# OOP Project Description - Information System of Research-Oriented University (Console-based)

It is a big project, but remember you have 8 arms in your team. Not the only one team leader :)
Project have various kinds of jobs, and it is the responsibility of a team leader to complete assignments effectively after careful examination of human resources.
It is a part of your amazing major! Good luck!

Your project - research-oriented university system. You should have classes (superclasses, subclasses, abstract classes), interfaces, enumerations, your own exceptions, patterns (to be studied next week), etc. - all techniques we have studied. Before coding you must design your system - create an architecture using **Use Case** and **UML class Diagrams.**

Your project costs 30 points - 10 pts in ATT2 **and** 20 of your final exam (other 20 points - oral exam. Team leaders automatically take 20 for this part).

## Attestation 2 Points distribution

Lab3 - 6 pts
Partic. points- 4 pts
End term - 10 pts
Project Part A (Use Case & UML) - 5 pts
Project Part B (part of the implemented classes, at least 10)- 5 pts

## 3 project parts

### ⚒️A. Diagrams

use case & class diagrams, submission: 14th week
Use GenMyModel (https://www.genmymodel.com/products/) or TopCoderUmlTool (https://github.com/topcoder-platform/topcoder-UML-Tool) or any other tool you like, but the only requirement is to be able to convert your class diagram to code. It is better to use these options; they are both free and have everything we need. GenMyModel can be used by every team member using one email (it works). Otherwise, it is not free. It also allows to change the transformation code to suit your needs. For a top coder installation, some Mac users will need to disable security settings; instructions are here https://www.youtube.com/watch?v=ZH8_XHzkKD4&ab_channel=kaboratech .
Upon finishing, generate code sources from your class diagram and fill the classes with method implementations.
Do not forget about reverse engineering - reflecting back changes to your diagrams.

### ⚒️B. Models (Classes)

Show a *draft* of your classes during the 15th week (practice time), the *final* version on the final exam

## ⚒️C. Demo (test class in console)

To be submitted during your final exam with finished and corrected parts A and B.

## ⚒️D. Project materials - documentation (HTML, for at least 5-10 classes), presentation, report.

**For your Final exam:**
SUBMIT YOUR REPORT(pdf only) and PROJECT (Zip only, with documentation) and PRESENTATION (pdf) IN COMMENTS UNDER TELEGRAM POST (you will see it later).
Indicate surnames and names of the team members in a message. *As a private message, the team leader must share with me a link to a telegram/ teams chat used all this period between team members to discuss and implement the project.*

**The report** must be complete, detailed, and well-structured. Include a detailed description of your classes, interfaces, etc.  Include code fragments and final versions of UML diagrams in your report. In addition, provide info about your problems and project management issues, screenshots of the teams, or a telegram group.

**The presentation** should have no more than 3-4 slides (what works/what does not work)

For the B part, you are not obliged to simulate the system at work; you just need to create models - classes, interfaces, enums, etc.

## REQUIREMENTS

I omit details description because that is not only a programming task but also a DESIGN task, so it is up to you which methods/fields your classes will have. And generally, which classes you will have :) Required are: ***User, Employee, Teacher, Manager, Student, GraduateStudent (can be master or PhD), Admin, Course, Mark, Lesson, Researcher, ResearchPaper, ResearchProject, News, Message***. Other classes are up to you.

## Your system must support:

- Lesson type - lecture/practice
- 3 languages - KZ, EN, RU
- Student - bachelor. Graduate Student - Master/PhD. Bachelor, Master, and PhD students must be represented by different classes.The difference in credits number, education years, etc. must be taken into account.

It is up to you which differences to include; they can be related to research, lecturers (e.g., a person without a PhD cant teach PhD students but can teach bachelor students), etc.

