Содержание

Введение 2

1 Постановка задачи 3

1.1 Общее представление 3

1.2 Существующие проблемы и пути их решения 4

1.3 Тестирование системы на виртуальной машине 4

1.4 Обновление изменений основной DL 8

2 Программно-файловая структура 11

2.1 Программные единицы 11

2.2 Используемые таблицы 16

3 Разработанные алгоритмы 20

3.1 Класс Teacher.java 20

3.2 Интегрированная статистика 22

3.3 Статистика по нескольким курсам 24

4 Технологии использования 26

4.1 Интегрированная статистика 26

4.2 Статистика по нескольким курсам 29

5 Полученные результаты 31

5.1 Интегрированная статистика 31

5.2 Статистика по нескольким курсам 33

5.3 Редактирование учеников учителя 37

5.4 Таблица учителей 38

Заключение 39

Список использованных источников 40

Приложение А 41

# Введение

В 1999 году сначала в сети ГГУ, а затем и сети Internet по адресу [http://dl.gsu.unibel.by](http://dl.gsu.unibel.by/) начал функционировать сервер «Дистанционное обучение в Беларуси» (Distance Learning Belarus). Своим появлением он предоставил тысячам людей со всего мира возможность, не выходя из дома получить новые знания или просто блеснуть своей сообразительностью и умом. Множество людей благодаря DL, смогли настолько усовершенствовать свое мастерство, что с легкостью преодолели рубежи республиканских соревнований и вышли на международный уровень.

С уверенностью можно заявить, что сервиса аналогичного DL в республике Беларусь как не существовало на 1999-ый год, так и не существует теперь. За прошедшие пять лет на базе этого сервера было проведено множество олимпиад и соревнований, а так же 5 недель компьютерных наук (GCSW). В последние годы все городские и областные олимпиады школьников по информатике так же проводились с использованием данного сервера, а также многочисленные сборы и тренировки.

В настоящее время в системе существуют сотни обучающих курсов, десятки тысяч задач. Каждый день в системе работают сотни пользователей. Также система постоянно совершенствуется и развивается, добавляются новые возможности и функционал.

Учителя в системе работают с большим количеством учеников, которые с каждым годом постоянно меняется, т.к. некоторые заканчивают обучение, а некоторые начинают.

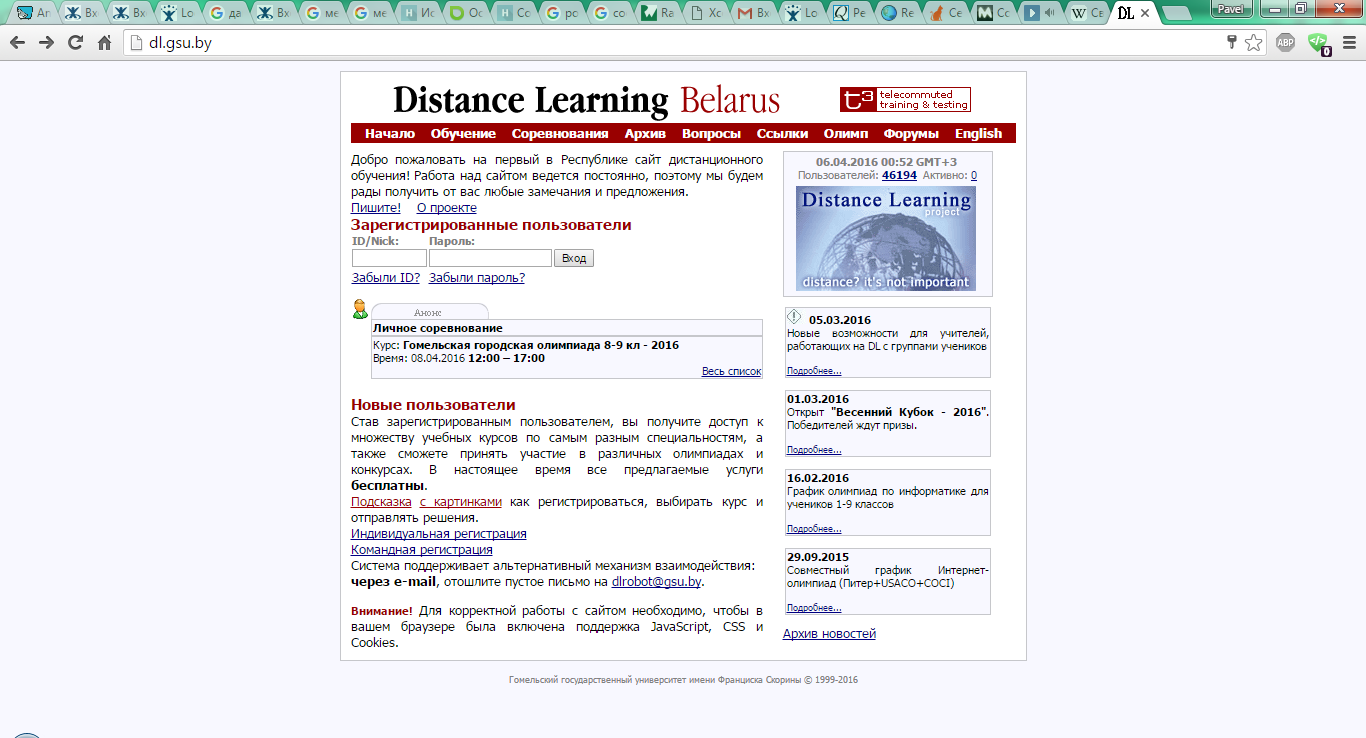
Магистерская диссертация преследовала следующие основные цели:

* Развитие средств отображения результатов обучения учеников;
* Добавление статистики по нескольким курсам учителей по их ученикам;
* Добавление и улучшение функционала для контроля учеников.

# 1 Постановка задачи

## 1.1 Общее представление

Сайт может работать с двумя типами пользователей: индивидуальный пользователь и команда. Команда при этом должна состоять из зарегистрированных пользователей.



**Рисунок 1 – Главная страница сайта**

Весь учебный материал распределен по учебным курсам. Каждый такой курс содержит соответствующий теоретический материал и задачи. Подписавшись на курс, пользователь получает доступ теории и соответствующим ей задачам. Курсы, задачи внутри курсов и теория организованы при помощи структуры «дерево».

Отдельный вид практических курсов - соревнования. Такие курсы, как правило, включают только задачи и период их "жизни" значительно меньше - обычно несколько часов. По истечении этого времени подводятся результаты и определяются победители.

Соревнования могут быть личные и командные (по правилам чемпионата ACM).

По желанию составителей некоторые курсы могут быть доступны только индивидуальным ученикам или только командам.

В зависимости от типа курса могут строиться таблицы результатов различных видов.

Привилегированные пользователи могут быть владельцами задач, редакторами курсов или администраторами.

## 1.2 Существующие проблемы и пути их решения

Система DL.GSU.BY насчитывает огромное количество учеников и учителей. Для более продуктивного отслеживание успеваемости учеников требуется внедрение функционала, отслеживающего успеваемость учеников в разные промежутки времени и по разным критериям.

Основные средства отображения успеваемости учеников, являются страницы олимпиад и кубков учеников по все различным курсам. На их основе и создавался дополнительный функционал для отслеживания успеваемости учеников.

Кроме отслеживания успеваемости учеников, так же есть необходимость для средств, позволяющих управлять группами учителей, редактирования и проверки введенной учеником информации о себе.

Разработка новых средств отображения результатов которые интегрируются с уже готовым функционалом, позволяют уменьшить количество действий совершаемые пользователями системы и помогает ученикам следить за своей активностью среди остальных учеников.

Актуальные задачи разработки:

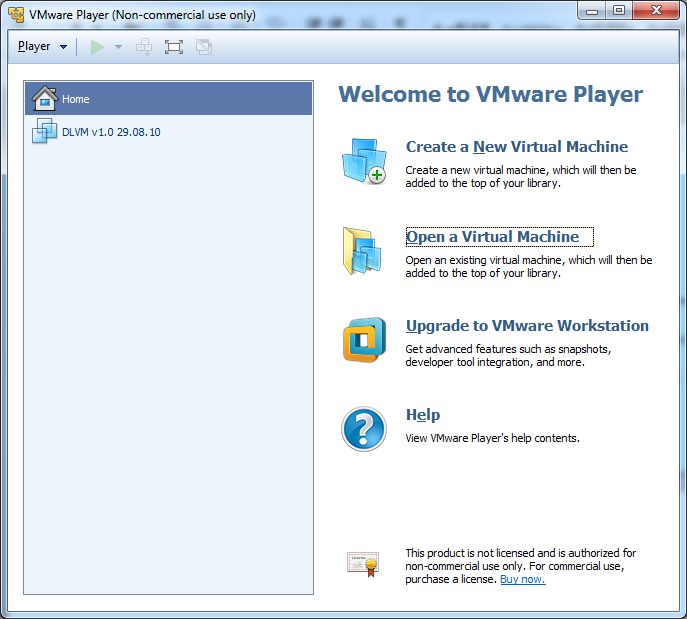
* Доработка готового функционала результатов учеников;
* Разработка и внедрение нового функционала успеваемости учеников;
* Развитие средств отображения результатов.

