clear()

self.direction\_combo\_box\_m.addItems(["Все", \*self.get\_directions("Магистратура")])

self.direction\_combo\_box\_m.blockSignals(False)

self.direction\_combo\_box\_s.blockSignals(True)

self.direction\_combo\_box\_s.clear()

self.direction\_combo\_box\_s.addItems(["Все", \*self.get\_directions("Специалитет")])

self.direction\_combo\_box\_s.blockSignals(False)

self.direction\_combo\_box\_CSET.blockSignals(True)

self.direction\_combo\_box\_CSET.clear()

self.direction\_combo\_box\_CSET.addItems(["Все", \*self.get\_directions("ККМТ")])

self.direction\_combo\_box\_CSET.blockSignals(False)

self.direction\_combo\_box\_TSTD.blockSignals(True)

self.direction\_combo\_box\_TSTD.clear()

self.direction\_combo\_box\_TSTD.addItems(["Все", \*self.get\_directions("ТТД")])

self.direction\_combo\_box\_TSTD.blockSignals(False)

self.year\_combo\_box\_b.blockSignals(True)

self.year\_combo\_box\_b.clear()

self.year\_combo\_box\_b.addItems(["Все", \*self.get\_years("Бакалавриат")])

self.year\_combo\_box\_b.blockSignals(False)

self.year\_combo\_box\_m.blockSignals(True)

self.year\_combo\_box\_m.clear()

self.year\_combo\_box\_m.addItems(["Все", \*self.get\_years("Магистратура")])

self.year\_combo\_box\_m.blockSignals(False)

self.year\_combo\_box\_s.blockSignals(True)

self.year\_combo\_box\_s.clear()

self.year\_combo\_box\_s.addItems(["Все", \*self.get\_years("Специалитет")])

self.year\_combo\_box\_s.blockSignals(False)

self.year\_combo\_box\_CSET.blockSignals(True)

self.year\_combo\_box\_CSET.clear()

self.year\_combo\_box\_CSET.addItems(["Все", \*self.get\_years("ККМТ")])

self.year\_combo\_box\_CSET.blockSignals(False)

self.year\_combo\_box\_TSTD.blockSignals(True)

self.year\_combo\_box\_TSTD.clear()

self.year\_combo\_box\_TSTD.addItems(["Все", \*self.get\_years("ТТД")])

self.year\_combo\_box\_TSTD.blockSignals(False)

def user\_settings(self):

"""Выбор пользователем шрифта для таблиц и его установка"""

icon\_main = QtGui.QIcon()

icon\_main.addPixmap(QtGui.QPixmap("Изображения/free-icon-binder-4624080.png"), QtGui.QIcon.Normal,

QtGui.QIcon.Off)

dialog = QtWidgets.QFontDialog()

dialog.setWindowIcon(icon\_main)

font, ok = dialog.getFont(self.table\_widget\_b.font())

if ok:

self.table\_widget\_b.setFont(font)

self.table\_widget\_m.setFont(font)

self.table\_widget\_s.setFont(font)

self.table\_widget\_CSET.setFont(font)

self.table\_widget\_TSTD.setFont(font)

self.table\_widget\_last\_month.setFont(font)

self.table\_widget\_last\_year.setFont(font)

def open\_dialog(self):

"""Открытие окна пользовательского отчета"""

self.dialog = QtWidgets.QDialog(flags=QtCore.Qt.WindowCloseButtonHint)

self.window = dialog\_ui.UiDialog(self.dialog)

self.dialog.show()

def open\_dialog\_simple(self, trigger):

"""Открытие окна для встроенных отчетов"""

self.dialog\_simple = QtWidgets.QDialog(flags=QtCore.Qt.WindowCloseButtonHint)

self.window\_simple = dialog\_simple\_ui.UiDialogSimple(self.dialog\_simple, trigger)

self.dialog\_simple.show()

**Листинг 6. db\_control.py**

import pyodbc

class DbControl:

def \_\_init\_\_(self):

"""Конструктор класса DbControl"""

super(DbControl, self).\_\_init\_\_()

\_driver = "Driver={SQL server}"

\_server = "Server={0}"

\_data\_base = "Database={Контингент}"

\_trust = "Trust\_Connection=yes"

\_config = open("Config.txt", "r")

\_data = \_config.readline()

\_config.close()

\_conn\_str = ';'.join([\_driver, \_server.format(\_data), \_data\_base, \_trust])

self.conn = pyodbc.connect(\_conn\_str)

self.cursor = self.conn.cursor()

def insert\_data(self, course, info, statistic):

"""Добавление данных из Excel в БД"""

self.cursor.execute("EXEC [Добавление данных] @dir\_id = ?, @dir\_name = ?, @foe\_name = ?, @loe\_name = ?, "

"@date = ?, @course = ?, @free\_in = ?, @disabled\_fi = ?, @target\_fi = ?, "

"@foreigners\_fi = ?, @pay\_in = ?, @disabled\_pi = ?, @target\_pi = ?, @foreigners\_pi = ?, "

"@free\_out = ?, @disabled\_fo = ?, @target\_fo = ?, @foreigners\_fo = ?, @pay\_out = ?, "

"@disabled\_po = ?, @target\_po = ?, @foreigners\_po = ?", info[1], info[2], info[3], info[0],

info[4], course, statistic[0], statistic[1], statistic[2], statistic[3], statistic[4],

statistic[5], statistic[6], statistic[7], statistic[8], statistic[9], statistic[10],

statistic[11], statistic[12], statistic[13], statistic[14], statistic[15])

self.cursor.commit()

def get\_data(self, loe, foe, direction, year, month, course=0):

"""Получение данных из БД по выбранным параметрам"""

if course == "Все":

course = 0

if month == "0" or month == "Все":

self.cursor.execute("EXEC [Получение данных] @loe\_name = ?, @foe\_name = ?, @dir\_name = ?, @year = ?, "

"@course = ?", loe, foe, direction, year, course)

else:

if len(month) < 2:

month = "0" + month

self.cursor.execute("EXEC [Получение данных] @loe\_name = ?, @foe\_name = ?, @dir\_name = ?, @year = ?, "

"@month = ?, @course = ?", loe, foe, direction, year, month, course)

data = self.cursor.fetchall()

return data

def get\_disabled(self, loe, foe, direction, year, month, course=0):

"""Получение данных из БД по инвалидам для отчета"""

if course == "Все":

course = 0

if month == "0" or month == "Все":

self.cursor.execute("EXEC [Получение данных об инвалидах] @loe\_name = ?, @foe\_name = ?, @dir\_name = ?, "

"@year = ?, @course = ?", loe, foe, direction, year, course)

else:

if len(month) < 2:

month = "0" + month

self.cursor.execute("EXEC [Получение данных об инвалидах] @loe\_name = ?, @foe\_name = ?, @dir\_name = ?, "

"@year = ?, @month = ?, @course = ?", loe, foe, direction, year, month, course)

data = self.cursor.fetchall()

return data

def get\_target(self, loe, foe, direction, year, month, course=0):

"""Получение данных из БД по целевикам для отчета"""

if course == "Все":

course = 0

if month == "0" or month == "Все":

self.cursor.execute("EXEC [Получение данных о целевиках] @loe\_name = ?, @foe\_name = ?, @dir\_name = ?, "

"@year = ?, @course = ?", loe, foe, direction, year, course)

else:

if len(month) < 2:

month = "0" + month

self.cursor.execute("EXEC [Получение данных о целевиках] @loe\_name = ?, @foe\_name = ?, @dir\_name = ?, "

"@year = ?, @month = ?, @course = ?", loe, foe, direction, year, month, course)

data = self.cursor.fetchall()

return data

def get\_foreigners(self, loe, foe, direction, year, month, course=0):

