АНОТАЦІЯ

Документ 1116130.00959-01 12 01 «Підсистема напівавтоматичного формування розкладу занять університету». Текст програми. Входить до складу програмної документації на програму, що реалізує підсистему напівавтоматичного формування розкладу занять університету.

# У даному документі представлений текст програми, написаної на мові java під платформу Java SE Runtime Environment 8. Об'єм пам'яті, що потребує програма складає 96 Мб оперативної пам'яті та 12.2 Мб дискового простору. В якості операційної системи дозволяється використовувати будь-яку систему під яку існує реалізація платформи Java SE Runtime Environment 8.

Програма розроблена в середовищі NetBeans IDE 8.1. В якості СУБД використовується Microsoft SQL Server 2012.

ЗМІСТ

[2 Схема взаємодії програмних модулів 5](#_Toc454789965)

[3 Текст програми 6](#_Toc454789966)

[3.1 Текст к модуля main 6](#_Toc454789967)

[3.1.1 Файл Main.java 6](#_Toc454789968)

[3.2 Текст модуля DatabaseTools: 6](#_Toc454789969)

[3.2.1 Файл DBTools.java 6](#_Toc454789970)

[3.2.2 Файл JDBCController.java 7](#_Toc454789971)

[3.3 Текст модуля DatabaseEntities 8](#_Toc454789972)

[3.3.1 Файл Class.java 8](#_Toc454789973)

[3.3.2 Файл Department.java 8](#_Toc454789974)

[3.3.3 Файл Faculty.java 9](#_Toc454789975)

[3.3.4 Файл GroupStudent.java 9](#_Toc454789976)

[3.3.5 Файл LoadDepartment.java 10](#_Toc454789977)

[3.3.6 Файл LoadDepartmentDoc.java 11](#_Toc454789978)

[3.3.7 Файл Requirements.java 12](#_Toc454789979)

[3.3.8 Файл RequirementsUniversityTeacher.java 12](#_Toc454789980)

[3.3.9 Файл Schedule.java 14](#_Toc454789981)

[3.3.10 Файл ScheduleClasses.java 14](#_Toc454789982)

[3.3.11 Файл Speciality.java 15](#_Toc454789983)

[3.3.12 Файл Stream.java 15](#_Toc454789984)

[3.3.13 Файл Subject.java 16](#_Toc454789985)

[3.3.14 Файл TimeClass.java 16](#_Toc454789986)

[3.3.15 Файл TypeClass.java 17](#_Toc454789987)

[3.3.16 Файл TypeWeek.java 17](#_Toc454789988)

[3.3.17 Файл UniversityTeacher.java 18](#_Toc454789989)

[3.3.18 Файл WeekDay.java 18](#_Toc454789990)

[3.4 Текст модуля DataAccessObject 19](#_Toc454789991)

[3.4.1 Файл ClassDAO.java 19](#_Toc454789992)

[3.4.2 Файл DepartmentDAO.java 21](#_Toc454789993)

[3.4.3 Файл FacultyDAO.java 22](#_Toc454789994)

[3.4.4 Файл GroupStudentDAO.java 23](#_Toc454789995)

[3.4.5 Файл LoadDepartmentDAO.java 24](#_Toc454789996)

[3.4.6 Файл LoadDepartmentDocDAO.java 26](#_Toc454789997)

[3.4.7 Файл RequirementsDAO.java 27](#_Toc454789998)

[3.4.8 Файл RequirementsUniversityTeacherDAO.java 28](#_Toc454789999)

[3.4.9 Файл ScheduleClassesDAO.java 30](#_Toc454790000)

[3.4.10 Файл ScheduleDAO.java 31](#_Toc454790001)

[3.4.11 Файл SpecialityDAO.java 34](#_Toc454790002)

[3.4.12 Файл StreamDAO.java 35](#_Toc454790003)

[3.4.13 Файл SubjectDAO.java 37](#_Toc454790004)

[3.4.14 Файл TimeClassDAO.java 37](#_Toc454790005)

[3.4.15 Файл TypeClassDAO.java 39](#_Toc454790006)

[3.4.16 Файл TypeWeekDAO.java 40](#_Toc454790007)

[3.4.17 Файл UniversityTeacherDAO.java 41](#_Toc454790008)

[3.4.18 Файл WeekDayDAO.java 42](#_Toc454790009)

[3.5 Текст модуля FormationOfShedule 43](#_Toc454790010)

[3.5.1 Файл FormationSchedule.java 43](#_Toc454790011)

[3.6 Текст модуля UI 47](#_Toc454790012)

[3.6.1 Файл AddNewRequirementForm.java 47](#_Toc454790013)

[3.6.2 Файл LoginForm.java 55](#_Toc454790014)

[3.6.3 Файл MainForm.java 58](#_Toc454790015)

[3.6.4 Файл RequirementsUniversityTeacherForm.java 60](#_Toc454790016)

[3.6.5 Файл ScheduleGenerateForm.java 68](#_Toc454790017)

[3.6.6 Файл ScheduleTableForm.java 71](#_Toc454790018)

# СХЕМА ВЗАЄМОДІЇ ПРОГРАМНИХ МОДУЛІВ

Програма складажться с заступних модулів

* модуля Main, який служить точкою входу у програму
* модуля DatabaseTools, який служить для встановлення підключення драйверу JDBC та уставновлення з’єднання з базою даних.
* модуля DatabaseEntities, який служить для відображення всіх сутностей з бази даних
* модуля DataAccessObject який надає абстрактний інтерфейс до будь якої сутності бази даних
* модуля FormationOfShedule, який реалізує алгоритм формування розкладу;
* модуля UI який який служить для відображення користувацького інтерфейсу.

На рис. 1.1 представлена схема взаємодії модулів програми і зв’язки модулів між собою.

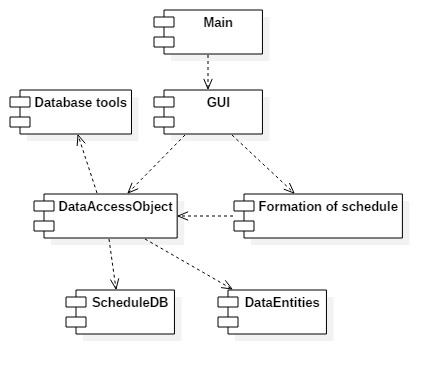


Рисунок 1.1 – Схема взаємодії модулів програми

# ТЕКСТ ПРОГРАМИ

## Текст к модуля main

### Файл Main.java

package Main;

import UI.LoginForm;

/\*\*

\* Класс вхождения в программу

\* Created by Dmitriy on 29.05.2016.

\*/

public class Main {

public static void main(String [] args){

LoginForm.main();

}

}

## Текст модуля DatabaseTools:

### Файл DBTools.java

package DatabaseTools;

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.SQLException;

/\*\*

\* <h2>Класс служащий для подключения к базе данных и получения connection.</h2>

\* Может выполнять подключение в 2-х режимах: <br>

\* <ol>

\* <li>Режим отладки (удобен во время написания и тестировки программы).

\* В этом режиме используются имя пользователя и пароль прописанные по умолчанию.

\* (для возможности использования этого метода необходимо знать пароль разработчика)</li>

\* <li>Режим 'пользовательский' имя пользователя и пароль должны вводиться пользователем приложения.</li>

\* </ol>

\*

\* Created by Dmitriy on 30.05.2016.

\*/

public class DBTools {

private static final String DB\_HOST = "jdbc:sqlserver://127.0.0.1:51003; databaseName = ScheduleDB;";

private static final String DB\_USER = "admin";

private static final String DB\_PASSWORD = "I951753i";

private static Connection dbConnection;

/\*\*

\* Устанавливаем коннект к бд со значениями по умолчанию.

\* @param SuperUserPassword ключ разработчика необходимый для выполнения данного метода (повышение безопасности).

\* @return check true/false - при успешном/неуспешном подключении к БД с предъявленными параметрами.

\*/

public static boolean connectToDatabase(String SuperUserPassword){

boolean check = false;

if(SuperUserPassword == "OpenDBSecurity\_22041994") {

try {

System.out.println("---------------------------------------------------------------------------------");

dbConnection = DriverManager.getConnection(DB\_HOST, DB\_USER, DB\_PASSWORD);

System.out.println("БД подключена с параметрами по умолчанию: ");

System.out.println("host: \t\t" + DB\_HOST);

System.out.println("username: \t" + DB\_USER);

System.out.println("password: \t" + DB\_PASSWORD);

System.out.println("---------------------------------------------------------------------------------");

check = true;

} catch (SQLException e) {

System.out.println("Method DBTools(String dbUser, String dpPassword)");

e.printStackTrace();

}

}else {

System.out.println("Некорректный пароль...");

}

return check;

}

/\*\*

\* Устанавливаем коннект к бд по используя переданные параметры: имя пользователя и пароль.

\* @param dbUser имя пользователя БД.

\* @param dpPassword пароль пользователя бд.

\* @return check true/false - при успешном/неуспешном подключении к БД с предъявленными параметрами.

\*/

public static boolean connectToDatabase(String dbUser, String dpPassword){

boolean check = false;

try {

System.out.println("-------------------------------------------------------------------------------------");

dbConnection = DriverManager.getConnection(DB\_HOST,dbUser,dpPassword);

System.out.println("БД подключена с пользовательскими параметрами: ");

System.out.println("host: \t\t" + DB\_HOST);

System.out.println("username: \t" + dbUser);

System.out.println("password: \t"+ dpPassword);

System.out.println("-------------------------------------------------------------------------------------");

check = true;

} catch (SQLException e) {

System.out.println("Method DBTools(String dbUser, String dpPassword)");

e.printStackTrace();

}

return check;

}

/\*\*

\* Возвращает текущее подключение к бд для этого объекта.

\* @return dbConnection - подключение для работы с ним в другом месте (может понадобится ^\_^)

\*/

public static Connection getConnection(){

try {

if(!dbConnection.isClosed())

return dbConnection;

} catch (SQLException e) {

System.out.println("Соединения ещё не установлено =,(");

System.out.println("Method getConnection()");

e.printStackTrace();

}

return null;

}

/\*\*

\* Закрываем соединение с БД.

\* @return true/false - при успешной/неуспешной операции закрытия соединения.

\*/

public static boolean closeConnection(){

boolean check = false;

try {

dbConnection.close();

if (dbConnection.isClosed()) {

System.out.println("Соединение успешно закрыто...");

check = true;

}

} catch (SQLException ex) {

System.out.println("Method closeConnection()");

ex.printStackTrace();

}

return check;

}

}

### Файл JDBCController.java

package DatabaseTools;

/\*\*

\* Created by Dmitriy on 29.05.2016.

\*/

public class JDBCController {

private static final String MSSQL\_DB\_DRIVER =

"com.microsoft.sqlserver.jdbc.SQLServerDriver";

public static boolean connectDriverToApplication(){

boolean check = false;

try{

java.lang.Class.forName(MSSQL\_DB\_DRIVER);

check = true;

} catch (ClassNotFoundException e) {

System.out.println("Не обнаруживает MsSQL JDBC Driver");

System.out.println(e.getMessage());

}

return check;

}

}

## Текст модуля DatabaseEntities

### Файл Class.java

package DatabaseEntities;

import java.sql.Time;

/\*\*

\* Created by Dmitriy on 13.06.2016.

\*/

public class Class {

int id;

TypeClass typeClass;

TimeClass timeClass;

Stream stream;

Subject subject;

UniversityTeacher universityTeacher;

public void setId(int id) {this.id = id;}

public void setTypeClass(TypeClass typeClass) {this.typeClass = typeClass;}

public void setTimeClass(TimeClass timeClass) {this.timeClass = timeClass;}

public void setStream(Stream stream) {this.stream = stream;}

public void setSubject(Subject subject) {this.subject = subject;}

public void setUniversityTeacher(UniversityTeacher universityTeacher) {this.universityTeacher = universityTeacher;}

public int getId() {return id;}

public TypeClass getTypeClass() {return typeClass;}

public TimeClass getTimeClass() {return timeClass;}

public Stream getStream() {return stream;}

public Subject getSubject() {return subject;}

public UniversityTeacher getUniversityTeacher() {return universityTeacher;}

@Override

public String toString() {

return "Class{" +

"\n id: " + id +

",\n typeClass: " + typeClass +

",\n timeClass: " + timeClass +

",\n stream: " + stream +

",\n subject: " + subject +

",\n universityTeacher: " + universityTeacher +

'}';

}

}

### Файл Department.java

package DatabaseEntities;

import java.sql.Date;

/\*\*

\* Класс описывающий сущность Кафедра

\* Created by Dmitriy on 10.06.2016.

\*/

public class Department {

int id;

String name;

String abbreviation;

int number;

Faculty faculty;

boolean isArchive;

Date archiveDate;

public void setId(int id) {this.id = id;}

public void setName(String name) {this.name = name;}

public void setAbbreviation(String abbreviation) {this.abbreviation = abbreviation;}

public void setNumber(int number) {this.number = number;}

public void setFaculty(Faculty faculty) {this.faculty = faculty;}

public void setIsArchive(boolean archive) {isArchive = archive;}

public void setArchiveDate(Date archiveDate) {this.archiveDate = archiveDate;}

public int getId() {return id;}

public String getName() {return name;}

public String getAbbreviation() {return abbreviation;}

public int getNumber() {return number;}

public Faculty getFaculty() {return faculty;}

public boolean getIsArchive() {return isArchive;}

public Date getArchiveDate() {return archiveDate;}

@Override

public String toString() {

return "Department{" +

"id: " + id +

", name: '" + name + '\'' +

", abbreviation: '" + abbreviation + '\'' +

", number: " + number +

", faculty: " + faculty +

", isArchive: " + isArchive +

", archiveDate: " + archiveDate +

'}';

}

}

### Файл Faculty.java

package DatabaseEntities;

import java.sql.Date;

/\*\*

\* Класс описывающий сущность Факультет

\* Created by Dmitriy on 10.06.2016.

\*/

public class Faculty {

int id;

String name;

String abbreviation;

int number;

boolean isArchive;

Date archiveDate;

public void setId(int id) {this.id = id;}

public void setName(String name) {this.name = name;}

public void setAbbreviation(String abbreviation) {this.abbreviation = abbreviation;}

public void setNumber(int number) {this.number = number;}

public void setIsArchive(boolean archive) {isArchive = archive;}

public void setArchiveDate(Date archiveDate) {this.archiveDate = archiveDate;}

public int getId() {return id;}

public String getName() {return name;}

public String getAbbreviation() {return abbreviation;}

public int getNumber() {return number;}

public boolean getIsArchive() {return isArchive;}

public Date getArchiveDate() {return archiveDate;}

@Override

public String toString() {

return "Faculty{" +

"id: " + id +

", name:'" + name + '\'' +

", abbreviation: '" + abbreviation + '\'' +

", number: " + number +

", isArchive: " + isArchive +

", archiveDate: " + archiveDate +

'}';

}

}

### Файл GroupStudent.java

package DatabaseEntities;

import java.sql.Date;

import java.util.List;

/\*\*

\* Created by Dmitriy on 13.06.2016.

\*/

public class GroupStudent {

int id;

String name;

int countStudent;

Speciality speciality;

int course;

boolean isArchive;

Date archiveDate;

public void setId(int id) {this.id = id;}

public void setName(String name) {this.name = name;}

public void setCountStudent(int countStudent) {this.countStudent = countStudent;}

public void setSpeciality(Speciality speciality) {this.speciality = speciality;}

public void setCourse(int course) {this.course = course;}

public void setIsArchive(boolean archive) {isArchive = archive;}

public void setArchiveDate(Date archiveDate) {this.archiveDate = archiveDate;}

public int getId() {return id;}

public String getName() {return name;}

public int getCountStudent() {return countStudent;}

public Speciality getSpeciality() {return speciality;}

public int getCourse() {return course;}

public boolean isArchive() {return isArchive;}

public Date getArchiveDate() {return archiveDate;}

@Override

public String toString() {

return "\n GroupStudent{" +

" id: " + id +

", name: '" + name + '\'' +

", countStudent: " + countStudent +

", speciality: " + speciality +

", course: " + course +

", isArchive: " + isArchive +

", archiveDate: " + archiveDate +

'}';

}

}

### Файл LoadDepartment.java

package DatabaseEntities;

import java.util.ArrayList;

import java.util.List;

/\*\*

\* Created by Dmitriy on 13.06.2016.

\*/

public class LoadDepartment {

int id;

LoadDepartmentDoc loadDepartmentDoc;

Subject subject;

boolean isHalfSemester;

GroupStudent groupStudent;

int countHourLecture;

int countHourLabWork;

int countHoutPracticWork;

UniversityTeacher teacherLecture;

UniversityTeacher teacherLabWork;

UniversityTeacher teacherPracticWork;

UniversityTeacher teacherLabWorkSecond;

UniversityTeacher teacherPracticWorkSecond;

public void setId(int id)

{this.id = id;}

public void setLoadDepartmentDoc(LoadDepartmentDoc loadDepartmentDoc)

{this.loadDepartmentDoc = loadDepartmentDoc;}

public void setSubject(Subject subject)

{this.subject = subject;}

public void setIsHalfSemester(boolean halfSemester)

{isHalfSemester = halfSemester;}

public void setGroupStudent(GroupStudent groupStudent)

{this.groupStudent = groupStudent;}

public void setCountHourLecture(int countHourLecture)

{this.countHourLecture = countHourLecture;}

public void setCountHourLabWork(int countHourLabWork)

{this.countHourLabWork = countHourLabWork;}

public void setCountHoutPracticWork(int countHoutPracticWork)

{this.countHoutPracticWork = countHoutPracticWork;}

public void setTeacherLecture(UniversityTeacher teacherLecture)

{this.teacherLecture = teacherLecture;}

public void setTeacherLabWork(UniversityTeacher teacherLabWork)

{this.teacherLabWork = teacherLabWork;}

public void setTeacherPracticWork(UniversityTeacher teacherPracticWork)

{this.teacherPracticWork = teacherPracticWork;}

public void setTeacherLabWorkSecond(UniversityTeacher teacherLabWorkSecond)

{this.teacherLabWorkSecond = teacherLabWorkSecond;}

public void setTeacherPracticWorkSecond(UniversityTeacher teacherPracticWorkSecond)

{this.teacherPracticWorkSecond = teacherPracticWorkSecond;}

public int getId() {return id;}

public LoadDepartmentDoc getLoadDepartmentDoc() {return loadDepartmentDoc;}

public Subject getSubject() {return subject;}

public boolean getIsHalfSemester() {return isHalfSemester;}

public GroupStudent getGroupStudent() {return groupStudent;}

public int getCountHourLecture() {return countHourLecture;}

public int getCountHourLabWork() {return countHourLabWork;}

public int getCountHoutPracticWork() {return countHoutPracticWork;}

public UniversityTeacher getTeacherLecture() {return teacherLecture;}

public UniversityTeacher getTeacherLabWork() {return teacherLabWork;}

public UniversityTeacher getTeacherPracticWork() {return teacherPracticWork;}

public UniversityTeacher getTeacherLabWorkSecond() {return teacherLabWorkSecond;}

public UniversityTeacher getTeacherPracticWorkSecond() {return teacherPracticWorkSecond;}

/\*\*

\* Метод возвращающий перечень всех дисциплин которые есть в наборе LoadDepartment

\* @param loadDepartments записи из одной предполагаем нагрузки по кафедре

\* @return список дисциплин которые есть в наборе LoadDepartment

\*/

public static List<Subject> getAllSubjects(List <LoadDepartment> loadDepartments){

List <Subject> result = new ArrayList<Subject>();

if(loadDepartments.size() > 0){

int previousSubject = -1;

int currentSubject;

for(LoadDepartment ld: loadDepartments){

currentSubject = ld.getSubject().getId();

if(currentSubject != previousSubject){

result.add(ld.getSubject());

previousSubject = currentSubject;

}

}

}

return result;

}

@Override

public String toString() {

return "\nLoadDepartment{" +

"\n id: \t" + id +

",\n loadDepartmentDoc: \t" + loadDepartmentDoc +

",\n subject: \t" + subject +

",\n isHalfSemester: \t" + isHalfSemester +

",\n groupStudent: \t" + groupStudent +

",\n countHourLecture: \t" + countHourLecture +

",\n countHourLabWork: \t" + countHourLabWork +

",\n countHoutPracticWork: \t" + countHoutPracticWork +

",\n teacherLecture: \t" + teacherLecture +

",\n teacherLabWork: \t" + teacherLabWork +

",\n teacherPracticWork: \t" + teacherPracticWork +

",\n teacherLabWorkSecond: \t" + teacherLabWorkSecond +

",\n teacherPracticWorkSecond: \t" + teacherPracticWorkSecond +

'}';

}

}

### Файл LoadDepartmentDoc.java

package DatabaseEntities;

import java.sql.Date;

/\*\*

\* Created by Dmitriy on 13.06.2016.

\*/

public class LoadDepartmentDoc {

int id;

int yearReception;

int semester;

Department department;

boolean isArchive;

Date archiveDate;

public void setId(int id) {this.id = id;}

public void setYearReception(int yearReception) {this.yearReception = yearReception;}

public void setSemester(int semester) {this.semester = semester;}

public void setDepartment(Department department) {this.department = department;}

public void setIsArchive(boolean archive) {isArchive = archive;}

public void setArchiveDate(Date archiveDate) {this.archiveDate = archiveDate;}

public int getId() {return id;}

public int getYearReception() {return yearReception;}

public int getSemester() {return semester;}

public Department getDepartment() {return department;}

public boolean getIsArchive() {return isArchive;}

public Date getArchiveDate() {return archiveDate;}

@Override

public String toString() {

return "LoadDepartmentDoc{" +

" id: " + id +

", yearReception: " + yearReception +

", semester: " + semester +

", department: " + department +

", isArchive: " + isArchive +

", archiveDate: " + archiveDate +

'}';

}

}

### Файл Requirements.java

package DatabaseEntities;

/\*\*

\* Created by Dmitriy on 12.06.2016.

\*/

public class Requirements {

int id;

String name;

public void setId(int id) {this.id = id;}

public void setName(String name) {this.name = name;}

public int getId() {return id;}

public String getName() {return name;}

@Override

public String toString() {

return "Requirements{" +

"id: " + id +

", name: " + name +

'}';

}

}

### Файл RequirementsUniversityTeacher.java

package DatabaseEntities;

import java.sql.Date;

import java.util.ArrayList;

import java.util.List;

/\*\*

\* Created by Dmitriy on 12.06.2016.

\*/

public class RequirementsUniversityTeacher {

int id;

UniversityTeacher universityTeacher;

int yearReception;

int semester;

WeekDay weekDay;

int numberClass;

Requirements requirements;

boolean isArchive;

Date archiveDate;

public void setId(int id) {this.id = id;}

public void setUniversityTeacher(UniversityTeacher universityTeacher) {this.universityTeacher = universityTeacher;}

public void setYearReception(int yearReception) {this.yearReception = yearReception;}

public void setSemester(int semester) {this.semester = semester;}

public void setWeekDay(WeekDay weekDay) {this.weekDay = weekDay;}

public void setNumberClass(int numberClass) {this.numberClass = numberClass;}

public void setRequirements(Requirements requirements) {this.requirements = requirements;}

public void setIsArchive(boolean archive) {isArchive = archive;}

public void setArchiveDate(Date archiveDate) {this.archiveDate = archiveDate;}

public int getId() {return id;}

public UniversityTeacher getUniversityTeacher() {return universityTeacher;}

public int getYearReception() {return yearReception;}

public int getSemester() {return semester;}

public WeekDay getWeekDay() {return weekDay;}

public int getNumberClass() {return numberClass;}

public Requirements getRequirements() {return requirements;}

public boolean getIsArchive() {return isArchive;}

public Date getArchiveDate() {return archiveDate;}

public static List<RequirementsUniversityTeacher> returnOnlyWithThisTeacher(

List <RequirementsUniversityTeacher> rutList, UniversityTeacher ut){

RequirementsUniversityTeacher rut = new RequirementsUniversityTeacher();

List <RequirementsUniversityTeacher> rutResultList = new ArrayList<RequirementsUniversityTeacher>();

for(int indx = 0; indx < rutList.size(); indx++){

rut = rutList.get(indx);

if(rut.getUniversityTeacher().getAllTeacherName().equals(ut.getAllTeacherName())){

rutResultList.add(rut);

}

}

return rutResultList;

}

public static List<RequirementsUniversityTeacher> returnWithBoolFilter(

List <RequirementsUniversityTeacher> rutList, boolean check){

RequirementsUniversityTeacher rut = new RequirementsUniversityTeacher();

List <RequirementsUniversityTeacher> rutResultList = new ArrayList<RequirementsUniversityTeacher>();

for(int indx = 0; indx < rutList.size(); indx++){

rut = rutList.get(indx);

if(rut.getIsArchive() == check){

rutResultList.add(rut);

}

}

return rutResultList;

}

public static List<RequirementsUniversityTeacher> returnListOnlyWithThisYear(

List <RequirementsUniversityTeacher> rutList, int year){

RequirementsUniversityTeacher rut = new RequirementsUniversityTeacher();

List <RequirementsUniversityTeacher> rutResultList = new ArrayList<RequirementsUniversityTeacher>();

for(int indx = 0; indx < rutList.size(); indx++){

rut = rutList.get(indx);

if(rut.getYearReception() == year){

rutResultList.add(rut);

}

}

return rutResultList;

}

@Override

public String toString() {

return "RequirementsUniversityTeacher{" +

"\n id: " + id +

",\n universityTeacher: " + universityTeacher +

",\n yearReception: " + yearReception +

",\n semester: " + semester +

",\n weekDay: " + weekDay +

",\n numberClass: " + numberClass +

",\n requirements: " + requirements +

",\n isArchive: " + isArchive +

",\n archiveDate: " + archiveDate +

'}';

}

}

### Файл Schedule.java

package DatabaseEntities;

import java.util.List;

/\*\*

\* Created by Dmitriy on 13.06.2016.

