**CS 311 HW7 Graph Algorithms Part 2**

**MST and Shortest Path (based on week 11 – 12)**

**==========================================================**

**DUE: Week 14 Thursday at the beginning of class**

**TOTAL 20 points Your score: (EC is available. See below)**

**\*NAME: Justin Goulet**

**\*DATE SUBMITTED: 3 December 2015**

**----------------------------------------------------**

**Purpose: To be able to implement DFS**

**----------------------------------------------------**

**==================================================**

**HW7 Implementation DFS of a Graph**

**==================================================**

**[20pts] Your score:**

**Now that you have a directed graph class from HW6, you can implement DFS.**

**You also need a stack class (from HW1) so that you can push vertex names onto a stack. (Where do you need to include stack.h? What do you compile?)**

**You need to add the following functions to the graph class:**

* **void visit(int, char) which will enter the given visit number for a given vertex**
  + **this is to indicate the order in which vertices were visited.**
* **bool isMarked(char) which returns true if a given vertex was already visited**

**(0 means not visited)**

**Make sure displayGraph now displays the visit numbers as well.**

**Your client will implement the DFS algorithm from Notes-11A.doc using the stack class and the graph class functions as follows:**

**Visit (1, ‘A’)**

**isMarked(‘A’) is then yes**

Display the graph before DFS begins.

**Mark/visit A (\*\*), the start vertex visit number as 1.**

**Get the adjacency list of A and push adjacent vertices onto the stack.**

**Display the stack**

**While the stack is not empty do**

**{**

**Remove a vertex v from the stack.**

**Display the vertex name.**

**If v is not marked yet (visit number is 0) then**

* **mark it (visit it \*\*) and inform the user E.G. “visited B”**
* **get its adjacency list and put adjacent ones on the stack.**
* **display the stack clearly labeling it has the stack**

**}**

**Display the Graph with visit numbers for all vertices.**

**Do not display unused entries of the Gtable.**

**(\*\*) visit numbers will start at 1 and increase as you traverse.**

**Add many labeling cout messages to make your output understandable.**

**Testing: Use the same input file as for HW6.**

**Submit the output for starting at vertex A.**

1. **state of the program statement [2pts] Works as expected**

* **Does your program compile without errors?**
  1. **Yes**
* **List any bugs you are aware of, or state “No bugs”:**
  1. **No bugs**

**Submit these files:**

* + - 1. **This assignment sheet with your answers**

1. **All graph class files : header, implementation, and client (commented well)**
2. **test results showing what you compiled and ran**
3. **Keep this set (HW7 files) of files for your future classes.**
4. **Congratulations! You have just finished writing a program that uses stack, linked list and graph classes.**

**Graphics Extra Credit [10 pts]**

**Do this version separately from the regular HW (i.e. finish the regular version first)**

**Must work perfectly to receive any points.**

**You will have to present your work on Week 15 Tuesday to receive points.**

**At the beginning of the program and at the end of the program call graphical display function to display the graph using graphics. All nodes must have names and visit numbers.**

**Email me all the files by Week 14 Thursday to schedule the presentation.**

**Presentation Requirement:**

* **Explain what tool you used and why.**
* **Show the graphical display function code and explain how it works.**
* **Compile and run the program.**