CS 594/690, Graph Algorithms, Applications and Implementations
Spring 2017, Homework 7

1. Add helpful comments to your code from homework 5 so other students can follow your algorithm. If your code was incorrect or ran slowly on some of the graphs, correct it and/or modify it to run faster.

2. Here are the first, second and third choices of presentation dates for the twelve CS 594 groups. Exactly two groups must be assigned to each of the six possible dates.

    1. March 29, April 5, April 12
    2. March 22, March 29, April 5
    3. March 29, April 5, April 12
    4. April 5, March 29, April 12
    5. April 19, April 12, April 5
    6. April 5, April 12, March 29
    7. March 29, April 5, March 22
    8. April 19, April 12, April 26
    9. March 29, April 5, April 12
    10. April 5, April 12, April 19
    11. March 29, April 5, April 12
    12. April 26, April 19, April 12

    a. Give an assignment of groups to dates that represents a maximum bipartite matching using a weight of 3 for a group's first choice, 2 for their second choice, and 1 for their third choice.
    b. Give an assignment that instead uses weights of 1000 for first choice, 2 for second choice and 1 for third choice.

3. Write a program to compute the Prüfer sequence of a tree. Your program should take the name of a graph file as a command line argument and output the Prüfer sequence of the graph if it is a tree. If the graph is not a tree, your program should output a message to that effect, and no sequence. (Note that forests are not trees, so simply checking that there are no remaining degree one vertices will not be sufficient by itself to test whether a graph is a tree.) For this assignment, we are assuming that the vertices of the graph are labeled 1 to *n* rather than 0 to *n*-1. The output should look very similar to the following two examples.

```
>./findprufersequence pufergraph1.txt
1 3 3 1
>

>./findprufersequence pufergraph2.txt
not a tree
>
```

4. Write a program to input a Prüfer sequence and output the unique tree for that sequence. Your program should input the sequence on standard input (do not use command line arguments or input from a file). Assume the input will be space-separated numbers. The output should be in the unweighted graph format we have been using this semester, except with vertices numbered 1 to *n* rather than 0 to *n*-1. The output should look very similar to the following.

```
>./makeprufertree < 1 3 3 1
6    5
1    2
3    4
3    5
1    3
1    6
```

The following apply as usual.

- You may choose any programming language you wish, as long as your program compiles and runs when invoked from the Linux command line on the EECS Linux machines, using only software currently installed. I will test your code on one of the Hydra Lab machines.
- Do not use any library routines specifically designed for graphs (e.g. Boost).
- Include an example how to compile and/or invoke your program in a README.txt file.

Submit your program by emailing all files necessary to compile and run your code to cphill25@utk.edu prior to the beginning of class next Wednesday, February 8. Bring a hard copy (either printed or handwritten) of your answers to question 2 to class. If you have any questions, please do not hesitate to email me or drop by during office hours.