**Group #9: ANONYMOUS**

**Group members:**

**Yerdaulet Tastanbek**

**Aidos Satan**

**Daniyar Mussa**

**Dana Aubakirova**

**Introduction**  
Timetable generating is one of the non-deterministic polynomial-time (NP) hard problems. Numerous algorithms can deal with this problem. However, understanding and implementing them is not a simple task. One of algorithm is heuristic search algorithm to find the optimal solution during making a class Schedule. However, it only works for simple cases. For more complex requirements, criteria and inputs, finding best solution is hard, even sometimes impossible. In our case as the requirement and inputs are not so complex we considered to you use Random algorithm. In this report, there will be information about the concepts of this algorithm, how do we use it to our program, challenges that we have during coding and how do we overcome them.   
  
**Background**  
During generating timetable many requirements must be taken into consideration (number of professors, students, classes, classrooms, capacity of room, computer or laboratory equipment in room and etc.). By the importance of the requirements they can be divided into several categories:   
• Class can be placed only in an empty, spare room   
• No professor and student groups have more than one class at the given time   
• The capacity of class should be enough to accommodate number of all student that attend to class   
• There is a lunchtime between 1PM and 2PM   
• Classes will be from 9AM – 6PM   
• If class requires lab equipment or computer it should be placed in the room which has the lab equipment or computer   
• Duration of classes.   
  
**Classes and their objects**   
In this problem, there are different type of data structures will be used for Room, Student Groups, Professors, Courses and Class constructed from this data. The most effective data type was using object-oriented programming instead of working with multi-dimensional arrays. There were used four different classes class Professor, class StudentGroup, class Room and class CourseClass. For the course, used only arrays, as there is no method will be used. Objects of these classes: 

Class Professor ProfessorObj[MaxProf];   
Class StudentGroup StudentGroupObj[MaxGroups];   
Class Room RoomObj[MaxRoom];   
Class CourseClass CourseClassObj[MaxClass]; 

**Challenges**

During the coding we faced a lot challenges. And then decided to write any challenge in one txt file and later tried to solve them. Here is the list of challenges that we have faced during the creating Timetable project:

1. Implementing proper data structure. We did not know how different data types we can use for the same array.
2. Reading from one input file. At the beginning it was decided that all values for initialization of attributes of classes should be read from one input file.
3. Initializing the members of classes. We did not know how we will use constructors to our project. Because it will run automatically when objects are created. Therefore, data from input files can not be initialized immediately when the objects arrays of classes are created.
4. Choosing an effective algorithm. We did a lot of research in this aspect. From the internet, academic articles, book we search to acquire any information about the Timetable geniting program. Most of them were about Genetic algorithm.
5. Working with large project. We cannot work only with one function main.
6. Working with memory allocation. Pointer and Reference. When we worked with functions and took the ofstream and ifstream files as their parameters and there was a compilation error. Because we observed that functions could not take the files itself. Because it needs a direct access to the data in the files and ability to modify them.
7. Duration of classes. As the duration of some classes are 2 hours, of some classes are 3 hours. And they should be held consequently. Meaning that if there is a tutorial of one course with 2 hours duration program should not generate it as one hour of class is in one-time slot another hour in different time slot of different day etc.
8. In this case there were a problem when even if we considered the duration of classes, and they will be consequently. There was problem that one part of class held in the last time slot of one day from 17:00 - 17:50 and left part held on the first-time slots of next day from 9:00-9:50.
9. No professor or student group can have more than one class at the same time.
10. Difference between Lab class and Tutorial Lab classes need lab equipment, so the room that this lab class will be held a should have a computers or equipment.
11. Lectures. In the lecture all groups should be attended. So that the program firstly takes a sum of all students in all groups and search room with enough capacity.   
    Secondly, all groups should have the same class (Lecture) at a given timeslot.   
    For each cycle of loop for different groups professor and room should not be free at that time.

**How did we overcome challenges?**  
1. We used Object Oriented Programming and Classes.

2. Reading values from different files of input. There are four different files of input: Professor.txt, Room.txt, Course.txt, StudentGroup.txt.

3. We considered not to use constructors. We parsed all the data from input files with different functions.  
4. Genetic algorithm use biology principles of chromosomes to get better solution.

5. Modular programming. As we work with big project in a group. To get the solution, we had to divide the project into small unit of functions and every member of group could work with it individually.

6. We considered using reference of files and work with pointers.

7. The loop is used to make class by taking into account its duration.  
8. To solve this problem, we divided one day for 2 parts. before lunch 4 hours and after lunch 4 hours and if our random time slot for class is more than 4 program breaks and starts from beginning.

9. There used 2D Boolean arrays to indicate whether ith Professor or Group is free in jth time.

10. In the input file there were added info to the class whether it need lab equipment and to room whether it has lab equipment when comparing them gives true lab will be held only in lab rooms.

