# BLG 336E Analysis of Algorithms II

## Project 3

Monday 24th April, 2017

## Due: 9th May, 2017, 23:59

In this project, you are asked to find an efficient algorithm to assign publications for a group of academicians (reviewers) to review a paper.

For the acceptance of a paper to be published on any platform, e.g. journal, book, or conference, they should be reviewed by different academicians. However, each platform can have its own conditions. For example, if a writer wants to be published his/her paper on a journal, the paper should be reviewed by 2 reviewers. Also, in some publication platforms, only specific kind of titled academician's reviews are accepted.

In this project, suppose you have *m* reviewers and *n* publications. Each reviewer is labeled from 1 to *m* and similarly, each publication is marked from 1 to *n*.

Also, there are 3 types of publication platforms, which are *journal*, *conference* and *workshop* and publications are grouped by requested publication platform.

Each publication *i* must be reviewed **exactly certain number** of reviewers based on the conditions of requested publication platform:

* A paper must be reviewed by 2 reviewers to be published in a journal.
* A paper must be reviewed by 3 reviewers to be published in conference proceeding.
* A paper must be reviewed by 2 reviewers to be published in workshop proceeding.

Moreover, reviewers are grouped by their titles because some publications must be evaluated by a specific titled academician. In this project, 3 types of titles exist: *Professor*, *Assistant Professor* and *Research Assistant.*

These titles also identify for which publication platform they can make review. The following table shows this information:

| | Journal | Conference Proceeding | Workshop Proceeding |
|---|---|---|---|
| Professor | + | + | + |
| Assistant Professor | - | + | + |
| Research Assistant | - | - | + |

Each reviewer *j* can study on **just one publication** at a time.

You have the information for each reviewer (***reviewer.txt***), describing the how much time in total they can spend for reviewing. Also, ***publication.txt*** gives the list of all publications. First line of these files indicate the number of reviewers and publications, respectively.

An output file (***output.txt***) should be generated at the end of the execution that contains the assignment for each publication. If a valid assignment does not exist, **an error message** ("***Publication #i cannot be assigned***.") should be written to the file. Note that, your algorithm may be tested on larger sized inputs as well.

Your program should compile and run using the following commands:

```
g++ yourStudentID.cpp –o project3
./project3 reviewer.txt publication.txt
```

## Code (60 points)

Implement an efficient Network Flow algorithm that either finds a feasible assignment of all publications or determines that no such assignment exists.

## Report (40 points)

- Formulate the problem properly as a Network Flow problem. Visualize your network by stating flow values. (20 points)
- Describe the method that you implemented for the task allocation problem **in detail** and give the complexity of the algorithm in big O notation. (20 points)

## Submission

You should be aware that the Ninova system clock may not be synchronized with your computer, watch, or cell phone. Do not e-mail the teaching assistant or the instructors your submission after the Ninova site submission has closed. If you have submitted to Ninova once and want to make any changes to your report, you should do it before the Ninova submission system closes. Your changes will not be accepted by e-mail. Connectivity problems to the Internet or to Ninova in the last few minutes are not valid excuses for being unable to submit. You should not risk leaving your submission to the last few minutes. After uploading to Ninova, check to make sure that your project appears there.

**Policy**: You may discuss the problem addressed by the project at an abstract level with your classmates, but you should not share or copy code from your classmates or from the Internet. You should submit your own, individual project. Plagiarism and any other forms of cheating will have serious consequences, including failing the course.

**Submission Instructions:** Please submit your homework through Ninova. Please zip and upload all your files. In the archived file, you must include your completed report file and all your program and header files. All your code must be written in C++, and we must be able to compile and run on it on ITUs Linux Server (you can access it through SSH) using g++. You should supply one source code file that calls necessary routines for all questions (Multiple files are acceptable, as long as you state the compilation instructions in your report). When you write your code, follow an object-oriented methodology with well-chosen variable, method, and class names and comments where necessary. Your code must compile without any errors; otherwise, you may get a grade of zero on the assignment.

*If the question is not clear, please let the teaching assistant, **Tuğba PAMAY**, known by email (**pamay@itu.edu.tr**)*