CSCE350 Programming Assignment 3

--Due Date on dropbox.cse.sc.edu--

Please submit *just* Graph.h. Do not zip the entire folder. All your code goes in that one file.

Description:

- You will code Floyd's and Warshall's
- This is a little bit shorter than usual *because you may need more time to get your environment setup (you need Linux).*

Notes:

- We will be doing this in C++.
- You will build with CMake and Make.
- You will need to do this in Linux, so use either the department lab or work from an Ubuntu install. There are various ways to use Ubuntu without overwriting your OS.
- Included is a copy of Google Test (gtest). Cmake should automatically compile everything for you so as long as all you change is the Graph.h file.
- Once you're done you might consider checking your HW9 answer – look in targetgtest.cpp and just ad your own tests (just cout your output).
- Your code should go inside the given functions – if you change the signature (return type, parameters, it won't even compile).
- If you don't have a favorite text editor, popular choices are emacs, vim, and Sublime.
- Don't forget to *save* your file before you **make** it (**make** is what actually compiles and links it). **cmake** is what generates a **Makefile** for **make** to use.

Setup:

- Download the program assignment to a convenient directory (outside of of Downloads probably)
- Run the following commands:
  - **tar -xvzf Program_4_350.tar.gz**
  - **cd Program_4_350**
  - **mkdir build**
  - **cd build**
  - **cmake ..**
  - **make**
  - **./runUnitTests**
- You should see all four tests fail.

Warshall's – Transitive Closure:

- Fill out getTransitiveClosure() in Graph.h
- Make sure you're looking at the pseudocode on pg 307
- Do this "place" – modify the original Boolean adjacency matrix
- You index a vector just like an array.
- The adjacency matrices are in row, column order so R[2] is the third row of R (where R is an (Boolean) adjacency matrix) and R[2][3] is the fourth item on the third row (0-indexed).
- Skim, at least, the Graph code and make sure you know how it works – look at the

targetgtest.cpp for more, too.

- Code it up and the first two test should pass.
- The tests are from the slides (look at the names)

Floyd's – All Pairs Shortest Paths

- Code up Floyd's in getAllPairsShortestPaths()
- Pseudocode is on pg. 310
- Run the tests and Floyd's should work as well (it has two tests as well, also from the slides).

Submission

- upload **Graph.h** to dropbox.cse.sc.edu
- that's it