11. The combating with last challenge was little bit complex. But we solved this problem firstly by indicating whether the course is lecture or not. And created 2 similar loops for solving similar problem to lectures and tutorial separately.   
  
  
**Algorithm**  
Algorithm used many nested loops for generating best solution, by comparing that all the conditions (criteria) are satisfied.   
1. During the algorithm 3 conditions will be checked:   
1) if room is empty at the given time, have enough capacity and if has lab equipment   
2) if professor is free the given time   
3) if the student group is free at the given time   
2. Algorithm starts searching best solution for each of student group. It means there is a for loop running through the each of student group and their data (that will attend courses, capacity). This for loop is the main loop and all other statements are in this for loop.   
3. Then algorithm go through the list courses that the Student group will attend. For this also used for loop as the number of iterations are known (from 0 till size of vector<int> course). For each of courses that Student bunch will attend following algorithms will be run.   
4. After choosing the course, program checks if this course is lecture or tutorial. For this, we implemented special algorithm that reads last four letters of courses and compare whether it is “\_lec”. The main reason of checking is that for lectures and tutorials should be implemented different algorithms. For example, more than one student groups can attend to the lecture at the same time. In addition, the room capacity should be greater than the total number of students of each group that attend to this lecture.   
5. If the course is lecture, number students of all student group attending to this lecture will be taken. Moreover, there is a checking code for whether this course is already set to the class of this group in any time of the week. Because, when program detects that the course is lecture it will firstly, search from other student groups if they attend to this lecture, if so it takes id of this group and set to the vector groups. Further, during setting timetable it will set the lecture course to this entire student group in the same room, with the same prof at the same time. Therefore, when program runs this algorithm to the second group it does need to set its lectures, because they are already set during iteration for the first group.   
6. While loop further used to determine if the all criteria for lecture is satisfied, meaning student group is free, professor is free, and room is empty at a given time. In each cycle, it takes random time and then check all above conditions to this time until they get the best solution. The room is also chosen randomly. The best condition for room is that the number of students should be less than or equal to the capacity of room and room should be empty a given random time interval. If all of the conditions are satisfied the while loop will be broken, and the Boolean indicator should be set to 1 meaning that groups, prof, room are not free at that time.   
7. All above for lectures, if the course is not lecture there is an else condition should be started.   
8. In the tutorials the main difference is that it will check the following conditions to only one student group, and in one cycle tutorials of only one group should will be set. It means that no longer professor and rooms are free at that time intervals for other groups and their tutorials should set in such a way that they do not overlap each other.   
9. Then the “while loop” similar to the 6th point were used to get best solution to the tutorial. But only thing which is different from lecture is that it will consider only one group, search lab room for lab tutorials, capacity of only one group and professor and room is not available for other groups in that time.

10. Finally when getting solution for only one class all information about this class will be stored in the ClassCourse class. And all steps above are repeated for other groups.

11. When time table was created for all student groups and stored in ClassCourse object. They will be printed out to the TimeTable.txt file.