## 1.3 Тестирование системы на виртуальной машине

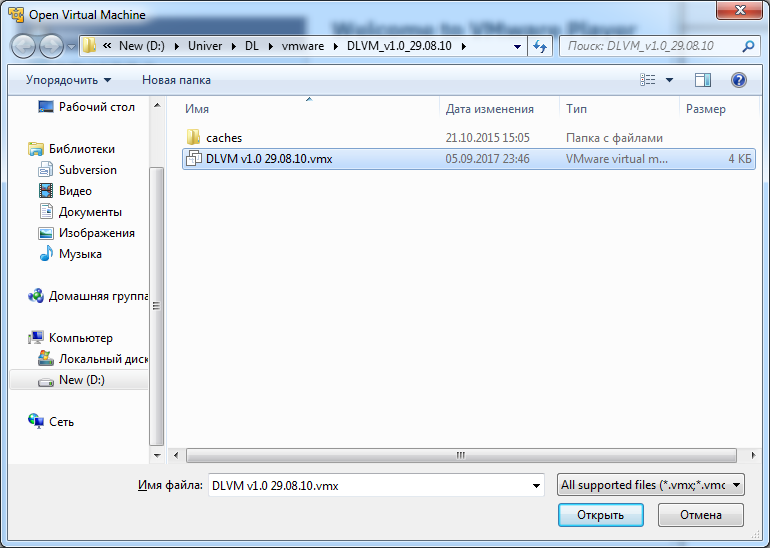
Для тестирования изменений на локальной машине используется программа VMware Player, которая запускает уже готовую виртуальную машину DL. Ссылки на скачивание VMware Player можно найти в интернете. Ссылки на скачивание архива с виртуальной машиной DL можно найти по этой ссылке <http://confluence.newit.gsu.by/display/dldev/DLVM>.

После установки и запуска VMware Player в главном окне требуется нажать на кнопку «Open a Virtual Machine» и в диалоговом окне указать путь к файлу виртуальной машины DL. После открытия виртуальной машины в левой панеле появится строка с названием виртуальной машины, по нажатию на которую, можно запустить виртуальную машину, нажав «Play virtual machine» или настроить, нажав «Edit virtual machine settings». Все нужные пароли виртуальной машины можно найти в файле passwords.txt который лежит в архиве с виртуальной машиной.

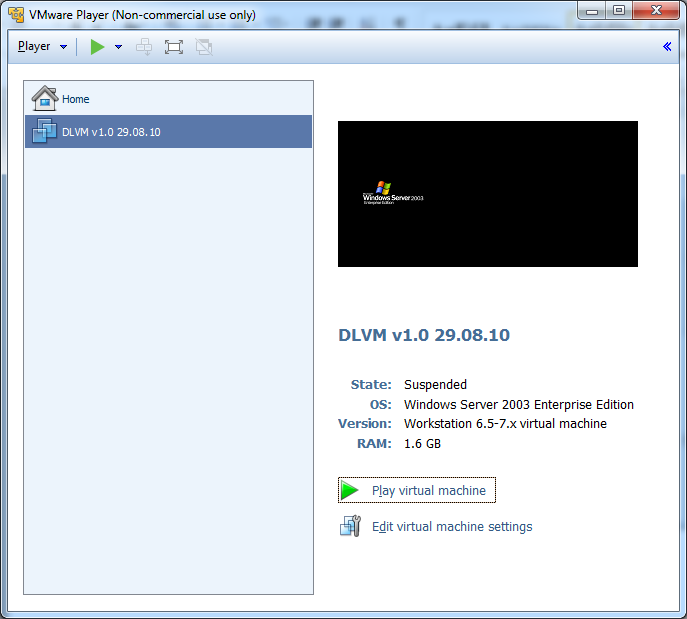
Действия по открытию и запуску DLVM показаны на рисунках 4-6.



**Рисунок 4 – Окно VMware Player**



**Рисунок 5 – Выбор файла виртуальной машины**

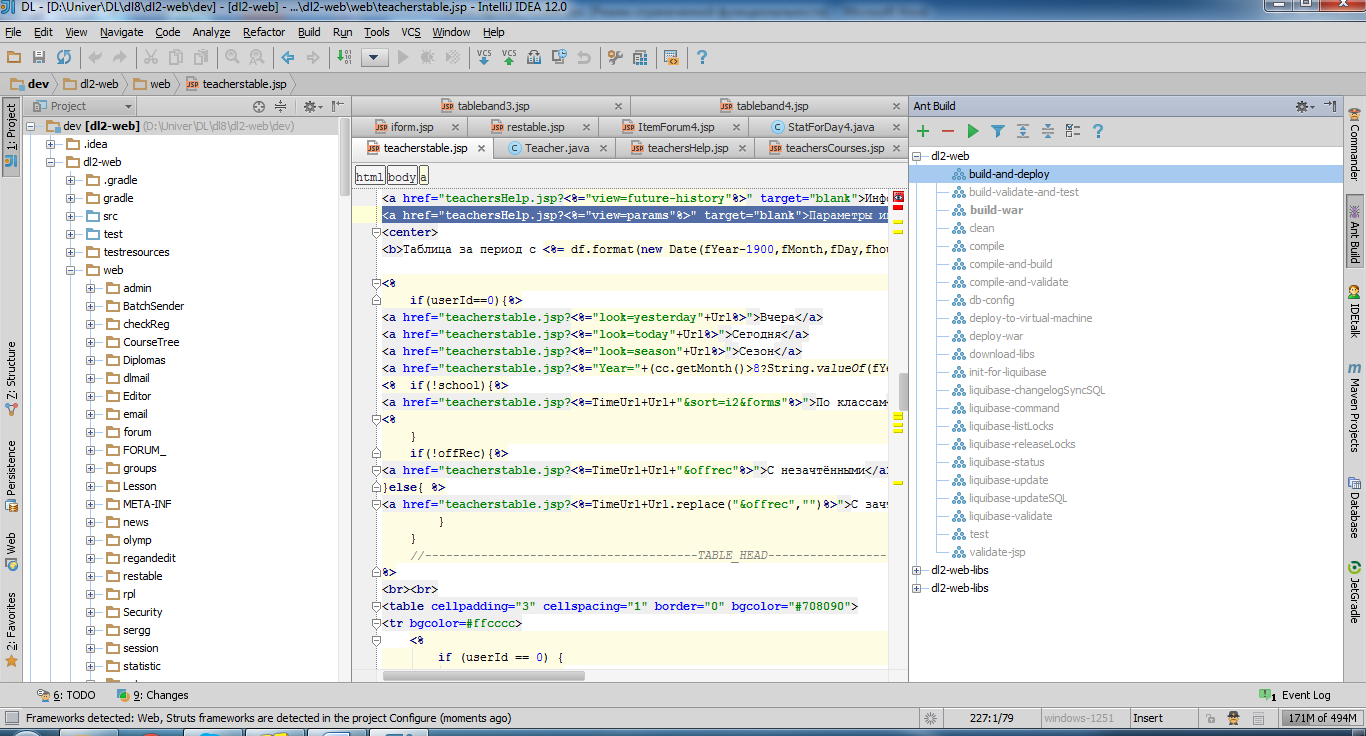


**Рисунок 6 – Запуск виртуальной машины**

После запуска виртуальной машины появляется возможность тестирования внесенных изменений в java коде путем построения и деплоя war архива на виртуальную машину, так же появляется возможность просмотра таблиц и тестирования SQL запросов в среде Microsoft SQL Management Studio. Теперь сайт DL, который расположен на виртуальной машине, доступен по адресу <http://dl>.

Для внесения изменений java части в среде Intelij IDEA используется Ant Built. Кнопка панели Ant Build находится справа главного окна в IDEA, при нажатии на которую открывается список заранее подготовленных действий Ant. Чтобы обновить DL на виртуальной машине, нужно выбрать секцию «dl-web» и в открывшемся списке нажать «build-and-deploy», Ant Build автоматически соберет и задеплоит war архив на виртуальную машину.

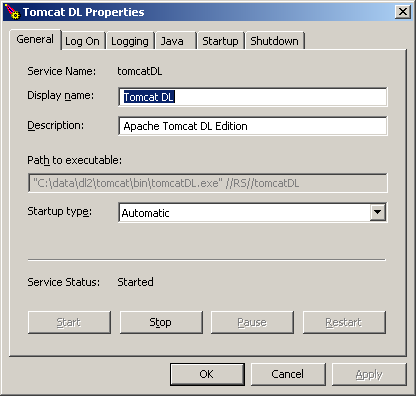
Выбор действия по обновлению виртуальной машины показан на рисунке 7.



**Рисунок 7 – Окно Intelij IDEA и панели Ant Build**

Изменения на виртуальной машине происходят не сразу, а спустя 1-3ех минут после деплоя. Чтобы увидеть изменения на сайте достаточно обновить страницу сайта в браузере. Иногда, при частых изменениях, требуется перезапустить Tomcat на виртуальной машине. Панель управления Tomcat находится в папке c:\data\dl2\tomcat\bin\tomcatDL.exe. Для перезапуска требуется остановить томкат кнопкой «Stop» и нажать на кнопку «Start» для запуска. Так же в панели можно посмотреть статус Tomcat.

Панель Tomcat показана на рисунке 8.