\*/

public class Schedule {

int id;

int semester;

int yearReception;

int quarter;

String name;

List<DatabaseEntities.Class> classList;

public void setId(int id) {this.id = id;}

public void setSemester(int semester) {this.semester = semester;}

public void setYearReception(int yearReception) {this.yearReception = yearReception;}

public void setClassList(List<DatabaseEntities.Class> classList) {this.classList = classList;}

public void setQuarter(int quoter) {this.quarter = quoter;}

public void setName(String name) {this.name = name;}

public int getId() {return id;}

public int getSemester() {return semester;}

public int getYearReception() {return yearReception;}

public List<DatabaseEntities.Class> getClassList() {return classList;}

public int getQuarter() {return quarter;}

public String getName() {return name;}

/\*\*

\* Создаем запись о новом расписании (2 семестра по 2 четверти каждый)

\*/

@Override

public String toString() {

return "\nSchedule{" +

"id=" + id +

", semester=" + semester +

", yearReception=" + yearReception +

", quarter=" + quarter +

", name='" + name + '\'' +

'}';

}

}

### Файл ScheduleClasses.java

package DatabaseEntities;

import DatabaseEntities.Schedule;

import java.lang.\*;

import java.util.List;

/\*\*

\* Created by Dmitriy on 13.06.2016.

\*/

public class ScheduleClasses {

int id;

int idSchedule;

int idClass;

public void setId(int id) {this.id = id;}

public void setIdSchedule(int idSchedule) {this.idSchedule = idSchedule;}

public void setIdClass(int idClass) {this.idClass = idClass;}

public int getId() {return id;}

public int getIdSchedule() {return idSchedule;}

public int getIdClass() {return idClass;}

@Override

public String toString() {

return "ScheduleClasses{" +

"id: " + id +

", idSchedule: " + idSchedule +

", idClass: " + idClass +

'}';

}

}

### Файл Speciality.java

package DatabaseEntities;

import java.sql.Date;

/\*\*

\* Created by Dmitriy on 13.06.2016.

\*/

public class Speciality {

int id;

String name;

String abbreviation;

String code;

Department department;

boolean isArchive;

Date archiveDate;

public void setId(int id) {this.id = id;}

public void setName(String name) {this.name = name;}

public void setAbbreviation(String avvreviation) {this.abbreviation = avvreviation;}

public void setCode(String code) {this.code = code;}

public void setDepartment(Department department) {this.department = department;}

public void setIsArchive(boolean archive) {isArchive = archive;}

public void setArchiveDate(Date archiveDate) {this.archiveDate = archiveDate;}

public int getId() {return id;}

public String getName() {return name;}

public String getAbbreviation() {return abbreviation;}

public String getCode() {return code;}

public Department getDepartment() {return department;}

public boolean getIsArchive() {return isArchive;}

public Date getArchiveDate() {return archiveDate;}

@Override

public String toString() {

return "Speciality{" +

"id: " + id +

", name: '" + name + '\'' +

", abbreviation: '" + abbreviation + '\'' +

", code: '" + code + '\'' +

", department: " + department +

", isArchive: " + isArchive +

", archiveDate: " + archiveDate +

'}';

}

}

### Файл Stream.java

package DatabaseEntities;

import java.sql.Date;

import java.util.List;

/\*\*

\* Created by Dmitriy on 13.06.2016.

\*/

public class Stream {

int id;

int volume;

int yearReception;

int semester;

Subject subject;

List <GroupStudent> groupStudentList;

boolean isArchive;

Date archiveDate;

public void setId(int id) {this.id = id;}

public void setVolume(int volume) {this.volume = volume;}

public void setYearReception(int yearReception) {this.yearReception = yearReception;}

public void setSemester(int semester) {this.semester = semester;}

public void setSubject(Subject subject) {this.subject = subject;}

public void setGroupStudentList(List<GroupStudent> groupStudentList) {this.groupStudentList = groupStudentList;}

public void setIsArchive(boolean archive) {isArchive = archive;}

public void setArchiveDate(Date archiveDate) {this.archiveDate = archiveDate;}

public int getId() {return id;}

public int getVolume() {return volume;}

public int getYearReception() {return yearReception;}

public int getSemester() {return semester;}

public Subject getSubject() {return subject;}

public List<GroupStudent> getGroupStudentList() {return groupStudentList;}

public boolean getIsArchive() {return isArchive;}

public Date getArchiveDate() {return archiveDate;}

@Override

public String toString() {

return "Stream{" +

" id: " + id +

", volume: " + volume +

", yearReception: " + yearReception +

", semester: " + semester +

", subject: " + subject +

", groupStudentList: " + groupStudentList +

", isArchive: " + isArchive +

", archiveDate: " + archiveDate +

'}';

}

}

### Файл Subject.java

package DatabaseEntities;

import java.sql.Date;

/\*\*

\* Created by Dmitriy on 12.06.2016.

\*/

public class Subject {

int id;

String name;

String abbreviation;

boolean isArchive;

Date archiveDate;

public void setId(int id) {this.id = id;}

public void setName(String name) {this.name = name;}

public void setAbbreviation(String abbreviation) {this.abbreviation = abbreviation;}

public void setIsArchive(boolean archive) {isArchive = archive;}

public void setArchiveDate(Date archiveDate) {this.archiveDate = archiveDate;}

public int getId() {return id;}

public String getName() {return name;}

public String getAbbreviation() {return abbreviation;}

public boolean getIsArchive() {return isArchive;}

public Date getArchiveDate() {return archiveDate;}

@Override

public String toString() {

return "\nSubject{" +

"id: " + id +

", name: '" + name + '\'' +

", abbreviation: '" + abbreviation + '\'' +

", isArchive: " + isArchive +

", archiveDate: " + archiveDate +

'}';

}

}

### Файл TimeClass.java

package DatabaseEntities;

/\*\*

\* Класс описывающий сущность Время занятия

\* Created by Dmitriy on 07.06.2016.

\*/

public class TimeClass {

int id;

TypeWeek typeWeek; //тип недели

WeekDay weekDay; //день недели

int numberClass; //номер занятия

public void setId(int id) {this.id = id;}

public void setTypeWeek(TypeWeek typeWeek) {this.typeWeek = typeWeek;}

public void setWeekDay(WeekDay weekDay) {this.weekDay = weekDay;}

public void setNumberClass(int numberClass) {this.numberClass = numberClass;}

public int getId() {return id;}

public TypeWeek getTypeWeek() {return typeWeek;}

public WeekDay getWeekDay() {return weekDay;}

public int getNumberClass() {return numberClass;}

@Override

public String toString() {

return getClass().getSimpleName() + "{" +

"\t id: " + id +

",\t typeWeek: " + typeWeek.name +

",\t weekDay: " + weekDay.name +

",\t numberClass: " + numberClass +

'}';

}

}

### Файл TypeClass.java

package DatabaseEntities;

/\*\*

\* Класс описывающий сущность Тип занятия

\* Created by Dmitriy on 07.06.2016.

\*/

public class TypeClass {

int id;

String name;

public void setId(int id) {this.id = id;}

public void setName(String name) {this.name = name;}

public int getId() {return id;}

public String getName() {return name;}

@Override

public String toString() {

return getClass().getSimpleName() + "{" +

"id: " + id +

", name: " + name +

'}';

}

}

### Файл TypeWeek.java

package DatabaseEntities;

/\*\*

\* Класс описывающий сущность Тип недели

\* Created by Dmitriy on 07.06.2016.

\*/

public class TypeWeek {

int id;

String name;

public void setName(String name) {this.name = name;}

public void setId(int id) {this.id = id;}

public int getId() {return id;}

public String getName() {return name;}

@Override

public String toString() {

return getClass().getSimpleName() + "{" +

"id: " + id +

", name: " + name +

'}';

}

}

### Файл UniversityTeacher.java

package DatabaseEntities;

import java.sql.Date;

/\*\*

\* Класс описывающий сущность Преподаватель университета

\* Created by Dmitriy on 11.06.2016.

\*/

public class UniversityTeacher {

int id;

String secondName;

String firstName;

String middleName;

String department;

String Position;

boolean isArchive;

Date archiveDate;

public void setId(int id) {this.id = id;}

public void setSecondName(String secondName) {this.secondName = secondName;}

public void setFirstName(String firstName) {this.firstName = firstName;}

public void setMiddleName(String middleName) {this.middleName = middleName;}

public void setDepartment(String department) {this.department = department;}

public void setPosition(String position) {Position = position;}

public void setIsArchive(boolean archive) {isArchive = archive;}

public void setArchiveDate(Date archiveDate) {this.archiveDate = archiveDate;}

public int getId() {return id;}

public String getSecondName() {return secondName;}

public String getFirstName() {return firstName;}

public String getMiddleName() {return middleName;}

public String getDepartment() {return department;}

public String getPosition() {return Position;}

public boolean getIsArchive() {return isArchive;}

public Date getArchiveDate() {return archiveDate;}

public String getAllTeacherName(){

return secondName + " " + firstName + " " + middleName;

}

@Override

public String toString() {

return secondName + " " + firstName + " " + middleName;

}

}

### Файл WeekDay.java

package DatabaseEntities;

/\*\*

\* Класс описывающий сущность День денели

\* Created by Dmitriy on 07.06.2016.

\*/

public class WeekDay {

int id;

String name;

public void setId(int id) {this.id = id;}

public void setName(String name) {this.name = name;}

public int getId() {return id;}

public String getName() {return name;}

@Override

public String toString() {

return getClass().getSimpleName() + "{" +

"id: " + id +

", name: " + name +

'}';

}

}

## Текст модуля DataAccessObject

### Файл ClassDAO.java

package DataAccessObject;

import DatabaseEntities.\*;

import DatabaseEntities.Class;

import java.sql.\*;

import java.util.ArrayList;

import java.util.List;

/\*\*

\* Created by Dmitriy on 13.06.2016.

\*/

public class ClassDAO {

private final Connection con;

public ClassDAO(Connection connection) {this.con = connection;}

private Class getClassFromRS(ResultSet rs){

Class result = new Class();

try {

result.setId(rs.getInt("id"));

result.setTypeClass(new TypeClassDAO(con).getTypeClass(rs.getInt("idTypeClass")));

result.setTimeClass(new TimeClassDAO(con).getTimeClass(rs.getInt("idTimeClass")));

result.setStream(new StreamDAO(con).getStream(rs.getInt("idStream")));

result.setSubject(new SubjectDAO(con).getSubject(rs.getInt("idSubject")));

result.setUniversityTeacher(new UniversityTeacherDAO(con).

getUniversityTeacher(rs.getInt("idUniversityTeacher")));

} catch (SQLException e) {

System.out.println("Method getClassFromRS()");

e.printStackTrace();

}

return result;

}

public Class getClass(int id){

Class result = null;

try {

PreparedStatement ps = con.prepareStatement("SELECT \* FROM Class WHERE id = ?");

ps.setInt(1, id);

ResultSet rs = ps.executeQuery();

if(rs.next()){

result = getClassFromRS(rs);

}

rs.close();

ps.close();

} catch (SQLException e) {

System.out.println("Method getClass()");

e.printStackTrace();

}

return result;

}

public List<Class> getClassList(){

List <Class> result = new ArrayList<Class>();

try {

ResultSet rs = con.createStatement().executeQuery("SELECT \* FROM Class");

while(rs.next()){

result.add(getClassFromRS(rs));

}

} catch (SQLException e) {

System.out.println("Method getClassList()");

e.printStackTrace();

}

return result;

}

public void updateClass(Class \_class){

try {

PreparedStatement ps = null;

TypeClass typeClass = \_class.getTypeClass();

TimeClass timeClass = \_class.getTimeClass();

Stream stream = \_class.getStream();

Subject subject = \_class.getSubject();

UniversityTeacher universityTeacher = \_class.getUniversityTeacher();

int idClass = \_class.getId();

if(idClass > 0){

String querry = "UPDATE Class SET " +

"idTypeClass = ?" +

", idTimeClass = ?" +

", idStream = ?" +

", idSubject = ?" +

", idUniversityTeacher = ? WHERE id = ?";

ps = con.prepareStatement(querry);

ps.setInt(6, idClass);

ps.setInt(1, typeClass.getId());

ps.setInt(2, timeClass.getId());

ps.setInt(3, stream.getId());

ps.setInt(4, subject.getId());

ps.setInt(5, universityTeacher.getId());

}else{

String querry = "INSERT INTO Class " +

"(idTypeClass" +

", idTimeClass" +

", idStream" +

", idSubject" +

", idUniversityTeacher) " +

"VALUES(?, ?, ?, ?, ?)";

ps = con.prepareStatement(querry);

ps.setInt(1, typeClass.getId());

ps.setInt(2, timeClass.getId());

ps.setInt(3, stream.getId());

ps.setInt(4, subject.getId());

ps.setInt(5, universityTeacher.getId());

}

ps.executeUpdate();

ps.close();

}catch (SQLException e) {

System.out.println("Method updateClass()");

e.printStackTrace();

}

}

public int getIdForLastRecord(){

int result = 0;

try {

ResultSet rs = con.createStatement().

executeQuery("SELECT id from Class WHERE id = (SELECT MAX(id) From Class)");

if(rs.next()){

result = rs.getInt("id");

}

} catch (SQLException e) {

System.out.println("Method getIdForLastRecord()");

e.printStackTrace();

}

return result;

}

public void deleteClass(int id){

try {

PreparedStatement ps = con.prepareStatement("DELETE FROM Class WHERE id = ?");

ps.setInt(1, id);

ps.executeUpdate();

ps.close();

} catch (SQLException e) {

System.out.println("Method deleteClass()");

e.printStackTrace();

}

}

public void deleteAllClass(){

try {

Statement st = con.createStatement();

st.executeUpdate("DELETE FROM Class");

st.close();

} catch (SQLException e) {

System.out.println("Method deleteAllClass()");

e.printStackTrace();

}

}

}

### Файл DepartmentDAO.java

package DataAccessObject;

import DatabaseEntities.Department;

import DatabaseEntities.Faculty;

import java.sql.Connection;

import java.sql.PreparedStatement;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.util.ArrayList;

import java.util.List;

import java.util.logging.Level;

import java.util.logging.Logger;

/\*\*

\* Created by Dmitriy on 11.06.2016.

\*/

public class DepartmentDAO {

private final Connection con;

public DepartmentDAO(Connection connection){con = connection;}

private Department getDepartmentFromRS(ResultSet rs){

Department result = new Department();

try {

result.setId(rs.getInt("id"));

result.setName(rs.getString("name"));

result.setAbbreviation(rs.getString("abbreviation"));

result.setNumber(rs.getInt("number"));

result.setFaculty(new FacultyDAO(con).getFaculty(rs.getInt("idFaculty")));

result.setIsArchive(rs.getBoolean("isArchive"));

result.setArchiveDate(rs.getDate("ArchiveDate"));

} catch (SQLException e) {

System.out.println("Method getDepartmentFromRS()");

e.printStackTrace();

}

return result;

}

public Department getDepartment(int id){

Department result = null;

try {

PreparedStatement ps = con.prepareStatement("SELECT \* FROM Department WHERE id = ? AND isArchive = 0");

ps.setInt(1, id);

ResultSet rs = ps.executeQuery();

if(rs.next()){

result = getDepartmentFromRS(rs);

}

rs.close();

ps.close();

} catch (SQLException e) {

System.out.println("Method getDepartment()");

e.printStackTrace();

}

return result;

}

public List <Department> getDepartmentListByFaculty(Faculty faculty){

List <Department> result = new ArrayList<Department>();

try {

int idFaculty = faculty.getId();

String querry = "SELECT \* FROM Department WHERE idFaculty = ? AND isArchive = 0";

PreparedStatement ps = con.prepareStatement(querry);

ps.setInt(1, idFaculty);

ResultSet rs = ps.executeQuery();

while(rs.next()){

result.add(getDepartmentFromRS(rs));

}

rs.close();

ps.close();

} catch (SQLException e) {

e.printStackTrace();

}

return result;

}

public List <Department> getDepartmentList(){

List <Department> result = new ArrayList <Department> ();

try {

ResultSet rs = con.createStatement().executeQuery("SELECT \* FROM Department WHERE isArchive = 0 ORDER BY number");

while(rs.next()){

result.add(getDepartmentFromRS(rs));

}

rs.close();

} catch (SQLException e) {

System.out.println("Method getDepartmentList()");

e.printStackTrace();

}

return result;

}

}

### Файл FacultyDAO.java

package DataAccessObject;

import DatabaseEntities.Faculty;

import java.sql.Connection;

import java.sql.PreparedStatement;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.util.ArrayList;

import java.util.List;

/\*\*

\* Created by Dmitriy on 10.06.2016.

\*/

public class FacultyDAO {

private final Connection con;

public FacultyDAO(Connection con) {this.con = con;}

private Faculty getFacultyFromRS(ResultSet rs){

Faculty result = new Faculty();

try {

result.setId(rs.getInt("id"));

result.setName(rs.getString("name"));

result.setAbbreviation(rs.getString("abbreviation"));

result.setNumber(rs.getInt("number"));

result.setIsArchive(rs.getBoolean("isArchive"));

result.setArchiveDate(rs.getDate("archiveDate"));

} catch (SQLException e) {

System.out.println("Method getFacultyFromRS()");

e.printStackTrace();

}

return result;

}

public Faculty getFaculty(int id){

Faculty result = null;

try {

PreparedStatement ps = con.prepareStatement("SELECT \* FROM Faculty WHERE id = ? AND isArchive = 0");

ps.setInt(1, id);

ResultSet rs = ps.executeQuery();

if(rs.next()){

result = getFacultyFromRS(rs);

}

rs.close();

ps.close();

} catch (SQLException e) {

System.out.println("Method getFaculty()");

e.printStackTrace();

}

return result;

}

public List <Faculty> getFacultyList(){

List <Faculty> result = new ArrayList<Faculty>();

try {

ResultSet rs = con.createStatement().executeQuery("SELECT \* FROM Faculty WHERE isArchive = 0");

while (rs.next()){

result.add(getFacultyFromRS(rs));

}

rs.close();

} catch (SQLException e) {

System.out.println("Method getFacultyList()");

e.printStackTrace();

}

return result;

}

}

### Файл GroupStudentDAO.java

package DataAccessObject;

import DatabaseEntities.Department;

import DatabaseEntities.GroupStudent;

import java.sql.Connection;

import java.sql.PreparedStatement;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.util.ArrayList;

import java.util.List;

import java.util.logging.Level;

import java.util.logging.Logger;

/\*\*

\* Created by Dmitriy on 13.06.2016.

\*/

public class GroupStudentDAO {

private final Connection con;

public GroupStudentDAO(Connection con) {this.con = con;}

private GroupStudent getGroupStudentFromRS(ResultSet rs){

GroupStudent result = new GroupStudent();

try {

result.setId(rs.getInt("id"));

result.setName(rs.getString("name"));

result.setCountStudent(rs.getInt("countStudent"));

result.setSpeciality(new SpecialityDAO(con).getSpeciality(rs.getInt("idSpeciality")));

result.setCourse(rs.getInt("course"));

result.setIsArchive(rs.getBoolean("isArchive"));

result.setArchiveDate(rs.getDate("ArchiveDate"));

} catch (SQLException e) {

System.out.println("Method getGroupStudentFromRS()");

e.printStackTrace();

}

return result;

}

public GroupStudent getGroupStudent(int id){

GroupStudent result = null;

try {

PreparedStatement ps = con.prepareStatement("SELECT \* FROM GroupStudent WHERE id = ? AND isArchive = 0");

ps.setInt(1, id);

ResultSet rs = ps.executeQuery();

if(rs.next()){

result = getGroupStudentFromRS(rs);

}

rs.close();

ps.close();

} catch (SQLException e) {

System.out.println("Method getFaculty()");

e.printStackTrace();

}

return result;

}

public List<GroupStudent> getGroupStudentList(){

List <GroupStudent> result = new ArrayList<GroupStudent>();

try {

ResultSet rs = con.createStatement().executeQuery("SELECT \* FROM GroupStudent WHERE isArchive = 0");

while (rs.next()){

result.add(getGroupStudentFromRS(rs));

}

rs.close();

} catch (SQLException e) {

System.out.println("Method getFacultyList()");

e.printStackTrace();

}

return result;

}

public List<GroupStudent> getGroupStudentListForDepartments(List <Department> departments){

List <GroupStudent> result = new ArrayList<GroupStudent>();

int idDepartment;

String querry = "SELECT \* FROM GroupStudent\n" +

"INNER JOIN Speciality ON GroupStudent.idSpeciality = Speciality.id\n" +

"WHERE idDepartment = ? AND GroupStudent.isArchive = 0";

for(Department department: departments){

idDepartment = department.getId();

try {

PreparedStatement ps = con.prepareStatement(querry);

ps.setInt(1, idDepartment);

ResultSet rs = ps.executeQuery();

while(rs.next()){

result.add(getGroupStudentFromRS(rs));

}

rs.close();

ps.close();

} catch (SQLException e) {

System.out.println("Method getGroupStudentListForDepartments()");

e.printStackTrace();

}

}

return result;

}

}

### Файл LoadDepartmentDAO.java

package DataAccessObject;

import DatabaseEntities.GroupStudent;

import DatabaseEntities.LoadDepartment;

import DatabaseEntities.LoadDepartmentDoc;

import DatabaseEntities.Subject;

import java.sql.Connection;

import java.sql.PreparedStatement;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.util.ArrayList;

import java.util.List;

/\*\*

\* Created by Dmitriy on 13.06.2016.