12. End of Algorithm.

**Output**

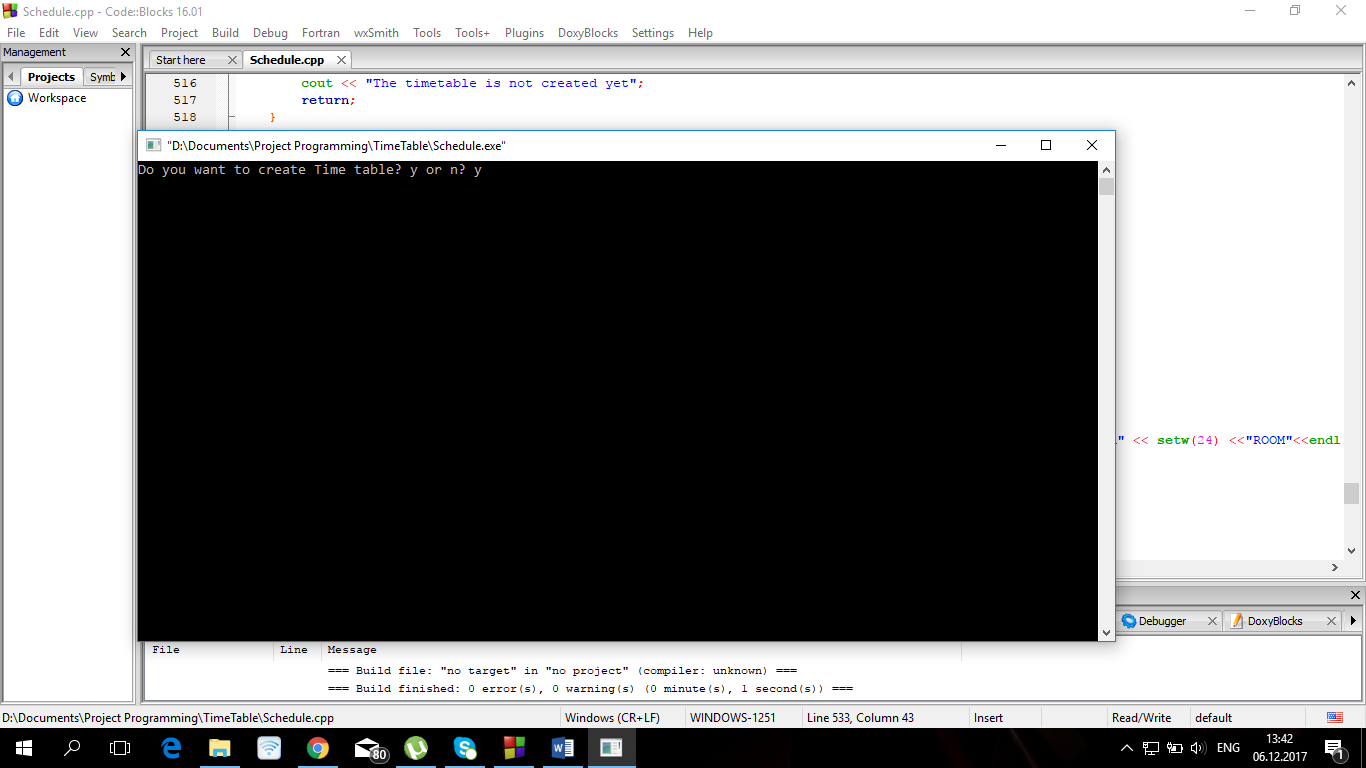
Final output will be for each of Student Group.

**Groupname**

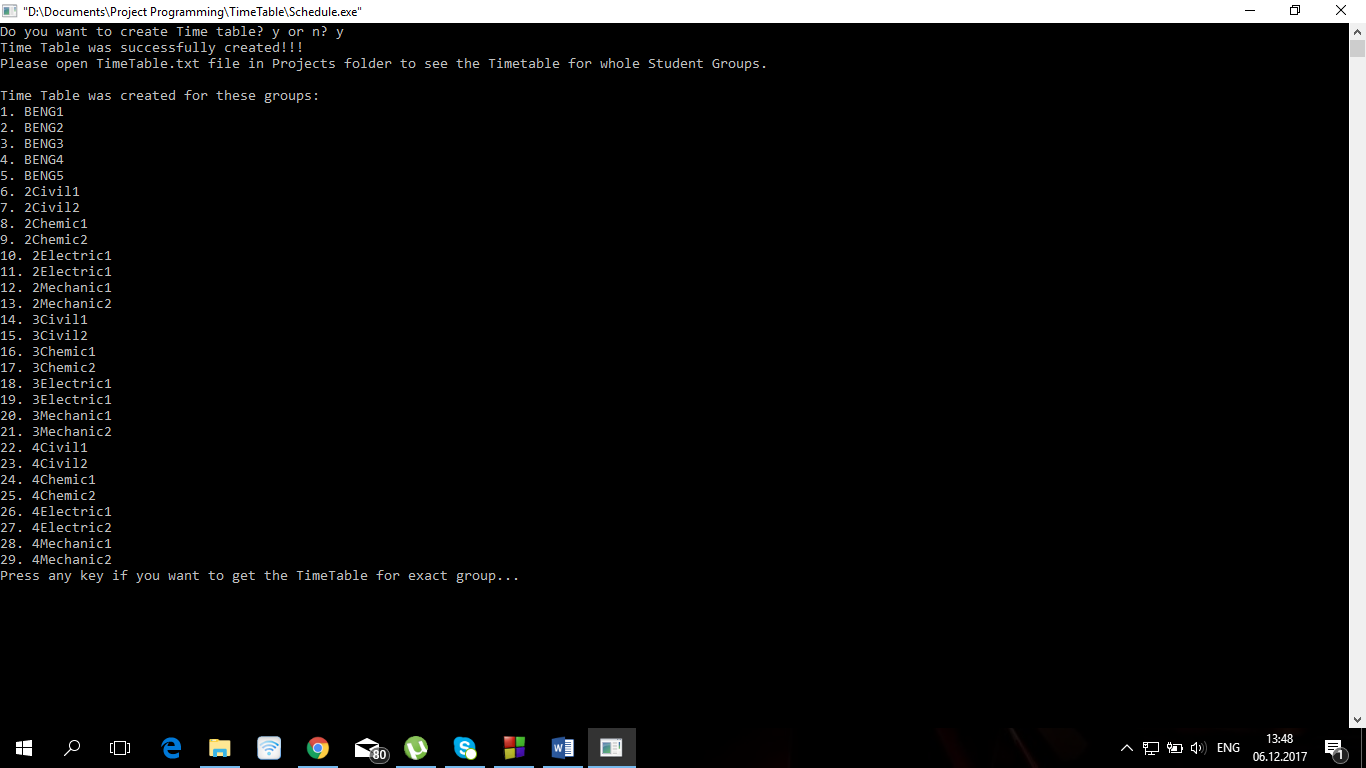
**Weekname |Timeslot | Coursename | Professor | Room |**

