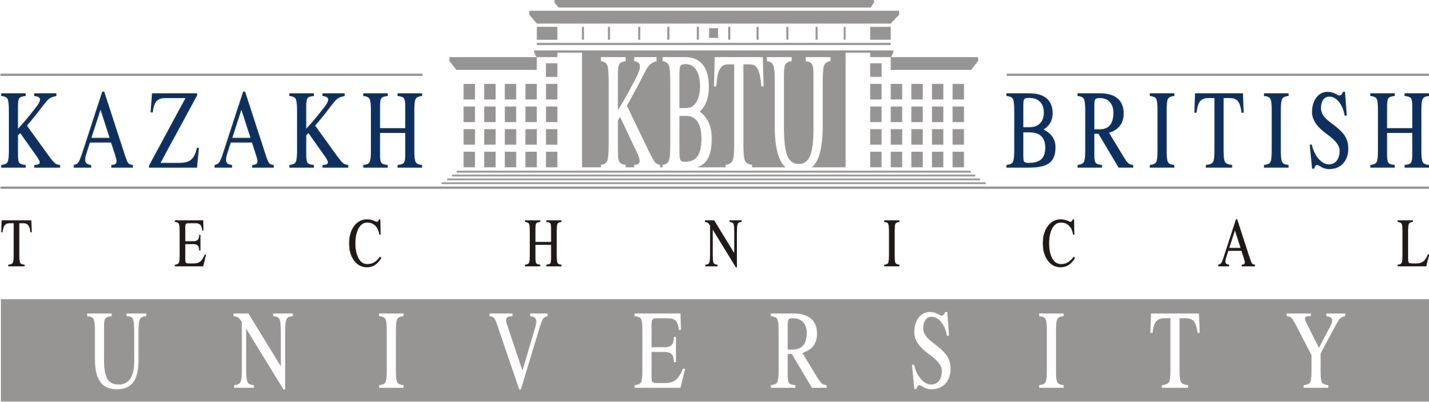
**MINISTRY OF EDUCATION AND SCIENCE**

**OF THE REPUBLIC OF KAZAKHSTAN**

FACULTY OF INFROMATION TECHNOLOGIES

COMPUTER ENGINEERING DEPARTMENT

Object-Oriented Programming and Design

**STUDENT INDEPENDENT STUDY**

**“University Study System (Intranet)”**

**Done by:**

“Belka, Strelka & Nurba” team

Aigerim Sadir

Kulyash Konyrova

Nurbolat Mustafin

**Checked by:**

**Pakita Shamoi**

Almaty, 2017

Table of contents

1. Introduction……………………………………………………………......…3
2. Objective of the Project………………………………………………………4
3. Main part

Creating UML diagrams………………………………………………...........5

1. Use Case Diagram………………..…………………………………....5
2. Class Diagram…………………………………………………………6

Creating code……………………………………………………………....…8

1. Conclusion…………………………………………………………………..10

**Objective of the Project**

**The task:** Create University Study System (Intranet) realized using Object-Oriented Programming.

**General requirements:**

* -  OOP style
* -  Properly Serialization
* -  Intuitive usage
* -  Consistency with UML
* -  Comparable, Cloneable, Serializable, equals(), hashcode(), toString()...
* -  Proper and logically consistent usage of Collections
* -  Usage of java api (standard classes). Do not reinvent the bicycle.

**Introduction**

The aim of this project was to create our own version of Intranet, using knowledge that we got on OOP lessons during this semester. This report presents the steps, that we followed to finish this work and discusses some problems, which we were confronted with.

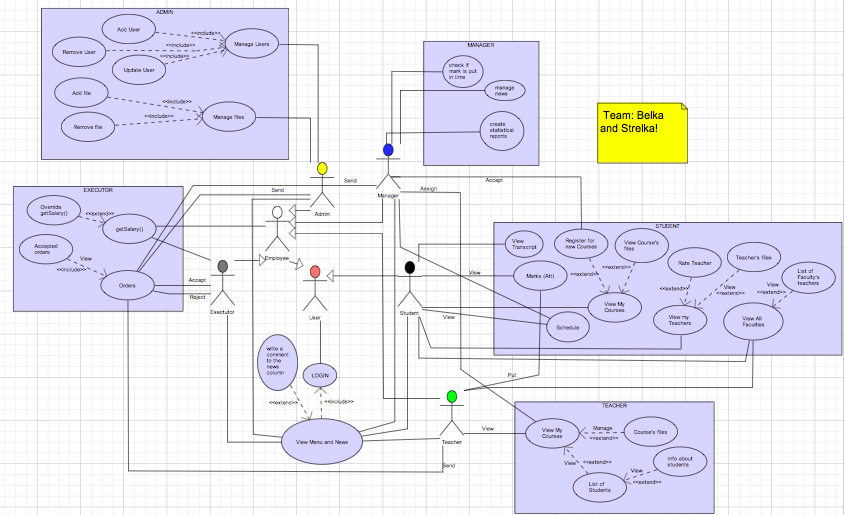
In general, we could divide this work to 3 huge parts: 1) understanding the whole concept and sharing ideas with teammates; 2) creating use case and class diagrams; 3) writing code.