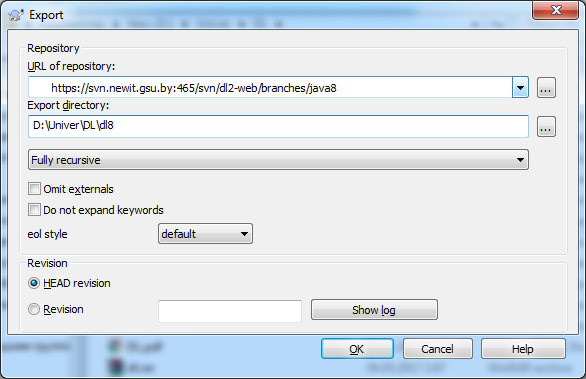


**Рисунок 8 – Панель Tomcat на виртуальной машине**

## 1.4 Обновление изменений основной DL

Для обновления изменений основной DL используется программа контроля версий Tortoise SVN скачать которую можно в интернете. Чтобы загрузить папку с кодом проекта DL необходимо в файловой системе нажать правую кнопку и в появившемся окне нажать кнопку TortoiseSVN->Export  
в появившемся окне указать адрес репозитория, имя папки куда будет загружен проект, а так же имя и пароль для доступа к репозиторию. Для веб части адрес репозитория будет https://svn.newit.gsu.by:465/svn/dl2-web/branches/java8.

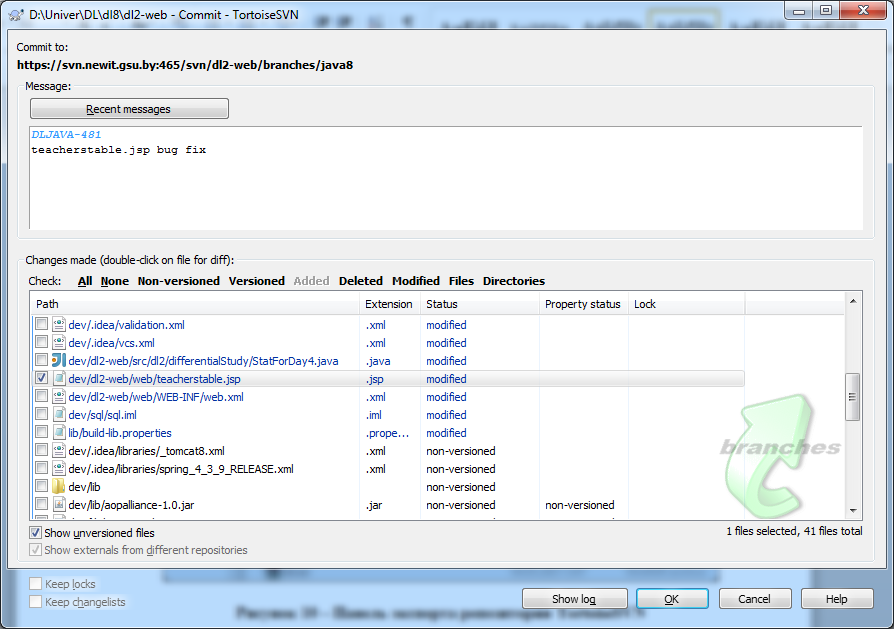
Окно экспорта репозитория показано на рисунке 9.



**Рисунок 9 – Панель экспорта репозитория TortoiseSVN**

После внесения изменений в проекте, для отправки изменений на основную DL требуется нажать правой кнопкой на файле проекта и на появившейся панеле нажать «SVN Commit». В появившемся окне написать номер задачи и краткое описание внесенных изменений, а в нижней панели выбрать файлы которые требуется обновить. После отправки изменений требуется их деплой, который выполняет администратор после проверки.

Пример окна отправки изменений показан на рисунке 10.



**Рисунок 10 – Пример окна отправки изменений на основную DL**

# 2 Программно-файловая структура

Основная логика компонентов содержится в java-классах и JSP страницах, которые в свою очередь получают данные из базы данных, потом с помощью JSP страниц генерируется отображение и отправляется клиенту.  
 Схема взаимодействия клиента и сервера показана на рисунке 11.

Сервер  
DL Server

База  
Данных

Java Servlet

JDBC Driver

Клиент

Браузер

**Рисунок 11 – Схема взаимодействия клиента и сервера**

## 2.1 Программные единицы

Основная логика работы с учителями описана в классе Teacher.java, который расположен в модуле dl2-web по пути ***dl2-web*** *->* ***src*** *->* ***dl2*** *->* ***Teacher.java***. Основное использование этого класса это получение списка учителей и их информацию.

Класс Teachers.java использует следующие таблицы:

* Pupils **–** получение информации об учениках и учителях, выборка учеников по классам;
* Groups **–** работа с группами учителей и кандидатов;
* ObjectsGroups **–** добавление учеников к учителям, учителей в группу учителей и кандидатов. Получение информации по всем объектам той или иной группы.
* vwUsers **–** отображение пользователей для более удобного получения информации без обращения к промежуточным таблицам. Например: город, школа, буква и номер класса.

Разработанный в данной дипломной работе класс Teachers.java представлен в таблицах 1-3.

**Таблица 1 − Поля класса**

|  |  |  |
| --- | --- | --- |
| Название | Тип | Описание |
| id | int | Уникальный идентификатор пользователя |
| name | String | Имя и фамилия учителя |
| eName | String | Имя и фамилия учителя на английском |
| midle | String | Отчество |
| eMidle | String | Отчество на английском языке |
| groupId | int | Id группы учителя |
| city | String | Город учителя |
| school | String | Школа учителя |
| email | String | Email учителя |
| PupilCount | int | Количество учеников |
| TaskCount | int | Количество решенных задач учениками |
| courses | Map<String,Integer> | Мап курсов с количеством активных учеников учителя, используется для страницы  teachersCourses.jsp |
| schoolCount | int | Количество школьников |
| sum | int | Сумма активных учеников по курсам, используется для страницы TeachersInfo.jsp |

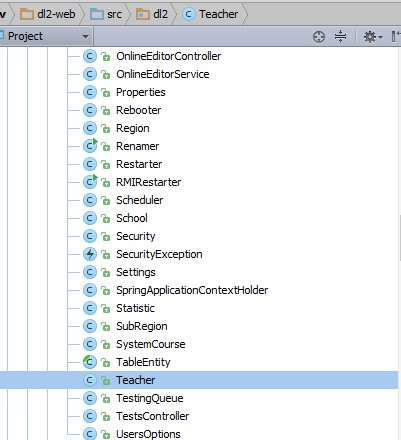
**Таблица 2 − Методы класса**

|  |  |  |
| --- | --- | --- |
| Название | Тип | Описание |
| getGroupID() | int | Возвращает id группы учителя |
| getName(int lang) | String | Возвращает имя и фамилию в зависимости от языка |
| getFullName(int lang) | String | Возвращает ФИО в зависимости от языка английском |
| getPupilCount() | int | Возвращает количество учеников |
| setPupilCount() | int | Задает количество учеников |
| getTaskCount() | int | Возвращает количество задач |
| setTaskCount() | int | Задает количество задач |
| addPupil(String id) | void | Добавляет ученика к учителю |
| Название | Тип | Описание |
| deletePupil(String id) | void | Удаляет ученика из группы учителя |
| getTeacherGroupId() | void | Получает группу учителя из базы данных и заносит значение в GroupId |
| isTeacher(String id) | boolean | Проверяет, учит ли учитель ученика, по id ученика |
| getPupilCounts() | int | Получает количество всех учеников учителя из базы данных |
| load(ResultSet rs) | void | Задает значения полей класса из строки базы данных |
| loadSchoolCount() | void | Получение количество всех школьников из базы |

**Таблица 3 − Статические методы класса**

|  |  |  |
| --- | --- | --- |
| Название | Тип | Описание |
| getTeachers(String sort) | Teacher[] | Возвращает массив объектов класса Teacher.java по указанной сортировке |
| getTeacherByUserId(int userId) | Teacher | Получение обьекта учителя из базы. |
| loadTeachers(String sort) | void | Получает список учителей из базы данных по указанной сортировке |
| loadCountByCourses(Teacher[] teachers,String[] courses, String fDate, String tDate, Map courseForms) | void | Загружает данные о количестве учеников по курсам, а так же количество школьников и общее количество учеников |
| updateObjectsGroups(int objID,int groupID,boolean insert) | void | В зависимости от значения insert либо удаляет запись в таблице ObjectsGroups, либо добавлет запись |
| getTeachersGroup() | int | Получает id группы учителей из базы данных |
| loadCandidateGroup() | int | Получает id группы кандидатов в учителя из базы данных |
| Название | Тип | Описание |
| addTeacher(String id) | void | Делает пользователя учителем по id без лишней работы для администратора |
| deleteTeacher(String id) | void | Удаляет учителя по id без лишней работы для администратора |
| addCandidate(String id) | void | Добавляет пользователя в кандидаты по id |
| deleteCandidate(String id) | void | Удаляет пользователя из группы кандидатов по id |
| createTeacherGroup(String id,String str) | void | Создает группу учителя для учеников по id учителя |
| getNameById(String id, int lang) | String | Получает имя пользователя из базы данных по id и значению языка |
| CheckTeach(String id) | boolean | Проверяет является ли пользователь учителем |
| CheckID(String id) | void | Проверяет существует ли такой id пользователя |
| getFormId(String name) | int | Возвращает id класса по значению класса |
| getTeacherByTeachId(Teacher[] teachers,int group) | Teacher | Получает объект Teacher из массива учителей по id группы учителя |
| getNameById(String id,int lang) | String | Получение имени юзера по его id и языку |

Расположение класса показано на рисунке 12.