\*/

public class LoadDepartmentDAO {

private final Connection con;

public LoadDepartmentDAO(Connection connection) {this.con = connection;}

private LoadDepartment getLoadDepartmentFromRS(ResultSet rs){

LoadDepartment result = new LoadDepartment();

try {

result.setId(rs.getInt("id"));

result.setLoadDepartmentDoc(new LoadDepartmentDocDAO(con).

getLoadDepartmentDoc(rs.getInt("idLoadDepartmentDoc")));

result.setSubject(new SubjectDAO(con).getSubject(rs.getInt("idSubject")));

result.setIsHalfSemester(rs.getBoolean("isHalfSemester"));

result.setGroupStudent(new GroupStudentDAO(con).

getGroupStudent(rs.getInt("idGroupStudent")));

result.setCountHourLecture(rs.getInt("countHourLecture"));

result.setCountHourLabWork(rs.getInt("countHourLabWork"));

result.setCountHoutPracticWork(rs.getInt("countHourPracticWork"));

int idUniversityTeacherLecture = rs.getInt("idUniversityTeacherLecture");

int idUniversityTeacherLabWork = rs.getInt("idUniversityTeacherLabWork");

int idUniversityTeacherPracticWork = rs.getInt("idUniversityTeacherPracticWork");

int idUniversityTeacherLabWorkSecond = rs.getInt("idUniversityTeacherLabWorkSecond");

int idUniversityTeacherPracticWorkSecond = rs.getInt("idUniversityTeacherPracticWorkSecond");

result.setTeacherLecture(new UniversityTeacherDAO(con).

getUniversityTeacher(idUniversityTeacherLecture));

result.setTeacherLabWork(new UniversityTeacherDAO(con).

getUniversityTeacher(idUniversityTeacherLabWork));

result.setTeacherPracticWork(new UniversityTeacherDAO(con).

getUniversityTeacher(idUniversityTeacherPracticWork));

result.setTeacherLabWorkSecond(new UniversityTeacherDAO(con).

getUniversityTeacher(idUniversityTeacherLabWorkSecond));

result.setTeacherPracticWorkSecond(new UniversityTeacherDAO(con).

getUniversityTeacher(idUniversityTeacherPracticWorkSecond));

} catch (SQLException e) {

System.out.println("Method getLoadDepartmentFromRS()");

e.printStackTrace();

}

return result;

}

public LoadDepartment getLoadDepartment(int id){

LoadDepartment result = null;

try {

PreparedStatement ps = con.prepareStatement("SELECT \* FROM LoadDepartment WHERE id = ?");

ps.setInt(1, id);

ResultSet rs = ps.executeQuery();

if(rs.next()){

result = getLoadDepartmentFromRS(rs);

}

rs.close();

ps.close();

} catch (SQLException e) {

System.out.println("Method getLoadDepartment()");

e.printStackTrace();

}

return result;

}

public List<LoadDepartment> getLoadDepartmentList(){

List<LoadDepartment> result = new ArrayList<LoadDepartment>();

try {

ResultSet rs = con.createStatement().executeQuery("SELECT \* FROM LoadDepartment");

while(rs.next()){

result.add(getLoadDepartmentFromRS(rs));

}

rs.close();

} catch (SQLException e) {

System.out.println("Method getLoadDepartmentList()");

e.printStackTrace();

}

return result;

}

public List <LoadDepartment> getLoadDepartmentForSubjectByDocId(Subject subject, LoadDepartmentDoc doc){

List <LoadDepartment> result = new ArrayList<LoadDepartment>();

try {

String querry = "SELECT \* FROM LoadDepartment WHERE idLoadDepartmentDoc = ? AND idSubject = ?";

PreparedStatement ps = con.prepareStatement(querry);

ps.setInt(1, doc.getId());

ps.setInt(2, subject.getId());

ResultSet rs = ps.executeQuery();

while(rs.next()){

result.add(getLoadDepartmentFromRS(rs));

}

rs.close();

ps.close();

} catch (SQLException e) {

System.out.println("Method getLoadDepartmentBySubjectAndDocId()");

e.printStackTrace();

}

return result;

}

public List<LoadDepartment> getLoadDepartmentListByDocId(int idLoadDepartmentDoc){

List<LoadDepartment> result = new ArrayList<LoadDepartment>();

try {

String querry = "SELECT \* FROM LoadDepartment WHERE idLoadDepartmentDoc = ?";

PreparedStatement ps = con.prepareStatement(querry);

ps.setInt(1, idLoadDepartmentDoc);

ResultSet rs = ps.executeQuery();

while(rs.next()){

result.add(getLoadDepartmentFromRS(rs));

}

rs.close();

ps.close();

} catch (SQLException e) {

System.out.println("Method getLoadDepartmentListByDocId()");

e.printStackTrace();

}

return result;

}

}

### Файл LoadDepartmentDocDAO.java

package DataAccessObject;

import DatabaseEntities.LoadDepartmentDoc;

import java.sql.Connection;

import java.sql.PreparedStatement;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.util.ArrayList;

import java.util.List;

/\*\*

\* Created by Dmitriy on 13.06.2016.

\*/

public class LoadDepartmentDocDAO {

private final Connection con;

public LoadDepartmentDocDAO(Connection connection) {this.con = connection;}

private LoadDepartmentDoc getLoadDepartmentDocFromRS(ResultSet rs){

LoadDepartmentDoc result = new LoadDepartmentDoc();

try {

result.setId(rs.getInt("id"));

result.setYearReception(rs.getInt("yearReception"));

result.setSemester(rs.getInt("semester"));

result.setDepartment(new DepartmentDAO(con).getDepartment(rs.getInt("idDepartment")));

result.setIsArchive(rs.getBoolean("isArchive"));

result.setArchiveDate(rs.getDate("archiveDate"));

} catch (SQLException e) {

System.out.println("Method getLoadDepartmentDocFromRS");

e.printStackTrace();

}

return result;

}

public LoadDepartmentDoc getLoadDepartmentDoc(int id){

LoadDepartmentDoc result = null;

try {

PreparedStatement ps = con.prepareStatement("SELECT \* FROM LoadDepartmentDoc WHERE id = ? AND isArchive = 0");

ps.setInt(1 , id);

ResultSet rs = ps.executeQuery();

if(rs.next()){

result = getLoadDepartmentDocFromRS(rs);

}

rs.close();

ps.close();

} catch (SQLException e) {

System.out.println("Method getLoadDepartmentDoc()");

e.printStackTrace();

}

return result;

}

public List <LoadDepartmentDoc> getLoadDepartmentDocList(){

List<LoadDepartmentDoc> result = new ArrayList<LoadDepartmentDoc>();

try {

String querry = "SELECT \* FROM LoadDepartmentDoc WHERE isArchive = 0 ORDER BY idDepartment";

ResultSet rs = con.createStatement().executeQuery(querry);

while(rs.next()){

result.add(getLoadDepartmentDocFromRS(rs));

}

rs.close();

} catch (SQLException e) {

System.out.println("Method getLoadDepartmentDocList()");

e.printStackTrace();

}

return result;

}

public List <LoadDepartmentDoc> getLoadDepartmentDocListByYearAndSemester(int year, int semester){

List<LoadDepartmentDoc> result = new ArrayList<LoadDepartmentDoc>();

try {

String querry = "SELECT \* FROM LoadDepartmentDoc " +

"where (yearReception=? AND semester=?) AND isArchive=0 " +

"ORDER BY idDepartment";

PreparedStatement ps = con.prepareStatement(querry);

ps.setInt(1 , year);

ps.setInt(2, semester);

ResultSet rs = ps.executeQuery();

while(rs.next()){

result.add(getLoadDepartmentDocFromRS(rs));

}

rs.close();

ps.close();

} catch (SQLException e) {

System.out.println("Method getLoadDepartmentDocList()");

e.printStackTrace();

}

return result;

}

}

### Файл RequirementsDAO.java

package DataAccessObject;

import DatabaseEntities.Requirements;

import java.sql.Connection;

import java.sql.PreparedStatement;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.util.ArrayList;

import java.util.List;

/\*\*

\* Created by Dmitriy on 12.06.2016.

\*/

public class RequirementsDAO {

private final Connection con;

public RequirementsDAO(Connection connection) {this.con = connection;}

private Requirements getRequirementsFromRS(ResultSet rs){

Requirements result = new Requirements();

try {

result.setId(rs.getInt("id"));

result.setName(rs.getString("name"));

} catch (SQLException e) {

System.out.println("Method getRequirementsFromRS()");

e.printStackTrace();

}

return result;

}

public Requirements getRequirements(int id){

Requirements result = null;

try {

PreparedStatement ps = con.prepareStatement("SELECT \* FROM Requirements WHERE id = ?");

ps.setInt(1, id);

ResultSet rs = ps.executeQuery();

if(rs.next()){

result = getRequirementsFromRS(rs);

}

rs.close();

ps.close();

} catch (SQLException e) {

System.out.println("Method getRequirements()");

e.printStackTrace();

}

return result;

}

public List<Requirements> getRequirementsList(){

List<Requirements> result = new ArrayList<Requirements>();

try {

ResultSet rs = con.createStatement().executeQuery("SELECT \* FROM Requirements");

while(rs.next()){

result.add(getRequirementsFromRS(rs));

}

rs.close();

} catch (SQLException e) {

System.out.println("Method getRequirementsList()");

e.printStackTrace();

}

return result;

}

}

### Файл RequirementsUniversityTeacherDAO.java

package DataAccessObject;

import DatabaseEntities.Requirements;

import DatabaseEntities.RequirementsUniversityTeacher;

import DatabaseEntities.UniversityTeacher;

import DatabaseEntities.WeekDay;

import java.sql.Connection;

import java.sql.PreparedStatement;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.util.ArrayList;

import java.util.List;

import java.util.logging.Level;

import java.util.logging.Logger;

/\*\*

\* Created by Dmitriy on 12.06.2016.

\*/

public class RequirementsUniversityTeacherDAO {

private final Connection con;

public RequirementsUniversityTeacherDAO(Connection connection) {this.con = connection;}

private RequirementsUniversityTeacher getRequirementsUniversityTeacherFromRS(ResultSet rs){

RequirementsUniversityTeacher result = new RequirementsUniversityTeacher();

try {

result.setId(rs.getInt("id"));

result.setUniversityTeacher(new UniversityTeacherDAO(con).

getUniversityTeacher(rs.getInt("idUniversityTeacher")));

result.setYearReception(rs.getInt("yearReception"));

result.setSemester(rs.getInt("semester"));

result.setWeekDay(new WeekDayDAO(con).getWeekDay(rs.getInt("idWeekDay")));

result.setNumberClass(rs.getInt("numberClass"));

result.setRequirements(new RequirementsDAO(con).getRequirements(rs.getInt("idRequirements")));

result.setIsArchive(rs.getBoolean("isArchive"));

result.setArchiveDate(rs.getDate("archiveDate"));

} catch (SQLException e) {

System.out.println("Method getRequirementsUniversityTeacherFromRS()");

e.printStackTrace();

}

return result;

}

public RequirementsUniversityTeacher getRequirementsUniversityTeacher(int id){

RequirementsUniversityTeacher result = null;

try {

PreparedStatement ps = con.prepareStatement("SELECT \* FROM RequirementsUniversityTeacher WHERE id = ?");

ps.setInt(1, id);

ResultSet rs = ps.executeQuery();

if(rs.next()){

result = getRequirementsUniversityTeacherFromRS(rs);

}

rs.close();

ps.close();

} catch (SQLException e) {

System.out.println("Method getRequirementsUniversityTeacher()");

e.printStackTrace();

}

return result;

}

public List<RequirementsUniversityTeacher> getRequirementsUniversityTeacherList(){

List <RequirementsUniversityTeacher> result = new ArrayList<RequirementsUniversityTeacher>();

try {

ResultSet rs = con.createStatement().executeQuery("SELECT \* FROM RequirementsUniversityTeacher");

while(rs.next()){

result.add(getRequirementsUniversityTeacherFromRS(rs));

}

rs.close();

} catch (SQLException e) {

System.out.println("Method getRequirementsUniversityTeacherList()");

e.printStackTrace();

}

return result;

}

public List <Integer> getAllYear(){

List <Integer> result = new ArrayList<Integer>();

try {

String querry = "SELECT DISTINCT yearReception FROM RequirementsUniversityTeacher";

ResultSet rs = con.createStatement().executeQuery(querry);

while(rs.next()){

result.add(rs.getInt("yearReception"));

}

} catch (SQLException e) {

System.out.println("Method getAllYear()");

e.printStackTrace();

}

return result;

}

public void updateRequirementsUniversityTeacher(RequirementsUniversityTeacher rut){

try {

PreparedStatement ps;

UniversityTeacher universityTeacher = rut.getUniversityTeacher();

WeekDay weekDay = rut.getWeekDay();

Requirements requirements = rut.getRequirements();

int rutId = rut.getId();

if(rutId > 0) {

String querry = "UPDATE RequirementsUniversityTeacher " +

"SET idUniversityTeacher = ?" +

", yearReception = ?" +

", semester = ? " +

", idWeekDay = ?" +

", numberClass = ?" +

", idRequirements = ?" +

", isArchive = ?" +

", ArchiveDate = ? WHERE id = ?";

ps = con.prepareStatement(querry);

ps.setInt(9, rutId);

ps.setInt(1, universityTeacher.getId());

ps.setInt(2, rut.getYearReception());

ps.setInt(3, rut.getSemester());

ps.setInt(4, weekDay.getId());

ps.setInt(5, rut.getNumberClass());

ps.setInt(6, requirements.getId());

ps.setBoolean(7, rut.getIsArchive());

ps.setDate(8, rut.getArchiveDate());

}

else{

String querry = "INSERT INTO RequirementsUniversityTeacher " +

"(idUniversityTeacher" +

", yearReception" +

", semester" +

", idWeekDay" +

", numberClass" +

", idRequirements" +

", isArchive" +

", ArchiveDate)" +

"VALUES (?, ?, ?, ?, ?, ?, ?, ?)";

ps = con.prepareStatement(querry);

ps.setInt(1, universityTeacher.getId());

ps.setInt(2, rut.getYearReception());

ps.setInt(3, rut.getSemester());

ps.setInt(4, weekDay.getId());

ps.setInt(5, rut.getNumberClass());

ps.setInt(6, requirements.getId());

ps.setBoolean(7, rut.getIsArchive());

ps.setDate(8, rut.getArchiveDate());

}

ps.executeUpdate();

ps.close();

} catch (SQLException e){

System.out.println("Method updateRequirementsUniversityTeacher()");

e.printStackTrace();

}

}

public void deleteRequirementsUniversityTeacher(int id){

try {

PreparedStatement ps = con.prepareStatement("DELETE FROM RequirementsUniversityTeacher WHERE id = ?");

ps.setInt(1, id);

ps.executeUpdate();

ps.close();

} catch (SQLException e) {

System.out.println("Method deleteRequirementsUniversityTeacher()");

e.printStackTrace();

}

}

}

### Файл ScheduleClassesDAO.java

package DataAccessObject;

import DatabaseEntities.Class;

import DatabaseEntities.ScheduleClasses;

import java.sql.\*;

import java.util.ArrayList;

import java.util.List;

/\*\*

\* Created by Dmitriy on 13.06.2016.

\*/

public class ScheduleClassesDAO {

private final Connection con;

public ScheduleClassesDAO(Connection connection) {this.con = connection;}

private ScheduleClasses getScheduleClassesFromRS(ResultSet rs) {

ScheduleClasses result = new ScheduleClasses();

try {

result.setId(rs.getInt("id"));

result.setIdSchedule(rs.getInt("idSchedule"));

result.setIdClass(rs.getInt("idClass"));

} catch (SQLException e) {

System.out.println("Method getStreamFromRS()");

e.printStackTrace();

}

return result;

}

public ScheduleClasses getScheduleClasses(int id){

ScheduleClasses result = null;

try {

PreparedStatement ps = con.prepareStatement("SELECT \* FROM ScheduleClasses WHERE id = ?");

ps.setInt(1, id);

ResultSet rs = ps.executeQuery();

if(rs.next()){

result = getScheduleClassesFromRS(rs);

}

rs.close();

ps.close();

} catch (SQLException e) {

System.out.println("Method getScheduleClasses()");

e.printStackTrace();

}

return result;

}

public List<ScheduleClasses> getScheduleClassesList(){

List <ScheduleClasses> result = new ArrayList<ScheduleClasses>();

try {

ResultSet rs = con.createStatement().executeQuery("SELECT \* FROM ScheduleClasses");

while(rs.next()){

result.add(getScheduleClassesFromRS(rs));

}

rs.close();

} catch (SQLException e) {

System.out.println("Method getScheduleClassesList()");

e.printStackTrace();

}

return result;

}

public void updateScheduleClasses(ScheduleClasses scheduleClasses){

try {

PreparedStatement ps = null;

int idScheduleClasses = scheduleClasses.getId();

if(idScheduleClasses > 0){

String querry = "UPDATE ScheduleClasses SET idSchedule = ?, idClass = ? WHERE id = ?";

ps = con.prepareStatement(querry);

ps.setInt(3, idScheduleClasses);

ps.setInt(1, scheduleClasses.getIdSchedule());

ps.setInt(2, scheduleClasses.getIdClass());

}else{

String querry = "INSERT INTO ScheduleClasses (idSchedule, idClass) VALUES (?, ?)";

ps = con.prepareStatement(querry);

ps.setInt(1, scheduleClasses.getIdSchedule());

ps.setInt(2, scheduleClasses.getIdClass());

}

ps.executeUpdate();

ps.close();

} catch (SQLException e) {

System.out.println("Method updateScheduleClasses()");

e.printStackTrace();

}

}

public void deleteScheduleClasses(int id){

try {

PreparedStatement ps = con.prepareStatement("DELETE FROM ScheduleClasses WHERE id = ?");

ps.setInt(1, id);

ps.executeUpdate();

ps.close();

} catch (SQLException e) {

System.out.println("Method deleteScheduleClasses()");

e.printStackTrace();

}

}

public void deleteAllScheduleClasses(){

try {

Statement st = con.createStatement();

st.executeUpdate("DELETE FROM ScheduleClasses");

st.close();

} catch (SQLException e) {

System.out.println("Method deleteAllScheduleClasses()");

e.printStackTrace();

}

}

}

### Файл ScheduleDAO.java

package DataAccessObject;

import DatabaseEntities.Schedule;

import java.sql.Connection;

import java.sql.PreparedStatement;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.util.ArrayList;

import java.util.List;

/\*\*

\* Created by Dmitriy on 13.06.2016.

\*/

public class ScheduleDAO {

private final Connection con;

public ScheduleDAO(Connection connection) {this.con = connection;}

private List<DatabaseEntities.Class> getListOfClassesFromScheduleClasses(int idSchedule){

List <DatabaseEntities.Class> classes = new ArrayList<DatabaseEntities.Class>();

try {

PreparedStatement ps = con.prepareStatement("SELECT \* FROM ScheduleClasses WHERE idSchedule = ?");

ps.setInt(1, idSchedule);

ResultSet rs = ps.executeQuery();

while (rs.next()) {

classes.add(new ClassDAO(con).getClass(rs.getInt("idClass")));

}

rs.close();

ps.close();

} catch (SQLException e) {

System.out.println("Method getListOfClassesFromScheduleClasses()");

e.printStackTrace();

}

return classes;

}

private Schedule getScheduleFromRS(ResultSet rs){

Schedule result = new Schedule();

try {

int idSchedule = rs.getInt("id");

result.setId(idSchedule);

result.setSemester(rs.getInt("semester"));

result.setYearReception(rs.getInt("yearReception"));

result.setClassList(getListOfClassesFromScheduleClasses(idSchedule));

result.setQuarter(rs.getInt("quarter"));

result.setName(rs.getString("name"));

} catch (SQLException e) {

System.out.println("Method getScheduleFromRS()");

e.printStackTrace();

}

return result;

}

public Schedule getSchedule(int id){

Schedule result = null;

try {

PreparedStatement ps = con.prepareStatement("SELECT \* FROM Schedule WHERE id = ?");

ps.setInt(1, id);

ResultSet rs = ps.executeQuery();

if(rs.next()){

result = getScheduleFromRS(rs);

}

rs.close();

ps.close();

} catch (SQLException e) {

System.out.println("Method getSchedule()");

e.printStackTrace();

}

return result;

}

public List<Schedule> getSheduleList(){

List <Schedule> result = new ArrayList<Schedule>();

try {

ResultSet rs = con.createStatement().executeQuery("SELECT \* FROM Schedule");

while(rs.next()){

result.add(getScheduleFromRS(rs));

}

rs.close();

} catch (SQLException e) {

System.out.println("Method getClassList()");

e.printStackTrace();

}

return result;

}

public List<Schedule> getThisSheduleList(int year, int semester, String name, int quarter){

List <Schedule> result = new ArrayList<Schedule>();

try {

String querry = "SELECT \* FROM Schedule WHERE yearReception = ? AND semester = ? AND name = ? AND quarter = ?";

PreparedStatement ps = con.prepareStatement(querry);

ps.setInt(1, year);

ps.setInt(2, semester);

ps.setString(3, name);

ps.setInt(4, quarter);

ResultSet rs = ps.executeQuery();

while(rs.next()){

result.add(getScheduleFromRS(rs));

}

rs.close();

ps.close();

} catch (SQLException e) {

System.out.println("Method getThisSheduleList()");

e.printStackTrace();

}

return result;

}

public List<Schedule> getThisSheduleList(int year, int semester, String name){

List <Schedule> result = new ArrayList<Schedule>();

try {

String querry = "SELECT \* FROM Schedule WHERE yearReception = ? AND semester = ? AND name = ?";

PreparedStatement ps = con.prepareStatement(querry);

ps.setInt(1, year);

ps.setInt(2, semester);

ps.setString(3, name);

ResultSet rs = ps.executeQuery();

while(rs.next()){

result.add(getScheduleFromRS(rs));

}

rs.close();

ps.close();

} catch (SQLException e) {

System.out.println("Method getThisSheduleList()");

e.printStackTrace();

}

return result;

}

public List<Schedule> getThisSheduleList(int year, String name){

List <Schedule> result = new ArrayList<Schedule>();

try {

String querry = "SELECT \* FROM Schedule WHERE yearReception = ? AND name = ?";

PreparedStatement ps = con.prepareStatement(querry);

ps.setInt(1, year);

ps.setString(2, name);

ResultSet rs = ps.executeQuery();

while(rs.next()){

result.add(getScheduleFromRS(rs));

}

rs.close();

ps.close();

} catch (SQLException e) {

System.out.println("Method getThisSheduleList()");

e.printStackTrace();

}

return result;

}

public List <String> getAllNameOfSchedulesByYear(int yearReception){

List <String> result = new ArrayList<String>();

try {

String querry = "SELECT DISTINCT name FROM Schedule WHERE yearReception = ?";

PreparedStatement ps = con.prepareStatement(querry);

ps.setInt(1, yearReception);

ResultSet rs = ps.executeQuery();

while(rs.next()){

result.add(rs.getString("name"));

}

rs.close();

} catch (SQLException e) {

System.out.println("Method getAllNameOfSchedulesByYear()");

e.printStackTrace();

}

return result;

}

public List <Integer> getAllYearsOfSchedules(){

List <Integer> result = new ArrayList<Integer>();

try {

String querry = "SELECT DISTINCT yearReception FROM Schedule";

ResultSet rs = con.createStatement().executeQuery(querry);

while(rs.next()){

result.add(rs.getInt("yearReception"));

}

rs.close();

} catch (SQLException e) {

System.out.println("Method getAllNameOfSchedules()");

e.printStackTrace();

}

return result;

}

public void updateSchedule(Schedule schedule){

try {

PreparedStatement ps;

int scheduleId = schedule.getId();

if(scheduleId > 0) {

String querry = "UPDATE Schedule " +

"SET semester = ?" +

", yearReception = ?" +

", quoter = ?" +

", name = ? WHERE id = ?";

ps = con.prepareStatement(querry);

ps.setInt(5, scheduleId);

ps.setInt(1, schedule.getSemester());

ps.setInt(2, schedule.getYearReception());

ps.setInt(3, schedule.getQuarter());

ps.setString(4, schedule.getName());

}

else {

String querry = "INSERT INTO Schedule" +

"(semester, yearReception, quarter, name)" +

"VALUES (?,?,?,?)";

ps = con.prepareStatement(querry);

ps.setInt(1, schedule.getSemester());

ps.setInt(2, schedule.getYearReception());

ps.setInt(3, schedule.getQuarter());

ps.setString(4, schedule.getName());

}

ps.executeUpdate();

ps.close();

} catch (SQLException e) {

System.out.println("Method updateSchedule()");

e.printStackTrace();

}

}

public void deleteSchedule(int id){

//yourCode

}

public void deleteAllSchedule(){

//yourCode

}

}

### Файл SpecialityDAO.java

package DataAccessObject;

import DatabaseEntities.Speciality;

import java.sql.Connection;

import java.sql.PreparedStatement;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.util.ArrayList;

import java.util.List;

/\*\*

\* Created by Dmitriy on 13.06.2016.

\*/

public class SpecialityDAO {

private final Connection con;

public SpecialityDAO(Connection connection) {this.con = connection;}

private Speciality getSpecialityFromRS(ResultSet rs){

Speciality result = new Speciality();

try {

result.setId(rs.getInt("id"));

result.setName(rs.getString("name"));

result.setAbbreviation(rs.getString("abbreviation"));

result.setCode(rs.getString("code"));

result.setDepartment(new DepartmentDAO(con).getDepartment(rs.getInt("idDepartment")));

result.setIsArchive(rs.getBoolean("isArchive"));

result.setArchiveDate(rs.getDate("archiveDate"));

} catch (SQLException e) {

System.out.println("Method getSpecialityFromRS()");

e.printStackTrace();

}

return result;

}

public Speciality getSpeciality(int id){

Speciality result = null;

try {

PreparedStatement ps = con.prepareStatement("SELECT \* FROM Speciality WHERE id = ?");

ps.setInt(1, id);

ResultSet rs = ps.executeQuery();

if(rs.next()){

result = getSpecialityFromRS(rs);

}

rs.close();

ps.close();

} catch (SQLException e) {

System.out.println("Method getSpeciality()");

e.printStackTrace();

}

return result;

}

public List<Speciality> getSpecialityList(){

List<Speciality> result = new ArrayList<Speciality>();

try {

ResultSet rs = con.createStatement().executeQuery("SELECT \* FROM Speciality");

while(rs.next()){

result.add(getSpecialityFromRS(rs));

}

rs.close();

} catch (SQLException e) {

System.out.println("Method getSpecialityList()");

e.printStackTrace();

}

return result;

}

}

### Файл StreamDAO.java

package DataAccessObject;

import DatabaseEntities.GroupStudent;

import DatabaseEntities.Stream;

import DatabaseEntities.Subject;

import java.sql.Connection;

import java.sql.PreparedStatement;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.util.ArrayList;

import java.util.List;

/\*\*

\* Created by Dmitriy on 13.06.2016.