"""Получение данных из БД по иностранцам для отчета"""

if course == "Все":

course = 0

if month == "0" or month == "Все":

self.cursor.execute("EXEC [Получение данных об иностранцах] @loe\_name = ?, @foe\_name = ?, @dir\_name = ?, "

"@year = ?, @course = ?", loe, foe, direction, year, course)

else:

if len(month) < 2:

month = "0" + month

self.cursor.execute("EXEC [Получение данных об иностранцах] @loe\_name = ?, @foe\_name = ?, @dir\_name = ?, "

"@year = ?, @month = ?, @course = ?", loe, foe, direction, year, month, course)

data = self.cursor.fetchall()

return data

def get\_directions(self, loe):

"""Получение всех направлений подготовки определенного уровня обучения"""

self.cursor.execute("EXEC [Получение направлений] @loe\_name = ?", loe)

data = self.cursor.fetchall()

data = list(map("".join, list(map(list, data))))

return data

def get\_years(self, loe):

"""Получение списка лет за которые хранятся данные определенного уровня обучения"""

self.cursor.execute("EXEC [Получение лет] @loe\_name = ?", loe)

data = self.cursor.fetchall()

info = list()

for i in data:

info.append(str(\*i))

return info

def get\_months(self, loe):

"""Получение списка лет за которые хранятся данные определенного уровня обучения"""

self.cursor.execute("EXEC [Получение месяцев] @loe\_name = ?", loe)

data = self.cursor.fetchall()

info = list()

for i in data:

info.append(str(\*i))

return info

**Листинг 7. excel\_control.py**

import db\_control

import pandas as pd

from PyQt5 import QtWidgets, QtGui

class ExcelControl(db\_control.DbControl):

def \_\_init\_\_(self):

"""Конструктор класса ExcelControl"""

super(ExcelControl, self).\_\_init\_\_()

def get\_excel\_data(self):

"""Получение данных из файлов выбранных пользователем"""

sheets = ["Контингент (очно)", "Контингент (очно-заочно)", "Контингент (заочно)"]

files = self.get\_file\_names()

if files[len(files) - 1]:

for f in range(len(files) - 1): # Проход по всем файлам

is\_wrong = 0

for i in range(3): # Проход по всем листам в файле

info = []

left\_side = 0

ride\_side = 0

data = pd.read\_excel('{0}'.format(''.join(files[f])), sheet\_name="{0}".format(sheets[i]),

engine="openpyxl")

for j in range(5): # Считывание информации для распределения

pars = data.iloc[[2], [j]]

info.append(\*pars[pars.columns.tolist()[0]].tolist())

for course in range(6): # Проход по всем курсам

statistic = []

for k in range(5 + left\_side, 29 + ride\_side):

pars = data.iloc[[2], [k]]

statistic.append(\*pars[pars.columns.tolist()[0]].tolist())

if info[0] == info[0]: # Добавление существующих данных в БД

self.insert\_data(course + 1, info, statistic[4:12] + statistic[16:24])

else:

is\_wrong = 1

left\_side += 24

ride\_side += 24

if is\_wrong:

msg = QtWidgets.QMessageBox()

icon\_main = QtGui.QIcon()

icon\_main.addPixmap(QtGui.QPixmap("Изображения/free-icon-binder-4624080.png"), QtGui.QIcon.Normal,

QtGui.QIcon.Off)

msg.setWindowTitle("Ошибка")

msg.setWindowIcon(icon\_main)

msg.setText("Файл " + ''.join(files[f]) + " заполнен не правильно")

msg.setIcon(QtWidgets.QMessageBox.Critical)

msg.exec\_()

msg = QtWidgets.QMessageBox()

icon\_main = QtGui.QIcon()

icon\_main.addPixmap(QtGui.QPixmap("Изображения/free-icon-binder-4624080.png"), QtGui.QIcon.Normal,

QtGui.QIcon.Off)

msg.setWindowTitle("Успешно")

msg.setWindowIcon(icon\_main)

msg.setText("Информация считана")

msg.setIcon(QtWidgets.QMessageBox.Information)

msg.exec\_()

def get\_file\_names(self):

"""Получение всех файлов выбранных пользователем"""

filenames, ok = QtWidgets.QFileDialog.getOpenFileNames()

return filenames, ok

def set\_file\_name(self):

"""Получение пути сохранения файла и его названия"""

filename, ok = QtWidgets.QFileDialog.getSaveFileName(filter=".xlsx")

return filename, ok

def export\_data(self, sheet\_1, foe, sheet\_2=None, sheet\_3=None):

"""Экспорт данных в Excel"""

files = self.set\_file\_name()

if files[len(files) - 1]:

if foe == "Все":

data\_1 = pd.DataFrame(sheet\_1)

data\_2 = pd.DataFrame(sheet\_2)

data\_3 = pd.DataFrame(sheet\_3)

sheets = {"Контингент (очно)": data\_1,

"Контингент (очно-заочно)": data\_2,

"Контингент (заочно)": data\_3}

writer = pd.ExcelWriter("{0}{1}".format(\*files), engine="xlsxwriter")

for sheet\_name in sheets.keys():

sheets[sheet\_name].to\_excel(writer, sheet\_name=sheet\_name, index=False)

writer.save()

elif foe == "Очная":

data\_1 = pd.DataFrame(sheet\_1)

data\_1.to\_excel("{0}{1}".format(\*files), sheet\_name="Контингент (очно)", index=False)

elif foe == "Очно-заочная":

data\_1 = pd.DataFrame(sheet\_1)

data\_1.to\_excel("{0}{1}".format(\*files), sheet\_name="Контингент (очно-заочно)", index=False)

elif foe == "Заочная":

data\_1 = pd.DataFrame(sheet\_1)

data\_1.to\_excel("{0}{1}".format(\*files), sheet\_name="Контингент (заочно)", index=False)

msg = QtWidgets.QMessageBox()

icon\_main = QtGui.QIcon()

icon\_main.addPixmap(QtGui.QPixmap("Изображения/free-icon-binder-4624080.png"), QtGui.QIcon.Normal,

QtGui.QIcon.Off)

msg.setWindowTitle("Успешно")

msg.setWindowIcon(icon\_main)

msg.setText("Экспорт данных завершен")

msg.setIcon(QtWidgets.QMessageBox.Information)

msg.exec\_()

**Приложение 2. Код БД**

**Листинг 8. DB\_script.sql**

USE [master]

GO

/\*\*\*\*\*\* Object: Database [Контингент] Script Date: 11.06.2021 10:00:41 \*\*\*\*\*\*/

CREATE DATABASE [Контингент]

CONTAINMENT = NONE

ON PRIMARY

( NAME = N'Контингент', FILENAME = N'C:\Program Files\Microsoft SQL Server\MSSQL15.SQLEXPRESS\MSSQL\DATA\Контингент.mdf' , SIZE = 8192KB , MAXSIZE = UNLIMITED, FILEGROWTH = 65536KB )

LOG ON

( NAME = N'Контингент\_log', FILENAME = N'C:\Program Files\Microsoft SQL Server\MSSQL15.SQLEXPRESS\MSSQL\DATA\Контингент\_log.ldf' , SIZE = 8192KB , MAXSIZE = 2048GB , FILEGROWTH = 65536KB )