**Рисунок 12 – Расположение файла Teacher.java**

Страницы JSP используются в основном для отображения информации на клиенте, так же в этих файлах может быть логика и обращение к базе данных. Все используемые в исследовательской работе JSP страницы находятся в dl2-web модуле в папке ***dl2-web*** *->* ***web***.

Программно-файловая структура JSP страниц с используемыми таблицами базы данных и классов представлена в таблице 4.

**Таблица 4 − Программно-файловая структура JSP страниц**

|  |  |  |  |
| --- | --- | --- | --- |
| Имя JSP страницы | Описание страницы | Таблицы базы данных | Дополнительные файлы |
| teacherstable.jsp | Интегрированная статистика учеников и учителей | TaskTrees Tasks  TestingLog  Pupils  vwUsers  UsersCourses  ObjectGroups  Courses | Teacher.java |
| TeachersInfo.jsp | Информация о всех учителях по курсам | Courses | Teacher.java |
| teachersCourses.jsp | Статистика активности и количества учеников по нескольким курсам | TestingLog  vwUsers  Pupils  ObjectGroups | Teacher.java |
| teachersgroups.jsp | Работа учителя с группой учителя, администрирование кандидатов и учителей | Pupils ObjectGroups | Teacher.java |
| teacherEdit.jsp | Редактирование учеников учителями | - | Pupils.java Teacher.java |
| teachersHelp.jsp | Справка по страницам учителей | - | - |

## 2.2 Используемые таблицы

Работа с базами данных в Java реализована через специальный интерфейс, который носит название JDBC.

В системе DL существует огромное множество таблиц и хранимых процедур. Большинство их них взаимосвязаны и служат для представления необходимой информации. Рассмотрим более подробно часть таблиц, которые используются в данном проекте.

Все пользователи, зарегистрированные в системе DL.GSU.BY, имеют представление в базе данных в виде таблицы 5.

**Таблица 5 − Pupils**

|  |  |  |
| --- | --- | --- |
| Атрибут | Тип | Описание |
| ID | Int | Уникальный идентификатор ученика |
| Last | Varchar(50) | Фамилия |
| ELast | Varchar(50) | Фамилия на английском языке |
| First | Varchar(50) | Имя |
| EFirst | Varchar(50) | Имя на английском языке |
| Midle | Varchar(50) | Отчество |
| EMidle | Varchar(50) | Отчество на английском языке |
| Country | Int | Id страны |
| City | Int | Id города |
| Form | Int | Id класса |
| Address | Varchar(225) | Адрес |
| Birth | Datetime | Дата рождения |
| Password | Varchar(50) | MD5 пароль |
| Email | Varchar(50) | Email пользователя |
| Phone | Varchar(50) | Номер телефона |
| Joined | Datetime | Дата регистрации |
| Question | Varchar(50) | Вопрос |
| Answer | Varchar(50) | Ответ на вопрос |
| Subscribed | bit | Поделится ли информацией о регистрации на форуме |
| AdditionalInfo | text | Дополнительная информация |
| School | Varchar(50) | Наименование учебного заведение |
| ESchool | Varchar(50) | Наименование учебного заведение на английском |
| Nick | Varchar(50) | Ник пользователя на форуме |

Таблица 6 предназначена для представления информации о группах учеников.

**Таблица 6 − Groups**

|  |  |  |
| --- | --- | --- |
| Атрибут | Тип | Описание |
| ID | Int | Уникальный идентификатор группы |
| Type | Int | Id типа группы |
| Name | Varchar(255) | Имя группы |
| EName | Varchar(255) | Имя группы на английском |
| PublicGroup | bit | Публичная ли группа |
| OwnerUserID |  | Id организатора группы |
| OwnerSchoolID |  | Id учебного заведения |
| Year | Datetime | Дата создания |

Таблица 7 предназначена для организации учеников в группы.

**Таблица 7 − ObjectsGroups**

|  |  |  |
| --- | --- | --- |
| Атрибут | Тип | Описание |
| ObjectID | Int | Id ученика из таблицы Pupils |
| GroupID | Int | Id группы |

В таблице 8 представлена информация о таблице TestingLog, которая используется для хранения отправленных решений учениками.

**Таблица 8 − TestingLog**

|  |  |  |
| --- | --- | --- |
| Атрибут | Тип | Описание |
| ID | Int | Уникальный идентификатор записи |
| NodeId | Int | Id записи задачи из таблицы TaskTrees |
| CourseId | Int | Id курса |
| TaskId | Int | Id задачи |
| UserId | Int | Id юзера отправившего решение |
| Result | Int | Количество пройденных тестов |
| SubmitTime | Datetime | Время отправки решения |
| QueueId | Int | Id очереди |

В таблице 9 представлена информация о таблице Tasks, которая используется для хранения заданий.

**Таблица 9 − Tasks**

|  |  |  |
| --- | --- | --- |
| Атрибут | Тип | Описание |
| ID | Int | Уникальный идентификатор записи |
| Name | Varchar(255) | Имя задачи |
| EName | Varchar(255) | Имя задачи на английском |
| TypeId | Int | Id типа задачи |
| Cost | Int | Количество тестов необходимое для решения задачи |

В таблице 10 представлена информация о таблице TaskTrees, которая используется для формирования списков заданий.

**Таблица 10 − TaskTrees**

|  |  |  |
| --- | --- | --- |
| Атрибут | Тип | Описание |
| ID | Int | Уникальный идентификатор записи |
| Deep | Int | Глубина вложения задачи |
| TaskId | Int | Id задачи из таблицы Tasks |
| ParentId | Int | Id родительской записи |

В таблице 11 представлена информация о таблице TaskTrees, которая используется для формирования списков заданий.

**Таблица 11 − TaskTrees**

|  |  |  |
| --- | --- | --- |
| Атрибут | Тип | Описание |
| ID | Int | Уникальный идентификатор записи |
| Deep | Int | Глубина вложения задачи |
| TaskId | Int | Id задачи из таблицы Tasks |
| ParentId | Int | Id родительской записи |

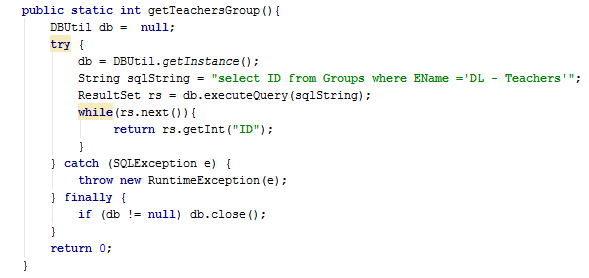
# 3 Разработанные алгоритмы

## 3.1 Класс Teacher.java

Все пользователи системы, учителя и ученики, хранятся в одной таблице. Необходимо было отделить логику учителей от логики учеников. Для решения данной задачи была использована готовая система групп для учеников, и разработан класс Teachers.java .

Для создания списка учителей администратору системы необходимо создать группу с английским названием «DL – Teachers», класс Teachers.java находит id группы учителей по английскому имени, тем самым получая id учителей. Так же для каждого учителя необходимо создать группу для хранения его учеников. Например если учитель Долинский Михаил, то группа учителя должна называться «DL – учитель – Долинский Михаил». Разработанный класс Teachers.java находит id группы учителя, тем самым получая список учеников.

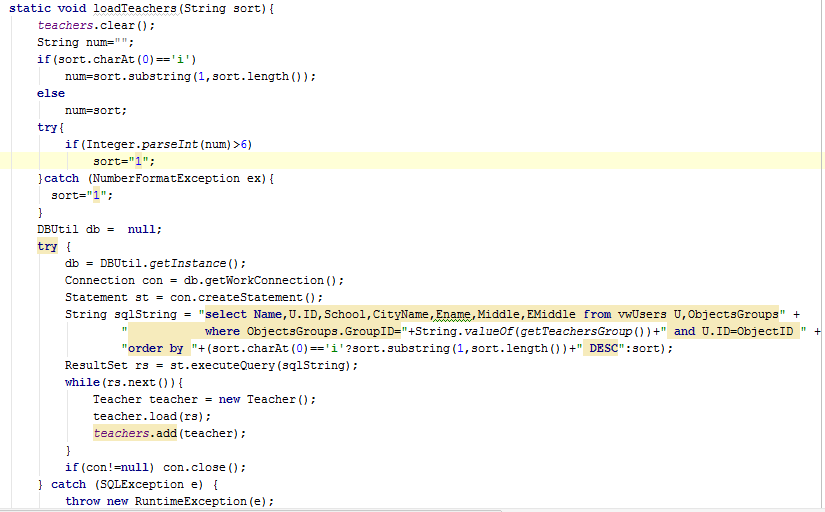
Метод с запросом получения группы учителей показан на рисунке 12.



**Рисунок 12 – Метод получения группы учителей**

Данный запрос возвращает id группы учителей, с помощью которого можно получить список имен учителей, используется в методе getTeacherGroup().

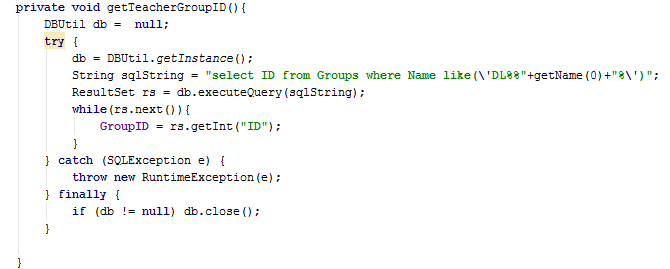
Метод загрузки учителей показан на рисунке 13.



