CMP SC 3050 Homework 4
Due: 11:59:59 pm, 03/24/2016

The fourth homework for CMP SC 3050 is a ***programming assignment***. There are 40 points possible for this assignment and will contribute towards 4% of your total grade. The first part of the homework specifies the exact problems that your submission should solve and the second part describes the constraints that you **must** follow (no exceptions). Absolutely no late work will be accepted.

Specification:

The assignment should:

- Read in one filename as input in the command line. The file is going to represent an input to the program. Do not hard-code any filenames in your program – doing so will result in a zero – no exceptions.

- The input file is going to store a **directed weighted graph** as follows. The first line of the file will represent the total number of vertices, and the remaining lines will be vertex pairs representing edges. We will assume that vertices are numbered as 1,2, …. An edge from two vertex numbered as x to vertex numbered as y with weight w will be represented as (x,y,w) (with no whitespaces). In the input file, edges shall be separated by newlines. A sample input file can be found on Blackboard. We assume that weights are positive.

- For the rest of the assignment, we will call the weight of the shortest weighted path from vertex s to vertex t to be the distance of the vertex t from vertex s.

  The output of your program should be written to stdout. Write a procedure that outputs each vertex and its distance from vertex 1. Specifically, the output should be formatted with the vertex number, followed by a single whitespace, then its distance from vertex 1 and a newline. If the vertex is unreachable from 1, please write -1 for the distance. The vertices should be listed in increasing order, that is the first line should list vertex 1 and its distance from vertex 1, the second line should list vertex 2 and its distance from vertex 1 and so on. A sample output file can be found on Blackboard.

Constraints:

- The assignment must be completed in C or C++.
- Built-in data structures and external libraries must not be used for this assignment. If the program requires a stack, linked list, or any other structure, it is up to you to provide it. If you have any doubt on whether or not any technique you wish to use is acceptable, do not hesitate to ask.
- A moderate amount of error checking and resource management is required. Even if you do some error recovery, you must report errors in the input files. Your application should ensure that each line from the input file is properly formatted, that the file is successfully opened, the file is successfully closed upon reading of the file and that all allocated space is de-allocated at the exit.
- For reporting input format error, please use the enum type provided in input_error.h. Please do not alter the files input_error.h. You need to include input_error.h in your main program. The file input_error.h contains documentation of how to use input_error.h.
- A moderate amount of formatting and documentation is required. Comments should be descriptive and used to illustrate the purpose and inner workings of an algorithm or function; they should not be used to annotate each line or self-evident logic.
- The assignment must be submitted via Blackboard. Please compress the folder before submitting using zip or tar. Please do not submit a rar file.
- The input size is unknown, and the input range is 0 to UINT_MAX. This means that your program should not make any assumptions about the range or size of the input. Assuming an input size will result in a grade of zero – no exceptions.

Timing Constraints

A good algorithm for this homework should run in O(m log n) where n is the total number of vertices and m the total number of edges in the input graph. While we shall not be grading the exact time complexity of your solution, we shall be timing your solution and grading efficiency. As a guideline, on a graph with m+n =10^6, your program should run in under a minute of system time to receive full credit for efficiency.

Grading:

There are 40 points possible for this assignment. The grade breakdown is as follows:

- 3 points for error checking and resource management.
- 5 points for general programming style and adherence to the constraints.
- 27 points for correctly outputting the distances.
- 5 points for efficiency.
- **Failure to adhere to the EXACT output format will result in a grade of zero – no exceptions**

If the program fails to compile or crashes due to a runtime exception, a grade of zero will be assigned.

Additional comments:

- Sometimes Windows machines read end of line from a file created on other systems as a carriage return. So please check for both carriage returns (\r) and end-of-lines (\n) when parsing the input file.
- For Windows machines, please use MinGW as the compiler (and not CygWin).
- Please use gnu online manuals in case you are unsure as to what a C library function does.