WITH CATALOG\_COLLATION = DATABASE\_DEFAULT

GO

ALTER DATABASE [Контингент] SET COMPATIBILITY\_LEVEL = 150

GO

IF (1 = FULLTEXTSERVICEPROPERTY('IsFullTextInstalled'))

begin

EXEC [Контингент].[dbo].[sp\_fulltext\_database] @action = 'enable'

end

GO

ALTER DATABASE [Контингент] SET ANSI\_NULL\_DEFAULT OFF

GO

ALTER DATABASE [Контингент] SET ANSI\_NULLS OFF

GO

ALTER DATABASE [Контингент] SET ANSI\_PADDING OFF

GO

ALTER DATABASE [Контингент] SET ANSI\_WARNINGS OFF

GO

ALTER DATABASE [Контингент] SET ARITHABORT OFF

GO

ALTER DATABASE [Контингент] SET AUTO\_CLOSE OFF

GO

ALTER DATABASE [Контингент] SET AUTO\_SHRINK OFF

GO

ALTER DATABASE [Контингент] SET AUTO\_UPDATE\_STATISTICS ON

GO

ALTER DATABASE [Контингент] SET CURSOR\_CLOSE\_ON\_COMMIT OFF

GO

ALTER DATABASE [Контингент] SET CURSOR\_DEFAULT GLOBAL

GO

ALTER DATABASE [Контингент] SET CONCAT\_NULL\_YIELDS\_NULL OFF

GO

ALTER DATABASE [Контингент] SET NUMERIC\_ROUNDABORT OFF

GO

ALTER DATABASE [Контингент] SET QUOTED\_IDENTIFIER OFF

GO

ALTER DATABASE [Контингент] SET RECURSIVE\_TRIGGERS OFF

GO

ALTER DATABASE [Контингент] SET DISABLE\_BROKER

GO

ALTER DATABASE [Контингент] SET AUTO\_UPDATE\_STATISTICS\_ASYNC OFF

GO

ALTER DATABASE [Контингент] SET DATE\_CORRELATION\_OPTIMIZATION OFF

GO

ALTER DATABASE [Контингент] SET TRUSTWORTHY OFF

GO

ALTER DATABASE [Контингент] SET ALLOW\_SNAPSHOT\_ISOLATION OFF

GO

ALTER DATABASE [Контингент] SET PARAMETERIZATION SIMPLE

GO

ALTER DATABASE [Контингент] SET READ\_COMMITTED\_SNAPSHOT OFF

GO

ALTER DATABASE [Контингент] SET HONOR\_BROKER\_PRIORITY OFF

GO

ALTER DATABASE [Контингент] SET RECOVERY SIMPLE

GO

ALTER DATABASE [Контингент] SET MULTI\_USER

GO

ALTER DATABASE [Контингент] SET PAGE\_VERIFY CHECKSUM

GO

ALTER DATABASE [Контингент] SET DB\_CHAINING OFF

GO

ALTER DATABASE [Контингент] SET FILESTREAM( NON\_TRANSACTED\_ACCESS = OFF )

GO

ALTER DATABASE [Контингент] SET TARGET\_RECOVERY\_TIME = 60 SECONDS

GO

ALTER DATABASE [Контингент] SET DELAYED\_DURABILITY = DISABLED

GO

ALTER DATABASE [Контингент] SET ACCELERATED\_DATABASE\_RECOVERY = OFF

GO

ALTER DATABASE [Контингент] SET QUERY\_STORE = OFF

GO

USE [Контингент]

GO

/\*\*\*\*\*\* Object: Table [dbo].[Движение контингента] Script Date: 11.06.2021 10:00:41 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE TABLE [dbo].[Движение контингента](

[Код направления подготовки] [nchar](8) NOT NULL,

[ID уровня образования] [int] NOT NULL,

[ID формы обучения] [int] NOT NULL,

[Дата записи] [date] NOT NULL,

[Курс] [int] NOT NULL,

[Зачислено за счет бюджетных ассигнований бюджета субъекта РФ] [int] NOT NULL,

[Из них лица с ОВЗ, инвалиды, дети-инвалиды (зачислено бюджет)] [int] NOT NULL,

[Из них на места в рамках квоты целевого приема (зачислено бюджет)] [int] NOT NULL,

[Из них иностранные граждане (зачислено бюджет)] [int] NOT NULL,

[Зачислено по договорам об оказании платных образовательных услуг] [int] NOT NULL,

[Из них лица с ОВЗ, инвалиды, дети-инвалиды (зачислено платное)] [int] NOT NULL,

[Из них на места в рамках квоты целевого приема (зачислено платное)] [int] NOT NULL,

[Из них иностранные граждане (зачислено платное)] [int] NOT NULL,

[Отчислено за счет бюджетных ассигнований бюджета субъекта РФ] [int] NOT NULL,

[Из них лица с ОВЗ, инвалиды, дети-инвалиды (отчислено бюджет)] [int] NOT NULL,

[Из них на места в рамках квоты целевого приема (отчислено бюджет)] [int] NOT NULL,

[Из них иностранные граждане (отчислено бюджет)] [int] NOT NULL,

[Отчислено по договорам об оказании платных образовательных услуг] [int] NOT NULL,

[Из них лица с ОВЗ, инвалиды, дети-инвалиды (отчислено платное)] [int] NOT NULL,

[Из них на места в рамках квоты целевого приема (отчислено платное)] [int] NOT NULL,

[Из них иностранные граждане (отчислено платное)] [int] NOT NULL

) ON [PRIMARY]

GO

/\*\*\*\*\*\* Object: Table [dbo].[Направления] Script Date: 11.06.2021 10:00:42 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE TABLE [dbo].[Направления](

[Код направления подготовки] [nchar](8) NOT NULL,

[Название направления подготовки] [nvarchar](100) NOT NULL,

[ID уровня образования] [int] NOT NULL,

CONSTRAINT [PK\_Направления] PRIMARY KEY CLUSTERED

(

[Код направления подготовки] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY]

GO

/\*\*\*\*\*\* Object: Table [dbo].[Уровни образования] Script Date: 11.06.2021 10:00:42 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE TABLE [dbo].[Уровни образования](

[ID уровня образования] [int] IDENTITY(1,1) NOT NULL,

[Название уровня образования] [nvarchar](50) NOT NULL,

CONSTRAINT [PK\_Уровни образования] PRIMARY KEY CLUSTERED

(

[ID уровня образования] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY]

GO

/\*\*\*\*\*\* Object: Table [dbo].[Формы обучения] Script Date: 11.06.2021 10:00:42 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE TABLE [dbo].[Формы обучения](

[ID формы обучения] [int] IDENTITY(1,1) NOT NULL,

[Название формы обучения] [nvarchar](50) NOT NULL,

CONSTRAINT [PK\_Формы обучения] PRIMARY KEY CLUSTERED

(

[ID формы обучения] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY]

GO

ALTER TABLE [dbo].[Движение контингента] WITH CHECK ADD CONSTRAINT [FK\_Движение контингента\_Направления] FOREIGN KEY([Код направления подготовки])

REFERENCES [dbo].[Направления] ([Код направления подготовки])

ON UPDATE CASCADE

ON DELETE CASCADE

GO

ALTER TABLE [dbo].[Движение контингента] CHECK CONSTRAINT [FK\_Движение контингента\_Направления]

GO

ALTER TABLE [dbo].[Движение контингента] WITH CHECK ADD CONSTRAINT [FK\_Движение контингента\_Уровни образования] FOREIGN KEY([ID уровня образования])

REFERENCES [dbo].[Уровни образования] ([ID уровня образования])

ON UPDATE CASCADE

ON DELETE CASCADE

GO

ALTER TABLE [dbo].[Движение контингента] CHECK CONSTRAINT [FK\_Движение контингента\_Уровни образования]

GO

ALTER TABLE [dbo].[Движение контингента] WITH CHECK ADD CONSTRAINT [FK\_Движение контингента\_Формы обучения] FOREIGN KEY([ID формы обучения])

REFERENCES [dbo].[Формы обучения] ([ID формы обучения])

