# CS 337: Algorithms: Design & Practice

## Lab 7: Simple Graph Algorithms

### The W-O-M-A-N Puzzle – March to Washington, DC

In this lab represent graphs and implement some simple graph algorithms to solve a quintessential Bryn Mawr Puzzle. First the puzzle:

You need to travel, by land, from the state of Washington to Washington, DC. You are only allowed to travel through states whose names begin with any of the letters in the word W-O-M-A-N. That is, you can travel through states like, **A**rkansas, **M**innesota, **W**yoming, etc. but not through **C**alifornia, **T**exas, **U**tah. Is it possible? If so, what would be the path?

**Graphs:**. You can (and should) solve the puzzle above by modeling it as a graph search problem. (See Chapters 5 & 6 from Cormen)

Given a graph of all the states in the United States and their neighbors, you can model the problem as a graph search problem where Washington is the start state and Washington, DC the goal state. A database of states and neighbors is available in a text file:

**Link:** http://cs.brynmawr.edu/cs337/USStates.csv

You may read this file from the URL or download it to you computer and work with it there. If you are working on the deparment's UNIX machines, the file is also available at /home/gtowell/Public/337/USStates.csv

You can get this file to your computer using scp:

scp YOU@MACHINE:/home/gtowell/Public/337/USStates.csv .

Here are some of the entries in the file:

```
Colorado, Arizona, Kansas, Nebraska, New Mexico, Oklahoma, Utah, Wyoming
Connecticut, Massachusetts, New York, Rhode Island
District of Columbia, Maryland, Virginia
Delaware, Maryland, New Jersey, Pennsylvania
Florida, Alabama, Georgia
Georgia, Alabama, Florida, North Carolina, South Carolina, Tennessee
Hawaii,
Idaho, Montana, Nevada, Oregon, Utah, Washington, Wyoming
Illinois, Indiana, Iowa, Michigan, Kentucky, Missouri, Wisconsin
```

Each line starts with a state; which is followed by all its neighbors. So from the above data we can see that Florida has two neighbors: Alabama and Georgia.

**Task#1:** Create a graph ADT (Abstract Data Type) to input and store the above graph. Use the ADT for write the following program. First, print out the state with the most neighbors. And then, use the graph as the foundation of an interactive program that allows users to enter a state name and learn about that state's neighbors. For instance, Here is a sample of the interaction in a program I wrote:

```
Enter a state: Florida
Florida has the following neighbors: Alabama, Georgia
```

```
Enter a state: Idaho
Idaho has the following neighbors: Montana, Nevada, Oregon,
Utah, Washington, Wyoming

Enter a state: Hawaii
Hawaii has no neighbors.
```

**Task#2:** Solve the W-O-M-A-N Puzzle. Your program should first say "Yes" or "No" whether there is a way to go from Washington to Washington, DC. If the answer is "Yes" it should print out the names of the states the path goes through to get from Washington to Washington, DC (District of Columbia). That is:

```
No. There is no way to get from Washington to District of
Columbia.
```

Or

```
Yes. To get from Washington to District of Columbia, march as
follows:
Washington, Oregon, Nevada, …, District of Columbia
```

Note that the above may not be correct. Your implementation should make judicious use of already implemented data types and structures in the library of the language of your choice.

Your program will be graded based not just on correctness, but also on choice of algorithm(s), programming style, and use of proper abstractions. (I will look closely at your code.)

**Extra Credit:** Suppose we change the destination to Pennsylvania (it does not begin with a 'W'). For each state in the US, print out whether it would be possible to travel from there to Pennsylvania (i.e. to Bryn Mawr College) and the path, if possible. The constraints of traveling through W-O-M-A-N states remain the same except for the start and end states (e.g. starting in Georgia is it possible to travel to Pennsylvania by traveling only through W-O-M-A-N states?).

**What to submit:**

Create a written report containing the following:

Make you secret identity two — non-fictional, non-extinct — animals

1. Discussion on this lab. Your writeup should include a figure(s) depicting the graph data structure you used. Your text should well-describe this data structure. Also in your discussion: describe and name the graph search algorithm employed (justify why that search algorithm makes sense), and the specific data structures/libraries you used.
2. A discussion summarizing the results from Extra Credit if you did it.
3. Appendices containing:

a. A printout of a sample run from Task #1 showing the output on at least 3 states and one non-state.
b. A printout of the output from Task#2.
c. A printout from Extra Credit if you did it.
d. A complete listing of your code from Task #2.