\*/

public class StreamDAO {

private final Connection con;

public StreamDAO(Connection connection) {this.con = connection;}

private List <GroupStudent> getListOfGroupsFromStreamGroup(int idStream){

List <GroupStudent> groupStudents = new ArrayList<GroupStudent>();

try {

PreparedStatement ps = con.prepareStatement("SELECT \* FROM StreamGroup WHERE idStream = ?");

ps.setInt(1, idStream);

ResultSet rs = ps.executeQuery();

while (rs.next()) {

groupStudents.add(new GroupStudentDAO(con).getGroupStudent(rs.getInt("idGroupStudent")));

}

ps.close();

} catch (SQLException e) {

System.out.println("Method getSetOfGroupsFromStreamGroup()");

e.printStackTrace();

}

return groupStudents;

}

private Stream getStreamFromRS(ResultSet rs) {

Stream result = new Stream();

try {

int idStream = rs.getInt("id");

result.setId(idStream);

result.setVolume(rs.getInt("volume"));

result.setYearReception(rs.getInt("yearReception"));

result.setSemester(rs.getInt("semester"));

result.setSubject(new SubjectDAO(con).getSubject(rs.getInt("idSubject")));

result.setGroupStudentList(getListOfGroupsFromStreamGroup(idStream));

result.setIsArchive(rs.getBoolean("isArchive"));

result.setArchiveDate(rs.getDate("ArchiveDate"));

} catch (SQLException e) {

System.out.println("Method getStreamFromRS()");

e.printStackTrace();

}

return result;

}

public Stream getStream(int id){

Stream result = null;

try {

PreparedStatement ps = con.prepareStatement("SELECT \* FROM Stream WHERE id = ?");

ps.setInt(1, id);

ResultSet rs = ps.executeQuery();

if(rs.next()){

result = getStreamFromRS(rs);

}

rs.close();

ps.close();

} catch (SQLException e) {

System.out.println("Method getStream()");

e.printStackTrace();

}

return result;

}

public List <Stream> searchStreamsForThisLecture(Subject subject, int semester, int year, int volume){

List <Stream> result = new ArrayList<Stream>();

try {

int idSubject = subject.getId();

String querry = "SELECT \* FROM Stream WHERE idSubject=? AND (SEMESTER=? AND yearReception=?) AND volume=?";

PreparedStatement ps = con.prepareStatement(querry);

ps.setInt(1, idSubject);

ps.setInt(2, semester);

ps.setInt(3, year);

ps.setInt(4, volume);

ResultSet rs = ps.executeQuery();

while(rs.next()){

result.add(getStreamFromRS(rs));

}

} catch (SQLException e) {

System.out.println("Method searchStreamsForThisLecture()");

e.printStackTrace();

}

return result;

}

public List<Stream> getStreamList(){

List <Stream> result = new ArrayList<Stream>();

try {

ResultSet rs = con.createStatement().executeQuery("SELECT \* FROM Stream");

while(rs.next()){

result.add(getStreamFromRS(rs));

}

rs.close();

} catch (SQLException e) {

System.out.println("Method getStreamList()");

e.printStackTrace();

}

return result;

}

}

### Файл SubjectDAO.java

package DataAccessObject;

import DatabaseEntities.Subject;

import java.sql.Connection;

import java.sql.PreparedStatement;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.util.ArrayList;

import java.util.List;

/\*\*

\* Created by Dmitriy on 12.06.2016.

\*/

public class SubjectDAO {

private final Connection con;

public SubjectDAO(Connection connection) {this.con = connection;}

private Subject getSubjectFromRS(ResultSet rs){

Subject result = new Subject();

try {

result.setId(rs.getInt("id"));

result.setName(rs.getString("name"));

result.setAbbreviation(rs.getString("abbreviation"));

result.setIsArchive(rs.getBoolean("isArchive"));

result.setArchiveDate(rs.getDate("archiveDate"));

} catch (SQLException e) {

System.out.println("Method getSubjectFromRS()");

e.printStackTrace();

}

return result;

}

public Subject getSubject(int id){

Subject result = null;

try {

PreparedStatement ps = con.prepareStatement("SELECT \* FROM Subject WHERE id = ? AND isArchive = 0");

ps.setInt(1, id);

ResultSet rs = ps.executeQuery();

if(rs.next()){

result = getSubjectFromRS(rs);

}

rs.close();

ps.close();

} catch (SQLException e) {

System.out.println("Method getSubject");

e.printStackTrace();

}

return result;

}

public List <Subject> getSubjectList(){

List<Subject> result = new ArrayList<Subject>();

try {

ResultSet rs = con.createStatement().executeQuery("SELECT \* FROM Subject WHERE isArchive = 0");

while (rs.next()){

result.add(getSubjectFromRS(rs));

}

rs.close();

} catch (SQLException e) {

System.out.println("Method getSubjectList()");

e.printStackTrace();

}

return result;

}

}

### Файл TimeClassDAO.java

package DataAccessObject;

import DatabaseEntities.TimeClass;

import DatabaseEntities.TypeWeek;

import DatabaseEntities.WeekDay;

import java.sql.\*;

import java.util.ArrayList;

import java.util.List;

/\*\*

\* Created by Dmitriy on 07.06.2016.

\*/

public class TimeClassDAO {

private final Connection con;

public TimeClassDAO(Connection connection) {this.con = connection;}

private TimeClass getTimeClassFromRS( ResultSet rs ){

TimeClass result = new TimeClass();

try {

result.setId(rs.getInt("id"));

result.setTypeWeek(new TypeWeekDAO(con).getTypeWeek(rs.getInt("idTypeWeek")));

result.setWeekDay(new WeekDayDAO(con).getWeekDay(rs.getInt("idWeekDay")));

result.setNumberClass(rs.getInt("numberClass"));

} catch (SQLException e) {

System.out.println("Метод getTimeClassFromRS()");

e.printStackTrace();

}

return result;

}

public TimeClass getTimeClass(int id){

TimeClass result = null;

try {

PreparedStatement ps = con.prepareStatement("SELECT \* FROM TimeClass WHERE id = ?");

ps.setInt(1, id);

ResultSet rs = ps.executeQuery();

if(rs.next()){

result = getTimeClassFromRS(rs);

}

ps.close();

} catch (SQLException e) {

System.out.println("Метод getTimeClass()");

e.printStackTrace();

}

return result;

}

public List <TimeClass> getTimeClassList(){

List <TimeClass> result = new ArrayList <TimeClass> ();

try {

ResultSet rs = con.createStatement().executeQuery("SELECT \* FROM TimeClass");

while(rs.next()){

result.add(getTimeClassFromRS(rs));

}

rs.close();

} catch (SQLException e) {

System.out.println("Метод getTimeClassList()");

e.printStackTrace();

}

return result;

}

public void updateTimeClass(TimeClass timeClass){

try{

PreparedStatement ps;

TypeWeek typeWeek = timeClass.getTypeWeek(); //извлекаем объект ТипНедели, для получения его id

WeekDay weekDay = timeClass.getWeekDay(); //извлекаем объект ДеньНедели, для получения его id

if(timeClass.getId() > 0){

String querry = "UPDATE TimeClass SET idTypeWeek = ?, idWeekDay = ?, numberClass = ? WHERE id = ?";

ps = con.prepareStatement(querry);

ps.setInt(4, timeClass.getId());

ps.setInt(1, typeWeek.getId());

ps.setInt(2, weekDay.getId());

ps.setInt(3, timeClass.getNumberClass());

}else{

String querry = "INSERT INTO TimeClass (idTypeWeek, idWeekDay, numberClass) VALUES (?, ?, ?)";

ps = con.prepareStatement(querry);

ps.setInt(1, typeWeek.getId());

ps.setInt(2, weekDay.getId());

ps.setInt(3, timeClass.getNumberClass());

}

ps.executeUpdate();

ps.close();

}catch(SQLException e){

System.out.println("Метод updateTimeClass()");

e.printStackTrace();

}

}

public void deleteTimeClass(int id){

try {

PreparedStatement ps = con.prepareStatement("DELETE FROM TimeClass WHERE id = ?");

ps.setInt(1, id);

ps.executeUpdate();

ps.close();

} catch (SQLException e) {

System.out.println("Method deleteTimeClass()");

e.printStackTrace();

}

}

public void deleteAllTimeClass(){

try {

Statement st = con.createStatement();

st.executeUpdate("DELETE FROM TimeClass");

st.close();

} catch (SQLException e) {

System.out.println("Method deleteAllTimeClass()");

e.printStackTrace();

}

}

}

### Файл TypeClassDAO.java

package DataAccessObject;

import DatabaseEntities.TypeClass;

import java.sql.Connection;

import java.sql.PreparedStatement;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.util.ArrayList;

import java.util.List;

/\*\*

\* Created by Dmitriy on 09.06.2016.

\*/

public class TypeClassDAO {

private final Connection con;

public TypeClassDAO(Connection con) {this.con = con;}

private TypeClass getTypeClassFromRS(ResultSet rs ){

TypeClass result = new TypeClass();

try{

result.setId(rs.getInt("id"));

result.setName(rs.getString("name"));

}catch (SQLException e) {

System.out.println("Метод getTypeClassFromRS()");

e.printStackTrace();

}

return result;

}

public TypeClass getTypeClass(int id){

TypeClass result = null;

try{

PreparedStatement ps = con.prepareStatement("SELECT \* FROM TypeClass WHERE id = ?");

ps.setInt( 1, id );

ResultSet rs = ps.executeQuery();

if(rs.next()){

result = getTypeClassFromRS(rs);

}

ps.close();

}catch(SQLException e){

System.out.println("Метод getTypeClass()");

e.printStackTrace();

}

return result;

}

public List<TypeClass> getTypeClassList(){

List <TypeClass> result = new ArrayList<TypeClass>();

try{

ResultSet rs = con.createStatement().executeQuery("SELECT \* FROM TypeClass");

while(rs.next()){

result.add(getTypeClassFromRS(rs));

}

rs.close();

}catch(SQLException e) {

System.out.println("getTypeClassList()");

e.printStackTrace();

}

return result;

}

}

### Файл TypeWeekDAO.java

package DataAccessObject;

import DatabaseEntities.TypeWeek;

import java.sql.Connection;

import java.sql.PreparedStatement;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.util.ArrayList;

import java.util.List;

/\*\*

\* Created by Dmitriy on 07.06.2016.

\*/

public class TypeWeekDAO {

private final Connection con;

public TypeWeekDAO(Connection connection) {

this.con = connection;

}

private TypeWeek getTypeWeekFromRS( ResultSet rs ){

TypeWeek result = new TypeWeek();

try{

result.setId(rs.getInt("id"));

result.setName(rs.getString("name"));

}catch (SQLException e) {

System.out.println("Метод getTypeWeekFromRS()");

e.printStackTrace();

}

return result;

}

public TypeWeek getTypeWeek(int id){

TypeWeek result = null;

try{

PreparedStatement ps = con.prepareStatement( "SELECT \* FROM TypeWeek WHERE id = ?" );

ps.setInt( 1, id );

ResultSet rs = ps.executeQuery();

if(rs.next()){

result = getTypeWeekFromRS( rs );

}

ps.close();

}catch(SQLException e){

System.out.println("Метод getTypeWeek()");

e.printStackTrace();

}

return result;

}

public List<TypeWeek> getTypeWeekList(){

List <TypeWeek> result = new ArrayList <TypeWeek> ();

try{

ResultSet rs = con.createStatement().executeQuery("SELECT \* FROM TypeWeek");

while(rs.next()){

result.add(getTypeWeekFromRS(rs));

}

rs.close();

}catch(SQLException e) {

System.out.println("getTypeWeekList()");

e.printStackTrace();

}

return result;

}

}

### Файл UniversityTeacherDAO.java

package DataAccessObject;

import DatabaseEntities.UniversityTeacher;

import java.sql.Connection;

import java.sql.PreparedStatement;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.util.ArrayList;

import java.util.List;

/\*\*

\* Created by Dmitriy on 11.06.2016.

\*/

public class UniversityTeacherDAO {

private final Connection con;

public UniversityTeacherDAO(Connection con) {this.con = con;}

private UniversityTeacher getUniversityTeacherFromRS(ResultSet rs){

UniversityTeacher result = new UniversityTeacher();

try {

//заглушка на извлечение Position.name

int idPosition = rs.getInt("idPosition");

PreparedStatement ps = con.prepareStatement("SELECT \* FROM Position WHERE id = ?");

ps.setInt(1, idPosition);

ResultSet rsFromPosition = ps.executeQuery();

// поехали )

result.setId(rs.getInt("id"));

result.setSecondName(rs.getString("secondName"));

result.setFirstName(rs.getString("firstName"));

result.setMiddleName(rs.getString("middleName"));

result.setDepartment(new DepartmentDAO(con).getDepartment(rs.getInt("idDepartment")).getName());

if(rsFromPosition.next()) {

result.setPosition(rsFromPosition.getString("name")); //забираем из заглушки "Должность.название"

}

result.setIsArchive(rs.getBoolean("isArchive"));

result.setArchiveDate(rs.getDate("ArchiveDate"));

rsFromPosition.close();

ps.close();

} catch (SQLException e) {

System.out.println("Method getUniversityTeacherFromRS()");

e.printStackTrace();

}

return result;

}

public UniversityTeacher getUniversityTeacher(int id){

UniversityTeacher result = null;

try {

PreparedStatement ps = con.prepareStatement("SELECT \* FROM UniversityTeacher WHERE id = ? AND isArchive = 0");

ps.setInt(1, id);

ResultSet rs = ps.executeQuery();

if(rs.next()){

result = getUniversityTeacherFromRS(rs);

}

ps.close();

} catch (SQLException e) {

System.out.println("Method getUniversityTeacher()");

e.printStackTrace();

}

return result;

}

public List <UniversityTeacher> getUniversityTeachersList(){

List <UniversityTeacher> result = new ArrayList <UniversityTeacher> ();

try {

String querry = "SELECT \* FROM UniversityTeacher WHERE isArchive = 0 ORDER BY secondName";

ResultSet rs = con.createStatement().executeQuery(querry);

while (rs.next()){

result.add(getUniversityTeacherFromRS(rs));

}

rs.close();

} catch (SQLException e) {

System.out.println("Method getUniversityTeachersList()");

e.printStackTrace();

}

return result;

}

}

### Файл WeekDayDAO.java

package DataAccessObject;

import DatabaseEntities.WeekDay;

import java.sql.Connection;

import java.sql.PreparedStatement;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.util.ArrayList;

import java.util.List;

/\*\*

\* Created by Dmitriy on 07.06.2016.

\*/

public class WeekDayDAO {

private final Connection con;

public WeekDayDAO(Connection connection) {

this.con = connection;

}

private WeekDay getWeekDayFromRS( ResultSet rs ){

WeekDay result = new WeekDay();

try{

result.setId(rs.getInt("id"));

result.setName(rs.getString("name"));

}catch (SQLException e) {

System.out.println("Метод getWeekDayFromRS()");

e.printStackTrace();

}

return result;

}

public WeekDay getWeekDay(int id){

WeekDay result = null;

try{

PreparedStatement ps = con.prepareStatement( "SELECT \* FROM WeekDay WHERE id = ?" );

ps.setInt( 1, id );

ResultSet rs = ps.executeQuery();

if(rs.next()){

result = getWeekDayFromRS( rs );

}

ps.close();

}catch(SQLException e){

System.out.println("Метод getWeekDay()");

e.printStackTrace();

}

return result;

}

public List <WeekDay> getWeekDayList(){

List <WeekDay> result = new ArrayList <WeekDay> ();

try{

ResultSet rs = con.createStatement().executeQuery("SELECT \* FROM WeekDay");

while(rs.next()){

result.add(getWeekDayFromRS(rs));

}

rs.close();

}catch(SQLException e) {

System.out.println("Метод getCarList()");

e.printStackTrace();

}

return result;

}

}

## Текст модуля FormationOfShedule

### Файл FormationSchedule.java

package Automation;

import DataAccessObject.\*;

import DatabaseEntities.\*;

import DatabaseEntities.Class;

import java.util.ArrayList;

import java.util.List;

import java.util.Random;

import static DatabaseEntities.LoadDepartment.getAllSubjects;

import static DatabaseTools.DBTools.getConnection;

/\*\*

\* Created by Dmitriy on 18.06.2016.

\*/

public class FormationSchedule{

////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////

private int scheduleYear;

private String scheduleName;

private int scheduleSemester;

private TypeClass typeClassLecture = new TypeClassDAO(getConnection()).getTypeClass(1);

private TypeClass typeClassLabWork = new TypeClassDAO(getConnection()).getTypeClass(2);

private TypeClass typeClassPracticWork = new TypeClassDAO(getConnection()).getTypeClass(3);

private LoadDepartmentDoc departmentDoc;

private LoadDepartmentDocDAO loadDepartmentDocDAO = new LoadDepartmentDocDAO(getConnection());

private List <LoadDepartment> loadDepartmentsByDocId;

private List <LoadDepartment> loadDepartmentsForOneSubject;

private LoadDepartmentDAO loadDepartmentDAO = new LoadDepartmentDAO(getConnection());

private List <Schedule> thisSchedule;

private ScheduleDAO scheduleDAO = new ScheduleDAO(getConnection());

private ClassDAO classDAO = new ClassDAO(getConnection());

private TimeClassDAO timeClassDAO = new TimeClassDAO(getConnection());

private Stream streamForLecture;

private StreamDAO streamDAO = new StreamDAO(getConnection());

private ScheduleClassesDAO scheduleClassesDAO = new ScheduleClassesDAO(getConnection());

public FormationSchedule(int year, String name){

scheduleName = name;

this.scheduleYear = year;

}

////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////

public void makeSchedule(){

//Записываем начальную информацию о создаваем расписании

//год, и два семестра по 2 четверти каждый

recordAboutNewSchedule();

//начинаем последовательн выбор предполагаем нагрузки по семестру и году

for(int indxSemester = 1; indxSemester < 3; indxSemester++){

this.scheduleSemester = indxSemester;

thisSchedule = scheduleDAO.getThisSheduleList(scheduleYear, indxSemester, scheduleName);

List<LoadDepartmentDoc> loadDepartmentDocListByYear =

loadDepartmentDocDAO.getLoadDepartmentDocListByYearAndSemester(scheduleYear, scheduleSemester);

//проходим поочередно по каждой нагрузке для этого (семестр, год)

for(LoadDepartmentDoc ldd: loadDepartmentDocListByYear){

departmentDoc = ldd;

//Извлекаем id для составления списка loadDepartmenr по полю idLoadDepartment

loadDepartmentsByDocId = loadDepartmentDAO.getLoadDepartmentListByDocId(ldd.getId());

//loadDepartmentsByDocId - Записи одной нагрузки

workWithOneLoadDepartment();

}

}

}

private void recordAboutNewSchedule(){

Schedule schedule = new Schedule();

schedule.setYearReception(scheduleYear);

schedule.setName(scheduleName);

int semesterCount = 2;

int quarter = 1;

int step;

for(int semesterIndx = 1; semesterIndx <= semesterCount; semesterIndx++){

step = 0;

while(step < 2 ){

schedule.setSemester(semesterIndx);

schedule.setQuarter(quarter++);

scheduleDAO.updateSchedule(schedule);

step = step + 1;

}

}

}

/\*\*

\* Фукнция принимает LoadDepartmentList который описывает

\* эквивалент документа "Распределение учебной нагрузки по кафедре"

\*/

private void workWithOneLoadDepartment(){

//получаем все предметы которые есть в данной нагружке

List <Subject> subjectsForOneLoad = getAllSubjects(loadDepartmentsByDocId);

boolean halfSemester;

for(Subject subject: subjectsForOneLoad){ //проходим по каждому предмету из нагрузки

List <GroupStudent> groupsForHalfSemester = new ArrayList<GroupStudent>();

List <GroupStudent> groupStudents = new ArrayList<GroupStudent>();

//выбираем записи нагрузки для конкретного предмета

loadDepartmentsForOneSubject = loadDepartmentDAO.getLoadDepartmentForSubjectByDocId(subject, departmentDoc);

//----------------------------------------------------------------------------------------------------------

halfSemester = false;

for(LoadDepartment note: loadDepartmentsForOneSubject){ //проходим по нагрузке конкретного предмета

groupStudents.add(note.getGroupStudent());

if(note.getIsHalfSemester()){

groupsForHalfSemester.add(note.getGroupStudent());

halfSemester = true;

}

}

//устанавливаем лекцию

workWithLecture(subject, groupStudents, groupsForHalfSemester, halfSemester);

//----------------------------------------------------------------------------------------------------------

//устанавливаем лабораторные

int countHourLabWork = 0;

for(LoadDepartment note: loadDepartmentsForOneSubject){ //проходим по нагрузке конкретного предмета

countHourLabWork = note.getCountHourLabWork();

if(countHourLabWork > 0){

workWithLabWork(subject, note.getGroupStudent(), note.getIsHalfSemester(),

countHourLabWork, note.getTeacherLabWork());

}

}

//----------------------------------------------------------------------------------------------------------

//устанавливаем практику

int countHourPracticWork;

for(LoadDepartment note: loadDepartmentsForOneSubject){//проходим по нагрузке конкретного предмета

countHourPracticWork = note.getCountHoutPracticWork();

if(countHourPracticWork > 0){

workWithPracticWork(subject, note.getGroupStudent(), note.getIsHalfSemester(),

countHourLabWork, note.getTeacherLabWork());

}

}

//----------------------------------------------------------------------------------------------------------

}

}

////////////////////////////////////////////////////////////////////////////////////////////////////////////////////

private void workWithLecture(Subject subject, List<GroupStudent>groupStudents,

List<GroupStudent>groupsForHalfSemester, boolean halfSemester){

int volume = 0;

for(GroupStudent group: groupStudents){ //считаем кол-во студентов для поиска соотв потока

volume = volume + group.getCountStudent();

}

Class aClass = new Class();

List <Stream> streamsForLecture =

streamDAO.searchStreamsForThisLecture(subject, scheduleSemester, scheduleYear,volume);

for(Stream stream: streamsForLecture){

if(stream.getVolume() == volume){

aClass.setStream(stream);

}

}

aClass.setSubject(subject);

aClass.setTypeClass(typeClassLecture);

aClass.setUniversityTeacher(loadDepartmentsForOneSubject.get(0).getTeacherLecture());

int countHourLecture = loadDepartmentsForOneSubject.get(0).getCountHourLecture();

setTime(countHourLecture, halfSemester, aClass);

}

//------------------------------------------------------------------------------------------------------------------

private void workWithLabWork(Subject subject, GroupStudent groupStudents,

boolean isHalfSemester, int countHourLabWork, UniversityTeacher teacher){

int volume = groupStudents.getCountStudent();

Class aClass = new Class();

List <Stream> streamsForDiscipline =

streamDAO.searchStreamsForThisLecture(subject, scheduleSemester, scheduleYear,volume);

List <GroupStudent> groupStudentList;

for(Stream stream: streamsForDiscipline){

if(stream.getVolume() == volume){

groupStudentList = stream.getGroupStudentList();

if(groupStudentList.get(0).getId() == groupStudents.getId()){

aClass.setStream(stream);

}

}

}

aClass.setTypeClass(typeClassLabWork);

aClass.setSubject(subject);

aClass.setUniversityTeacher(teacher);

setTime(countHourLabWork, isHalfSemester, aClass);

}

//------------------------------------------------------------------------------------------------------------------

private void workWithPracticWork(Subject subject, GroupStudent groupStudents,

boolean isHalfSemester, int countHourPracticWork, UniversityTeacher teacher){

int volume = groupStudents.getCountStudent();

Class aClass = new Class();

List <Stream> streamsForDiscipline =

streamDAO.searchStreamsForThisLecture(subject, scheduleSemester, scheduleYear,volume);

List <GroupStudent> groupStudentList;

for(Stream stream: streamsForDiscipline){

if(stream.getVolume() == volume){

groupStudentList = stream.getGroupStudentList();

if(groupStudentList.get(0).getId() == groupStudents.getId()){

aClass.setStream(stream);

}

}

}

aClass.setTypeClass(typeClassPracticWork);

aClass.setSubject(subject);

aClass.setUniversityTeacher(teacher);

setTime(countHourPracticWork, isHalfSemester, aClass);

}

////////////////////////////////////////////////////////////////////////////////////////////////////////////////////

private void setTime(int countHourLecture, boolean isHalfSemester, Class aClass){

TimeClass timeClass = null;

Random rand = new Random(System.currentTimeMillis());

int randomVolume;

List<TimeClass> timeClasses = timeClassDAO.getTimeClassList();

int quorter1 = thisSchedule.get(0).getQuarter();

int quorter2 = thisSchedule.get(1).getQuarter();

int randQuorter;

ScheduleClasses scheduleClasses = new ScheduleClasses();

boolean checkStandardLimitsQuorter1;

boolean checkStandardLimitsQuorter2;

for(int i = 0; i < countHourLecture; i++) {

randomVolume = rand.nextInt(timeClasses.size());