ON UPDATE CASCADE

ON DELETE CASCADE

GO

ALTER TABLE [dbo].[Движение контингента] CHECK CONSTRAINT [FK\_Движение контингента\_Формы обучения]

GO

ALTER TABLE [dbo].[Направления] WITH CHECK ADD CONSTRAINT [FK\_Направления\_Уровни образования] FOREIGN KEY([ID уровня образования])

REFERENCES [dbo].[Уровни образования] ([ID уровня образования])

GO

ALTER TABLE [dbo].[Направления] CHECK CONSTRAINT [FK\_Направления\_Уровни образования]

GO

/\*\*\*\*\*\* Object: StoredProcedure [dbo].[Добавление данных] Script Date: 11.06.2021 10:00:42 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE PROCEDURE [dbo].[Добавление данных]

@dir\_id nchar(8),

@dir\_name nvarchar(100),

@foe\_name nvarchar(50),

@loe\_name nvarchar(50),

@date date,

@course int,

@free\_in int,

@disabled\_fi int,

@target\_fi int,

@foreigners\_fi int,

@pay\_in int,

@disabled\_pi int,

@target\_pi int,

@foreigners\_pi int,

@free\_out int,

@disabled\_fo int,

@target\_fo int,

@foreigners\_fo int,

@pay\_out int,

@disabled\_po int,

@target\_po int,

@foreigners\_po int

AS

SET NOCOUNT ON

DECLARE @foe\_id int;

DECLARE @loe\_id int;

SET @foe\_id = (SELECT [ID формы обучения] FROM [Формы обучения] WHERE @foe\_name = [Название формы обучения])

IF EXISTS (SELECT [Код направления подготовки] FROM [Движение контингента] WHERE [Код направления подготовки] = @dir\_id AND [Дата записи] = @date

AND [Курс] = @course AND [ID формы обучения] = @foe\_id)

BEGIN

UPDATE [Движение контингента]

SET [Зачислено за счет бюджетных ассигнований бюджета субъекта РФ] = @free\_in, [Из них лица с ОВЗ, инвалиды, дети-инвалиды (зачислено бюджет)] = @disabled\_fi, [Из них на места в рамках квоты целевого приема (зачислено бюджет)] = @target\_fi,

[Из них иностранные граждане (зачислено бюджет)] = @foreigners\_fi, [Зачислено по договорам об оказании платных образовательных услуг] = @pay\_in, [Из них лица с ОВЗ, инвалиды, дети-инвалиды (зачислено платное)] = @disabled\_pi,

[Из них на места в рамках квоты целевого приема (зачислено платное)] = @target\_pi, [Из них иностранные граждане (зачислено платное)] = @foreigners\_pi, [Отчислено за счет бюджетных ассигнований бюджета субъекта РФ] = @free\_out,

[Из них лица с ОВЗ, инвалиды, дети-инвалиды (отчислено бюджет)] = @disabled\_fo, [Из них на места в рамках квоты целевого приема (отчислено бюджет)] = @target\_fo, [Из них иностранные граждане (отчислено бюджет)] = @foreigners\_fo,

[Отчислено по договорам об оказании платных образовательных услуг] = @pay\_out, [Из них лица с ОВЗ, инвалиды, дети-инвалиды (отчислено платное)] = @disabled\_po, [Из них на места в рамках квоты целевого приема (отчислено платное)] = @target\_po, [Из них иностранные граждане (отчислено платное)] = @foreigners\_po

END

ELSE

BEGIN

IF EXISTS (SELECT [ID формы обучения] FROM [Формы обучения] WHERE @foe\_name = [Название формы обучения])

BEGIN

SET @foe\_id = (SELECT [ID формы обучения] FROM [Формы обучения] WHERE @foe\_name = [Название формы обучения])

END

ELSE

BEGIN

INSERT INTO [Формы обучения] ([Название формы обучения])

VALUES (@foe\_name)

SET @foe\_id = (SELECT [ID формы обучения] FROM [Формы обучения] WHERE @foe\_name = [Название формы обучения])

END

IF EXISTS (SELECT [ID уровня образования] FROM [Уровни образования] WHERE @loe\_name = [Название уровня образования])

BEGIN

SET @loe\_id = (SELECT [ID уровня образования] FROM [Уровни образования] WHERE @loe\_name = [Название уровня образования])

END

ELSE

BEGIN

INSERT INTO [Уровни образования] ([Название уровня образования])

VALUES (@loe\_name)

SET @loe\_id = (SELECT [ID уровня образования] FROM [Уровни образования] WHERE @loe\_name = [Название уровня образования])

END

IF EXIStS (SELECT [Код направления подготовки] FROM [Направления] WHERE [Код направления подготовки] = @dir\_id)

BEGIN

INSERT INTO [Движение контингента] ([Код направления подготовки], [ID уровня образования], [ID формы обучения], [Дата записи], Курс, [Зачислено за счет бюджетных ассигнований бюджета субъекта РФ], [Из них лица с ОВЗ, инвалиды, дети-инвалиды (зачислено бюджет)], [Из них на места в рамках квоты целевого приема (зачислено бюджет)], [Из них иностранные граждане (зачислено бюджет)], [Зачислено по договорам об оказании платных образовательных услуг], [Из них лица с ОВЗ, инвалиды, дети-инвалиды (зачислено платное)], [Из них на места в рамках квоты целевого приема (зачислено платное)], [Из них иностранные граждане (зачислено платное)], [Отчислено за счет бюджетных ассигнований бюджета субъекта РФ], [Из них лица с ОВЗ, инвалиды, дети-инвалиды (отчислено бюджет)], [Из них на места в рамках квоты целевого приема (отчислено бюджет)], [Из них иностранные граждане (отчислено бюджет)], [Отчислено по договорам об оказании платных образовательных услуг], [Из них лица с ОВЗ, инвалиды, дети-инвалиды (отчислено платное)], [Из них на места в рамках квоты целевого приема (отчислено платное)], [Из них иностранные граждане (отчислено платное)])

VALUES (@dir\_id, @loe\_id, @foe\_id, @date, @course, @free\_in, @disabled\_fi, @target\_fi, @foreigners\_fi, @pay\_in, @disabled\_pi, @target\_pi, @foreigners\_pi, @free\_out, @disabled\_fo, @target\_fo, @foreigners\_fo, @pay\_out, @disabled\_po, @target\_po, @foreigners\_po)

END

ELSE

BEGIN

INSERT INTO [Направления] ([Код направления подготовки], [Название направления подготовки], [ID уровня образования])

VALUES (@dir\_id, @dir\_name, @loe\_id)

END

END

GO

/\*\*\*\*\*\* Object: StoredProcedure [dbo].[Получение данных] Script Date: 11.06.2021 10:00:42 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE PROCEDURE [dbo].[Получение данных]

@loe\_name nvarchar(50) = 'Все',

@foe\_name nvarchar(50) = 'Все',

@dir\_name nvarchar(100) = 'Все',

@year nchar(4) = 'Все',

@month nchar(2) = '00',

@course int = 0

AS

SET NOCOUNT ON

DECLARE @sql\_condition nvarchar(4000) = 'SELECT mk.[Код направления подготовки], dir.[Название направления подготовки], loe.[Название уровня образования], foe.[Название формы обучения], mk.[Дата записи], mk.Курс,

(mk.[Зачислено за счет бюджетных ассигнований бюджета субъекта РФ] + mk.[Зачислено по договорам об оказании платных образовательных услуг]) as [Всего зачислено],

(mk.[Из них лица с ОВЗ, инвалиды, дети-инвалиды (зачислено бюджет)] + mk.[Из них лица с ОВЗ, инвалиды, дети-инвалиды (зачислено платное)]) as [Из них лица с ОВЗ, инвалиды, дети-инвалиды (всего зачислено)],

