Video Tutorial Transcript

Uh, Hi!
Welcome to the video tutorial on how to use this timetable optimisation program.

So first we need to download the source code of the program which we do from moodle.
Here we go...
Now let's unzip the folder and check if it has what we need.
The cpp file is here. Now in order to build and run, we need Code::blocks IDE. Yes, so, we go there open the cpp file from the unzipped folder and build it and run it.
So yeah, this is how it looks...
Now before we start let me make you familiar with the premises of the program that we are working on: it is a set of constraints that we will introduce you to...we'll go to a word doc, wait a second.

In this application, we get an optimized timetables for students divided into 4 divisions-D1,D2, D3 and D4.For the Professors, and TA's both, we have considered a situation where 1 Instructor handles 1 Subject for 2 Divisions. In case of Lab TA, 1 Lab TA handles 1 Subject Lab batch, which comprises of half of the students of any division.

We have 5 theory courses –MA106,PH107 ,CH107,BB101, and IC102(Subject Codes), with number of lectures pre-determined by the Institute. MA106 and PH107 have 1 tutorial per week managed by Tutorial TA's, who also fill in their preferences. Also, there are PH117 and CH117 Labs which are pre-determined to be in morning for D1 and D2 divisions, and in the afternoon for D3 and D4 divisions. Corresponding Lab TA's also fill in their preferences according to their availability. So basically no division has a lecture slot scheduled in the time allotted for Practical Slot.

The Professors and TA's conducting the course will fill in their preferences in integer values from 1 to 5 for 5 week days on which they will be working(Monday to Friday), where magnitude of number is directly proportional to the preference level for that day. Also, just after the preference value for the day is entered, the Professor or TA enters the slot, either forenoon(slot 1) or afternoon (slot 2) or both(slot 0), when he would be happy to come and teach.

So now that that has been done, let's go back to the application window.
There are 3 options, with the last one being exit.
Let's check out the first one, default cases. This case was added for ease of experimentation and verification but still serves the purpose of checking some default cases. Yes, now we have these cases...

This is relatively less time consuming and has been designed in such a manner as to cover most of the common preference orders. We basically have 4 such default cases which will be explained next.

Enter option 2 and press Enter.
Here we are presented with the four default cases as options with a short description of how the preferences are assigned to each of the professors and teaching assistants. The default cases are such that:
1. In this default case, all the professors and teaching assistants are assumed to have the same preference order for all days in the week, that is Monday being the most preferred and Friday being the least, with all of them preferring morning slots.
2. Over here,
3. Here, all the teachers are assumed to have the preference order similar to option number one but with the change in preference of the teaching slots, with all of them preferring alternate slots on consecutive days. That is if the professors prefer morning slot on Monday, they will prefer afternoon slot on Tuesday.
4. Here, once again all the professors and teaching assistants are being allotted the same preference order, but the day of maximum preference is randomly chosen every single time the program is run, with the next day getting successively a lesser preference.

Perhaps we choose to go with the second default case.
Type 2 and press enter.

Our program internally generates a timetable and optimizes it.
We can now see the timetable generated for each division.
In our scenario, we are managing four divisions, namely D1, D2, D3 and D4.

Now, we have all the timetables in front of us and if we examine them carefully, we will observe that there is no slot clash for either the teachers or the students. Even the optimization in the timetable is quite evident, in the sense that first five professors were allotted the morning slots and the next five afternoon as was the need denoted by our input.

Now let's check the second case, which is input by the professor. Here, you are required to type in your name; And enter your preferences, basically it calls you to rank (5 for the highest and 1 for lowest) the days based on your ease of availability.

Here we manually have to enter the preferences of every professor and TA individually, which is tedious and time consuming , considering the fact that there are 10 professors, 8 Lab TAs and 4 Tutorial TAs. And each teacher will have to enter the preference order for all the days in the week along with their preference level for the morning slots and afternoon slots. Hence, we devised the next case.

But anyways we'll continue doing it, yeah, you might have to wait for while till the input gets over, let's skip that part and see the end result.

Yeah, so I guess that's it.

We can see that in this manner, our program is generating an optimum timetable for various preferences.
Thank You for listening.