//выбираем время для проведения занятия

timeClass = timeClasses.get(randomVolume);

if (isHalfSemester) {

//выбираем случайным образом (1/2)(3/4) четверть

randQuorter = thisSchedule.get(rand.nextInt(2)).getQuarter();

//выполняем проверку для этого занятия в данной четверти

checkStandardLimitsQuorter1 = checkForStandardLimits(

timeClass, aClass.getUniversityTeacher(), aClass.getStream(), randQuorter);

if (checkStandardLimitsQuorter1) {

aClass.setTimeClass(timeClass);

classDAO.updateClass(aClass);

int idClass = classDAO.getIdForLastRecord();

//добавляем только в 1 выбранную случайно четверть

scheduleClasses.setIdClass(idClass);

scheduleClasses.setIdSchedule(thisSchedule.get(randQuorter).getId());

scheduleClassesDAO.updateScheduleClasses(scheduleClasses);

} else {

timeClasses.remove(randomVolume);

i--;

}

} else {

//занятие будет длиться 2 четверти

//проверяем можно ли исп. наше time одновременно для двух четвертей

checkStandardLimitsQuorter1 = checkForStandardLimits(

timeClass, aClass.getUniversityTeacher(), aClass.getStream(), quorter1);

checkStandardLimitsQuorter2 = checkForStandardLimits(

timeClass, aClass.getUniversityTeacher(), aClass.getStream(), quorter2);

if (checkStandardLimitsQuorter1 && checkStandardLimitsQuorter2) {

aClass.setTimeClass(timeClass);

classDAO.updateClass(aClass);

int idClass = classDAO.getIdForLastRecord();

//добавляем в 1-ю четверть

scheduleClasses.setIdClass(idClass);

scheduleClasses.setIdSchedule(thisSchedule.get(0).getId());

scheduleClassesDAO.updateScheduleClasses(scheduleClasses);

//добавляем в 2-ю четветр

scheduleClasses.setIdClass(idClass);

scheduleClasses.setIdSchedule(thisSchedule.get(1).getId());

scheduleClassesDAO.updateScheduleClasses(scheduleClasses);

} else {

timeClasses.remove(randomVolume);

i--;

}

}

}

}

////////////////////////////////////////////////////////////////////////////////////////////////////////////////////

/\*\*

\* Метод проверяющий что бы не поставить

\* 1 преподавателю: 2 занятия в одно и тоже время

\* 1 группе: 2 занятия в одно и тоже время

\* @param timeClass

\* @param teacher

\* @param stream

\* @param quarter

\* @return возвращает true/false в случае если (нет нарушения/есть нарушение)

\*/

private boolean checkForStandardLimits(TimeClass timeClass, UniversityTeacher teacher, Stream stream, int quarter){

boolean result = true;

List <GroupStudent> students = stream.getGroupStudentList();

List <GroupStudent> studentsFromClass;

List <Class> classes;

Stream streamFromClass;

List <Schedule> scheduleInDB = new ScheduleDAO(getConnection()).

getThisSheduleList(scheduleYear,scheduleSemester,scheduleName, quarter);

for(Schedule noteSchedule: scheduleInDB){

classes = noteSchedule.getClassList();

for(Class noteClass: classes){

if(noteClass.getTimeClass().getId() == timeClass.getId()){

if(noteClass.getUniversityTeacher().getId() == teacher.getId()){

System.out.println("Cтандартное ограничение:");

System.out.println("noteClass" + noteClass.getTimeClass().getId());

System.out.println("timeClass" + timeClass.getId());

System.out.println("noteTeacher" + noteClass.getUniversityTeacher().getId()); //todo поменять что бы выводило преподавателя

System.out.println("teacher" + teacher.getId());

result = false;

break;

}

streamFromClass = noteClass.getStream();

studentsFromClass = streamFromClass.getGroupStudentList();

for(GroupStudent groupFromClass :studentsFromClass){

for(GroupStudent group: students){

if(groupFromClass.getId() == group.getId()){

System.out.println("Cтандартное ограничение:");

System.out.println("noteClass:\t" + noteClass.getTimeClass().getId());

System.out.println("timeClass:\t" + timeClass.getId());

System.out.println("groupFromClass:\t" + groupFromClass.getId());

System.out.println("group:\t" + group.getId());

result = false;

break;

}

}

}

}

}

}

return result;

}

}

## Текст модуля UI

### Файл AddNewRequirementForm.java

package UI;

import java.sql.Connection;

import java.util.ArrayList;

import java.util.Calendar;

import java.util.List;

import javax.swing.DefaultListModel;

import javax.swing.JOptionPane;

import DataAccessObject.RequirementsDAO;

import DataAccessObject.RequirementsUniversityTeacherDAO;

import DataAccessObject.UniversityTeacherDAO;

import DataAccessObject.WeekDayDAO;

import DatabaseEntities.Requirements;

import DatabaseEntities.RequirementsUniversityTeacher;

import DatabaseEntities.UniversityTeacher;

import DatabaseEntities.WeekDay;

import static DatabaseTools.DBTools.getConnection;

/\*\*

\*

\* @author Dmitriy

\*/

public class AddNewRequirementForm extends javax.swing.JFrame {

Connection con = getConnection();

private UniversityTeacher universityTeacher;

List<UniversityTeacher> universityTeacherList =

new ArrayList<UniversityTeacher>();

private UniversityTeacherDAO universityTeacherDAO =

new UniversityTeacherDAO(con);

private Requirements requirements;

private List<Requirements> requirementsList =

new ArrayList<Requirements>();

private RequirementsDAO requirementsDAO =

new RequirementsDAO(con);

private WeekDay weekDay;

private List <WeekDay> weekDayList = new ArrayList<WeekDay>();

private WeekDayDAO weekDayDAO = new WeekDayDAO(con);

private RequirementsUniversityTeacherDAO rutDAO =

new RequirementsUniversityTeacherDAO(con);

/\*\*

\* Creates new form AddNewRequirement

\*/

public AddNewRequirementForm() {

initComponents();

this.setLocationRelativeTo(null);

DefaultListModel dlm = new DefaultListModel();

requirementsJList.setModel(dlm);

universityTeacherList = universityTeacherDAO.getUniversityTeachersList();

requirementsList = requirementsDAO.getRequirementsList();

weekDayList = weekDayDAO.getWeekDayList();

for(UniversityTeacher ut: universityTeacherList){

teacherComboBox.addItem(ut.getAllTeacherName());

}

for(Requirements r: requirementsList){

dlm.addElement(r.getName());

}

}

/\*\*

\* This method is called from within the constructor to initialize the form.

\* WARNING: Do NOT modify this code. The content of this method is always

\* regenerated by the Form Editor.

\*/

@SuppressWarnings("unchecked")

// <editor-fold defaultstate="collapsed" desc="Generated Code">//GEN-BEGIN:initComponents

private void initComponents() {

buttonGroup1 = new javax.swing.ButtonGroup();

buttonGroup2 = new javax.swing.ButtonGroup();

buttonGroup3 = new javax.swing.ButtonGroup();

jLabel1 = new javax.swing.JLabel();

teacherComboBox = new javax.swing.JComboBox<>();

saveRequitementButton = new javax.swing.JButton();

cancell = new javax.swing.JButton();

jPanel2 = new javax.swing.JPanel();

jLabel3 = new javax.swing.JLabel();

semester1CheckBox = new javax.swing.JCheckBox();

semester2CheckBox = new javax.swing.JCheckBox();

jPanel3 = new javax.swing.JPanel();

jLabel4 = new javax.swing.JLabel();

lesson1CheckBox = new javax.swing.JCheckBox();

lesson2CheckBox = new javax.swing.JCheckBox();

lesson3CheckBox = new javax.swing.JCheckBox();

lesson4CheckBox = new javax.swing.JCheckBox();

lesson5CheckBox = new javax.swing.JCheckBox();

lesson6CheckBox = new javax.swing.JCheckBox();

jPanel4 = new javax.swing.JPanel();

jScrollPane1 = new javax.swing.JScrollPane();

requirementsJList = new javax.swing.JList<>();

jLabel5 = new javax.swing.JLabel();

jPanel5 = new javax.swing.JPanel();

jLabel2 = new javax.swing.JLabel();

day1CheckBox = new javax.swing.JCheckBox();

day2CheckBox = new javax.swing.JCheckBox();

day3CheckBox = new javax.swing.JCheckBox();

day4CheckBox = new javax.swing.JCheckBox();

day5CheckBox = new javax.swing.JCheckBox();

Clear = new javax.swing.JButton();

setDefaultCloseOperation(javax.swing.WindowConstants.DISPOSE\_ON\_CLOSE);

setLocation(new java.awt.Point(0, 0));

setMinimumSize(null);

setSize(new java.awt.Dimension(0, 0));

addWindowListener(new java.awt.event.WindowAdapter() {

public void windowOpened(java.awt.event.WindowEvent evt) {

formWindowOpened(evt);

}

});

jLabel1.setFont(new java.awt.Font("Tahoma", 0, 18)); // NOI18N

jLabel1.setText("Преподаватель:");

teacherComboBox.setFont(new java.awt.Font("Tahoma", 0, 18)); // NOI18N

saveRequitementButton.setFont(new java.awt.Font("Tahoma", 0, 18)); // NOI18N

saveRequitementButton.setLabel("Сохранить пожелание");

saveRequitementButton.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

saveRequitementButtonActionPerformed(evt);

}

});

cancell.setFont(new java.awt.Font("Tahoma", 0, 18)); // NOI18N

cancell.setLabel("Отмена");

cancell.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

cancellActionPerformed(evt);

}

});

jPanel2.setBorder(javax.swing.BorderFactory.createMatteBorder(1, 1, 1, 1, new java.awt.Color(0, 102, 102)));

jLabel3.setFont(new java.awt.Font("Tahoma", 0, 18)); // NOI18N

jLabel3.setText("Семестр:");

semester1CheckBox.setFont(new java.awt.Font("Tahoma", 0, 14)); // NOI18N

semester1CheckBox.setLabel("Первый");

semester2CheckBox.setFont(new java.awt.Font("Tahoma", 0, 14)); // NOI18N

semester2CheckBox.setLabel("Второй");

javax.swing.GroupLayout jPanel2Layout = new javax.swing.GroupLayout(jPanel2);

jPanel2.setLayout(jPanel2Layout);

jPanel2Layout.setHorizontalGroup(

jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(jPanel2Layout.createSequentialGroup()

.addContainerGap()

.addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addComponent(semester2CheckBox)

.addComponent(jLabel3)

.addComponent(semester1CheckBox))

.addContainerGap(30, Short.MAX\_VALUE))

);

jPanel2Layout.setVerticalGroup(

jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(jPanel2Layout.createSequentialGroup()

.addContainerGap()

.addComponent(jLabel3)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addComponent(semester1CheckBox)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addComponent(semester2CheckBox)

.addContainerGap(javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE))

);

jPanel3.setBorder(javax.swing.BorderFactory.createMatteBorder(1, 1, 1, 1, new java.awt.Color(0, 102, 102)));

jPanel3.setFont(new java.awt.Font("Tahoma", 0, 14)); // NOI18N

jLabel4.setFont(new java.awt.Font("Tahoma", 0, 18)); // NOI18N

jLabel4.setText("Номер занятия:");

lesson1CheckBox.setFont(new java.awt.Font("Tahoma", 0, 14)); // NOI18N

lesson1CheckBox.setLabel("1");

lesson2CheckBox.setFont(new java.awt.Font("Tahoma", 0, 14)); // NOI18N

lesson2CheckBox.setLabel("2");

lesson3CheckBox.setFont(new java.awt.Font("Tahoma", 0, 14)); // NOI18N

lesson3CheckBox.setLabel("3");

lesson4CheckBox.setFont(new java.awt.Font("Tahoma", 0, 14)); // NOI18N

lesson4CheckBox.setLabel("4");

lesson5CheckBox.setFont(new java.awt.Font("Tahoma", 0, 14)); // NOI18N

lesson5CheckBox.setLabel("5");

lesson6CheckBox.setFont(new java.awt.Font("Tahoma", 0, 14)); // NOI18N

lesson6CheckBox.setLabel("6");

javax.swing.GroupLayout jPanel3Layout = new javax.swing.GroupLayout(jPanel3);

jPanel3.setLayout(jPanel3Layout);

jPanel3Layout.setHorizontalGroup(

jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(jPanel3Layout.createSequentialGroup()

.addContainerGap()

.addGroup(jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addComponent(lesson6CheckBox)

.addComponent(lesson5CheckBox)

.addComponent(lesson4CheckBox)

.addComponent(lesson3CheckBox)

.addComponent(lesson2CheckBox)

.addComponent(jLabel4)

.addComponent(lesson1CheckBox))

.addContainerGap(javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE))

);

jPanel3Layout.setVerticalGroup(

jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(jPanel3Layout.createSequentialGroup()

.addContainerGap()

.addComponent(jLabel4)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addComponent(lesson1CheckBox)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addComponent(lesson2CheckBox)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addComponent(lesson3CheckBox)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addComponent(lesson4CheckBox)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addComponent(lesson5CheckBox)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addComponent(lesson6CheckBox)

.addContainerGap(27, Short.MAX\_VALUE))

);

jPanel4.setBorder(javax.swing.BorderFactory.createMatteBorder(1, 1, 1, 1, new java.awt.Color(0, 102, 102)));

jPanel4.setFont(new java.awt.Font("Tahoma", 0, 14)); // NOI18N

requirementsJList.setFont(new java.awt.Font("Tahoma", 0, 18)); // NOI18N

requirementsJList.setSelectionMode(javax.swing.ListSelectionModel.SINGLE\_SELECTION);

jScrollPane1.setViewportView(requirementsJList);

requirementsJList.getAccessibleContext().setAccessibleParent(this);

jLabel5.setFont(new java.awt.Font("Tahoma", 0, 18)); // NOI18N

jLabel5.setText("Пожелание:");

javax.swing.GroupLayout jPanel4Layout = new javax.swing.GroupLayout(jPanel4);

jPanel4.setLayout(jPanel4Layout);

jPanel4Layout.setHorizontalGroup(

jPanel4Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(jPanel4Layout.createSequentialGroup()

.addContainerGap()

.addGroup(jPanel4Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(jPanel4Layout.createSequentialGroup()

.addComponent(jLabel5)

.addGap(0, 60, Short.MAX\_VALUE))

.addComponent(jScrollPane1, javax.swing.GroupLayout.Alignment.TRAILING, javax.swing.GroupLayout.PREFERRED\_SIZE, 0, Short.MAX\_VALUE))

.addContainerGap())

);

jPanel4Layout.setVerticalGroup(

jPanel4Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(javax.swing.GroupLayout.Alignment.TRAILING, jPanel4Layout.createSequentialGroup()

.addContainerGap()

.addComponent(jLabel5)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addComponent(jScrollPane1)

.addContainerGap())

);

jPanel5.setBorder(javax.swing.BorderFactory.createMatteBorder(1, 1, 1, 1, new java.awt.Color(0, 102, 102)));

jLabel2.setFont(new java.awt.Font("Tahoma", 0, 18)); // NOI18N

jLabel2.setText("День:");

day1CheckBox.setFont(new java.awt.Font("Tahoma", 0, 14)); // NOI18N

day1CheckBox.setLabel("Понедельник");

day2CheckBox.setFont(new java.awt.Font("Tahoma", 0, 14)); // NOI18N

day2CheckBox.setLabel("Вторник");

day3CheckBox.setFont(new java.awt.Font("Tahoma", 0, 14)); // NOI18N

day3CheckBox.setLabel("Среда");

day4CheckBox.setFont(new java.awt.Font("Tahoma", 0, 14)); // NOI18N

day4CheckBox.setLabel("Четверг");

day5CheckBox.setFont(new java.awt.Font("Tahoma", 0, 14)); // NOI18N

day5CheckBox.setLabel("Пятница");

javax.swing.GroupLayout jPanel5Layout = new javax.swing.GroupLayout(jPanel5);

jPanel5.setLayout(jPanel5Layout);

jPanel5Layout.setHorizontalGroup(

jPanel5Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(jPanel5Layout.createSequentialGroup()

.addContainerGap()

.addGroup(jPanel5Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addComponent(day5CheckBox)

.addComponent(day4CheckBox)

.addComponent(day3CheckBox)

.addComponent(day2CheckBox)

.addComponent(jLabel2)

.addComponent(day1CheckBox))

.addContainerGap(16, Short.MAX\_VALUE))

);

jPanel5Layout.setVerticalGroup(

jPanel5Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(jPanel5Layout.createSequentialGroup()

.addContainerGap()

.addComponent(jLabel2)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addComponent(day1CheckBox)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addComponent(day2CheckBox)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addComponent(day3CheckBox)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addComponent(day4CheckBox)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addComponent(day5CheckBox)

.addContainerGap(javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE))

);

Clear.setFont(new java.awt.Font("Tahoma", 0, 18)); // NOI18N

Clear.setLabel("Сброс");

Clear.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

ClearActionPerformed(evt);

}

});

javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());

getContentPane().setLayout(layout);

layout.setHorizontalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addContainerGap()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addComponent(jLabel1, javax.swing.GroupLayout.PREFERRED\_SIZE, 130, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addComponent(teacherComboBox, 0, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE))

.addGroup(layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING, false)

.addGroup(layout.createSequentialGroup()

.addComponent(jPanel5, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addComponent(jPanel2, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addComponent(saveRequitementButton, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING, false)

.addComponent(jPanel3, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)

.addComponent(Clear, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addComponent(cancell, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)

.addComponent(jPanel4, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE))))

.addContainerGap())

);

layout.setVerticalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addContainerGap()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(jLabel1)

.addComponent(teacherComboBox, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addComponent(jPanel5, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)

.addComponent(jPanel2, javax.swing.GroupLayout.Alignment.TRAILING, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)

.addComponent(jPanel4, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)

.addComponent(jPanel3, javax.swing.GroupLayout.Alignment.TRAILING, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(Clear)

.addComponent(cancell)

.addComponent(saveRequitementButton))

.addContainerGap())

);

pack();

}// </editor-fold>//GEN-END:initComponents

private void cancellActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event\_cancellActionPerformed

;

dispose();

}//GEN-LAST:event\_cancellActionPerformed

private void saveRequitementButtonActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event\_saveRequitementButtonActionPerformed

if(requirementsJList.getSelectedIndex() < 0){

String message = "Выберите пожалуйста пожелание.";

JOptionPane.showMessageDialog(null, message);

}else{

boolean checkDayEntered = false;

boolean checkSemesterEntered = false;

boolean checkNumLessonEntered = false;

Boolean [] days = {false,false,false,false,false};

Boolean [] semesters = {false,false};

Boolean [] numLessons = {false,false,false,false,false,false};

//записываем какие дни недели выбраны-------------------------------

if(day1CheckBox.isSelected()){

days[0] = true;

checkDayEntered = true;

}

if(day2CheckBox.isSelected()){

days[1] = true;

checkDayEntered = true;

}

if(day3CheckBox.isSelected()){

days[2] = true;

checkDayEntered = true;

}

if(day4CheckBox.isSelected()){

days[3] = true;

checkDayEntered = true;

}

if(day5CheckBox.isSelected()){

days[4] = true;

checkDayEntered = true;

}

//Записываем какие семестры выбраны---------------------------------

if(semester1CheckBox.isSelected()){

checkSemesterEntered = true;

semesters[0] = true;

}

if(semester2CheckBox.isSelected()){

checkSemesterEntered = true;

semesters[1] = true;

}

//Записываем какие номера занятий выбраны---------------------------

if(lesson1CheckBox.isSelected()){

checkNumLessonEntered = true;

numLessons[0] = true;

}

if(lesson2CheckBox.isSelected()){

checkNumLessonEntered = true;

numLessons[1] = true;

}

if(lesson3CheckBox.isSelected()){

checkNumLessonEntered = true;

numLessons[2] = true;

}

if(lesson4CheckBox.isSelected()){

checkNumLessonEntered = true;

numLessons[3] = true;

}

if(lesson5CheckBox.isSelected()){

checkNumLessonEntered = true;

numLessons[4] = true;

}

if(lesson6CheckBox.isSelected()){

checkNumLessonEntered = true;

numLessons[5] = true;

}

boolean closeOperation = false;

String messageDay = "Выберите день недели.\n";

String messageSemester = "Выберите семестр.\n";

String messageNumLesson = "Выберите номер занятия.\n";

String allMessage = "Внимание:\n";

if(checkDayEntered == false){

allMessage = allMessage.concat(messageDay);

closeOperation = true;

}

if(checkSemesterEntered == false){

allMessage = allMessage.concat(messageSemester);

closeOperation = true;

}

if(checkNumLessonEntered == false){

allMessage = allMessage.concat(messageNumLesson);

closeOperation = true;

}

if(closeOperation){

JOptionPane.showMessageDialog(null, allMessage);

}else{

RequirementsUniversityTeacher rut = new RequirementsUniversityTeacher();

Calendar calendar = Calendar.getInstance();

universityTeacher = universityTeacherList.

get(teacherComboBox.getSelectedIndex());

requirements = requirementsList.get(requirementsJList.getSelectedIndex());

rut.setUniversityTeacher(universityTeacher);

rut.setYearReception(calendar.get(Calendar.YEAR));

rut.setRequirements(requirements);

rut.setIsArchive(false);

for(int indxDay = 0; indxDay < 5; indxDay++){

for(int indxSemester = 0; indxSemester < 2; indxSemester++){

for(int indxLesson = 0; indxLesson < 6; indxLesson++){

if(days[indxDay] &&(semesters[indxSemester])&& numLessons[indxLesson]){

rut.setSemester(indxSemester + 1);

weekDay = weekDayList.get(indxDay);

rut.setWeekDay(weekDay);

rut.setNumberClass(indxLesson + 1);

rutDAO.updateRequirementsUniversityTeacher(rut);

}

}

}

}

JOptionPane.showMessageDialog(null, "Готово!");

}

}

}//GEN-LAST:event\_saveRequitementButtonActionPerformed

private void ClearActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event\_ClearActionPerformed

day1CheckBox.setSelected(false);

day2CheckBox.setSelected(false);

day3CheckBox.setSelected(false);

day4CheckBox.setSelected(false);

day5CheckBox.setSelected(false);

semester1CheckBox.setSelected(false);

semester2CheckBox.setSelected(false);

lesson1CheckBox.setSelected(false);

lesson2CheckBox.setSelected(false);

lesson3CheckBox.setSelected(false);

lesson4CheckBox.setSelected(false);

lesson5CheckBox.setSelected(false);

lesson6CheckBox.setSelected(false);

}//GEN-LAST:event\_ClearActionPerformed

private void formWindowOpened(java.awt.event.WindowEvent evt) {//GEN-FIRST:event\_formWindowOpened

this.setSize(622, 326);

}//GEN-LAST:event\_formWindowOpened

public static void main() {

/\* Set the Nimbus look and feel \*/

//<editor-fold defaultstate="collapsed" desc=" Look and feel setting code (optional) ">

/\* If Nimbus (introduced in Java SE 6) is not available, stay with the default look and feel.

\* For details see http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html

\*/

try {

for (javax.swing.UIManager.LookAndFeelInfo info : javax.swing.UIManager.getInstalledLookAndFeels()) {

if ("Nimbus".equals(info.getName())) {

javax.swing.UIManager.setLookAndFeel(info.getClassName());

break;

}

}

} catch (ClassNotFoundException ex) {

java.util.logging.Logger.getLogger(AddNewRequirementForm.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (InstantiationException ex) {

java.util.logging.Logger.getLogger(AddNewRequirementForm.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (IllegalAccessException ex) {

java.util.logging.Logger.getLogger(AddNewRequirementForm.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (javax.swing.UnsupportedLookAndFeelException ex) {

java.util.logging.Logger.getLogger(AddNewRequirementForm.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

}

//</editor-fold>

//</editor-fold>

/\* Create and display the form \*/

java.awt.EventQueue.invokeLater(new Runnable() {

public void run() {

new AddNewRequirementForm().setVisible(true);

}

});

}

// Variables declaration - do not modify//GEN-BEGIN:variables

private javax.swing.JButton Clear;

private javax.swing.ButtonGroup buttonGroup1;

private javax.swing.ButtonGroup buttonGroup2;

private javax.swing.ButtonGroup buttonGroup3;

private javax.swing.JButton cancell;

private javax.swing.JCheckBox day1CheckBox;

private javax.swing.JCheckBox day2CheckBox;

private javax.swing.JCheckBox day3CheckBox;

private javax.swing.JCheckBox day4CheckBox;

private javax.swing.JCheckBox day5CheckBox;

private javax.swing.JLabel jLabel1;

private javax.swing.JLabel jLabel2;

private javax.swing.JLabel jLabel3;

private javax.swing.JLabel jLabel4;

private javax.swing.JLabel jLabel5;

private javax.swing.JPanel jPanel2;

private javax.swing.JPanel jPanel3;

private javax.swing.JPanel jPanel4;

private javax.swing.JPanel jPanel5;

private javax.swing.JScrollPane jScrollPane1;

private javax.swing.JCheckBox lesson1CheckBox;

private javax.swing.JCheckBox lesson2CheckBox;

private javax.swing.JCheckBox lesson3CheckBox;

private javax.swing.JCheckBox lesson4CheckBox;

private javax.swing.JCheckBox lesson5CheckBox;

private javax.swing.JCheckBox lesson6CheckBox;

private javax.swing.JList<String> requirementsJList;

private javax.swing.JButton saveRequitementButton;

private javax.swing.JCheckBox semester1CheckBox;

private javax.swing.JCheckBox semester2CheckBox;

private javax.swing.JComboBox<String> teacherComboBox;

// End of variables declaration//GEN-END:variables

}

### Файл LoginForm.java

/\*

\* To change this license header, choose License Headers in Project Properties.