**|Timeslot | Coursename | Professor | Room |   
|Timeslot | Coursename | Professor | Room |   
—----------------------------------------------------------**

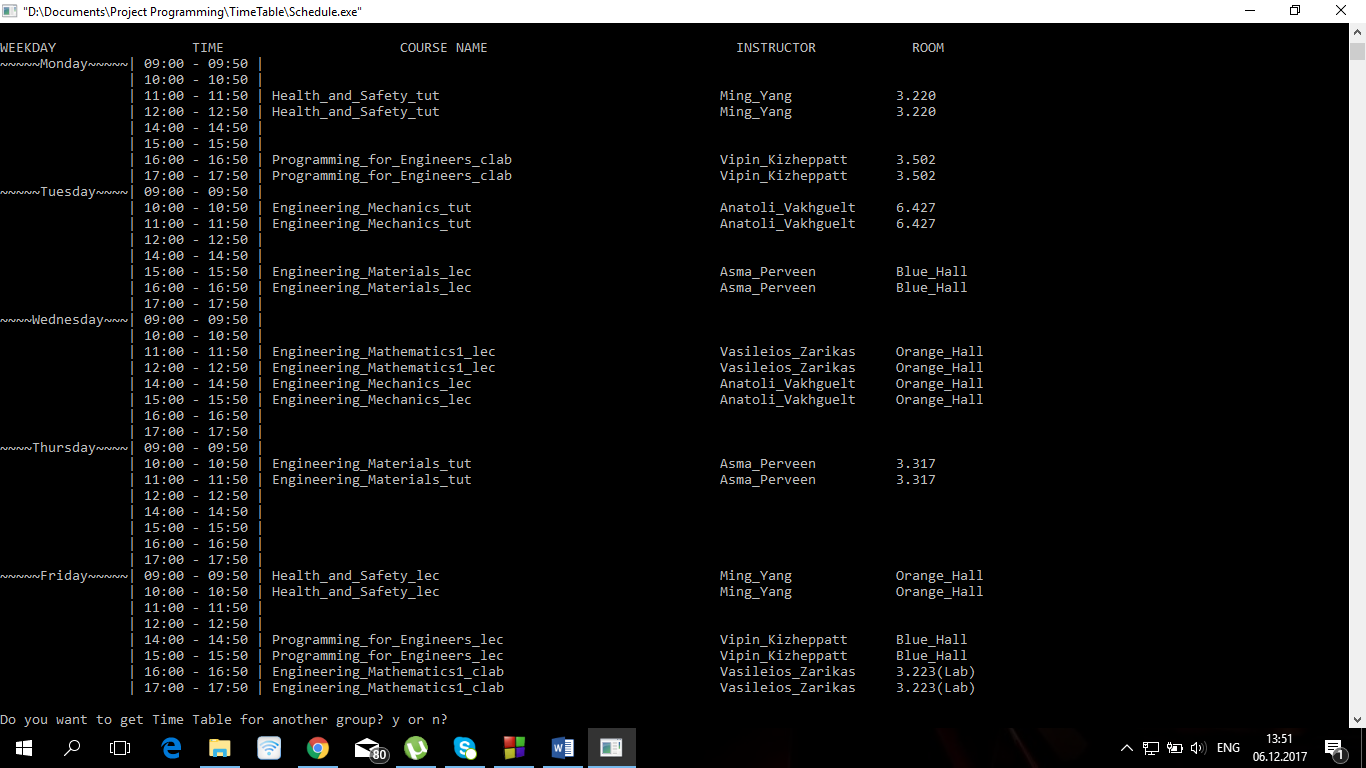
When user runs the program it firstly asks whether he wants to create new timetable or work with timetable which has already been created.



If the answer is yes, program will print to screen message if the Timetable created successfully or there went something wrong. If it is created successfully the list of Student groups will be printed to the screen.



If the user wants to see the time table for exact Student Group program asks to press any key, and write the name of group for which schedule should be printed out.



When time table for one group is printed out, program asks whether the user wants to continue search. If user agrees and wants to get timetable for whole student groups he can write “All” and the schedule for whole group will be printed out to the screen.

Remember that all timetable when created it save to the TimeTable.txt file in project’s folder, and you can see the schedule for whole group there. 