(mk.[Из них на места в рамках квоты целевого приема (зачислено бюджет)] + mk.[Из них на места в рамках квоты целевого приема (зачислено платное)]) as [Из них на места в рамках квоты целевого приема (всего зачислено)],

(mk.[Из них иностранные граждане (зачислено бюджет)] + mk.[Из них иностранные граждане (зачислено платное)]) as [Из них иностранные граждане (всего зачислено)],

(mk.[Отчислено за счет бюджетных ассигнований бюджета субъекта РФ] + mk.[Отчислено по договорам об оказании платных образовательных услуг]) as [Всего отчислено],

(mk.[Из них лица с ОВЗ, инвалиды, дети-инвалиды (отчислено бюджет)] + mk.[Из них лица с ОВЗ, инвалиды, дети-инвалиды (отчислено платное)]) as [Из них лица с ОВЗ, инвалиды, дети-инвалиды (всего отчислено)],

(mk.[Из них на места в рамках квоты целевого приема (отчислено бюджет)] + mk.[Из них на места в рамках квоты целевого приема (отчислено платное)]) as [Из них на места в рамках квоты целевого приема (всего отчислено)],

(mk.[Из них иностранные граждане (отчислено бюджет)] + mk.[Из них иностранные граждане (отчислено платное)]) as [Из них иностранные граждане (всего отчислено)],

mk.[Зачислено за счет бюджетных ассигнований бюджета субъекта РФ], mk.[Из них лица с ОВЗ, инвалиды, дети-инвалиды (зачислено бюджет)], mk.[Из них на места в рамках квоты целевого приема (зачислено бюджет)],

mk.[Из них иностранные граждане (зачислено бюджет)], mk.[Зачислено по договорам об оказании платных образовательных услуг], mk.[Из них лица с ОВЗ, инвалиды, дети-инвалиды (зачислено платное)],

mk.[Из них на места в рамках квоты целевого приема (зачислено платное)], mk.[Из них иностранные граждане (зачислено платное)], mk.[Отчислено за счет бюджетных ассигнований бюджета субъекта РФ],

mk.[Из них лица с ОВЗ, инвалиды, дети-инвалиды (отчислено бюджет)], mk.[Из них на места в рамках квоты целевого приема (отчислено бюджет)], mk.[Из них иностранные граждане (отчислено бюджет)],

mk.[Отчислено по договорам об оказании платных образовательных услуг], mk.[Из них лица с ОВЗ, инвалиды, дети-инвалиды (отчислено платное)], mk.[Из них на места в рамках квоты целевого приема (отчислено платное)],

mk.[Из них иностранные граждане (отчислено платное)]

FROM [Движение контингента] as mk

INNER JOIN [Направления] as dir ON (mk.[Код направления подготовки] = dir.[Код направления подготовки])

INNER JOIN [Уровни образования] as loe ON (mk.[ID уровня образования] = loe.[ID уровня образования])

INNER JOIN [Формы обучения] as foe on (mk.[ID формы обучения] = foe.[ID формы обучения]) WHERE'

DECLARE @sql nvarchar(4000) = 'SELECT mk.[Код направления подготовки], dir.[Название направления подготовки], loe.[Название уровня образования], foe.[Название формы обучения], mk.[Дата записи], mk.Курс,

(mk.[Зачислено за счет бюджетных ассигнований бюджета субъекта РФ] + mk.[Зачислено по договорам об оказании платных образовательных услуг]) as [Всего зачислено],

(mk.[Из них лица с ОВЗ, инвалиды, дети-инвалиды (зачислено бюджет)] + mk.[Из них лица с ОВЗ, инвалиды, дети-инвалиды (зачислено платное)]) as [Из них лица с ОВЗ, инвалиды, дети-инвалиды (всего зачислено)],

(mk.[Из них на места в рамках квоты целевого приема (зачислено бюджет)] + mk.[Из них на места в рамках квоты целевого приема (зачислено платное)]) as [Из них на места в рамках квоты целевого приема (всего зачислено)],

(mk.[Из них иностранные граждане (зачислено бюджет)] + mk.[Из них иностранные граждане (зачислено платное)]) as [Из них иностранные граждане (всего зачислено)],

(mk.[Отчислено за счет бюджетных ассигнований бюджета субъекта РФ] + mk.[Отчислено по договорам об оказании платных образовательных услуг]) as [Всего отчислено],

(mk.[Из них лица с ОВЗ, инвалиды, дети-инвалиды (отчислено бюджет)] + mk.[Из них лица с ОВЗ, инвалиды, дети-инвалиды (отчислено платное)]) as [Из них лица с ОВЗ, инвалиды, дети-инвалиды (всего отчислено)],

(mk.[Из них на места в рамках квоты целевого приема (отчислено бюджет)] + mk.[Из них на места в рамках квоты целевого приема (отчислено платное)]) as [Из них на места в рамках квоты целевого приема (всего отчислено)],

(mk.[Из них иностранные граждане (отчислено бюджет)] + mk.[Из них иностранные граждане (отчислено платное)]) as [Из них иностранные граждане (всего отчислено)],

mk.[Зачислено за счет бюджетных ассигнований бюджета субъекта РФ], mk.[Из них лица с ОВЗ, инвалиды, дети-инвалиды (зачислено бюджет)], mk.[Из них на места в рамках квоты целевого приема (зачислено бюджет)],

mk.[Из них иностранные граждане (зачислено бюджет)], mk.[Зачислено по договорам об оказании платных образовательных услуг], mk.[Из них лица с ОВЗ, инвалиды, дети-инвалиды (зачислено платное)],

mk.[Из них на места в рамках квоты целевого приема (зачислено платное)], mk.[Из них иностранные граждане (зачислено платное)], mk.[Отчислено за счет бюджетных ассигнований бюджета субъекта РФ],

mk.[Из них лица с ОВЗ, инвалиды, дети-инвалиды (отчислено бюджет)], mk.[Из них на места в рамках квоты целевого приема (отчислено бюджет)], mk.[Из них иностранные граждане (отчислено бюджет)],

mk.[Отчислено по договорам об оказании платных образовательных услуг], mk.[Из них лица с ОВЗ, инвалиды, дети-инвалиды (отчислено платное)], mk.[Из них на места в рамках квоты целевого приема (отчислено платное)],

mk.[Из них иностранные граждане (отчислено платное)]

FROM [Движение контингента] as mk

INNER JOIN [Направления] as dir ON (mk.[Код направления подготовки] = dir.[Код направления подготовки])

INNER JOIN [Уровни образования] as loe ON (mk.[ID уровня образования] = loe.[ID уровня образования])

INNER JOIN [Формы обучения] as foe on (mk.[ID формы обучения] = foe.[ID формы обучения])'

DECLARE @loe\_id int

DECLARE @foe\_id int

DECLARE @dir\_id nchar(8)

IF @loe\_name != 'Все'

BEGIN

SET @loe\_id = (SELECT [ID уровня образования] FROM [Уровни образования] WHERE [Название уровня образования] = @loe\_name)

SET @sql\_condition = @sql\_condition + ' mk.[ID уровня образования] = ' + STR(@loe\_id, 1)

END

IF @foe\_name != 'Все'

BEGIN

SET @foe\_id = (SELECT [ID формы обучения] FROM [Формы обучения] WHERE [Название формы обучения] = @foe\_name)

IF @sql\_condition = (@sql + ' WHERE')

SET @sql\_condition = @sql\_condition + ' mk.[ID формы обучения] = ' + STR(@foe\_id, 1)

ELSE

SET @sql\_condition = @sql\_condition + ' AND mk.[ID формы обучения] = ' + STR(@foe\_id, 1)

END

IF @dir\_name != 'Все'