\* To change this template file, choose Tools | Templates

\* and open the template in the editor.

\*/

package UI;

import javax.swing.JOptionPane;

import static DatabaseTools.DBTools.\*;

/\*\*

\*

\* @author Dmitriy

\*/

public class LoginForm extends javax.swing.JFrame {

/\*\*

\* Creates new form LoginForm

\* @param tittle название формы

\*/

public LoginForm(String tittle) {

super(tittle);

initComponents();

}

/\*\*

\* This method is called from within the constructor to initialize the form.

\* WARNING: Do NOT modify this code. The content of this method is always

\* regenerated by the Form Editor.

\*/

@SuppressWarnings("unchecked")

// <editor-fold defaultstate="collapsed" desc="Generated Code">//GEN-BEGIN:initComponents

private void initComponents() {

jLabel1 = new javax.swing.JLabel();

jLabel2 = new javax.swing.JLabel();

rememberPasLabel = new javax.swing.JLabel();

enterButton = new javax.swing.JButton();

extiButton = new javax.swing.JButton();

jProgressBar1 = new javax.swing.JProgressBar();

loginField = new javax.swing.JTextField();

passwordField = new javax.swing.JPasswordField();

setDefaultCloseOperation(javax.swing.WindowConstants.EXIT\_ON\_CLOSE);

setTitle("Аутентификация");

setFont(new java.awt.Font("Comic Sans MS", 2, 14)); // NOI18N

setMaximumSize(null);

setMinimumSize(null);

setName("LoginForm"); // NOI18N

setUndecorated(true);

setResizable(false);

jLabel1.setFont(new java.awt.Font("Comic Sans MS", 0, 18)); // NOI18N

jLabel1.setLabelFor(loginField);

jLabel1.setText("Имя пользователя:");

jLabel2.setFont(new java.awt.Font("Comic Sans MS", 0, 18)); // NOI18N

jLabel2.setLabelFor(passwordField);

jLabel2.setText("Пароль:");

rememberPasLabel.setFont(new java.awt.Font("Consolas", 0, 18)); // NOI18N

rememberPasLabel.setForeground(new java.awt.Color(204, 0, 0));

rememberPasLabel.setText("Забыли пароль?");

rememberPasLabel.addMouseListener(new java.awt.event.MouseAdapter() {

public void mouseClicked(java.awt.event.MouseEvent evt) {

rememberPasLabelMouseClicked(evt);

}

});

enterButton.setFont(new java.awt.Font("Tahoma", 0, 18)); // NOI18N

enterButton.setIcon(new javax.swing.ImageIcon("D:\\cloud\\Программирование\\ДИПЛОМ\\ScheduleMaker\\src\\project\\images\\play1.png")); // NOI18N

enterButton.setText("Войти");

enterButton.setBorder(javax.swing.BorderFactory.createEtchedBorder(new java.awt.Color(0, 204, 204), new java.awt.Color(0, 153, 153)));

enterButton.setBorderPainted(false);

enterButton.setContentAreaFilled(false);

enterButton.setRolloverIcon(new javax.swing.ImageIcon("D:\\cloud\\Программирование\\ДИПЛОМ\\ScheduleMaker\\src\\project\\images\\play2.png")); // NOI18N

enterButton.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

enterButtonActionPerformed(evt);

}

});

extiButton.setFont(new java.awt.Font("Tahoma", 0, 18)); // NOI18N

extiButton.setIcon(new javax.swing.ImageIcon("D:\\cloud\\Программирование\\ДИПЛОМ\\ScheduleMaker\\src\\project\\images\\exit1.png")); // NOI18N

extiButton.setText("Выйти");

extiButton.setBorder(javax.swing.BorderFactory.createEmptyBorder(1, 1, 1, 1));

extiButton.setContentAreaFilled(false);

extiButton.setRolloverIcon(new javax.swing.ImageIcon("D:\\cloud\\Программирование\\ДИПЛОМ\\ScheduleMaker\\src\\project\\images\\exit2.png")); // NOI18N

extiButton.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

extiButtonActionPerformed(evt);

}

});

jProgressBar1.setBackground(new java.awt.Color(0, 153, 153));

jProgressBar1.setForeground(new java.awt.Color(0, 153, 153));

jProgressBar1.setIndeterminate(true);

loginField.setFont(new java.awt.Font("Tahoma", 0, 18)); // NOI18N

loginField.setForeground(new java.awt.Color(0, 102, 102));

passwordField.setFont(new java.awt.Font("Tahoma", 0, 18)); // NOI18N

passwordField.setForeground(new java.awt.Color(0, 102, 102));

javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());

getContentPane().setLayout(layout);

layout.setHorizontalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addGap(20, 20, 20)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING, false)

.addGroup(layout.createSequentialGroup()

.addComponent(rememberPasLabel, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)

.addGap(64, 64, 64)

.addComponent(enterButton, javax.swing.GroupLayout.PREFERRED\_SIZE, 100, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addComponent(jLabel2)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING, false)

.addComponent(passwordField, javax.swing.GroupLayout.Alignment.LEADING)

.addComponent(loginField, javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addComponent(jLabel1)

.addGap(100, 100, 100)

.addComponent(extiButton, javax.swing.GroupLayout.PREFERRED\_SIZE, 85, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addGap(8, 8, 8)))))

.addContainerGap(javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE))

.addComponent(jProgressBar1, javax.swing.GroupLayout.Alignment.TRAILING, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)

);

layout.setVerticalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addGap(51, 51, 51)

.addComponent(jLabel1))

.addGroup(layout.createSequentialGroup()

.addGap(19, 19, 19)

.addComponent(extiButton, javax.swing.GroupLayout.PREFERRED\_SIZE, 35, javax.swing.GroupLayout.PREFERRED\_SIZE)))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addComponent(loginField, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)

.addComponent(jLabel2)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addComponent(passwordField, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addGap(30, 30, 30)

.addComponent(rememberPasLabel))

.addGroup(layout.createSequentialGroup()

.addGap(19, 19, 19)

.addComponent(enterButton, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)))

.addGap(5, 5, 5)

.addComponent(jProgressBar1, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE))

);

pack();

setLocationRelativeTo(null);

}// </editor-fold>//GEN-END:initComponents

private void extiButtonActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event\_extiButtonActionPerformed

System.exit(0);

}//GEN-LAST:event\_extiButtonActionPerformed

private void enterButtonActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event\_enterButtonActionPerformed

jProgressBar1.setVisible(true);

enterButton.setEnabled(false);

String userName = loginField.getText();

String userPassword = new String (passwordField.getPassword());

System.out.println(userName);

System.out.println(userPassword);

boolean check = connectToDatabase(userName, userPassword);

if (check){

String message = "База данных успешно подключена. \n"

+ "Желаем вам приятного дня!";

JOptionPane.showMessageDialog(null, message);

jProgressBar1.setVisible(false);

MainForm.main();

dispose();

}else{

String message = "Неверно введер логин / пароль . . .";

JOptionPane.showMessageDialog(null, message);

enterButton.setEnabled(true);

jProgressBar1.setVisible(false);

}

}//GEN-LAST:event\_enterButtonActionPerformed

private void rememberPasLabelMouseClicked(java.awt.event.MouseEvent evt) {//GEN-FIRST:event\_rememberPasLabelMouseClicked

String message = "Имя пользователя: admin \n"

+ "Пароль: I951753i \n"

+ "Желаем вам хорошего дня! ;)";

JOptionPane.showMessageDialog(null, message);

}//GEN-LAST:event\_rememberPasLabelMouseClicked

public static void main() {

/\* Set the Nimbus look and feel \*/

//<editor-fold defaultstate="collapsed" desc=" Look and feel setting code (optional) ">

/\* If Nimbus (introduced in Java SE 6) is not available, stay with the default look and feel.

\* For details see http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html

\*/

try {

for (javax.swing.UIManager.LookAndFeelInfo info : javax.swing.UIManager.getInstalledLookAndFeels()) {

if ("Nimbus".equals(info.getName())) {

javax.swing.UIManager.setLookAndFeel(info.getClassName());

break;

}

}

// UIManager.setLookAndFeel("com.sun.java.swing.plaf.windows.WindowsLookAndFeel");

} catch (ClassNotFoundException ex) {

java.util.logging.Logger.getLogger(LoginForm.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (InstantiationException ex) {

java.util.logging.Logger.getLogger(LoginForm.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (IllegalAccessException ex) {

java.util.logging.Logger.getLogger(LoginForm.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (javax.swing.UnsupportedLookAndFeelException ex) {

java.util.logging.Logger.getLogger(LoginForm.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

}

//</editor-fold>

/\* Create and display the form \*/

java.awt.EventQueue.invokeLater(new Runnable() {

public void run() {

LoginForm form = new LoginForm("");

form.jProgressBar1.setVisible(false);

form.setVisible(true);

}

});

}

// Variables declaration - do not modify//GEN-BEGIN:variables

private javax.swing.JButton enterButton;

private javax.swing.JButton extiButton;

private javax.swing.JLabel jLabel1;

private javax.swing.JLabel jLabel2;

private javax.swing.JProgressBar jProgressBar1;

private javax.swing.JTextField loginField;

private javax.swing.JPasswordField passwordField;

private javax.swing.JLabel rememberPasLabel;

// End of variables declaration//GEN-END:variables

}

### Файл MainForm.java

package UI;

/\*\*

\*

\* @author Dmitriy

\*/

public class MainForm extends javax.swing.JFrame {

/\*\*

\* Creates new form MainForm

\*/

public MainForm() {

initComponents();

this.setLocationRelativeTo(null);

}

/\*\*

\* This method is called from within the constructor to initialize the form.

\* WARNING: Do NOT modify this code. The content of this method is always

\* regenerated by the Form Editor.

\*/

@SuppressWarnings("unchecked")

// <editor-fold defaultstate="collapsed" desc="Generated Code">//GEN-BEGIN:initComponents

private void initComponents() {

formScheduleButton = new javax.swing.JButton();

viewScheduleButton = new javax.swing.JButton();

jButton3 = new javax.swing.JButton();

javax.swing.JButton requirementsButton = new javax.swing.JButton();

jButton5 = new javax.swing.JButton();

jButton6 = new javax.swing.JButton();

jButton7 = new javax.swing.JButton();

setDefaultCloseOperation(javax.swing.WindowConstants.EXIT\_ON\_CLOSE);

formScheduleButton.setFont(new java.awt.Font("Tahoma", 0, 14)); // NOI18N

formScheduleButton.setLabel("Генерация расписания");

formScheduleButton.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

formScheduleButtonActionPerformed(evt);

}

});

viewScheduleButton.setFont(new java.awt.Font("Tahoma", 0, 14)); // NOI18N

viewScheduleButton.setLabel("Просмотр расписания");

viewScheduleButton.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

viewScheduleButtonActionPerformed(evt);

}

});

jButton3.setFont(new java.awt.Font("Tahoma", 0, 14)); // NOI18N

jButton3.setEnabled(false);

jButton3.setLabel("Формирование документации");

requirementsButton.setFont(new java.awt.Font("Tahoma", 0, 14)); // NOI18N

requirementsButton.setLabel("Пожелания преподавателей");

requirementsButton.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

requirementsButtonActionPerformed(evt);

}

});

jButton5.setFont(new java.awt.Font("Tahoma", 0, 14)); // NOI18N

jButton5.setEnabled(false);

jButton5.setLabel("Справочники");

jButton6.setFont(new java.awt.Font("Tahoma", 0, 14)); // NOI18N

jButton6.setEnabled(false);

jButton6.setLabel("Создание резервной копии базы данных");

jButton7.setFont(new java.awt.Font("Tahoma", 0, 14)); // NOI18N

jButton7.setEnabled(false);

jButton7.setLabel("Обновление из резервной копии базы данных");

javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());

getContentPane().setLayout(layout);

layout.setHorizontalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addContainerGap()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING, false)

.addComponent(jButton7, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)

.addComponent(jButton5, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)

.addComponent(jButton6, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)

.addComponent(requirementsButton, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)

.addComponent(jButton3, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)

.addComponent(viewScheduleButton, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)

.addComponent(formScheduleButton, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE))

.addContainerGap(javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE))

);

layout.setVerticalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addContainerGap()

.addComponent(formScheduleButton)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addComponent(viewScheduleButton)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addComponent(jButton3)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addComponent(requirementsButton)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addComponent(jButton5)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addComponent(jButton6)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addComponent(jButton7)

.addContainerGap(javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE))

);

pack();

}// </editor-fold>//GEN-END:initComponents

private void formScheduleButtonActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event\_formScheduleButtonActionPerformed

ScheduleGenerateForm.main();

}//GEN-LAST:event\_formScheduleButtonActionPerformed

private void requirementsButtonActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event\_requirementsButtonActionPerformed

RequirementsUniversityTeacherForm.main();

}//GEN-LAST:event\_requirementsButtonActionPerformed

private void viewScheduleButtonActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event\_viewScheduleButtonActionPerformed

ScheduleTableForm.main();

}//GEN-LAST:event\_viewScheduleButtonActionPerformed

public static void main() {

/\* Set the Nimbus look and feel \*/

//<editor-fold defaultstate="collapsed" desc=" Look and feel setting code (optional) ">

/\* If Nimbus (introduced in Java SE 6) is not available, stay with the default look and feel.

\* For details see http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html

\*/

try {

for (javax.swing.UIManager.LookAndFeelInfo info : javax.swing.UIManager.getInstalledLookAndFeels()) {

if ("Nimbus".equals(info.getName())) {

javax.swing.UIManager.setLookAndFeel(info.getClassName());

break;

}

}

} catch (ClassNotFoundException ex) {

java.util.logging.Logger.getLogger(MainForm.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (InstantiationException ex) {

java.util.logging.Logger.getLogger(MainForm.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (IllegalAccessException ex) {

java.util.logging.Logger.getLogger(MainForm.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (javax.swing.UnsupportedLookAndFeelException ex) {

java.util.logging.Logger.getLogger(MainForm.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

}

//</editor-fold>

/\* Create and display the form \*/

java.awt.EventQueue.invokeLater(new Runnable() {

public void run() {

new MainForm().setVisible(true);

}

});

}

// Variables declaration - do not modify//GEN-BEGIN:variables

private javax.swing.JButton formScheduleButton;

private javax.swing.JButton jButton3;

private javax.swing.JButton jButton5;

private javax.swing.JButton jButton6;

private javax.swing.JButton jButton7;

private javax.swing.JButton viewScheduleButton;

// End of variables declaration//GEN-END:variables

}

### Файл RequirementsUniversityTeacherForm.java

package UI;

import java.awt.Color;

import java.sql.Connection;

import java.util.ArrayList;

import java.util.List;

import javax.swing.table.DefaultTableModel;

import DataAccessObject.RequirementsDAO;

import DataAccessObject.RequirementsUniversityTeacherDAO;

import DataAccessObject.UniversityTeacherDAO;

import DatabaseEntities.Requirements;

import DatabaseEntities.RequirementsUniversityTeacher;

import static DatabaseEntities.RequirementsUniversityTeacher.\*;

import DatabaseEntities.UniversityTeacher;

import static DatabaseTools.DBTools.\*;

/\*\*

\*

\* @author Dmitriy

\*/

public class RequirementsUniversityTeacherForm extends javax.swing.JFrame {

Connection con = getConnection();

private UniversityTeacher universityTeacher;

List<UniversityTeacher> universityTeacherList;

private UniversityTeacherDAO universityTeacherDAO =

new UniversityTeacherDAO(con);

private RequirementsUniversityTeacher rut;

private List<RequirementsUniversityTeacher> rutList;

private List<RequirementsUniversityTeacher> rutResultList;

private List<Integer> years;

private RequirementsUniversityTeacherDAO rutDAO =

new RequirementsUniversityTeacherDAO(con);

private Requirements requirements;

private List<Requirements> requirementsList;

private RequirementsDAO requirementsDAO =

new RequirementsDAO(con);

/\*\*

\* Creates new form RequirementsUniversityTeacher

\*/

public RequirementsUniversityTeacherForm() {

initComponents();

this.setLocationRelativeTo(null);

universityTeacherList = universityTeacherDAO.getUniversityTeachersList();

rutList = rutDAO.getRequirementsUniversityTeacherList();

years = rutDAO.getAllYear();

for(UniversityTeacher ut: universityTeacherList){

teacherComboBox.addItem(ut.getAllTeacherName());

}

for(Integer year: years){

yearComboBox.addItem(year.toString());

}

}

/\*\*

\* This method is called from within the constructor to initialize the form.

\* WARNING: Do NOT modify this code. The content of this method is always

\* regenerated by the Form Editor.

\*/

@SuppressWarnings("unchecked")

// <editor-fold defaultstate="collapsed" desc="Generated Code">//GEN-BEGIN:initComponents

private void initComponents() {

goupForRadio = new javax.swing.ButtonGroup();

buttonGroup1 = new javax.swing.ButtonGroup();

jPanel1 = new javax.swing.JPanel();

allTeacherRadioButton = new javax.swing.JRadioButton();

oneTeacherRadioButton = new javax.swing.JRadioButton();

teacherComboBox = new javax.swing.JComboBox<>();

jPanel2 = new javax.swing.JPanel();

addingButton = new javax.swing.JButton();

activateButton = new javax.swing.JButton();

deactivateButton = new javax.swing.JButton();

jPanel3 = new javax.swing.JPanel();

jLabel1 = new javax.swing.JLabel();

showAllFilterRadioButton = new javax.swing.JRadioButton();

showActiveFilterRadioButton = new javax.swing.JRadioButton();

showDeactiveFilterRadioButton = new javax.swing.JRadioButton();

yearFilterCheckBoxBox = new javax.swing.JCheckBox();

yearComboBox = new javax.swing.JComboBox<>();

updateButton = new javax.swing.JButton();

jScrollPane1 = new javax.swing.JScrollPane();

jTable1 = new javax.swing.JTable();

setDefaultCloseOperation(javax.swing.WindowConstants.DISPOSE\_ON\_CLOSE);

setTitle("Пожелания преподавателей");

setLocation(new java.awt.Point(0, 0));

setLocationByPlatform(true);

addWindowListener(new java.awt.event.WindowAdapter() {

public void windowActivated(java.awt.event.WindowEvent evt) {

formWindowActivated(evt);

}

public void windowOpened(java.awt.event.WindowEvent evt) {

formWindowOpened(evt);

}

});

jPanel1.setBackground(new java.awt.Color(51, 51, 51));

allTeacherRadioButton.setBackground(new java.awt.Color(51, 51, 51));

goupForRadio.add(allTeacherRadioButton);

allTeacherRadioButton.setFont(new java.awt.Font("Tahoma", 0, 18)); // NOI18N

allTeacherRadioButton.setForeground(new java.awt.Color(255, 255, 255));

allTeacherRadioButton.setSelected(true);

allTeacherRadioButton.setLabel("Пожелания всех преподавателей");

allTeacherRadioButton.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

allTeacherRadioButtonActionPerformed(evt);

}

});

oneTeacherRadioButton.setBackground(new java.awt.Color(51, 51, 51));

goupForRadio.add(oneTeacherRadioButton);

oneTeacherRadioButton.setFont(new java.awt.Font("Tahoma", 0, 18)); // NOI18N

oneTeacherRadioButton.setForeground(new java.awt.Color(255, 255, 255));

oneTeacherRadioButton.setLabel("Пожелания конкретного преподавателя");

oneTeacherRadioButton.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

oneTeacherRadioButtonActionPerformed(evt);

}

});

teacherComboBox.setFont(new java.awt.Font("Tahoma", 0, 18)); // NOI18N

teacherComboBox.setEnabled(false);

teacherComboBox.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

teacherComboBoxActionPerformed(evt);

}

});

javax.swing.GroupLayout jPanel1Layout = new javax.swing.GroupLayout(jPanel1);

jPanel1.setLayout(jPanel1Layout);

jPanel1Layout.setHorizontalGroup(

jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(jPanel1Layout.createSequentialGroup()

.addContainerGap()

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addComponent(allTeacherRadioButton)

.addGroup(jPanel1Layout.createSequentialGroup()

.addComponent(oneTeacherRadioButton)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addComponent(teacherComboBox, javax.swing.GroupLayout.PREFERRED\_SIZE, 340, javax.swing.GroupLayout.PREFERRED\_SIZE)))

.addContainerGap(274, Short.MAX\_VALUE))

);

jPanel1Layout.setVerticalGroup(

jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(jPanel1Layout.createSequentialGroup()

.addContainerGap()

.addComponent(allTeacherRadioButton)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(oneTeacherRadioButton)

.addComponent(teacherComboBox, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addContainerGap(javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE))

);

jPanel2.setBackground(new java.awt.Color(0, 102, 102));

jPanel2.setForeground(new java.awt.Color(0, 51, 51));

addingButton.setFont(new java.awt.Font("Tahoma", 0, 18)); // NOI18N

addingButton.setForeground(new java.awt.Color(0, 102, 102));

addingButton.setIcon(new javax.swing.ImageIcon("D:\\cloud\\Программирование\\ДИПЛОМ\\ScheduleMaker\\src\\project\\images\\adding1.png")); // NOI18N

addingButton.setHorizontalAlignment(javax.swing.SwingConstants.LEFT);

addingButton.setIconTextGap(10);

addingButton.setLabel("Добавить");

addingButton.setRolloverIcon(new javax.swing.ImageIcon("D:\\cloud\\Программирование\\ДИПЛОМ\\ScheduleMaker\\src\\project\\images\\adding2\_1.png")); // NOI18N

addingButton.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

addingButtonActionPerformed(evt);

}

});

activateButton.setFont(new java.awt.Font("Tahoma", 0, 18)); // NOI18N

activateButton.setForeground(new java.awt.Color(0, 102, 102));

activateButton.setIcon(new javax.swing.ImageIcon("D:\\cloud\\Программирование\\ДИПЛОМ\\ScheduleMaker\\src\\project\\images\\checek3.png")); // NOI18N

activateButton.setHorizontalAlignment(javax.swing.SwingConstants.LEFT);

activateButton.setIconTextGap(10);

activateButton.setLabel("Активировать");

activateButton.setRolloverIcon(new javax.swing.ImageIcon("D:\\cloud\\Программирование\\ДИПЛОМ\\ScheduleMaker\\src\\project\\images\\checked4.png")); // NOI18N

activateButton.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

activateButtonActionPerformed(evt);

}

});

deactivateButton.setFont(new java.awt.Font("Tahoma", 0, 18)); // NOI18N

deactivateButton.setForeground(new java.awt.Color(0, 102, 102));

deactivateButton.setIcon(new javax.swing.ImageIcon("D:\\cloud\\Программирование\\ДИПЛОМ\\ScheduleMaker\\src\\project\\images\\cancel1.png")); // NOI18N

deactivateButton.setFocusTraversalPolicyProvider(true);

deactivateButton.setHorizontalAlignment(javax.swing.SwingConstants.LEFT);

deactivateButton.setIconTextGap(10);

deactivateButton.setLabel("Деактивировать");

deactivateButton.setRolloverIcon(new javax.swing.ImageIcon("D:\\cloud\\Программирование\\ДИПЛОМ\\ScheduleMaker\\src\\project\\images\\cancell2.png")); // NOI18N

deactivateButton.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

deactivateButtonActionPerformed(evt);

}

});

jPanel3.setBackground(new java.awt.Color(0, 102, 102));

jPanel3.setBorder(javax.swing.BorderFactory.createMatteBorder(1, 1, 1, 1, new java.awt.Color(0, 153, 153)));

jLabel1.setFont(new java.awt.Font("Tahoma", 0, 18)); // NOI18N

jLabel1.setForeground(new java.awt.Color(255, 255, 255));

jLabel1.setText("Фильтр:");

showAllFilterRadioButton.setBackground(new java.awt.Color(0, 102, 102));

buttonGroup1.add(showAllFilterRadioButton);

showAllFilterRadioButton.setFont(new java.awt.Font("Tahoma", 0, 14)); // NOI18N

showAllFilterRadioButton.setForeground(new java.awt.Color(255, 255, 255));

showAllFilterRadioButton.setText("Показать все");

showAllFilterRadioButton.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

showAllFilterRadioButtonActionPerformed(evt);

