Math 3134: Applied Combinatorics

Coding Project 3

This project is *optional*.

This project is optional. If you submit a satisfactory project, one of two things will happen, whichever is to your advantage:

1. I will use your score on this project to replace your lowest project grade.
2. I will count all three projects for 12.5% of the grade (and scale the rest of the percentages proportionally.) *I do not encourage you to invest much time on this project, for a relatively small benefit. Your time can probably be used in better ways!*

Instructions: Write a well-documented program which

1. reads in a network, in the form of a weighted, directed adjacency matrix (described below).
2. provides a partial implementation of the "greedy" Ford-Fulkerson algorithm to find a maximal flow in the network. You will not have to implement the portion of the algorithm which finds augmenting paths involving backflow.
3. Shows the flow paths and the capacity along each path, along with the total flow of the network.

Specifications: The input for your program will be a textfile containing in order:

- The number of vertices n, followed by a comma,
- The entries in the weighted, directed adjacency matrix, left-to-right, top-to-bottom, separated by commas.

Note that since the arrows in a network are one-way, the matrix will not be symmetric. In other words, for two vertices, i and j, there may be an arrow from i to j, but not from j to i.

You may assume that the first vertex listed is the source, and the last vertex listed is the sink.

For example, this is valid input:

```
6,0,5,0,0,0,0,0,0,3,3,0,0,0,0,0,2,2,0,0,0,0,0,4,0,0,0,0,0,0,6,0,0,0,0,0,0
```

You may write your code in any reasonably English-like language (Python, Java, Matlab, Mathematica are all fine but please not Befunge or Ook!). You may include libraries which provide matrix functions, but no graph-theory libraries or other functions which trivialize the assignment. You may discuss the project with other classmates, but may not share code in any form.

I will provide several input files on the course website—some will have the correct answers given so that you may test your code, and one will not have the answer given (you need to provide that answer as part of the assignment).

Grading: Any code which works correctly and is submitted on time will receive at least a passing grade of 70%. The remaining 30% will be based on somewhat subjective criteria including coding style (for example: Are there any variables which are defined, but never used? Are there large blocks of code which appear multiple times in the program and could have been modularized?), and reasonable standards of documentation (for example: Do the variables which store important data have names that accurately reflect their use?). My overarching subjective criterion is this: *Does the code indicate that the student understands the graph-theory definitions and algorithms covered in this project?*

You may email me your code at `heathdav@math.vt.edu` or submit a hardcopy in class.