**Рисунок 13 – Метод загрузки учителей**

Где параматр sort используется для начальной сортировки списка. Значения сорт от 1 до количества полей в выборке, если первый символ sort буква i то сортировка будет по убыванию.

После получения списка учителей в методе loadTeachers(String sort), для каждого учителя из списка идет получение их групп учителей.   
 Метод получения группы учителя показан на рисунке 14.

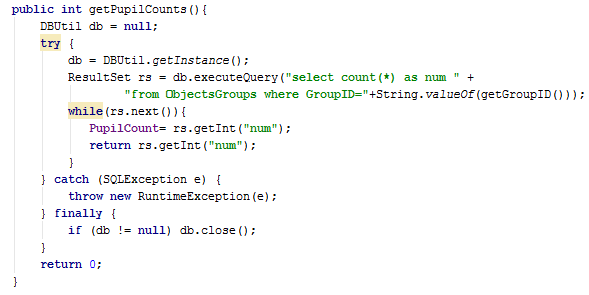


**Рисунок 14 – Метод получения группы учителя**

Где метод getName(int lang), возвращает фамилию и имя на русском языке. Данный запрос используется в методе getTeacherGroupID().

После получения id группы учителя, получение количества учеников производится с помощью таблицы Groups.

Метод получения количества учеников показан на рисунке 15.



**Рисунок 15 – Метод получения количества учеников**

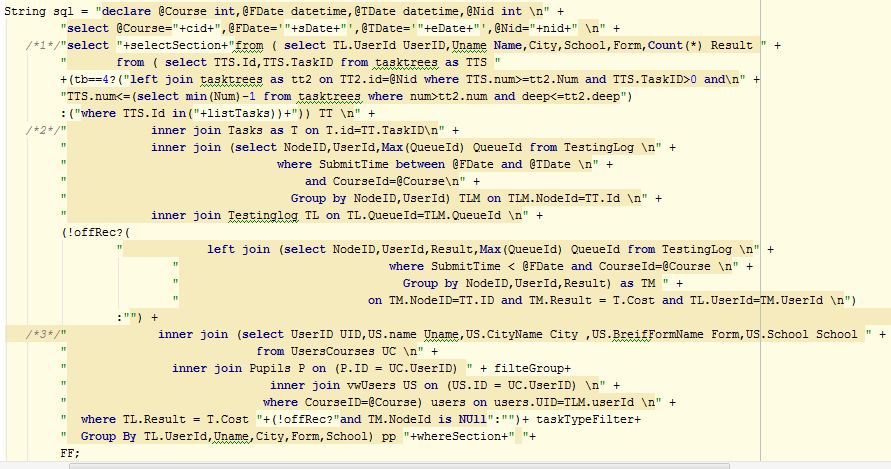
Где getGroupId() метод который возвращает id группы учителя. Данный запрос используется в методе getPupilCounts(). Это метод отличается от метода getPupilCount() тем что обращается к базе и загружает данные в объект Teacher, а этот метод просто возвращает число из объекта. Остальные запросы получения количества учеников по тем или иным параметрам, отличаются от данного запроса тем что перемножаются с нужными таблицами и выбираются по нужным критериям.

## 3.2 Интегрированная статистика

На странице teacherstable.jsp основная информация об учителях берется из класса Teachers.java, основная проблема была в получении количества решенных задач по курсам и промежутку времени.

Основой получения информации на данной странице является запрос в который подставляется строка selectSection и whereSection которые от указанных параметров меняет свое содержимое. Если присутствует параметр userId, выполняется отдельный запрос на получение статистики по ученику.

Сам запрос, а так же связи параметров и частей запроса показаны в таблице на рисунке 12 и таблице 12.



**Рисунок 12 – Расположение файла Teacher.java**

**Таблица 12 − Связи параметров и выполнения запроса**

|  |  |  |
| --- | --- | --- |
| Параметр | Select часть | Where часть |
| teachId==0 | GroupID,count(\*), sum(Result) | inner join ObjectsGroups on ObjectID=UserID and GroupID in("+tg+")  group by GroupID |
| teachId!=0 | \* | inner join ObjectsGroups on ObjectID=UserID and GroupID="+String.valueOf(teachId) |
| form==true | Form, count(\*) | inner join ObjectsGroups on ObjectID=UserID and GroupID="+String.valueOf(teachId)+ group by Form |
| school==true | \* | where School ='СШ 27' |

Основной запрос разделен на три части:

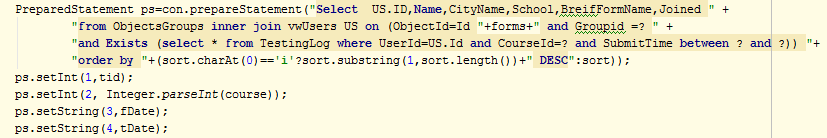
1. Получения списка задач по курсу и записям задач;
2. Получения всех решенных задач из логов тестирования вместе с id учеников;
3. Получение данных об учениках.

Для удобного изменения параметров на странице находятся ссылки которые делают переход на эту же страницу с нужными параметрами. Так же есть переходы на другие страницы.

## 3.3 Статистика по нескольким курсам

Вся информация на странице статистики по нескольким курсам, страница teachersCourses.jsp, берется из класса Teacher.java, кроме получения списка активных учеников за определенный промежуток времени. Активным учеником в данный период считается тот ученик, который отправил хотя бы одно решение на тестирование по заданному курсу.

Запрос на получение активных учеников показан на рисунке 17.



**Рисунок 17 – Запрос на получение активных учеников по курсу**

Где forms – строка, которая содержит id классов, по которым идет выборка.

Работу по загрузке учеников по каждому курсу выполняет класс Teacher.java, который загружает данные в массив учителей с помощью метода loadCountByCourses(Teacher[] teachers,String[] courses, String fDate, String tDate, Map courseForms), для более удобного построения страницы, так как все данные хранятся в объектах учителей.

Метод загрузки количества активных учеников по курсам показан на рисунке 18.



**Рисунок 18 – Метод получения активных учеников по курсу**

Курсы передаются в URL в виде строки id курсов разделенных запятыми. Для каждого могут указываться классы тоже в виде одной строки,

например: 15=6,5. В данной строке для 15 курса идет выборка по 6 и 7 классу. Данные о курсах и их классах заносятся в Map<String,String> courseForms где ключ курс, а значение строка с набором классов.

Так же как и в интегрированной статистике отображение информации зависит от параметров которые передаются в URL.

Связи параметров и отображения показаны в таблице 13.

**Таблица 13 − Связи параметров**

|  |  |  |
| --- | --- | --- |
| Параметр 1 | Параметр 2 | Отображение |
| tid==0 | - | Отображение информации по всем учителям и указанным курсам, количество активных и зарегистрированных учеников, количество школьников |
| tid!=0 | pupils==true | Списки активных учеников учителя по каждому курсу |
| tid!=0 | forms==true | Список количества зарегистрированных учеников учителя по классам |
| tid!=0 | register==true | Список количества зарегистрированных учеников учителя по годам и классам |
| tid!=0 | school==true | Список количества школьников учителя |
| tid!=0 | - | Список зарегистрированных учеников учителя |

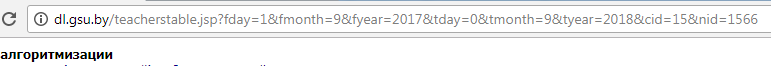
Для просмотра активных учеников по каждому курсу используется интегрированная статистика. По ссылке на количествах учеников уже используются готовые параметры для отображения нужной информации на странице. На данный момент цифры иногда могут различаться на этих страницах т.к. некоторые ученики могут отправить решение задачи по этому курсу которое не учитывается на странице интегрированной статистики, в будущем эту проблему надо решить.

# 4 Технологии использования

## 4.1 Интегрированная статистика

Интегрированная статистика позволяет смотреть информацию об учителях в разрезе количества учеников и их успеваемости. Получаемая информация строится на основе параметров переданных в URL.

Пример параметров URL показан на рисунке 19.



**Рисунок 19 – Пример URL с параметрами**

Список используемых параметров описано на странице teachersHelp.jsp по ссылке со страницы.

При загрузке страницы формируются ссылки с готовыми параметрами для изменения отображения страницы. Например если нажать на ссылку количества учеников, то в URL подставляется id группы учителя и формируется список учеников с количеством решенных задач данного учителя.

Примеры изменения отображения показаны на рисунках 20-21.



**Рисунок 20 – Страница teacherstable.jsp по всем учителям**



**Рисунок 21 – Страница teacherstable.jsp по указанному учителю**

Как видно из примера добавляется параметр tid, который отвечает за id группы учителя. Аналогично работают и остальные параметры.

Так же есть возможность выбрать классы, по которым будет делаться выборка. Классы указываются в параметре g в виде строки id классов, которые разделены запятыми. Например, строке g=10,7 соответствует сортировка по 11 и 8 классу.

