CSE 6010
Workshop 3

Due Date: 11:59pm on Thursday, October 12
Submit 'diameter.c', 'contrib.txt', and 'howthingswent.txt" to Gradescope through Canvas
48-hour grace period applies to all deadlines

There are several purposes for the workshop assignments:

- To provide some low-stakes programming practice in C where you can get real-time help from the TAs (if you attend the class in person);
- To cover small programming-related subjects that don't have a logical place among the course topics;
- To give you the opportunity to work together with other students, which can expose you to new ways of thinking about and writing C code; and
- To provide the opportunity for extra credit (if you submit more than 4 workshop assignments).

For this assignment, you may work in groups of 2-3 students. If you prefer, you also may choose to work independently.

If you choose to work as a team, you are expected to **work together for the full assignment**. A divide-and-conquer approach, where each team member works independently on some part of the assignment and the team only interacts when they combine their separate codes to submit, is not in the spirit of this assignment. If you don't want to work with anyone, you should submit independently.

The workshop is designed so that we expect most of you will be able to finish most of the assignment during the **75-minute class period**, possibly with a little extra time offline for cleanup and submission. Nevertheless, we know that different individuals and groups will have different levels of comfort/proficiency with C, and that programming does not always follow a timeline. Thus, it is not strictly required that every submission necessarily will include all topics. As part of your submission, please write a few sentences about how things went in a text file named **'howthingswent.txt'**. This is your opportunity to let us know if you got bogged down in a particular area and how you went about resolving any problems you may have encountered.

Contributions

We ask you to identify how each team member participated by filling in and including the contrib.txt file provided. For each problem, please assess the contribution level for each category according to the following scale.

- 3 = >50% (student did the majority of the work for that category – so there can only be a maximum of one "3" per category + problem combination)
- 2 = 20-50% (solid contribution)
- 1 = 1-20% (minor contribution)
- 0 = no contribution

The categories are:

- Ideas / planning
- Detailed code design
- Writing code / implementation
- Debugging / testing
- Documentation / comments

You can find the template 'contrib.txt' file on Canvas. When submitting to Gradescope, make sure your filename is **'contrib.txt'**. **Submit this file even if you are a "group" with one member;** delete the space for contributions by other participants when not needed.

Your assignment: Complete a provided C code "scaffold" to calculate the diameter of a graph represented as an adjacency matrix using the all-pairs shortest paths algorithm discussed in class. The code includes function prototypes and some of the basic organization.

- The program specifies the graph via the function `getGraph()` defined in the header file 'graph.h'. This function includes the adjacency matrix representing the graph as an array (called `mygraph` by the main program); the number of vertices in the graph is given as `NVERT` through a `#define` in 'graph.h'. Note that the array is a one-dimensional array (as in Workshop 2).
- After setting up the array to represent the graph, you will need to update the representation so that all "non-edges" between distinct vertices have a value of (essentially) +Infinity. To do so, you should use the system value `INT_MAX`, which you can use through the inclusion of `#include<limits.h>`. **You will need to write this segment of the code, given by the function `PrepareGraph()`.**
- You will need to perform a number of iterations of "matrix multiplication." The number of iterations is the ceiling of the base-2 log of $N - 1$. You can calculate the ceiling and the log by including `math.h` and using the flag -lm when compiling. Note that `log()` calculates a base-e log, so to calculate the base-2 log of x, you should calculate `log(x)/log(2)`. In addition, note that `ceil(x)` returns a double rather than an int, but if you store the result in a variable declared as an int, it will be truncated and stored as an int and not a double. In the scaffold code provided, the number of iterations is set to 0 to ensure that the code will compile and run. **You will need to update the number of iterations in the code, given by the function `calcIterations()`.**
- Within each iteration, you should perform the "matrix multiplication" all-pairs shortest paths algorithm as described in the slides. **You will need to write this segment of the code, given by the function `CalcShortestPaths()`.** Keep in mind that to run the algorithm, you will need a second array in which to store the temporary results obtained in each iteration. It is good practice to initialize the entries to 0.
- After calculating the all-pairs shortest distances, you will need to calculate the diameter as the maximum of these distances. **You will need to write this segment of the code, given by the function `CalcDiameter()`.**

- Debugging aids: A `PrintGraph()` function is already defined and may be useful during debugging. The code also includes conditional compilation through the use of a `DEBUG` variable defined by `#define DEBUG`. Check out how the conditional compilation works within the code. When you no longer wish to debug, just comment out or remove the `#define DEBUG` statement. (Of course, you also may comment out individual lines that are not helpful for you or add additional print statements to use when `DEBUG` is defined.)
- At the end, your program should only print out the diameter as an integer; nothing other than that value should be printed. (See previous comment for a way to do this easily while making it easy to continue debugging if needed!)
- When submitting to Gradescope, make sure your code filename remains '**diameter.c**'. For this assignment, please keep all your code including function prototypes and definitions in this one file. **You should not submit the 'graph.h' file.**

For the graph included in the file, you should find that the diameter is **10**.

Individual and group submissions

Submit your code file named '**diameter.c**' along with '**contrib.txt**' and '**howthingswent.txt**' on **Gradescope** under **Workshop 2**. If you don't know how to submit on Gradescope as a group, you can follow this tutorial on YouTube.