Project Report

Team name: *Tourists*
Team members: *Ruslan Bazhenov, Kainolda Yassawe, Ali Rakhimzhanov, Tamirlan Baigara*

Algorithm used in program

In order to implement the creation of timetable, the program uses the following classes: ‘course’, ‘faculty’, ‘group’, ‘room’, ‘student’, ‘time’ and ‘timeslot’. It also uses the inheritable class ‘reservable’ to indicate whether the object has timeslots that are already reserved, indicating that object (faculty, student, room) is not available.

One study week is divided to five study days (mon-fri), each having five timeslots. Therefore, there is 20 timeslots available for each particular room, group and faculty member.
The program checks availabilities of each room, faculty member and group every time it considering one case, and if they are available simultaneously and other conditions explained below are satisfied, it “bonds” them together.

Class ‘room’ has ‘complab’ and ‘lab’ parameters indicating whether each particular room has inventory to conduct labs or complabs there. Our program is based on assumptions that labs could be provided only in lab rooms, complabs only in complab rooms and tutorials could be conducted basically anywhere (even in Orange Hall if needed, for example).

The program uses class ‘course’ as the “reference frame”, meaning that it takes object of course, register it in accordance with other classes, then proceed to the next one. In first place it registers labs and complabs, only then lectures and, subsequently, tutorials.
When registering the course (creating the “chain” between group, time, course and room), each room is given its priority:

- Firstly, obviously, room should be available at this timeslot and have enough capacity for the group
- Secondly, for timeslots assigned for labs and complabs of each course, the first priority will be given to the room having non-zero parameters ‘lab’ or ‘complab’ (because we cannot conduct complab without computers or lab without equipment)
- Thirdly, if size of the room satisfies the number of students in group, then the smaller is the capacity of the room, the higher its priority. For example, if group consists of 20 students and we have two rooms, having capacities 30 and 300 accordingly, the group would be assigned to the former one (because we don’t want to waste Orange Hall to provide tutorial for 20 people, for example).

Rooms are sorted by their priority. First comes rooms with computers, then rooms with lab equipment and finally rooms with both computers and lab equipment. Inside these categories, rooms are sorted by their capacity in accordance with the principle shown above.

The number of labs, complabs and tutorials per week for each course is given in input.

Example of input for course:

“Programming
Vipin Kizheppatt
1L, 5CLb”.

After the “chains” are created, program sorts them by timeslots (from earliest to latest) and writes them in output with all corresponding information.

Reasons for choosing algorithm

1. Sorting rooms by their priority leads to minimization of used rooms, therefore in our algorithm the number of rooms reserved for courses will be minimum

2. In our algorithm, complabs will be conducted only in rooms in which it is possible to conduct them and it is similar with labs. Because of that, we avoid situations when wrong venue is chosen for the courses.

3. Unlike other algorithms, our one does not count labs, tutorials and lectures of the given course as distinct courses. Rather, they are represented as parameters of the course. This significantly helps to simplify input.

4. The approach used in this program is the most flexible and easy to follow. All characteristics are represented as different classes in different files and headers, so they can be easily removed or added to/from the program with the minimum intervention to the code. Furthermore, as the program is absolutely segmented, parts of it can be modified by members of team autonomously and at the same time.

Challenges

1. The uncertainty about how to arrange inputs occurred at the early stage of development. After several tries and discussions, we agreed to use three different input files for courses, rooms and groups instead of using just one input file.

2. Before proper method of sorting the rooms was implemented, it was a major source of mistakes. Program worked improperly and, for example, used to assign Orange Hall for tutorials in some cases.