BEGIN

SET @dir\_id = (SELECT [Код направления подготовки] FROM [Направления] WHERE [Название направления подготовки] = @dir\_name)

IF @sql\_condition = (@sql + ' WHERE')

SET @sql\_condition = @sql\_condition + ' mk.[Код направления подготовки] = ''' + @dir\_id + ''''

ELSE

SET @sql\_condition = @sql\_condition + ' AND mk.[Код направления подготовки] = ''' + @dir\_id + ''''

END

IF @year != 'Все' AND @month != '00'

BEGIN

IF @sql\_condition = (@sql + ' WHERE')

SET @sql\_condition = @sql\_condition + ' mk.[Дата записи] like ''' + @year + '-' + @month + '-%'''

ELSE

SET @sql\_condition = @sql\_condition + ' AND mk.[Дата записи] like ''' + @year + '-' + @month + '-%'''

END

ELSE

IF @year != 'Все'

IF @sql\_condition = (@sql + ' WHERE')

SET @sql\_condition = @sql\_condition + ' mk.[Дата записи] like ''' + @year + '-%-%'''

ELSE

SET @sql\_condition = @sql\_condition + ' AND mk.[Дата записи] like ''' + @year + '-%-%'''

ELSE

IF @month != '00'

IF @sql\_condition = (@sql + ' WHERE')

SET @sql\_condition = @sql\_condition + ' mk.[Дата записи] like ''%-' + @month + '-%'''

ELSE

SET @sql\_condition = @sql\_condition + ' AND mk.[Дата записи] like ''%-' + @month + '-%'''

IF @course != 0

BEGIN

IF @sql\_condition = (@sql + ' WHERE')

SET @sql\_condition = @sql\_condition + ' mk.[Курс] = ' + STR(@course, 1)

ELSE

SET @sql\_condition = @sql\_condition + ' AND mk.[Курс] = ' + STR(@course, 1)

END

IF @sql\_condition = (@sql + ' WHERE')

EXECUTE(@sql)

ELSE

EXECUTE(@sql\_condition)

GO

/\*\*\*\*\*\* Object: StoredProcedure [dbo].[Получение данных о целевиках] Script Date: 11.06.2021 10:00:42 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE PROCEDURE [dbo].[Получение данных о целевиках]

@loe\_name nvarchar(50) = 'Все',

@foe\_name nvarchar(50) = 'Все',

@dir\_name nvarchar(100) = 'Все',

@year nchar(4) = 'Все',

@month nchar(2) = '00',

@course int = 0

AS

SET NOCOUNT ON

DECLARE @sql\_condition nvarchar(4000) = 'SELECT dir.[Название направления подготовки], loe.[Название уровня образования], foe.[Название формы обучения], mk.[Дата записи], mk.Курс,

(mk.[Из них на места в рамках квоты целевого приема (зачислено бюджет)] + mk.[Из них на места в рамках квоты целевого приема (зачислено платное)]) as [Из них на места в рамках квоты целевого приема (всего зачислено)],

(mk.[Из них на места в рамках квоты целевого приема (отчислено бюджет)] + mk.[Из них на места в рамках квоты целевого приема (отчислено платное)]) as [Из них на места в рамках квоты целевого приема (всего отчислено)],

mk.[Из них на места в рамках квоты целевого приема (зачислено бюджет)], mk.[Из них на места в рамках квоты целевого приема (зачислено платное)],

mk.[Из них на места в рамках квоты целевого приема (отчислено бюджет)], mk.[Из них на места в рамках квоты целевого приема (отчислено платное)]

FROM [Движение контингента] as mk

INNER JOIN [Направления] as dir ON (mk.[Код направления подготовки] = dir.[Код направления подготовки])

INNER JOIN [Уровни образования] as loe ON (mk.[ID уровня образования] = loe.[ID уровня образования])

INNER JOIN [Формы обучения] as foe on (mk.[ID формы обучения] = foe.[ID формы обучения]) WHERE'

DECLARE @sql nvarchar(4000) = 'SELECT dir.[Название направления подготовки], loe.[Название уровня образования], foe.[Название формы обучения], mk.[Дата записи], mk.Курс,

(mk.[Из них на места в рамках квоты целевого приема (зачислено бюджет)] + mk.[Из них на места в рамках квоты целевого приема (зачислено платное)]) as [Из них на места в рамках квоты целевого приема (всего зачислено)],

(mk.[Из них на места в рамках квоты целевого приема (отчислено бюджет)] + mk.[Из них на места в рамках квоты целевого приема (отчислено платное)]) as [Из них на места в рамках квоты целевого приема (всего отчислено)],

mk.[Из них на места в рамках квоты целевого приема (зачислено бюджет)], mk.[Из них на места в рамках квоты целевого приема (зачислено платное)],

mk.[Из них на места в рамках квоты целевого приема (отчислено бюджет)], mk.[Из них на места в рамках квоты целевого приема (отчислено платное)]

FROM [Движение контингента] as mk

INNER JOIN [Направления] as dir ON (mk.[Код направления подготовки] = dir.[Код направления подготовки])

INNER JOIN [Уровни образования] as loe ON (mk.[ID уровня образования] = loe.[ID уровня образования])

INNER JOIN [Формы обучения] as foe on (mk.[ID формы обучения] = foe.[ID формы обучения])'

DECLARE @loe\_id int

DECLARE @foe\_id int

DECLARE @dir\_id nchar(8)

IF @loe\_name != 'Все'

BEGIN

SET @loe\_id = (SELECT [ID уровня образования] FROM [Уровни образования] WHERE [Название уровня образования] = @loe\_name)

SET @sql\_condition = @sql\_condition + ' mk.[ID уровня образования] = ' + STR(@loe\_id, 1)

END

IF @foe\_name != 'Все'

BEGIN

SET @foe\_id = (SELECT [ID формы обучения] FROM [Формы обучения] WHERE [Название формы обучения] = @foe\_name)

IF @sql\_condition = (@sql + ' WHERE')

SET @sql\_condition = @sql\_condition + ' mk.[ID формы обучения] = ' + STR(@foe\_id, 1)

ELSE

SET @sql\_condition = @sql\_condition + ' AND mk.[ID формы обучения] = ' + STR(@foe\_id, 1)

END

IF @dir\_name != 'Все'

BEGIN

SET @dir\_id = (SELECT [Код направления подготовки] FROM [Направления] WHERE [Название направления подготовки] = @dir\_name)

IF @sql\_condition = (@sql + ' WHERE')

SET @sql\_condition = @sql\_condition + ' mk.[Код направления подготовки] = ''' + @dir\_id + ''''

ELSE

SET @sql\_condition = @sql\_condition + ' AND mk.[Код направления подготовки] = ''' + @dir\_id + ''''

END

IF @year != 'Все' AND @month != '00'

BEGIN

IF @sql\_condition = (@sql + ' WHERE')

SET @sql\_condition = @sql\_condition + ' mk.[Дата записи] like ''' + @year + '-' + @month + '-%'''

ELSE

SET @sql\_condition = @sql\_condition + ' AND mk.[Дата записи] like ''' + @year + '-' + @month + '-%'''

END

ELSE

IF @year != 'Все'

IF @sql\_condition = (@sql + ' WHERE')

SET @sql\_condition = @sql\_condition + ' mk.[Дата записи] like ''' + @year + '-%-%'''

ELSE

SET @sql\_condition = @sql\_condition + ' AND mk.[Дата записи] like ''' + @year + '-%-%'''

ELSE

IF @month != '00'

IF @sql\_condition = (@sql + ' WHERE')

SET @sql\_condition = @sql\_condition + ' mk.[Дата записи] like ''%-' + @month + '-%'''

