Project Name: Console Spreadsheet Application
Author: Bryan Kline
Date: January 10, 2019

Development and Build Information:
The Console Spreadsheet Application is a program built and tested in a Linux environment, written in C++, and compiled with g++-7. The project contains a makefile and assumes a g++-7 compiler. The source code can be compiled separately for other compilers and environments. Instructions to do so are included in the README file.

The program requires at least one command line argument, the name of the input file containing the tab separated spreadsheet data to be processed, and may contain up to three additional arguments, the name of output file to generate, and two flags for debugging purposes which write to the terminal. If no output file name is provided, "output.txt" will be generated containing the results.

The program consists of a main driver, the Spreadsheet class header and implementation files, a makefile, the README file, and one test input file along with it as well as the test output.

Assumptions:
The assumptions made in the development of the program are that all equations are well-formed strings, i.e. always preceded by an equal sign, containing no intervening spaces, all column letters upper case, and no syntactically invalid use of operators and operands. Additionally, the spreadsheet is the size of the highest row by the highest column provided, any reference to cells outside of the bounds created by the input file are ignored. Due to time constraints, the bonuses were not implemented, neither unit testing nor end-to-end testing were employed, and the design of the program was not intended to be modular or extensible. If developing in a production environment, formal design patterns would have been used to produce a modular, well tested solution.

Program Structure:
The program opens the input file, determines the maximum size of the rows and columns, builds a two dimensional array of Cell structs (which have places for the value at the cell as well as status metadata), and the contents of the file are used to populate each cell. Next, a hash map is used to map the label of each cell to a pointer to that Cell struct for constant time look up. After the spreadsheet is built and filled with data, the program then iterates through it and if an equation is encountered, the equation is parsed and if references to other cells are contained in the equation the program recursively jumps from cell to cell until it resolves the value to an integer, keeping another map of cells visited in the recursive calls in order to detect circular references. Once the values for the references in an equation are resolved, the equation is first processed by removing the leading equal sign and differentiating between negative signs and minus signs if present, and then the equation is evaluated in the appropriate order based on operator precedence. Finally, once all cells have been processed, the results of all calculations in the spreadsheet are written to a tab separated file.