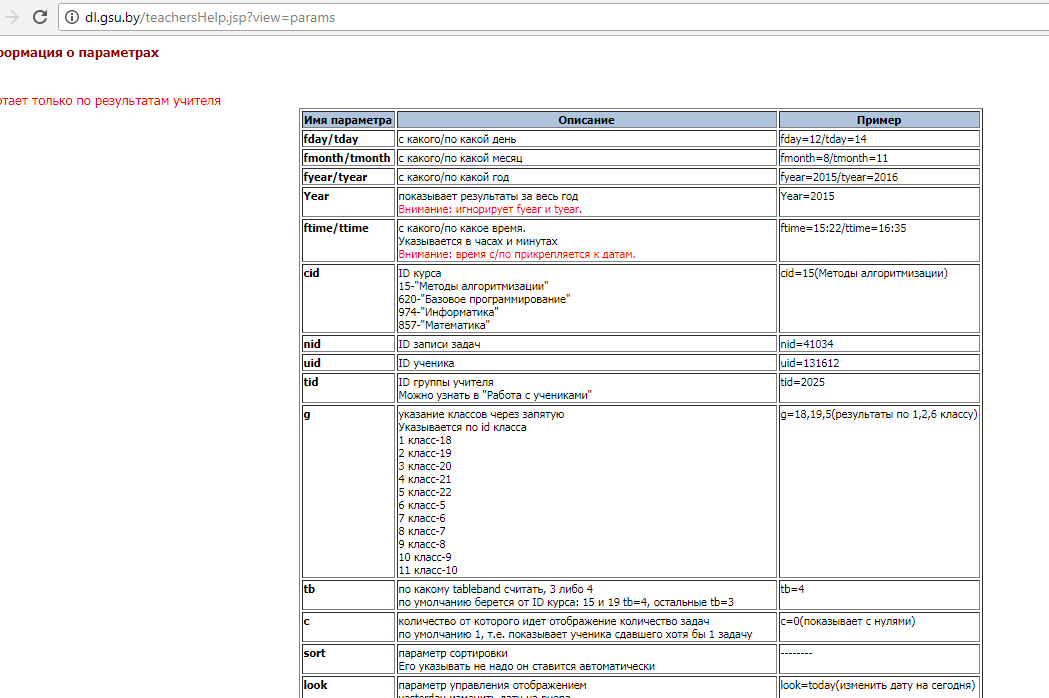
Пример фильтра по 11 и 8 классу показан на рисунке 22.



**Рисунок 22 – Использование фильтра по классам**

Чтобы не перегружать страницу интегрированной статистики лишней статической информацией, было решено перенести информацию о параметрах на отдельную страницу teachersInfo.jsp. Так же туда была перенесена информация разработки, которая содержит дальнейший план работ данного функционала. Перейти на страницу можно со страницы teacherstable.jsp по ссылкам “Параметры инфо” и “Инфо разработки”. На странице teachersHelp.jsp есть 2 вида отображения, которые задаются с помощью параметра view значениями params и future-history, которые отвечают за параметры и информацию о будущей разработке соответственно.

Страница teachersHelp.jsp с информацией о параметрах показана на рисунке 23.



**Рисунок 22 – Использование фильтра по классам**

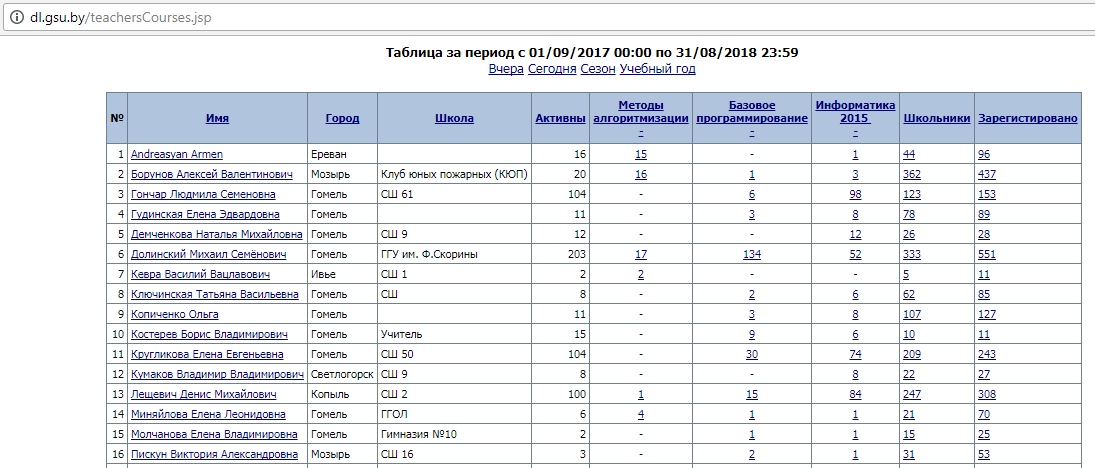
На данной странице также видно, по каким id класса указывать конкретные классы. Так же есть информация по часто используемым курсам.

## 4.2 Статистика по нескольким курсам

Страница статистики по нескольким курсам отображает информацию об активности учеников и количестве учеников учителя по нескольким курсам. Количество указанных курсов ограничивается максимальной длинной GET запроса так же необходимо учитывать остальные параметры. Использование параметров на странице статистики по нескольким курсам такое же, как и на странице интегрированной статистики, сама работа обработки некоторых параметров связанных с курсами отличается.

При открытии страницы без указания параметров по умолчанию подставляются 3 курса, и выберется промежуток времени за текущий учебный год. Параметры по умолчанию не будут отображены в строке URL после открытия, зато их можно будет увидеть при следующих обращениях к странице.

Страница статистики по курсам без указания параметров показана на рисунке 23.



**Рисунок 22 – Страница teachersCourses.jsp без указания параметров**

Как видно из рисунка параметры не показаны. На рисунке 24 показан URL, который был получен при обращении к странице. В URL будет текущий учебный год и 3 курса.



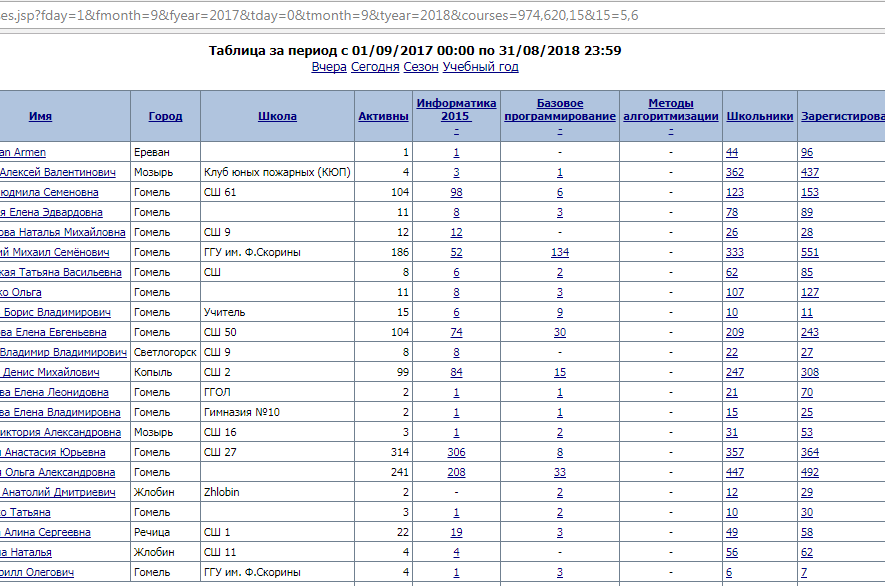
**Рисунок 23 – Полученный по умолчанию URL**

За выбранные курсы на странице отвечает параметр courses, курсы в нем указаны в виде строки, id курсов которые разделены запятыми. Последовательность отображения колонок курсов, такая же, как и в параметре courses.

Фильтр учеников по классам такой же, как и в интегрированной статистике, но указывать его нужно для определенного курса отдельно. Параметром фильтра определенного курса является его id. Например: 15=5,6

указывает, что для 15-го курса нужно делать выборку по 6 и 7-му классу.

Пример фильтра по 15-му курсу показан на рисунке 24.



**Рисунок 24 – Пример указания фильтра для курса**

Как видно из рисунка по 15 курсу нет учеников 6 и 7-го класса. Аналогично фильтры указываются и для остальных курсов.

# 5 Полученные результаты

## 5.1 Интегрированная статистика

Интегрированная статистика представляет собой информацию об успеваемости учеников в разрезе их учителей.

Страница интегрированной статистики показана на рисунке 25.



**Рисунок 25 – Страница teacherstable.jsp**

В правом верхнем углу расположены ссылки для помощи учителям. Ссылки над таблицей используются для управления параметрами страницы. Если нажать на ссылку «С незачтенными», страница покажет результаты, учитывая повторную отправку решений учениками.

Результат с неззачтенными решениями показан на рисунке 26.



**Рисунок 26 – Страница teacherstable.jsp с незачтенными решениями**

Как видно количество задач стало больше, а ссылка поменялась на противоположную. Также этот режим распространяется на результаты учеников, если перейти по ссылке количества учеников учителя.

