**Report**

**Group Name:** SHARK

**Group Members:** Kalamkas Zhagyparova , Alina Zakharova**,** Ruslan Zhagypar**,** Shaikhislam Zhaxylykov .

The **aim** of this project was to generate an automated timetable for School of Engineering students of NU. The program should accept inputs as list of courses, list of faculties teaching the above courses, list of student groups and their courses, list of class rooms and their capacity.

**Algorithm:**

Since there is no the most efficient and the fastest way to solve this problem, it can be categorized to NP-complete problem. Therefore, we used brute force search algorithm, which is a common technique for problem solving. By numerating all possible inputs, the chosen algorithm later checks whether each variant conforms the problem’s statement/condition.

The implementation of our code consists of two parts: reading data from input and solving the problem.

Implementation of the first part is pretty simple, we just create variables for data and read it from text files.

The second part is a bit complicated, not because of algorithm, but because of implementation.

Steps of algorithm implementation:

* go through all days and available (9:00-18:00 excluding lunch time) time gaps.
* go through all rooms and check whether the room is empty at that time gap.
* go through all courses and check whether that course wasn’t taken at that time gap.
* go through all faculties who is not busy and teaches that course.
* if the room type is lecture, count the number of students taking that course and compare with room capacity.
* if groups are not busy at that time gap, assign that room, faculty and course to that time gap and print it on terminal window.
* if the room type is lab, take one group, which is not busy and is taking that course, assign it to that time gap and room and print it.

**Assumptions:**

In order to simplify the code, we assumed that for each course there is a one lecture per week with duration of 2 hours for all groups and one lab (tutorial) per week with duration of 2 hours for each group.

Second assumption and limitation is that there are no professors who can teach different courses for students of different years of study and majors.

**Team members contribution:**

Alina Zakharova: taking inputs from the text file using freopen () function and vectors; arranging time slots, taking into consideration 1 hour for lunch break; assigning rooms to courses by comparing the number of students in groups to corresponding room capacities.

Shaikhislam Zhaxylykov: taking inputs from the text file using freopen () function and vectors; iteration through all rooms for checking availability of rooms and courses; recovering all variables (rooms, faculty, courses) for a new time slot.

Kalamkas Zhagyparova: the idea of possibility for one faculty member to teach more than one course and its implementation through nested for loops and vectors; assigning rooms for lectures, afterwards assigning rooms for labs by checking whether the group is busy by lectures.

Ruslan Zhagypar: determining which text file to use by asking user’s choice for the major and year of study; the idea of keep tracking whether group, faculty etc. is busy or not at the given time slot, which was implemented by using Boolean operators; assigning rooms for lectures, afterwards assigning rooms for labs by checking whether the group is busy by lectures.

The code was commented by all team members.

**Problems with program:**

1. The program is student-oriented, which means the timetable will be displayed for all student groups in one major and year of study. The problem here is in not taking into account the possibility of one professor to teach courses for different years or majors of study.
2. The timetables for different years and years of study are independent of each other. If there is a common room assigned for different majors or different years, there might be more than one course assigned to this room for at a given point in time.
3. The above discussed problems can be overcome by creating one text file for all engineering students, however the timetable that will be displayed will not be divided by years or majors of study.
4. The input does not accept space between words, underscore only.

**Challenges faced:**

1. Deciding the format of inputs and outputs
2. Deciding the foundation of algorithm: how to assign the business of certain group, room or faculty member.
3. The main challenge was to keep the code clean and readable, because it has a lot of data, loops and if-else statements.
4. One more challenge was to integrate different parts of codes into one holistic program on the grounds that 4 students were working on different aspects of program.
5. Organizing working process of the group was societal challenge that we faced during making project. This includes organizing meetings, dividing the responsibilities and coming into agreement when there are different ideas.

**Overcoming the challenges (respectively):**

1. By the method of trial and fail
2. Asking an advice from adviser (2nd year NU student of computer science), who suggested us to use vectors and maps and told us about the NP-completeness of the problem.
3. Structuring the code using logic and knowledge gained from programming lectures and computer laboratories.
4. Organizing regular group meetings with the project discussion.
5. Creating WhatsApp group, regular meetings during free-time, discussions during lecture or tutorial breaks.

**Sample output is shown below:**