}

});

showActiveFilterRadioButton.setBackground(new java.awt.Color(0, 102, 102));

buttonGroup1.add(showActiveFilterRadioButton);

showActiveFilterRadioButton.setFont(new java.awt.Font("Tahoma", 0, 14)); // NOI18N

showActiveFilterRadioButton.setForeground(new java.awt.Color(255, 255, 255));

showActiveFilterRadioButton.setSelected(true);

showActiveFilterRadioButton.setText("Только активные");

showActiveFilterRadioButton.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

showActiveFilterRadioButtonActionPerformed(evt);

}

});

showDeactiveFilterRadioButton.setBackground(new java.awt.Color(0, 102, 102));

buttonGroup1.add(showDeactiveFilterRadioButton);

showDeactiveFilterRadioButton.setFont(new java.awt.Font("Tahoma", 0, 14)); // NOI18N

showDeactiveFilterRadioButton.setForeground(new java.awt.Color(255, 255, 255));

showDeactiveFilterRadioButton.setText("Только неактивные");

showDeactiveFilterRadioButton.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

showDeactiveFilterRadioButtonActionPerformed(evt);

}

});

yearFilterCheckBoxBox.setBackground(new java.awt.Color(0, 102, 102));

yearFilterCheckBoxBox.setFont(new java.awt.Font("Tahoma", 0, 14)); // NOI18N

yearFilterCheckBoxBox.setForeground(new java.awt.Color(255, 255, 255));

yearFilterCheckBoxBox.setLabel("Конкретный год");

yearFilterCheckBoxBox.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

yearFilterCheckBoxBoxActionPerformed(evt);

}

});

yearComboBox.setFont(new java.awt.Font("Tahoma", 0, 14)); // NOI18N

yearComboBox.setEnabled(false);

yearComboBox.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

yearComboBoxActionPerformed(evt);

}

});

javax.swing.GroupLayout jPanel3Layout = new javax.swing.GroupLayout(jPanel3);

jPanel3.setLayout(jPanel3Layout);

jPanel3Layout.setHorizontalGroup(

jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(jPanel3Layout.createSequentialGroup()

.addContainerGap()

.addGroup(jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addComponent(yearComboBox, 0, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)

.addGroup(jPanel3Layout.createSequentialGroup()

.addGroup(jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addComponent(yearFilterCheckBoxBox)

.addComponent(showDeactiveFilterRadioButton)

.addComponent(showActiveFilterRadioButton)

.addComponent(jLabel1)

.addComponent(showAllFilterRadioButton))

.addGap(0, 40, Short.MAX\_VALUE)))

.addContainerGap())

);

jPanel3Layout.setVerticalGroup(

jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(jPanel3Layout.createSequentialGroup()

.addContainerGap()

.addComponent(jLabel1)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addComponent(showAllFilterRadioButton)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)

.addComponent(showActiveFilterRadioButton)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)

.addComponent(showDeactiveFilterRadioButton)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)

.addComponent(yearFilterCheckBoxBox)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addComponent(yearComboBox, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addContainerGap(javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE))

);

updateButton.setBackground(new java.awt.Color(0, 102, 102));

updateButton.setBorderPainted(false);

updateButton.setContentAreaFilled(false);

updateButton.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

updateButtonActionPerformed(evt);

}

});

javax.swing.GroupLayout jPanel2Layout = new javax.swing.GroupLayout(jPanel2);

jPanel2.setLayout(jPanel2Layout);

jPanel2Layout.setHorizontalGroup(

jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(jPanel2Layout.createSequentialGroup()

.addContainerGap()

.addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addComponent(jPanel3, javax.swing.GroupLayout.Alignment.TRAILING, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)

.addComponent(activateButton, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)

.addComponent(deactivateButton, javax.swing.GroupLayout.DEFAULT\_SIZE, 215, Short.MAX\_VALUE)

.addComponent(addingButton, javax.swing.GroupLayout.Alignment.TRAILING, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)

.addComponent(updateButton, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE))

.addContainerGap())

);

jPanel2Layout.setVerticalGroup(

jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(jPanel2Layout.createSequentialGroup()

.addContainerGap()

.addComponent(jPanel3, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addComponent(addingButton)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addComponent(activateButton)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addComponent(deactivateButton)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)

.addComponent(updateButton)

.addContainerGap())

);

jTable1.setBackground(new java.awt.Color(204, 255, 255));

jTable1.setFont(new java.awt.Font("Tahoma", 0, 14)); // NOI18N

jTable1.setModel(new javax.swing.table.DefaultTableModel(

new Object [][] {

{null, null, null, null, null, null, null},

{null, null, null, null, null, null, null},

{null, null, null, null, null, null, null},

{null, null, null, null, null, null, null},

{null, null, null, null, null, null, null},

{null, null, null, null, null, null, null},

{null, null, null, null, null, null, null},

{null, null, null, null, null, null, null},

{null, null, null, null, null, null, null},

{null, null, null, null, null, null, null},

{null, null, null, null, null, null, null},

{null, null, null, null, null, null, null},

{null, null, null, null, null, null, null},

{null, null, null, null, null, null, null},

{null, null, null, null, null, null, null},

{null, null, null, null, null, null, null},

{null, null, null, null, null, null, null},

{null, null, null, null, null, null, null},

{null, null, null, null, null, null, null},

{null, null, null, null, null, null, null},

{null, null, null, null, null, null, null},

{null, null, null, null, null, null, null}

},

new String [] {

"Преподаватель", "Семестр", "День", "№ Занятия", "Пожелание", "Год добавления", "Активность"

}

) {

boolean[] canEdit = new boolean [] {

false, false, false, false, false, false, false

};

public boolean isCellEditable(int rowIndex, int columnIndex) {

return canEdit [columnIndex];

}

});

jTable1.setRowHeight(20);

jScrollPane1.setViewportView(jTable1);

javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());

getContentPane().setLayout(layout);

layout.setHorizontalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addComponent(jPanel1, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)

.addGroup(layout.createSequentialGroup()

.addComponent(jScrollPane1, javax.swing.GroupLayout.DEFAULT\_SIZE, 736, Short.MAX\_VALUE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addComponent(jPanel2, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE))

);

layout.setVerticalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addComponent(jPanel1, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addComponent(jScrollPane1, javax.swing.GroupLayout.DEFAULT\_SIZE, 491, Short.MAX\_VALUE)

.addComponent(jPanel2, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)))

);

pack();

}// </editor-fold>//GEN-END:initComponents

private void formWindowOpened(java.awt.event.WindowEvent evt) {//GEN-FIRST:event\_formWindowOpened

activateButton.setEnabled(false);

jTable1.getColumnModel().getColumn(0).setMinWidth(230);

jTable1.getColumnModel().getColumn(1).setMinWidth(65);

jTable1.getColumnModel().getColumn(2).setMinWidth(80);

jTable1.getColumnModel().getColumn(3).setMinWidth(75);

jTable1.getColumnModel().getColumn(4).setMinWidth(85);

jTable1.getColumnModel().getColumn(5).setMinWidth(105);

jTable1.getColumnModel().getColumn(6).setMinWidth(75);

}//GEN-LAST:event\_formWindowOpened

private void oneTeacherRadioButtonActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event\_oneTeacherRadioButtonActionPerformed

if(oneTeacherRadioButton.isSelected()){

teacherComboBox.setEnabled(true);

updateButtonActionPerformed(evt);

}

}//GEN-LAST:event\_oneTeacherRadioButtonActionPerformed

private void allTeacherRadioButtonActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event\_allTeacherRadioButtonActionPerformed

teacherComboBox.setEnabled(false);

if(allTeacherRadioButton.isSelected()){

updateButtonActionPerformed(evt);

}

}//GEN-LAST:event\_allTeacherRadioButtonActionPerformed

private void teacherComboBoxActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event\_teacherComboBoxActionPerformed

if(teacherComboBox.isEnabled()){

updateButtonActionPerformed(evt);

}

}//GEN-LAST:event\_teacherComboBoxActionPerformed

private void addingButtonActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event\_addingButtonActionPerformed

AddNewRequirementForm.main();

}//GEN-LAST:event\_addingButtonActionPerformed

private void yearFilterCheckBoxBoxActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event\_yearFilterCheckBoxBoxActionPerformed

if(yearFilterCheckBoxBox.isSelected()){

yearComboBox.setEnabled(true);

updateButtonActionPerformed(evt);

}else{

yearComboBox.setEnabled(false);

updateButtonActionPerformed(evt);

}

}//GEN-LAST:event\_yearFilterCheckBoxBoxActionPerformed

/\*\*

\* Метод смотрит какой приподаватель и фильтр выбран,

\* формирует из списка пожеланий выборку и выводит результат в таблицу

\* @param evt

\*/

private void updateButtonActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event\_updateButtonActionPerformed

rutResultList = new ArrayList<RequirementsUniversityTeacher>();

if(oneTeacherRadioButton.isSelected()){

universityTeacher = universityTeacherList.get(teacherComboBox.getSelectedIndex());

rutResultList = returnOnlyWithThisTeacher(rutList, universityTeacher);

}else{

if(allTeacherRadioButton.isSelected()){

rutResultList = rutList;

}

}

if(showActiveFilterRadioButton.isSelected()){

rutResultList = returnWithBoolFilter(rutResultList, false);

}

if(showDeactiveFilterRadioButton.isSelected()){

rutResultList = returnWithBoolFilter(rutResultList, true);

}

if(yearFilterCheckBoxBox.isSelected()){

Integer year = new Integer((String)yearComboBox.getSelectedItem());

rutResultList = returnListOnlyWithThisYear(rutResultList, year);

}

enterRequirementsListToTable(rutResultList);

}//GEN-LAST:event\_updateButtonActionPerformed

private void enterRequirementsListToTable(List<RequirementsUniversityTeacher> resList){

DefaultTableModel dtm = (DefaultTableModel) jTable1.getModel();

dtm.setRowCount(0);

if(resList.size() > 0){

int howManyRows = resList.size();

dtm.setRowCount(howManyRows);

int column = 0;

int row = 0;

for(RequirementsUniversityTeacher rute: resList){

column = 0;

jTable1.setValueAt(rute.getUniversityTeacher().getAllTeacherName(), row, column++);

jTable1.setValueAt(rute.getSemester(), row, column++);

jTable1.setValueAt(rute.getWeekDay().getName(), row, column++);

jTable1.setValueAt(rute.getNumberClass(), row, column++);

jTable1.setValueAt(rute.getRequirements().getName(), row, column++);

jTable1.setValueAt(rute.getYearReception(), row, column++);

if(rute.getIsArchive() == true)

jTable1.setValueAt("Нет", row++, column);

else

jTable1.setValueAt("Да", row++, column);

}

}

}

private void showAllFilterRadioButtonActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event\_showAllFilterRadioButtonActionPerformed

deactivateButton.setEnabled(true);

activateButton.setEnabled(true);

jTable1.setBackground(new Color(255,255,255));

updateButtonActionPerformed(evt);

}//GEN-LAST:event\_showAllFilterRadioButtonActionPerformed

private void showActiveFilterRadioButtonActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event\_showActiveFilterRadioButtonActionPerformed

deactivateButton.setEnabled(true);

activateButton.setEnabled(false);

jTable1.setBackground(new Color(204,255,255));

updateButtonActionPerformed(evt);

}//GEN-LAST:event\_showActiveFilterRadioButtonActionPerformed

private void showDeactiveFilterRadioButtonActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event\_showDeactiveFilterRadioButtonActionPerformed

deactivateButton.setEnabled(false);

activateButton.setEnabled(true);

jTable1.setBackground(new Color(255,204,204));

updateButtonActionPerformed(evt);

}//GEN-LAST:event\_showDeactiveFilterRadioButtonActionPerformed

private void yearComboBoxActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event\_yearComboBoxActionPerformed

updateButtonActionPerformed(evt);

}//GEN-LAST:event\_yearComboBoxActionPerformed

private void activateButtonActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event\_activateButtonActionPerformed

if(jTable1.getSelectedRow() > -1){

rut = rutResultList.get(jTable1.getSelectedRow());

if(rut.getIsArchive() == true){

rut.setIsArchive(false);

rutDAO.updateRequirementsUniversityTeacher(rut);

updateButtonActionPerformed(evt);

}

}

}//GEN-LAST:event\_activateButtonActionPerformed

private void deactivateButtonActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event\_deactivateButtonActionPerformed

if(jTable1.getSelectedRow() > -1){

rut = rutResultList.get(jTable1.getSelectedRow());

if(rut.getIsArchive() == false){

rut.setIsArchive(true);

rutDAO.updateRequirementsUniversityTeacher(rut);

updateButtonActionPerformed(evt);

}

}

}//GEN-LAST:event\_deactivateButtonActionPerformed

private void formWindowActivated(java.awt.event.WindowEvent evt) {//GEN-FIRST:event\_formWindowActivated

rutList = rutDAO.getRequirementsUniversityTeacherList();

rutResultList = new ArrayList<RequirementsUniversityTeacher>();

if(oneTeacherRadioButton.isSelected()){

universityTeacher = universityTeacherList.get(teacherComboBox.getSelectedIndex());

rutResultList = returnOnlyWithThisTeacher(rutList, universityTeacher);

}else{

if(allTeacherRadioButton.isSelected()){

rutResultList = rutList;

}

}

if(showActiveFilterRadioButton.isSelected()){

rutResultList = returnWithBoolFilter(rutResultList, false);

}

if(showDeactiveFilterRadioButton.isSelected()){

rutResultList = returnWithBoolFilter(rutResultList, true);

}

if(yearFilterCheckBoxBox.isSelected()){

Integer year = new Integer((String)yearComboBox.getSelectedItem());

rutResultList = returnListOnlyWithThisYear(rutResultList, year);

}

enterRequirementsListToTable(rutResultList);

}//GEN-LAST:event\_formWindowActivated

public static void main() {

/\* Set the Nimbus look and feel \*/

//<editor-fold defaultstate="collapsed" desc=" Look and feel setting code (optional) ">

/\* If Nimbus (introduced in Java SE 6) is not available, stay with the default look and feel.

\* For details see http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html

\*/

try {

for (javax.swing.UIManager.LookAndFeelInfo info : javax.swing.UIManager.getInstalledLookAndFeels()) {

if ("Nimbus".equals(info.getName())) {

javax.swing.UIManager.setLookAndFeel(info.getClassName());

break;

}

}

} catch (ClassNotFoundException ex) {

java.util.logging.Logger.getLogger(RequirementsUniversityTeacherForm.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (InstantiationException ex) {

java.util.logging.Logger.getLogger(RequirementsUniversityTeacherForm.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (IllegalAccessException ex) {

java.util.logging.Logger.getLogger(RequirementsUniversityTeacherForm.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (javax.swing.UnsupportedLookAndFeelException ex) {

java.util.logging.Logger.getLogger(RequirementsUniversityTeacherForm.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

}

//</editor-fold>

//</editor-fold>

/\* Create and display the form \*/

java.awt.EventQueue.invokeLater(new Runnable() {

public void run() {

RequirementsUniversityTeacherForm form =

new RequirementsUniversityTeacherForm();

form.setVisible(true);

}

});

}

// Variables declaration - do not modify//GEN-BEGIN:variables

private javax.swing.JButton activateButton;

private javax.swing.JButton addingButton;

private javax.swing.JRadioButton allTeacherRadioButton;

private javax.swing.ButtonGroup buttonGroup1;

private javax.swing.JButton deactivateButton;

private javax.swing.ButtonGroup goupForRadio;

private javax.swing.JLabel jLabel1;

private javax.swing.JPanel jPanel1;

private javax.swing.JPanel jPanel2;

private javax.swing.JPanel jPanel3;

private javax.swing.JScrollPane jScrollPane1;

private javax.swing.JTable jTable1;

private javax.swing.JRadioButton oneTeacherRadioButton;

private javax.swing.JRadioButton showActiveFilterRadioButton;

private javax.swing.JRadioButton showAllFilterRadioButton;

private javax.swing.JRadioButton showDeactiveFilterRadioButton;

private javax.swing.JComboBox<String> teacherComboBox;

private javax.swing.JButton updateButton;

private javax.swing.JComboBox<String> yearComboBox;

private javax.swing.JCheckBox yearFilterCheckBoxBox;

// End of variables declaration//GEN-END:variables

}

### Файл ScheduleGenerateForm.java

package UI;

import Automation.FormationSchedule;

/\*\*

\*

\* @author Dmitriy

\*/

public class ScheduleGenerateForm extends javax.swing.JFrame {

/\*\*

\* Creates new form ScheduleMaker

\*/

public ScheduleGenerateForm() {

initComponents();

this.setLocationRelativeTo(null);

}

/\*\*

\* This method is called from within the constructor to initialize the form.

\* WARNING: Do NOT modify this code. The content of this method is always

\* regenerated by the Form Editor.

\*/

@SuppressWarnings("unchecked")

// <editor-fold defaultstate="collapsed" desc="Generated Code">//GEN-BEGIN:initComponents

private void initComponents() {

jLabel3 = new javax.swing.JLabel();

jPanel1 = new javax.swing.JPanel();

fieldNameSchedule = new javax.swing.JTextField();

jLabel1 = new javax.swing.JLabel();

fieldYearReception = new javax.swing.JTextField();

jLabel2 = new javax.swing.JLabel();

generateScheduleButton = new javax.swing.JButton();

Cancell = new javax.swing.JButton();

jLabel4 = new javax.swing.JLabel();

jProgressBar1 = new javax.swing.JProgressBar();

setDefaultCloseOperation(javax.swing.WindowConstants.DISPOSE\_ON\_CLOSE);

setTitle("Генерация расписания");

setResizable(false);

addWindowListener(new java.awt.event.WindowAdapter() {

public void windowOpened(java.awt.event.WindowEvent evt) {

formWindowOpened(evt);

}

});

jLabel3.setFont(new java.awt.Font("Tahoma", 0, 14)); // NOI18N

jLabel3.setText("Конфигурации составляемого расписания");

jLabel3.setToolTipText("");

jPanel1.setBorder(javax.swing.BorderFactory.createMatteBorder(1, 1, 1, 1, new java.awt.Color(204, 204, 204)));

jLabel1.setFont(new java.awt.Font("Tahoma", 0, 14)); // NOI18N

jLabel1.setText("Название:");

jLabel2.setFont(new java.awt.Font("Tahoma", 0, 14)); // NOI18N

jLabel2.setText("Год создания:");

generateScheduleButton.setFont(new java.awt.Font("Tahoma", 0, 14)); // NOI18N

generateScheduleButton.setText("Сгенерировать");

generateScheduleButton.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

generateScheduleButtonActionPerformed(evt);

}

});

Cancell.setFont(new java.awt.Font("Tahoma", 0, 14)); // NOI18N

Cancell.setLabel("Отмена");

Cancell.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

CancellActionPerformed(evt);

}

});

jLabel4.setText(".... доболнительные в магистерской работе");

javax.swing.GroupLayout jPanel1Layout = new javax.swing.GroupLayout(jPanel1);

jPanel1.setLayout(jPanel1Layout);

jPanel1Layout.setHorizontalGroup(

jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(jPanel1Layout.createSequentialGroup()

.addContainerGap()

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(jPanel1Layout.createSequentialGroup()

.addGap(0, 0, Short.MAX\_VALUE)

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(javax.swing.GroupLayout.Alignment.TRAILING, jPanel1Layout.createSequentialGroup()

.addComponent(jLabel1)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addComponent(fieldNameSchedule, javax.swing.GroupLayout.PREFERRED\_SIZE, 221, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addGroup(javax.swing.GroupLayout.Alignment.TRAILING, jPanel1Layout.createSequentialGroup()

.addComponent(jLabel2)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addComponent(fieldYearReception, javax.swing.GroupLayout.PREFERRED\_SIZE, 221, javax.swing.GroupLayout.PREFERRED\_SIZE))))

.addGroup(jPanel1Layout.createSequentialGroup()

.addComponent(generateScheduleButton)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)

.addComponent(Cancell)))

.addContainerGap())

.addGroup(jPanel1Layout.createSequentialGroup()

.addGap(27, 27, 27)

.addComponent(jLabel4, javax.swing.GroupLayout.PREFERRED\_SIZE, 270, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addContainerGap(javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE))

);

jPanel1Layout.setVerticalGroup(

jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(jPanel1Layout.createSequentialGroup()

.addContainerGap()

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(fieldNameSchedule, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addComponent(jLabel1))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(fieldYearReception, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addComponent(jLabel2))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, 44, Short.MAX\_VALUE)

.addComponent(jLabel4)

.addGap(28, 28, 28)

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(generateScheduleButton)

.addComponent(Cancell))

.addContainerGap())

);

jProgressBar1.setIndeterminate(true);

javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());

getContentPane().setLayout(layout);

layout.setHorizontalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addContainerGap()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addComponent(jLabel3)

.addComponent(jPanel1, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addContainerGap(javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE))

.addComponent(jProgressBar1, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)

);

layout.setVerticalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addContainerGap()

.addComponent(jLabel3)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addComponent(jPanel1, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addComponent(jProgressBar1, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE))

);

getAccessibleContext().setAccessibleName("Конфигурации генерируемого расписания");

pack();

}// </editor-fold>//GEN-END:initComponents

private void formWindowOpened(java.awt.event.WindowEvent evt) {//GEN-FIRST:event\_formWindowOpened

jProgressBar1.setVisible(false);

}//GEN-LAST:event\_formWindowOpened

private void CancellActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event\_CancellActionPerformed

dispose();

}//GEN-LAST:event\_CancellActionPerformed

private void generateScheduleButtonActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event\_generateScheduleButtonActionPerformed

jProgressBar1.setVisible(true);

String nameSchedule = fieldNameSchedule.getText();

String year = fieldYearReception.getText();

int yearReception = 0;

if( !(year.isEmpty()) ){

yearReception = Integer.parseInt(year);

}

if(!(nameSchedule.isEmpty()) && yearReception != 0){

FormationSchedule formationSchedule =

new FormationSchedule(yearReception, nameSchedule);

formationSchedule.makeSchedule();

}

jProgressBar1.setVisible(false);

}//GEN-LAST:event\_generateScheduleButtonActionPerformed

public static void main() {

/\* Set the Nimbus look and feel \*/

//<editor-fold defaultstate="collapsed" desc=" Look and feel setting code (optional) ">

/\* If Nimbus (introduced in Java SE 6) is not available, stay with the default look and feel.

\* For details see http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html

\*/

try {

for (javax.swing.UIManager.LookAndFeelInfo info : javax.swing.UIManager.getInstalledLookAndFeels()) {

if ("Nimbus".equals(info.getName())) {

javax.swing.UIManager.setLookAndFeel(info.getClassName());

break;

}

}

} catch (ClassNotFoundException ex) {

java.util.logging.Logger.getLogger(ScheduleGenerateForm.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (InstantiationException ex) {

java.util.logging.Logger.getLogger(ScheduleGenerateForm.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (IllegalAccessException ex) {

java.util.logging.Logger.getLogger(ScheduleGenerateForm.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (javax.swing.UnsupportedLookAndFeelException ex) {

java.util.logging.Logger.getLogger(ScheduleGenerateForm.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

}

//</editor-fold>

//</editor-fold>

//</editor-fold>

//</editor-fold>

/\* Create and display the form \*/

java.awt.EventQueue.invokeLater(new Runnable() {

public void run() {

new ScheduleGenerateForm().setVisible(true);

}

});

}

// Variables declaration - do not modify//GEN-BEGIN:variables

private javax.swing.JButton Cancell;

private javax.swing.JTextField fieldNameSchedule;

private javax.swing.JTextField fieldYearReception;

private javax.swing.JButton generateScheduleButton;

private javax.swing.JLabel jLabel1;

private javax.swing.JLabel jLabel2;

private javax.swing.JLabel jLabel3;

private javax.swing.JLabel jLabel4;

private javax.swing.JPanel jPanel1;

private javax.swing.JProgressBar jProgressBar1;

// End of variables declaration//GEN-END:variables

}

### Файл ScheduleTableForm.java

package UI;

import DatabaseEntities.\*;

import DataAccessObject.\*;

import static DatabaseTools.DBTools.getConnection;

import java.util.ArrayList;

import java.util.List;

import javax.swing.table.DefaultTableModel;

/\*\*

\*

\* @author Dmitriy

\*/

public class ScheduleTableForm extends javax.swing.JFrame {

Faculty faculty;

Department department;

List <GroupStudent> groupStudentList;

List <Faculty> facultyList;

List <Department> departmentList;

List <String> scheduleNamesList;

List <Integer> scheduleAllYears;

List <Schedule> scheduleClasses;

GroupStudentDAO groupStudentDAO = new GroupStudentDAO(getConnection());

FacultyDAO facultyDAO = new FacultyDAO(getConnection());

DepartmentDAO departmentDAO = new DepartmentDAO(getConnection());

ScheduleDAO scheduleDAO = new ScheduleDAO(getConnection());

/\*\*

\* Creates new form ScheduleTableForm

\*/

public ScheduleTableForm() {

initComponents();

facultyList = facultyDAO.getFacultyList();

scheduleAllYears = scheduleDAO.getAllYearsOfSchedules();

//заполняем бокс с факультетами

for(Faculty fac: facultyList){

jComboBoxFaculty.addItem(fac.getName());

}

//заполняем все существующие года расписаний

for(Integer year: scheduleAllYears){

jComboBoxYearReception.addItem(year.toString());

}

}

/\*\*

\* This method is called from within the constructor to initialize the form.