Main purpose of the project was to repeat and use on practice the material, that we learned on our lessons. The last laboratory work was a mini version of Intranet, so this helps us to understand the basic operations and idea.

**Main part**

**2.1. Creating UML diagrams**

1. Use Case Diagram

We started doing our project by creating use case diagram to understand what operations should happen and when. During doing this diagram we discussed the main methods of our project. This type of diagram is the most understandable, because of that it is often presented to people, who are bad in programming.

We have 5 actors in our diagram: 'Teacher', 'Student', 'Manager', 'Executor', 'Admin'. All of them, except Student, are inherited from actor Employee, beacuse all of them can get salary. Student and Employee are inherited from actor User and have common metods: Login and View Menu (which includes writing comments).

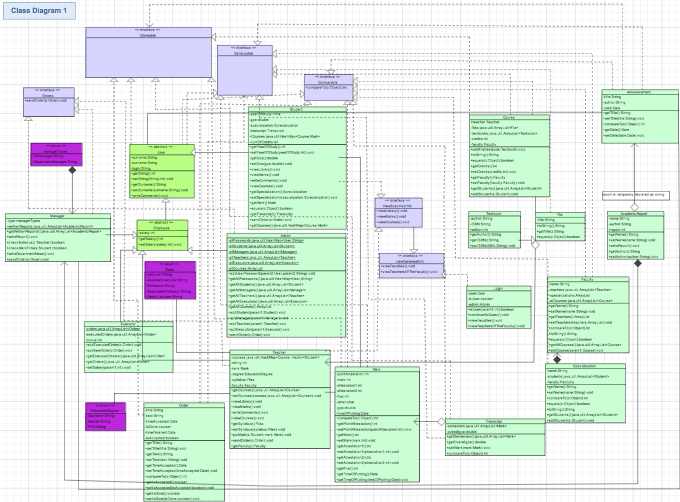
Student can view list his of courses (with registrarion for new ones and viewing course's files), view his marks and transcript, view list of faculties (with faculty's teachers) and view his teachers (also he could rate them).

Teacher has method to view his courses, then he could manage course's files or view list of students, that have this course, with infromation about them. Teacher could put marks to student's attestation or final exam.

Manager has 5 main functions: accept student's registration for a new course, assign courses to a teacher, check if marks were put on time, manage news and create statistical reports.

The basic function of Executor is accepting or rejecting orders, which were sended by all users (except Student). Also he has method getSalary(), that could be overridden if someone is working well.

Admin can manage users and manage files. Managing users includes adding a new user, removing or updating existing one. Manage file method includes add file and remove file methods.

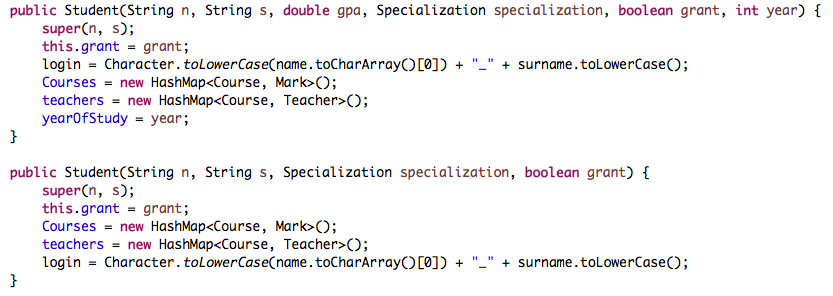
1. Class Diagram

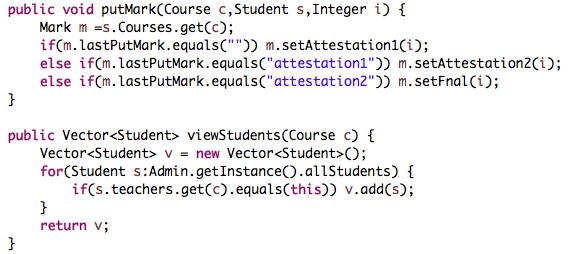
Creating use case diagram gave us incitement to do the next part: doing class diagram. Creating class diagram is a little bit more complicated than use case. Because it needs a good understanding of OOP concepts, abstract classes, interfaces, methods, connection between classes and etc. (in general, topics that we learned at first attestation). At first, we created too many classes, about 35. But then we deleted several of them, because it was possible to combine them into one.