ELSE

SET @sql\_condition = @sql\_condition + ' AND mk.[Дата записи] like ''%-' + @month + '-%'''

IF @course != 0

BEGIN

IF @sql\_condition = (@sql + ' WHERE')

SET @sql\_condition = @sql\_condition + ' mk.[Курс] = ' + STR(@course, 1)

ELSE

SET @sql\_condition = @sql\_condition + ' AND mk.[Курс] = ' + STR(@course, 1)

END

IF @sql\_condition = (@sql + ' WHERE')

EXECUTE(@sql)

ELSE

EXECUTE(@sql\_condition)

GO

/\*\*\*\*\*\* Object: StoredProcedure [dbo].[Получение данных об инвалидах] Script Date: 11.06.2021 10:00:42 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE PROCEDURE [dbo].[Получение данных об инвалидах]

@loe\_name nvarchar(50) = 'Все',

@foe\_name nvarchar(50) = 'Все',

@dir\_name nvarchar(100) = 'Все',

@year nchar(4) = 'Все',

@month nchar(2) = '00',

@course int = 0

AS

SET NOCOUNT ON

DECLARE @sql\_condition nvarchar(4000) = 'SELECT dir.[Название направления подготовки], loe.[Название уровня образования], foe.[Название формы обучения], mk.[Дата записи], mk.Курс,

(mk.[Из них лица с ОВЗ, инвалиды, дети-инвалиды (зачислено бюджет)] + mk.[Из них лица с ОВЗ, инвалиды, дети-инвалиды (зачислено платное)]) as [Из них лица с ОВЗ, инвалиды, дети-инвалиды (всего зачислено)],

(mk.[Из них лица с ОВЗ, инвалиды, дети-инвалиды (отчислено бюджет)] + mk.[Из них лица с ОВЗ, инвалиды, дети-инвалиды (отчислено платное)]) as [Из них лица с ОВЗ, инвалиды, дети-инвалиды (всего отчислено)],

mk.[Из них лица с ОВЗ, инвалиды, дети-инвалиды (зачислено бюджет)], mk.[Из них лица с ОВЗ, инвалиды, дети-инвалиды (зачислено платное)],

mk.[Из них лица с ОВЗ, инвалиды, дети-инвалиды (отчислено бюджет)], mk.[Из них лица с ОВЗ, инвалиды, дети-инвалиды (отчислено платное)]

FROM [Движение контингента] as mk

INNER JOIN [Направления] as dir ON (mk.[Код направления подготовки] = dir.[Код направления подготовки])

INNER JOIN [Уровни образования] as loe ON (mk.[ID уровня образования] = loe.[ID уровня образования])

INNER JOIN [Формы обучения] as foe on (mk.[ID формы обучения] = foe.[ID формы обучения]) WHERE'

DECLARE @sql nvarchar(4000) = 'SELECT dir.[Название направления подготовки], loe.[Название уровня образования], foe.[Название формы обучения], mk.[Дата записи], mk.Курс,

(mk.[Из них лица с ОВЗ, инвалиды, дети-инвалиды (зачислено бюджет)] + mk.[Из них лица с ОВЗ, инвалиды, дети-инвалиды (зачислено платное)]) as [Из них лица с ОВЗ, инвалиды, дети-инвалиды (всего зачислено)],

(mk.[Из них лица с ОВЗ, инвалиды, дети-инвалиды (отчислено бюджет)] + mk.[Из них лица с ОВЗ, инвалиды, дети-инвалиды (отчислено платное)]) as [Из них лица с ОВЗ, инвалиды, дети-инвалиды (всего отчислено)],

mk.[Из них лица с ОВЗ, инвалиды, дети-инвалиды (зачислено бюджет)], mk.[Из них лица с ОВЗ, инвалиды, дети-инвалиды (зачислено платное)],

mk.[Из них лица с ОВЗ, инвалиды, дети-инвалиды (отчислено бюджет)], mk.[Из них лица с ОВЗ, инвалиды, дети-инвалиды (отчислено платное)]

FROM [Движение контингента] as mk

INNER JOIN [Направления] as dir ON (mk.[Код направления подготовки] = dir.[Код направления подготовки])

INNER JOIN [Уровни образования] as loe ON (mk.[ID уровня образования] = loe.[ID уровня образования])

INNER JOIN [Формы обучения] as foe on (mk.[ID формы обучения] = foe.[ID формы обучения])'

DECLARE @loe\_id int

DECLARE @foe\_id int

DECLARE @dir\_id nchar(8)

IF @loe\_name != 'Все'

BEGIN

SET @loe\_id = (SELECT [ID уровня образования] FROM [Уровни образования] WHERE [Название уровня образования] = @loe\_name)

SET @sql\_condition = @sql\_condition + ' mk.[ID уровня образования] = ' + STR(@loe\_id, 1)

END

IF @foe\_name != 'Все'

BEGIN

SET @foe\_id = (SELECT [ID формы обучения] FROM [Формы обучения] WHERE [Название формы обучения] = @foe\_name)

IF @sql\_condition = (@sql + ' WHERE')

SET @sql\_condition = @sql\_condition + ' mk.[ID формы обучения] = ' + STR(@foe\_id, 1)

ELSE

SET @sql\_condition = @sql\_condition + ' AND mk.[ID формы обучения] = ' + STR(@foe\_id, 1)

END

IF @dir\_name != 'Все'

BEGIN

SET @dir\_id = (SELECT [Код направления подготовки] FROM [Направления] WHERE [Название направления подготовки] = @dir\_name)

IF @sql\_condition = (@sql + ' WHERE')

SET @sql\_condition = @sql\_condition + ' mk.[Код направления подготовки] = ''' + @dir\_id + ''''

ELSE

SET @sql\_condition = @sql\_condition + ' AND mk.[Код направления подготовки] = ''' + @dir\_id + ''''

END

IF @year != 'Все' AND @month != '00'

BEGIN

IF @sql\_condition = (@sql + ' WHERE')

SET @sql\_condition = @sql\_condition + ' mk.[Дата записи] like ''' + @year + '-' + @month + '-%'''

ELSE

SET @sql\_condition = @sql\_condition + ' AND mk.[Дата записи] like ''' + @year + '-' + @month + '-%'''

END

ELSE

IF @year != 'Все'

IF @sql\_condition = (@sql + ' WHERE')

SET @sql\_condition = @sql\_condition + ' mk.[Дата записи] like ''' + @year + '-%-%'''

ELSE

SET @sql\_condition = @sql\_condition + ' AND mk.[Дата записи] like ''' + @year + '-%-%'''

ELSE

IF @month != '00'

IF @sql\_condition = (@sql + ' WHERE')

SET @sql\_condition = @sql\_condition + ' mk.[Дата записи] like ''%-' + @month + '-%'''

ELSE

SET @sql\_condition = @sql\_condition + ' AND mk.[Дата записи] like ''%-' + @month + '-%'''

IF @course != 0

BEGIN

IF @sql\_condition = (@sql + ' WHERE')

SET @sql\_condition = @sql\_condition + ' mk.[Курс] = ' + STR(@course, 1)

ELSE

SET @sql\_condition = @sql\_condition + ' AND mk.[Курс] = ' + STR(@course, 1)

END

IF @sql\_condition = (@sql + ' WHERE')

EXECUTE(@sql)

ELSE

EXECUTE(@sql\_condition)

GO

/\*\*\*\*\*\* Object: StoredProcedure [dbo].[Получение данных об иностранцах] Script Date: 11.06.2021 10:00:42 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE PROCEDURE [dbo].[Получение данных об иностранцах]