- Teacher must have a method to send a complaint about student(s) to a dean with urgency level LOW, MEDIUM, HIGH
- all master's and PhD students have a research supervisor who is a Researcher. If a person whose h-index < 3 is assigned as a supervisor, a custom exception must be thrown.
- There can be more than 1 instructor per course
- In the system, Teachers and Students CAN be researchers. Those teachers, who are professors and PhD students and master students ***ARE always*** *researchers*. However, bachelor students and teachers who are not professors(e.g., tutors, senior lecturers, etc.) **can** also be researchers. And there can be an employee, who is neither a Teacher nor Student, but he is a Researcher. Researcher has a research projects(s), research papers (also an Object!) etc.
- Fields for a ResearchPaper can be chosen from here - https://ieeexplore.ieee.org/document/9766691 . Take 5-10 important ones, citations, name, authors, journal, pages, date , doi etc.
- Researcher must have a method `PrintPapers(Comparator c)` that prints his research papers in sorted order, dictated by the comparator - by date published or by citations or by the article length (use pages).
- Researcher must have a method to calculate h-index.
- The system must support printing research papers of all researcher in the university, also sorted by date published or by citations or by the article length.
- The system must support printing top cited researcher of the school, of the year (among all schools).
- ResearchPaper must have a method `String getCitation(Format f)`, where format can be either 'Plain Text' or 'Bibtex'.
- To see them, please follow the link above and click the 'Cite this' button.
- ResearchProject has a topic, published papers, and project participants. If someone who is not a Researcher tries to join the ResearchProject, custom Exception must be thrown.
- Report generation (about marks, just simple statistics)
- Служебные записки (work messages)
- Diploma projects. Diploma projects of master and phd students must have a list of published research paper(s).
- Major, minor, free elective courses. Please note, that for a SITE student some major course from Oil and Gas school can be a free elective.
- News with comments. News with a topic "Research" must be prioritized in order (pinned). When some Researcher publishes a paper, there must be an announcement. Also, don't forget to automatically generate news about top cited Researcher in the university.
- As it is a research university, it has its own university researh journals. All users in the university (not only researchers) can subscribe to some university journals. The system must notify readers when the new paper in published in

the journal they are subscribed. From time to time, new journals appear. Which pattern is this?

- 4 or more design patterns

## General requirements:

- **OOP style**
- **Usage of Comparable, Comparators, equals, hashcode, toString, etc.**
- **Properly working serialization (think about Data Storage and some pattern)**
- Any user should access the system via authentication
- Intuitive usage
- **Consistency with UML**
- Do not forget about proper usage of enumerations. You can use them to represent teachers' titles, for example - tutor, lector, senior_lector, professor, etc.
- Proper and logically consistent usage of Collections
- Usage of java api (standard classes). Do not reinvent the bicycle.
- **Documentation**
- Low coupling, high cohesion

**Approximate Checklist (for everything except research):**

- **Admin**
  Manage Users (Add, remove, update)
  See log files about user actions

- **Teacher**
  - View Courses
  - Manage Course
  - View Students, info about students
  - Put marks
  - Send  messages to other employees (actually, any employee can send the message to any employee), Send Complaints

- **Student**
  - View Courses, Register for Courses
  - View info about teacher of a specific course
  - View Marks
  - View Transcript
  - Rate teachers
  - Get Transcript
  - Student organizations (e.g., OSIT). Student can be a member/head.

- **Manager**
  - Approve students registration, Add courses for registration (specify for which major/year of study the course is intended)
  - Assign courses to teachers
  - Manager types – OR, department, etc. (use enumeration).
  - Create statistical reports on academic performance.
  - Manage news
  - View info about students and teachers (in different ways , e.g.,

sorted by gpa , alphabetically , etc)
    - View requests from employees (they have to be signed by dean/ rector)

- **Researcher - described above. Researcher is a mystery. Is it an interface? Abstract class? Created using Decorator pattern? Just employee? Figure it out. There is no single answer.**

**Important note:**
**Account for details! For example:**

- Students can't have more than 21 credits
- Students can't fail more than 3 times
- Mark consists of 1st, 2nd attestation, and final.
- Many more, really …

**MOST IMPORTANT FUNCTIONALITY - course registration, putting marks, research. Finish it first.**

We will study design patterns, collections streams in upcoming lectures.

**Bonus:** Take into account as many details as possible (for valuable extra features you will get extra points, e.g. Schedule generation (take into account room load, room type, etc.), Attendance, Report generation option for teacher (about marks), advanced search by regular expressions, startups, recommendation letters.