The main classes in our project are “Student”, “Teacher”, “Manager”, “Executor”, “Admin” and two big classes “Employee” and “User”. Class “Employee” extends “User” class and it is superclass for all classes except “Student”, because the student is the only one who doesn’t get salary. The others extend “Employee” and automatically extend “User”. We created User class because all of user could login, view menu and write comments to news menu, and only after login the program identifies is it student, teacher, manager, executor or admin. At first, we wrote getters, setters, constructors, equals(), compareTo() and toString() methods for each class. Only after that we could start writing specific methods for them.

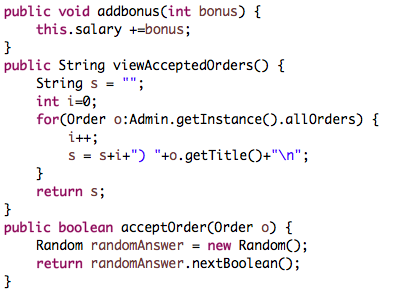
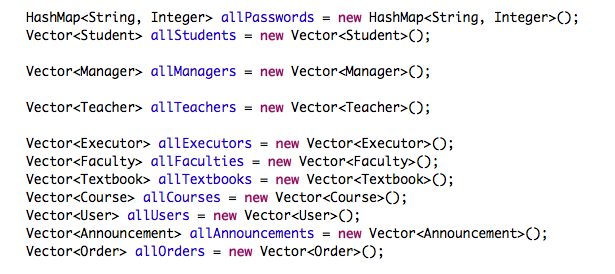
**2.2. Creating code**

The next step of our work was to start creating code. There is a description of classes and their methods below.

Class “Student” has fields: name and surname (that it extends from User), int yearOfStudy and numOfCredits, double GPA, transcript (object of a class Transcript), specialization (object of Specialization class), HashMap of courses, where key is the Mark that student has, and boolean grant for knowing is student studying for free or not. For Student we wrote such methods as viewCourses() and viewMarks(), that return hashmap Courses. Student also could register to a course by method addCourse(), which adds this course to Courses hashmap, if specializations of this course contains specialization of the student. Students could rateTeacher(), this method raises “rating” int in Teacher class.

Teacher extends name, surname and salary fields from Employee. There are hashmap courses, but containing Student vector, enums “Rank” and “EducationDegree” and vector of teacher’s own files “library”. Method viewCourses(), that prints hashmap courses and putMark(), that sets attestation1, attestation2 or final in Mark class, by identifying what was the last putted mark.

In class Manager user checks are marks putted on time and writes Announcements for news menu (adds a new announcement to it’s vector in Announcement class).

Executor class has fields that it extends from Employee (name, surname and salary), also it has 2 arraylists (orders and executed orders). All Employees could send orders to Executor and this order would be added to orders list. Then, after executing this order it is would be added to list “executedOrders”. Also, there is method addBonus(), which adds int bonus to Employee’s salary, if his rating is high. We have class “Course” to keep information about the disciplines. It has files and textbooks collections and fields like name, faculty, teacher, credits. Adverting to this class, managers could set a teacher or credits of the course and teachers could add files.

One of the biggest classes in our project is “Admin”, because this class manages the other classes work. There we keep a collection of all users, their passwords, courses, announcements, textbook and faculties (serialize and deserialize them). Method recodnizeUser() is responsible for identifying who logged into system. If user is an object of students class, it adds this user to students collection and etc.

Login class contains methods like: viewFaculties() (return the vector of allFaculties in Admin class), viewTeachers (same logic) and viewCourses(). Login implements this methods from viewGeneralInfo interface.



**Conclusion**

As a result, we created University Study System (Intranet) realized using Object-Oriented Programming. Before implementing it, we did 2 types of UML diagrams to understand the main concept.

To sum up, this project helps us to use knowledge that we got from our lesson on practice and raise our teamwork skills. Working together, we shared new information and some tips about difficult themes. Obviously, there are some things, that we could fix, improve and add to our Intranet (like attendance, for example). However, we thing that we handled with this task and did our best. Finally, we are glad to pass our first big project, because it was an interesting challenge and great chance to present our skills.