@loe\_name nvarchar(50) = 'Все',

@foe\_name nvarchar(50) = 'Все',

@dir\_name nvarchar(100) = 'Все',

@year nchar(4) = 'Все',

@month nchar(2) = '00',

@course int = 0

AS

SET NOCOUNT ON

DECLARE @sql\_condition nvarchar(4000) = 'SELECT dir.[Название направления подготовки], loe.[Название уровня образования], foe.[Название формы обучения], mk.[Дата записи], mk.Курс,

(mk.[Из них иностранные граждане (зачислено бюджет)] + mk.[Из них иностранные граждане (зачислено платное)]) as [Из них иностранные граждане (всего зачислено)],

(mk.[Из них иностранные граждане (отчислено бюджет)] + mk.[Из них иностранные граждане (отчислено платное)]) as [Из них иностранные граждане (всего отчислено)],

mk.[Из них иностранные граждане (зачислено бюджет)], mk.[Из них иностранные граждане (зачислено платное)],

mk.[Из них иностранные граждане (отчислено бюджет)], mk.[Из них иностранные граждане (отчислено платное)]

FROM [Движение контингента] as mk

INNER JOIN [Направления] as dir ON (mk.[Код направления подготовки] = dir.[Код направления подготовки])

INNER JOIN [Уровни образования] as loe ON (mk.[ID уровня образования] = loe.[ID уровня образования])

INNER JOIN [Формы обучения] as foe on (mk.[ID формы обучения] = foe.[ID формы обучения]) WHERE'

DECLARE @sql nvarchar(4000) = 'SELECT dir.[Название направления подготовки], loe.[Название уровня образования], foe.[Название формы обучения], mk.[Дата записи], mk.Курс,

(mk.[Из них иностранные граждане (зачислено бюджет)] + mk.[Из них иностранные граждане (зачислено платное)]) as [Из них иностранные граждане (всего зачислено)],

(mk.[Из них иностранные граждане (отчислено бюджет)] + mk.[Из них иностранные граждане (отчислено платное)]) as [Из них иностранные граждане (всего отчислено)],

mk.[Из них иностранные граждане (зачислено бюджет)], mk.[Из них иностранные граждане (зачислено платное)],

mk.[Из них иностранные граждане (отчислено бюджет)], mk.[Из них иностранные граждане (отчислено платное)]

FROM [Движение контингента] as mk

INNER JOIN [Направления] as dir ON (mk.[Код направления подготовки] = dir.[Код направления подготовки])

INNER JOIN [Уровни образования] as loe ON (mk.[ID уровня образования] = loe.[ID уровня образования])

INNER JOIN [Формы обучения] as foe on (mk.[ID формы обучения] = foe.[ID формы обучения])'

DECLARE @loe\_id int

DECLARE @foe\_id int

DECLARE @dir\_id nchar(8)

IF @loe\_name != 'Все'

BEGIN

SET @loe\_id = (SELECT [ID уровня образования] FROM [Уровни образования] WHERE [Название уровня образования] = @loe\_name)

SET @sql\_condition = @sql\_condition + ' mk.[ID уровня образования] = ' + STR(@loe\_id, 1)

END

IF @foe\_name != 'Все'

BEGIN

SET @foe\_id = (SELECT [ID формы обучения] FROM [Формы обучения] WHERE [Название формы обучения] = @foe\_name)

IF @sql\_condition = (@sql + ' WHERE')

SET @sql\_condition = @sql\_condition + ' mk.[ID формы обучения] = ' + STR(@foe\_id, 1)

ELSE

SET @sql\_condition = @sql\_condition + ' AND mk.[ID формы обучения] = ' + STR(@foe\_id, 1)

END

IF @dir\_name != 'Все'

BEGIN

SET @dir\_id = (SELECT [Код направления подготовки] FROM [Направления] WHERE [Название направления подготовки] = @dir\_name)

IF @sql\_condition = (@sql + ' WHERE')

SET @sql\_condition = @sql\_condition + ' mk.[Код направления подготовки] = ''' + @dir\_id + ''''

ELSE

SET @sql\_condition = @sql\_condition + ' AND mk.[Код направления подготовки] = ''' + @dir\_id + ''''

END

IF @year != 'Все' AND @month != '00'

BEGIN

IF @sql\_condition = (@sql + ' WHERE')

SET @sql\_condition = @sql\_condition + ' mk.[Дата записи] like ''' + @year + '-' + @month + '-%'''

ELSE

SET @sql\_condition = @sql\_condition + ' AND mk.[Дата записи] like ''' + @year + '-' + @month + '-%'''

END

ELSE

IF @year != 'Все'

IF @sql\_condition = (@sql + ' WHERE')

SET @sql\_condition = @sql\_condition + ' mk.[Дата записи] like ''' + @year + '-%-%'''

ELSE

SET @sql\_condition = @sql\_condition + ' AND mk.[Дата записи] like ''' + @year + '-%-%'''

ELSE

IF @month != '00'

IF @sql\_condition = (@sql + ' WHERE')

SET @sql\_condition = @sql\_condition + ' mk.[Дата записи] like ''%-' + @month + '-%'''

ELSE

SET @sql\_condition = @sql\_condition + ' AND mk.[Дата записи] like ''%-' + @month + '-%'''

IF @course != 0

BEGIN

IF @sql\_condition = (@sql + ' WHERE')

SET @sql\_condition = @sql\_condition + ' mk.[Курс] = ' + STR(@course, 1)

ELSE

SET @sql\_condition = @sql\_condition + ' AND mk.[Курс] = ' + STR(@course, 1)

END

IF @sql\_condition = (@sql + ' WHERE')

EXECUTE(@sql)

ELSE

EXECUTE(@sql\_condition)

GO

/\*\*\*\*\*\* Object: StoredProcedure [dbo].[Получение лет] Script Date: 11.06.2021 10:00:42 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE PROCEDURE [dbo].[Получение лет]

@loe\_name nvarchar(50)

AS

SET NOCOUNT ON

IF @loe\_name = 'Все'

BEGIN

SELECT DISTINCT YEAR([Дата записи]) FROM [Движение контингента]

END

ELSE

BEGIN

SELECT DISTINCT YEAR(mk.[Дата записи]) as Год

FROM [Движение контингента] as mk

INNER JOIN [Уровни образования] as loe ON (loe.[ID уровня образования] = mk.[ID уровня образования] AND loe.[Название уровня образования] = @loe\_name)

END

GO

/\*\*\*\*\*\* Object: StoredProcedure [dbo].[Получение месяцев] Script Date: 11.06.2021 10:00:42 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE PROCEDURE [dbo].[Получение месяцев]

@loe\_name nvarchar(50)

AS

SET NOCOUNT ON

IF @loe\_name = 'Все'

BEGIN

SELECT DISTINCT MONTH([Дата записи]) FROM [Движение контингента]

END

ELSE

BEGIN

SELECT DISTINCT MONTH(mk.[Дата записи]) as Год

FROM [Движение контингента] as mk

INNER JOIN [Уровни образования] as loe ON (loe.[ID уровня образования] = mk.[ID уровня образования] AND loe.[Название уровня образования] = @loe\_name)

END

GO

/\*\*\*\*\*\* Object: StoredProcedure [dbo].[Получение направлений] Script Date: 11.06.2021 10:00:42 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE PROCEDURE [dbo].[Получение направлений]

@loe\_name nvarchar(50)

AS

SET NOCOUNT ON

IF @loe\_name = 'Все'

BEGIN

SELECT [Название направления подготовки] FROM [Направления]

END

ELSE

BEGIN

SELECT dir.[Название направления подготовки]

FROM [Направления] as dir

INNER JOIN [Уровни образования] as loe ON (loe.[ID уровня образования] = dir.[ID уровня образования] AND loe.[Название уровня образования] = @loe\_name)

END

GO

USE [master]

GO

ALTER DATABASE [Контингент] SET READ\_WRITE

GO