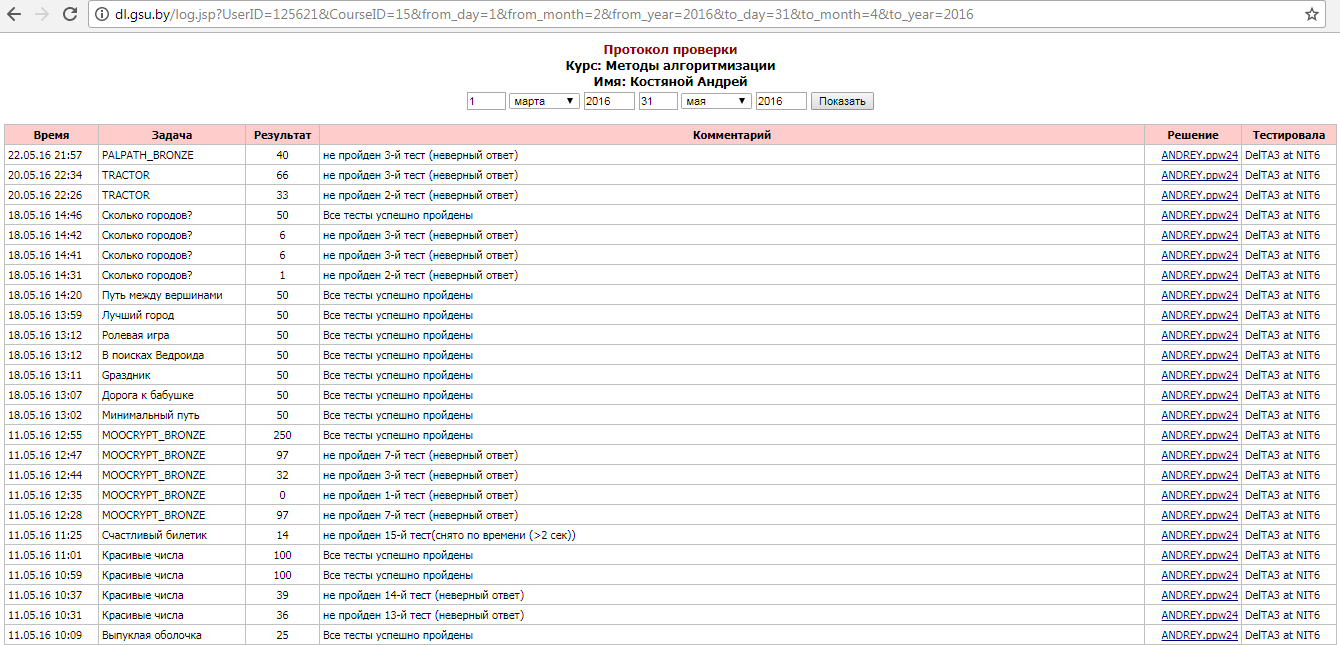
Список учеников учителя показан на рисунке 27.



**Рисунок 27 – Список успеваемости учеников учителя**

Как видно из рисунка, с права добавилась информация об учителе и над таблицей добавилась ссылка на просмотр количества учеников по классам, также добавились ссылки на лог учеников, который показывает отправляемые учеником решения.

Лог отправки учеником решений показан на рисунке 28.

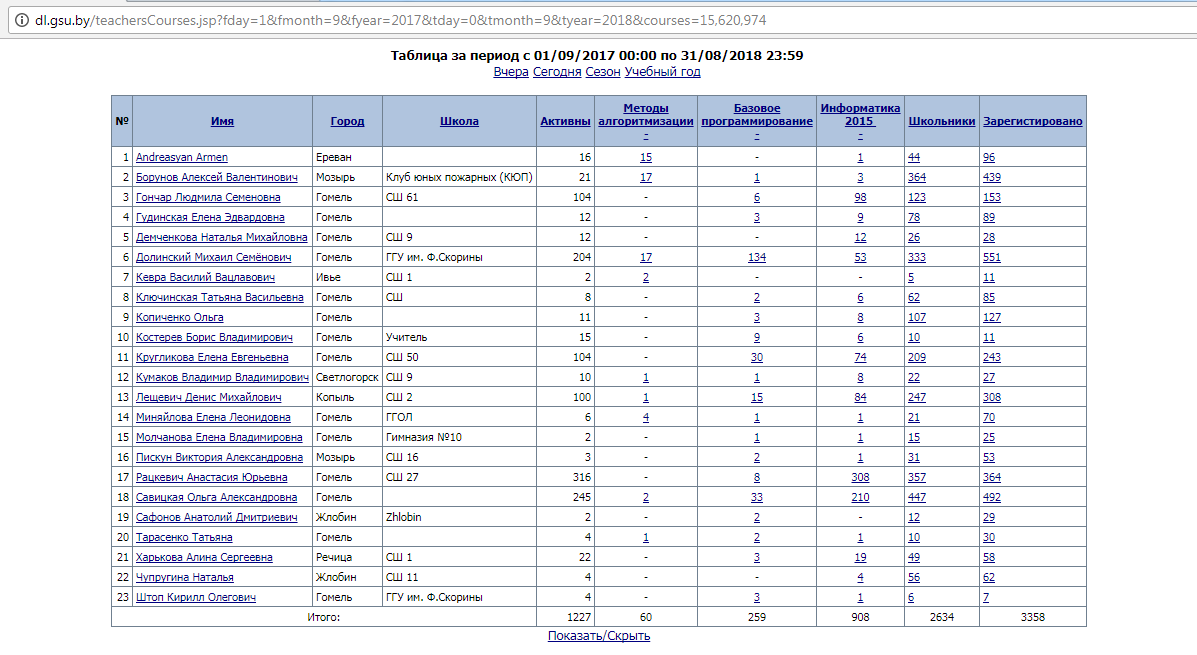


**Рисунок 28 – Лог отправляемых решений ученика**

## 5.2 Статистика по нескольким курсам

Самая объемная часть магистерской работы, была работа над статистикой по нескольким курсам, так как необходимо было учитывать затратность получения информации по одному курсу. Было принято решение получать информацию активности ученика в курсе, если ученик отправил хотя бы одно решение в конкретном курсе.

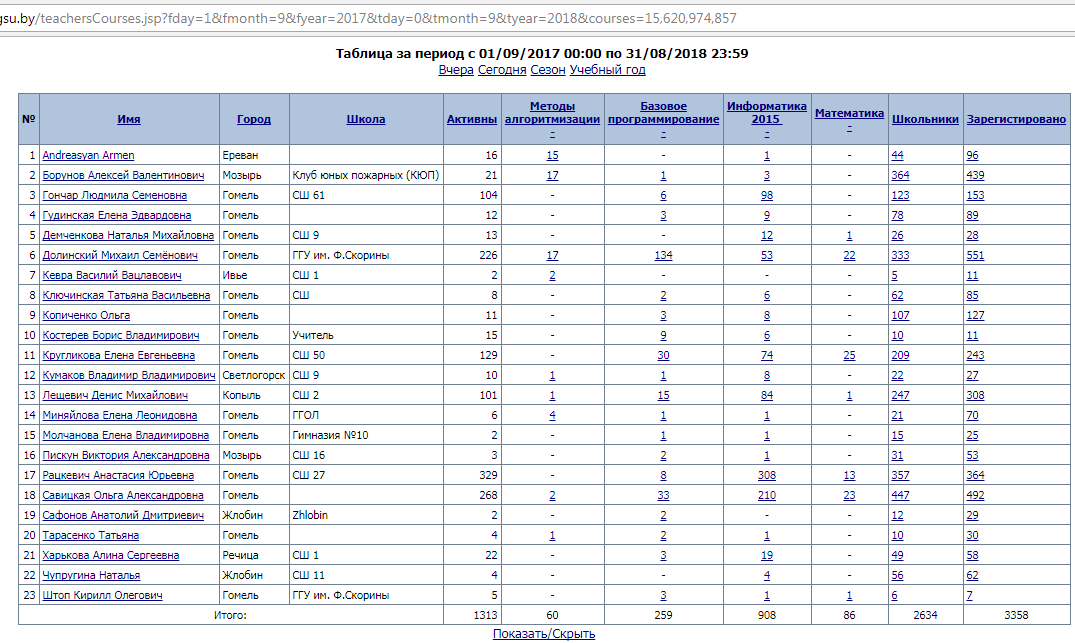
Страница статистики по нескольким курсам показа на рисунке 29.



**Рисунок 29 – Страница teachersCourses.jsp**

Как видно из рисунка информация на странице берется по 3-ем курсам, добавления и удаление курсов для получения информации задается параметром courses в котором указывается список всех требуемых курсов через запятую. Например, для добавления курса «Математика» необходимо добавить в параметр 857(id курса «Математика»).

Страница с курсом «Математика» показана на рисунке 30.



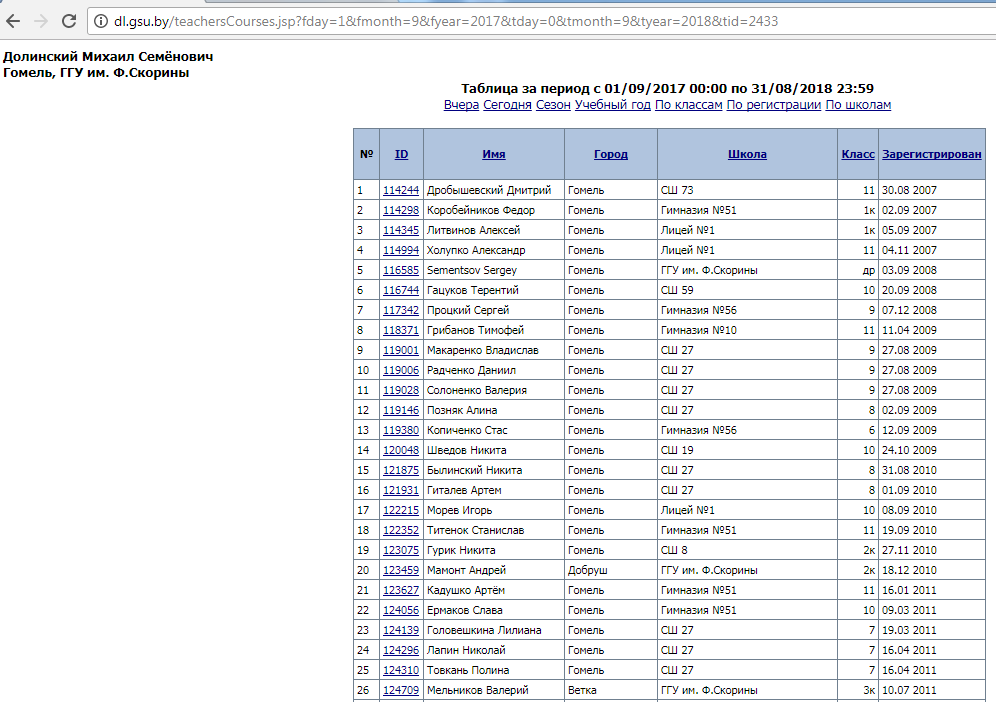
**Рисунок 30 – Страница teachersCourses.jsp**

Как видно добавилась колонка с новым курсом, порядок колонок курсов соответствует порядку в параметре courses.