\* WARNING: Do NOT modify this code. The content of this method is always

\* regenerated by the Form Editor.

\*/

@SuppressWarnings("unchecked")

// <editor-fold defaultstate="collapsed" desc="Generated Code">//GEN-BEGIN:initComponents

private void initComponents() {

jPanel1 = new javax.swing.JPanel();

jPanel3 = new javax.swing.JPanel();

jLabelFaculty = new javax.swing.JLabel();

jLabelSemester = new javax.swing.JLabel();

jLabelQuarter = new javax.swing.JLabel();

jComboBoxFaculty = new javax.swing.JComboBox<>();

jComboBoxSemester = new javax.swing.JComboBox<>();

jComboBoxQuarter = new javax.swing.JComboBox<>();

selectionFromSchedule = new javax.swing.JButton();

jComboBoxYearReception = new javax.swing.JComboBox<>();

jComboBoxSchedulesName = new javax.swing.JComboBox<>();

jLabel2 = new javax.swing.JLabel();

jLabel1 = new javax.swing.JLabel();

downloadScheduleByYearAndName = new javax.swing.JButton();

jScrollPane1 = new javax.swing.JScrollPane();

jTable1 = new javax.swing.JTable();

setDefaultCloseOperation(javax.swing.WindowConstants.DISPOSE\_ON\_CLOSE);

setTitle("Доска с расписанием");

setLocationByPlatform(true);

addWindowListener(new java.awt.event.WindowAdapter() {

public void windowOpened(java.awt.event.WindowEvent evt) {

formWindowOpened(evt);

}

});

jPanel1.setBackground(new java.awt.Color(51, 51, 51));

jPanel3.setBackground(new java.awt.Color(0, 102, 102));

jPanel3.setBorder(javax.swing.BorderFactory.createEtchedBorder(javax.swing.border.EtchedBorder.RAISED, new java.awt.Color(0, 204, 204), new java.awt.Color(0, 153, 153)));

jLabelFaculty.setFont(new java.awt.Font("Tahoma", 0, 18)); // NOI18N

jLabelFaculty.setForeground(new java.awt.Color(255, 255, 255));

jLabelFaculty.setText("Факультет");

jLabelSemester.setFont(new java.awt.Font("Tahoma", 0, 18)); // NOI18N

jLabelSemester.setForeground(new java.awt.Color(255, 255, 255));

jLabelSemester.setText("Семестр");

jLabelQuarter.setFont(new java.awt.Font("Tahoma", 0, 18)); // NOI18N

jLabelQuarter.setForeground(new java.awt.Color(255, 255, 255));

jLabelQuarter.setText("Четверть");

jComboBoxFaculty.setFont(new java.awt.Font("Tahoma", 0, 14)); // NOI18N

jComboBoxFaculty.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

jComboBoxFacultyActionPerformed(evt);

}

});

jComboBoxSemester.setFont(new java.awt.Font("Tahoma", 0, 14)); // NOI18N

jComboBoxSemester.setModel(new javax.swing.DefaultComboBoxModel<>(new String[] { "1", "2" }));

jComboBoxSemester.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

jComboBoxSemesterActionPerformed(evt);

}

});

jComboBoxQuarter.setFont(new java.awt.Font("Tahoma", 0, 14)); // NOI18N

jComboBoxQuarter.setModel(new javax.swing.DefaultComboBoxModel<>(new String[] { "1", "2" }));

jComboBoxQuarter.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

jComboBoxQuarterActionPerformed(evt);

}

});

selectionFromSchedule.setBorderPainted(false);

selectionFromSchedule.setContentAreaFilled(false);

selectionFromSchedule.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

selectionFromScheduleActionPerformed(evt);

}

});

jComboBoxYearReception.setFont(new java.awt.Font("Tahoma", 0, 14)); // NOI18N

jComboBoxYearReception.setFocusTraversalPolicyProvider(true);

jComboBoxYearReception.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

jComboBoxYearReceptionActionPerformed(evt);

}

});

jComboBoxSchedulesName.setFont(new java.awt.Font("Tahoma", 0, 14)); // NOI18N

jComboBoxSchedulesName.setFocusTraversalPolicyProvider(true);

jLabel2.setFont(new java.awt.Font("Tahoma", 0, 18)); // NOI18N

jLabel2.setForeground(new java.awt.Color(255, 255, 255));

jLabel2.setText("Год выпуска");

jLabel2.setFocusTraversalPolicyProvider(true);

jLabel1.setFont(new java.awt.Font("Tahoma", 0, 18)); // NOI18N

jLabel1.setForeground(new java.awt.Color(255, 255, 255));

jLabel1.setText("Название");

jLabel1.setFocusTraversalPolicyProvider(true);

downloadScheduleByYearAndName.setFont(new java.awt.Font("Tahoma", 0, 14)); // NOI18N

downloadScheduleByYearAndName.setText("Загрузить");

downloadScheduleByYearAndName.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

downloadScheduleByYearAndNameActionPerformed(evt);

}

});

javax.swing.GroupLayout jPanel3Layout = new javax.swing.GroupLayout(jPanel3);

jPanel3.setLayout(jPanel3Layout);

jPanel3Layout.setHorizontalGroup(

jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(jPanel3Layout.createSequentialGroup()

.addContainerGap()

.addGroup(jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addComponent(jLabel2)

.addComponent(jLabel1)

.addComponent(jLabelSemester)

.addComponent(jLabelQuarter))

.addGap(10, 10, 10)

.addGroup(jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(jPanel3Layout.createSequentialGroup()

.addComponent(jComboBoxSemester, javax.swing.GroupLayout.PREFERRED\_SIZE, 100, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addGap(0, 0, Short.MAX\_VALUE))

.addGroup(jPanel3Layout.createSequentialGroup()

.addGroup(jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING, false)

.addGroup(jPanel3Layout.createSequentialGroup()

.addComponent(jComboBoxQuarter, javax.swing.GroupLayout.PREFERRED\_SIZE, 100, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)

.addComponent(downloadScheduleByYearAndName))

.addGroup(jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addComponent(jComboBoxSchedulesName, javax.swing.GroupLayout.PREFERRED\_SIZE, 240, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addComponent(jComboBoxYearReception, javax.swing.GroupLayout.PREFERRED\_SIZE, 240, javax.swing.GroupLayout.PREFERRED\_SIZE)))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, 64, Short.MAX\_VALUE)

.addGroup(jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addComponent(selectionFromSchedule)

.addGroup(jPanel3Layout.createSequentialGroup()

.addComponent(jLabelFaculty)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addComponent(jComboBoxFaculty, javax.swing.GroupLayout.PREFERRED\_SIZE, 354, javax.swing.GroupLayout.PREFERRED\_SIZE)))

.addContainerGap())))

);

jPanel3Layout.setVerticalGroup(

jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(jPanel3Layout.createSequentialGroup()

.addContainerGap()

.addGroup(jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(jPanel3Layout.createSequentialGroup()

.addComponent(jComboBoxYearReception, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addComponent(jComboBoxSchedulesName, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addComponent(jComboBoxSemester, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addGap(7, 7, 7)

.addGroup(jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(jComboBoxQuarter, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addComponent(downloadScheduleByYearAndName))

.addGap(0, 0, Short.MAX\_VALUE))

.addGroup(jPanel3Layout.createSequentialGroup()

.addGroup(jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(jComboBoxFaculty, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addComponent(jLabel2)

.addComponent(jLabelFaculty))

.addGroup(jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(jPanel3Layout.createSequentialGroup()

.addGap(24, 24, 24)

.addComponent(selectionFromSchedule)

.addContainerGap(javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE))

.addGroup(jPanel3Layout.createSequentialGroup()

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addComponent(jLabel1)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addComponent(jLabelSemester)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, 12, Short.MAX\_VALUE)

.addComponent(jLabelQuarter)

.addContainerGap(23, Short.MAX\_VALUE))))))

);

jTable1.setFont(new java.awt.Font("Tahoma", 0, 12)); // NOI18N

jTable1.setModel(new javax.swing.table.DefaultTableModel(

new Object [][] {

{"", "", "ГРУПИ ->", null, null},

{"ПН", "1", "Чисельник", null, null},

{"", null, null, null, null},

{"ПН", "1", "Знаменник", null, null},

{null, null, null, null, null},

{"ПН", "2", "Чисельник", null, null},

{null, null, null, null, null},

{"ПН", "2", "Знаменник", null, null},

{null, null, null, null, null},

{"ПН", "3", "Чисельник", null, null},

{null, null, "", null, null},

{"ПН", "3", "Знаменник", null, null},

{null, null, null, null, null},

{"ПН", "4", "Чисельник", null, null},

{null, null, null, null, null},

{"ПН", "4", "Знаменник", null, null},

{null, null, null, null, null},

{"ПН", "5", "Чисельник", null, null},

{null, null, null, null, null},

{"ПН", "5", "Знаменник", null, null},

{null, null, "", null, null},

{"ПН", "6", "Чисельник", null, null},

{null, null, null, null, null},

{"ПН", "6", "Знаменник", null, null},

{null, null, null, null, null},

{"ВТ", "1", "Чисельник", null, null},

{null, null, "", null, null},

{"ВТ", "1", "Знаменник", null, null},

{null, null, "", null, null},

{"ВТ", "2", "Чисельник", null, null},

{null, null, null, null, null},

{"ВТ", "2", "Знаменник", null, null},

{null, null, null, null, null},

{"ВТ", "3", "Чисельник", null, null},

{null, null, null, null, null},

{"ВТ", "3", "Знаменник", null, null},

{null, null, null, null, null},

{"ВТ", "4", "Чисельник", null, null},

{null, null, null, null, null},

{"ВТ", "4", "Знаменник", null, null},

{null, null, null, null, null},

{"ВТ", "5", "Чисельник", null, null},

{null, null, null, null, null},

{"ВТ", "5", "Знаменник", null, null},

{"", null, null, null, null},

{"ВТ", "6", "Чисельник", null, null},

{null, null, null, null, null},

{"ВТ", "6", "Знаменник", null, null},

{null, null, null, null, null},

{"СР", "1", "Чисельник", null, null},

{null, null, null, null, null},

{"СР", "1", "Знаменник", null, null},

{null, null, null, null, null},

{"СР", "2", "Чисельник", null, null},

{null, null, null, null, null},

{"СР", "2", "Знаменник", null, null},

{null, null, "", null, null},

{"СР", "3", "Чисельник", null, null},

{null, null, null, null, null},

{"СР", "3", "Знаменник", null, null},

{null, null, null, null, null},

{"СР", "4", "Чисельник", null, null},

{null, null, null, null, null},

{"СР", "4", "Знаменник", null, null},

{null, "", null, null, null},

{"СР", "5", "Чисельник", null, null},

{null, null, null, null, null},

{"СР", "5", "Знаменник", null, null},

{null, null, null, null, null},

{"СР", "6", "Чисельник", null, null},

{null, null, null, null, null},

{"СР", "6", "Знаменник", null, null},

{null, null, null, null, null},

{"ЧТ", "1", "Чисельник", null, null},

{null, null, null, null, null},

{"ЧТ", "1", "Знаменник", null, null},

{null, null, null, null, null},

{"ЧТ", "2", "Чисельник", null, null},

{null, null, null, null, null},

{"ЧТ", "2", "Знаменник", null, null},

{null, null, null, null, null},

{"ЧТ", "3", "Чисельник", null, null},

{null, null, null, null, null},

{"ЧТ", "3", "Знаменник", null, null},

{null, null, null, null, null},

{"ЧТ", "4", "Чисельник", null, null},

{null, null, null, null, null},

{"ЧТ", "4", "Знаменник", null, null},

{null, null, null, null, null},

{"ЧТ", "5", "Чисельник", null, null},

{null, null, null, null, null},

{"ЧТ", "5", "Знаменник", null, null},

{null, null, null, null, null},

{"ЧТ", "6", "Чисельник", null, null},

{null, null, null, null, null},

{"ЧТ", "6", "Знаменник", null, null},

{null, null, "", null, null},

{"ПТ", "1", "Чисельник", null, null},

{null, null, null, null, null},

{"ПТ", "1", "Знаменник", null, null},

{null, null, null, null, null},

{"ПТ", "2", "Чисельник", null, null},

{null, null, null, null, null},

{"ПТ", "2", "Знаменник", null, null},

{"", null, null, null, null},

{"ПТ", "3", "Чисельник", null, null},

{null, null, null, null, null},

{"ПТ", "3", "Знаменник", null, null},

{null, null, null, null, null},

{"ПТ", "4", "Чисельник", null, null},

{null, null, null, null, null},

{"ПТ", "4", "Знаменник", null, null},

{null, null, null, null, null},

{"ПТ", "5", "Чисельник", null, null},

{null, null, null, null, null},

{"ПТ", "5", "Знаменник", null, null},

{null, null, null, null, null},

{"ПТ", "6", "Чисельник", null, null},

{null, null, null, null, null},

{"ПТ", "6", "Знаменник", null, null},

{null, null, null, null, null}

},

new String [] {

"", "", "", "", ""

}

));

jScrollPane1.setViewportView(jTable1);

javax.swing.GroupLayout jPanel1Layout = new javax.swing.GroupLayout(jPanel1);

jPanel1.setLayout(jPanel1Layout);

jPanel1Layout.setHorizontalGroup(

jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(jPanel1Layout.createSequentialGroup()

.addContainerGap()

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addComponent(jScrollPane1)

.addComponent(jPanel3, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE))

.addContainerGap())

);

jPanel1Layout.setVerticalGroup(

jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(jPanel1Layout.createSequentialGroup()

.addContainerGap()

.addComponent(jPanel3, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addComponent(jScrollPane1, javax.swing.GroupLayout.DEFAULT\_SIZE, 344, Short.MAX\_VALUE)

.addContainerGap())

);

javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());

getContentPane().setLayout(layout);

layout.setHorizontalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addComponent(jPanel1, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)

);

layout.setVerticalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addComponent(jPanel1, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)

);

pack();

}// </editor-fold>//GEN-END:initComponents

private void formWindowOpened(java.awt.event.WindowEvent evt) {//GEN-FIRST:event\_formWindowOpened

jTable1.getColumnModel().getColumn(0).setMinWidth(30);

jTable1.getColumnModel().getColumn(0).setMaxWidth(30);

jTable1.getColumnModel().getColumn(1).setMinWidth(30);

jTable1.getColumnModel().getColumn(1).setMaxWidth(30);

jTable1.getColumnModel().getColumn(2).setMinWidth(100);

jTable1.getColumnModel().getColumn(2).setMaxWidth(100);

}//GEN-LAST:event\_formWindowOpened

private void jComboBoxYearReceptionActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event\_jComboBoxYearReceptionActionPerformed

String year = jComboBoxYearReception.getSelectedItem().toString();

int yearReception = Integer.parseInt(year);

//В бокс с названиями расписания добавляем только те, которые были

//созданы для этого года

jComboBoxSchedulesName.removeAllItems();

scheduleNamesList = scheduleDAO.getAllNameOfSchedulesByYear(yearReception);

//заполняем бокс с названиями расписания

for(String name: scheduleNamesList){

jComboBoxSchedulesName.addItem(name);

}

}//GEN-LAST:event\_jComboBoxYearReceptionActionPerformed

private void downloadScheduleByYearAndNameActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event\_downloadScheduleByYearAndNameActionPerformed

String name = jComboBoxSchedulesName.getSelectedItem().toString();

String year = jComboBoxYearReception.getSelectedItem().toString();

int yearReception = Integer.parseInt(year);

int semester = jComboBoxSemester.getSelectedIndex() + 1;

int quarter = Integer.parseInt(jComboBoxQuarter.getSelectedItem().toString());

System.out.println(name);

System.out.println(yearReception);

System.out.println(semester);

System.out.println(quarter);

if(!(name.isEmpty()) && yearReception > 0 && semester > 0 && quarter > 0){

scheduleClasses = scheduleDAO.getThisSheduleList(yearReception, semester, name, quarter);

}

selectionFromScheduleActionPerformed(evt);

}//GEN-LAST:event\_downloadScheduleByYearAndNameActionPerformed

private void drowTable(List <GroupStudent> groups){

DefaultTableModel dtm = (DefaultTableModel) jTable1.getModel();

int size = groups.size();

int column = 3;

if(size == 0){

dtm.setColumnCount(3);

jTable1.getColumnModel().getColumn(0).setMinWidth(30);

jTable1.getColumnModel().getColumn(0).setMaxWidth(30);

jTable1.getColumnModel().getColumn(1).setMinWidth(30);

jTable1.getColumnModel().getColumn(1).setMaxWidth(30);

jTable1.getColumnModel().getColumn(2).setMinWidth(100);

jTable1.getColumnModel().getColumn(2).setMaxWidth(100);

}else{

dtm.setColumnCount(3 + size);

jTable1.getColumnModel().getColumn(0).setMinWidth(30);

jTable1.getColumnModel().getColumn(0).setMaxWidth(30);

jTable1.getColumnModel().getColumn(1).setMinWidth(30);

jTable1.getColumnModel().getColumn(1).setMaxWidth(30);

jTable1.getColumnModel().getColumn(2).setMinWidth(100);

jTable1.getColumnModel().getColumn(2).setMaxWidth(100);

for(GroupStudent group: groups){

jTable1.setValueAt(group.getName(), 0, column++);

}

}

}

private void enterScheduleToTable(){

List <DatabaseEntities.Class> classes = new ArrayList<DatabaseEntities.Class>();

TimeClass timeClass;

UniversityTeacher teacher;

List <GroupStudent> groupStudentList;

TypeClass typeClass;

Subject subject;

//todo привести scheduleClasses от списка к одной записи

for(Schedule schedule: scheduleClasses){

classes = schedule.getClassList();

for(DatabaseEntities.Class aClass: classes){

timeClass = aClass.getTimeClass();

typeClass = aClass.getTypeClass();

teacher = aClass.getUniversityTeacher();

groupStudentList = aClass.getStream().getGroupStudentList();

subject = aClass.getSubject();

for(GroupStudent group: groupStudentList){

enterToTable(group, timeClass, typeClass, teacher, subject);

}

}

}

}

private void enterToTable(GroupStudent groupStudent, TimeClass timeClass,

TypeClass typeClass, UniversityTeacher teacher, Subject subject){

if(groupStudent != null){

WeekDay weekDay = timeClass.getWeekDay();

TypeWeek typeWeek = timeClass.getTypeWeek();

int numberClass = timeClass.getNumberClass();

int column = 0;

int row = 0;

String groupFromTable;

String dayForTableAdapter;

String typeClassAdapter = null;

String dayFromTable;

int numberClassFromTable;

String typeWeekFromTable;

switch(weekDay.getName()){

case "Понеділок":

dayForTableAdapter = "ПН";

break;

case "Вівторок":

dayForTableAdapter = "ВТ";

break;

case "Середа":

dayForTableAdapter = "СР";

break;

case "Четверг":

dayForTableAdapter = "ЧТ";

break;

case "П'ятниця":

dayForTableAdapter = "ПТ";

break;

default: dayForTableAdapter = "";

}

switch(typeClass.getName()){

case "Лекція":

typeClassAdapter = "лекц: ";

break;

case "Практичне заняття":

typeClassAdapter = "практ: ";

break;

case "Лабораторне заняття":

typeClassAdapter = "лабор: ";

break;

}

//определяем колонку соотв группы

for(int i = 0, j = 2; j < jTable1.getColumnCount(); j++){

groupFromTable = jTable1.getValueAt(i, j).toString();

if(groupFromTable.equals(groupStudent.getName())){

column = j;

}

}

for(int i = 1, j = 0; i < jTable1.getRowCount(); i+=2){

dayFromTable = jTable1.getValueAt(i, j).toString();

if(dayFromTable.equals(dayForTableAdapter)){

numberClassFromTable = Integer.parseInt(jTable1.getValueAt(i, 1).toString()); // здесь была ошибка 1 вместо i

if(numberClass == numberClassFromTable){

typeWeekFromTable = jTable1.getValueAt(i, 2).toString();

if(typeWeekFromTable.equals(typeWeek.getName())){

row = i;

}

}

}

}

if(column > 2 && row > 0){

String subjectInfo = typeClassAdapter + subject.getName();

jTable1.setValueAt(subjectInfo, row, column);

jTable1.setValueAt(teacher.getAllTeacherName(), (row+1), column);

}

}

}

private void selectionFromScheduleActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event\_selectionFromScheduleActionPerformed

if(scheduleClasses != null){

int indxFaculty = jComboBoxFaculty.getSelectedIndex();

faculty = facultyList.get(indxFaculty);

departmentList = departmentDAO.getDepartmentListByFaculty(faculty);

groupStudentList = groupStudentDAO.getGroupStudentListForDepartments(departmentList);

drowTable(groupStudentList);

enterScheduleToTable();

}

}//GEN-LAST:event\_selectionFromScheduleActionPerformed

private void jComboBoxFacultyActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event\_jComboBoxFacultyActionPerformed

selectionFromScheduleActionPerformed(evt);

}//GEN-LAST:event\_jComboBoxFacultyActionPerformed

private void jComboBoxSemesterActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event\_jComboBoxSemesterActionPerformed

selectionFromScheduleActionPerformed(evt);

jComboBoxQuarter.removeAllItems();

int selectionSemester = jComboBoxSemester.getSelectedIndex() + 1;

if(selectionSemester == 1){

jComboBoxQuarter.addItem("1");

jComboBoxQuarter.addItem("2");

}else{

jComboBoxQuarter.addItem("3");

jComboBoxQuarter.addItem("4");

}

}//GEN-LAST:event\_jComboBoxSemesterActionPerformed

private void jComboBoxQuarterActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event\_jComboBoxQuarterActionPerformed

selectionFromScheduleActionPerformed(evt);

}//GEN-LAST:event\_jComboBoxQuarterActionPerformed

public static void main() {

/\* Set the Nimbus look and feel \*/

//<editor-fold defaultstate="collapsed" desc=" Look and feel setting code (optional) ">

/\* If Nimbus (introduced in Java SE 6) is not available, stay with the default look and feel.

\* For details see http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html

\*/

try {

for (javax.swing.UIManager.LookAndFeelInfo info : javax.swing.UIManager.getInstalledLookAndFeels()) {

if ("Nimbus".equals(info.getName())) {

javax.swing.UIManager.setLookAndFeel(info.getClassName());

break;

}

}

} catch (ClassNotFoundException ex) {

java.util.logging.Logger.getLogger(MainForm.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (InstantiationException ex) {

java.util.logging.Logger.getLogger(MainForm.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (IllegalAccessException ex) {

java.util.logging.Logger.getLogger(MainForm.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (javax.swing.UnsupportedLookAndFeelException ex) {

java.util.logging.Logger.getLogger(MainForm.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

}

//</editor-fold>

/\* Create and display the form \*/

java.awt.EventQueue.invokeLater(new Runnable() {

@Override

public void run() {

new ScheduleTableForm().setVisible(true);

}

});

}

// Variables declaration - do not modify//GEN-BEGIN:variables

private javax.swing.JButton downloadScheduleByYearAndName;

private javax.swing.JComboBox<String> jComboBoxFaculty;

private javax.swing.JComboBox<String> jComboBoxQuarter;

private javax.swing.JComboBox<String> jComboBoxSchedulesName;

private javax.swing.JComboBox<String> jComboBoxSemester;

private javax.swing.JComboBox<String> jComboBoxYearReception;

private javax.swing.JLabel jLabel1;

private javax.swing.JLabel jLabel2;

private javax.swing.JLabel jLabelFaculty;

private javax.swing.JLabel jLabelQuarter;

private javax.swing.JLabel jLabelSemester;

private javax.swing.JPanel jPanel1;

private javax.swing.JPanel jPanel3;

private javax.swing.JScrollPane jScrollPane1;

private javax.swing.JTable jTable1;

private javax.swing.JButton selectionFromSchedule;

// End of variables declaration//GEN-END:variables

}