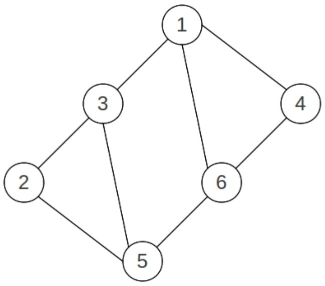
COT 4400: Analysis of Algorithms
Final Project (100 pts)
Due: Thursday, April 24, 2014

You are not allowed to use the internet or consult any references. The only people you can work with on this project are your group members.

1 Problem Description

Your input is an undirected graph, G, with n vertices and m edges connecting pairs of vertices. The object is to design an algorithm that splits the vertices into two sets so that the number of edges between vertices in different sets is maximized. Note that the two resulting sets do not need to have the same number of vertices.

Example:
Split the following graph into two sets.



Observe that a split of ({1,4,6}, {2,3,5}) results in 2 crossing edges, whereas a split of ({1,2,3}, {4,5,6}) results in 4 crossing edges, making the second split superior to the first.

Note that the problem is NP-Hard, so don't plan on getting a perfect solution. Your task is going to be to get as good a solution as possible using a a reasonable amount of computer time. Please note that my test examples my contain a large number of vertices (e.g., $n = 10,000$), so you should design an algorithm with polynomial complexity[1].

2 Input/Output

- The Input: (`input<#>.txt`)
  The first line contains two integers where the first integer denotes n, the number of vertices, and the second integer denotes e, the number of edges in the graph. The vertices of the graph may then be denoted by integers in the range $[1, n]$. The following e lines are used to denote each edge of the graph. An edge in the graph is denoted by two integers which represent the two vertices it connects.

- The output: (`<studentLastName><#>.txt`)
  For each test case, the output must follow the description below.

  The first line should contain the runtime, in milliseconds, of your algorithm for the test case. The second line should contain the number of crossing edges. The next two lines should contain integers, separated by spaces, to indicate the vertices in each set.

An example input/output file will be posted on Canvas.

[1] If your algorithm/code is too slow to handle the inputs of the prescribed size you will lose points for both your algorithm design grade AND your results grade.

3 Deliverables

Schedule a 20-minute meeting with me to demonstrate your project in one of the time slots that I will make available for this purpose. All group members must be present at this meeting. I will email input files in the format described above to all group members approximately twenty minutes before the scheduled time for your demo. *You will have to run your program on these inputs and output the results and the run time of your program in the given format.* The meeting will take place in my office (ENB-339). Be prepared to answer questions about your algorithm, its implementation, and what each group member's contribution was. The only written deliverable is the *outcomes* form that has been posted separately. Answer the questions on this form and use it to report your results. You will also electronically submit your code and an output text file for each of the inputs.

4 Grading

- 80 points: Algorithm Design, Analysis, and Implementation
  Develop a reasonable algorithm that is able to get as good a solution as possible using a a reasonable amount of computer time. Be able to discuss your algorithm, including design decisions, implementation, and analysis of your algorithm.
- 20 points: Results
  A portion of your grade will be based on how your solution compares to solutions generated by other groups.

Good Luck!!!