При нажатии на количество учеников в колонке курса, произойдет перенаправление на страницу интегрированной статистики учителя с заданными параметрами курса по незачтенным параметрам.

Ссылки по количеству учеников из колонок школьники и зарегистрировано, показывают списки школьников и количество всех зарегистрированных учеников учителя соответственно.

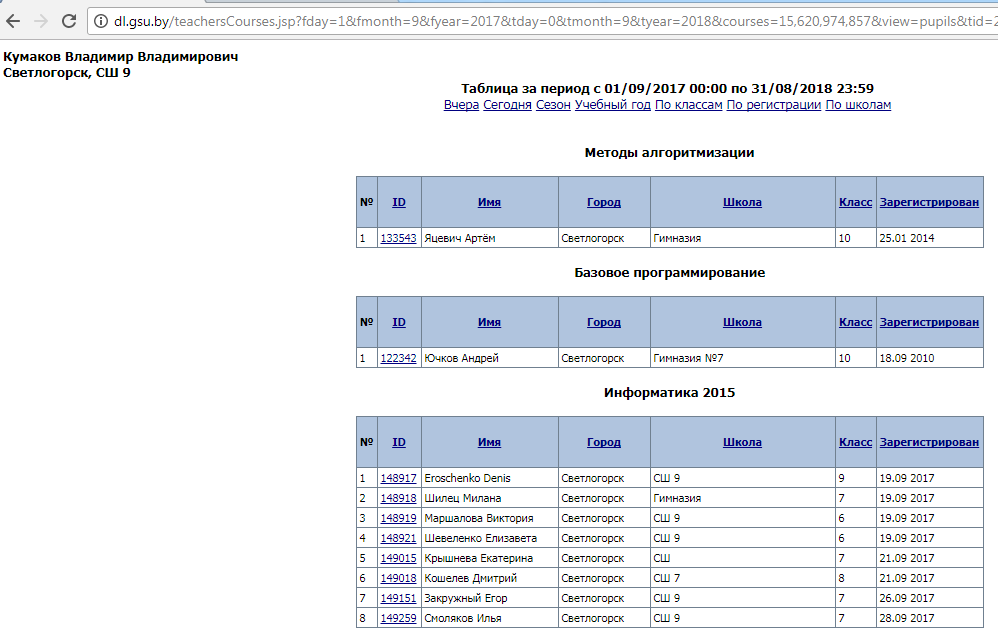
Список зарегистрированных учеников показан на рисунке 31.



**Рисунок 31 – Список зарегистрированных учеников учителя**

Со ссылки имени учителя, показываются списки всех активных учеников в таблицах выбранных курсов.

Список активных учеников по курсам показан на рисунке 32.



**Рисунок 31 – Список активных учеников учителя по выбранным курсам**

Кликнув по ссылке «По регистрации», можно будет показана таблица количества зарегистрированных учеников по классам в разрезе года их регистрации.

Таблица регистрации учеников по годам показана на рисунке 33.

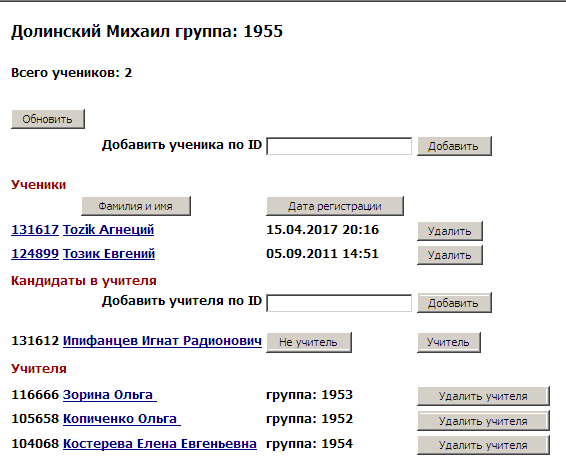


**Рисунок 32 – Таблица регистрации учеников по годам**

## 5.3 Редактирование учеников учителя

Для работы с учениками есть страница teachersGroups.jsp, на ней можно добавлять учеников по id, а так же увидеть список всех учеников с их id. У администратора еще есть возможность видеть списки кандидатов, добавлять учителей по id, удалять учителей и видеть список текущих учителей.

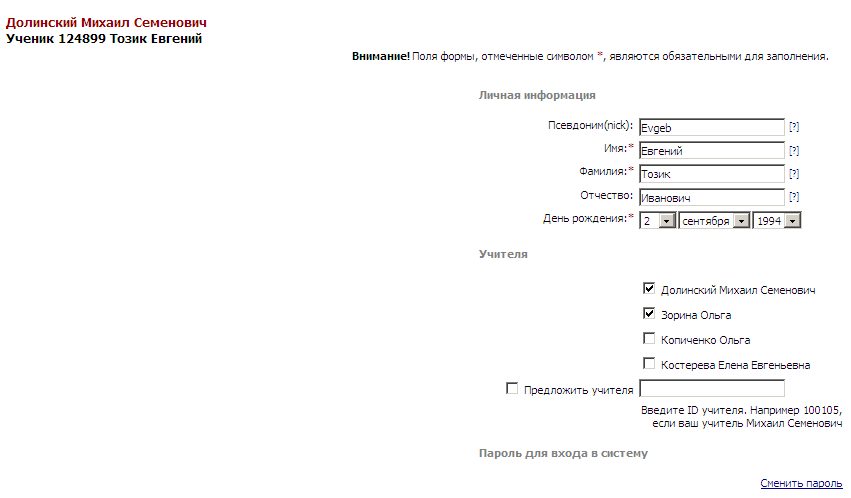
Страница работы с группой учителя показана на рисунке 34.



**Рисунок 34 – Страница teachersCourses.jsp**

Для редактирования информации ученика необходимо нажать на его id, после чего произойдет перенаправление на страницу teachersEdit.jsp где можно будет редактировать информацию об ученике.

Страница редактирования ученика показана на рисунке 35.



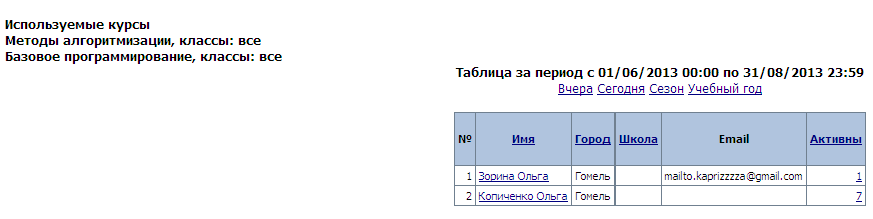
**Рисунок 35 – Страница редактирования ученика**

Логика страницы редактирования позаимствована со страницы регистрации сайта, немного изменив логику для работы учителя.

## 5.4 Таблица учителей

Таблица учителей находится на странице teachersInfo.jsp, логика страницы полностью взята со страницы статистики по нескольким курсам, кроме отображения. Отображается только информация об учителе и количестве его активных учеников. Страница доступна только администратору и служит в основном для формирования отчетов.

Страница таблицы учителей показана на рисунке 36.



**Рисунок 35 – Страница таблицы учителей**

# Заключение

Тема данной дипломной работы является весьма актуальной, т.к. она направлена на повышение уровня работоспособности системы дистанционного обучения DL.GSU.BY.

В ходе выполнения магистерской работы были добавлены следующие возможности:

* Просмотр ресзультатов интегрированной статистики с незачтенными задачами;
* Добавлена статистика активности по нескольким курсам;
* Добавлена возможность редактирования учеников учителями;
* Добавлена возможность просмотра протокола тестирования ученика со страницы интегрированной статистики;
* Добавлен и улучшен функционал для контроля учеников.

# Список использованных источников

1. Форум программистов [Электронный ресурс] // URL: <http://programmersforum.ru> (дата обращения: 12.10.2016).
2. Htmlbook.ru [Электронный ресурс] // URL: https:// Htmlbook.ru/ (дата обращения: 17.05.2017).
3. Шилдт, Г. Java. Полное руководство, 8-е издание [Текст] // Г. Шилдт. – СПб.: Вильямс, 2012. – 1104 с.
4. Блинов, И.Н. Java. Промышленное программирование [Текст] // И.Н. Блинов, В.С. Романчик. – Мн.: Универсал Пресс, 2007. – 124 с.
5. Эккель, Б. Философия Java. Библиотека программиста. 4-ое изд. [Текст] // Б. Эккель. – СПб.: Питер, 2009 – 640с.

Приложение А