Description of the Code

The code features three parts: the Calculator, the Parser and the main class PostFixProgram.

The parser is the first part used, to read a CSV file and translate it into a grid of strings. The first possible limitation of my project is the size of the grid of strings. I amended the code so that it would first parse the file to obtain size of the grid to create an array[][] of the correct size, but the limitation is that this required the file to be parsed twice, which could affect performance. However, I felt in the long run, it may be better to parse the file twice, than to use the other method of copying an array to a larger array for every iteration after the initial size.

The main loop code is the traverseCells method which runs two for loops in the hope of translating all the cells. The final limitation here is the use of two nested four loops, which isn’t ideal aesthetically or performance wise, however at the time I couldn’t think of a better way to go through the values again to insure in examples like the B1 B2, are picked up and accounted for, as they wouldn’t be at times before the B1 and B2 cells aren’t populated, which would be the case when the code first goes through the cells.

The logic of the code is the calculateEquations method, which first ensures a string isn’t empty, null or just a blank space and then splits it into an array of smaller strings, before choosing between whether it’s Cell information, in which it will replace the cell with the cells value and place that in a stack of floats, whether it’s an operator, in which it will perform a calculation and place the value within the stack of doubles, or whether it’s a number, in which it will parse the double and place It in the stack of doubles. At the end there should be only one value, which will be formatted to make sure it doesn’t show a decimal unless necessary (this has also been amended in the case there is a decimal passed as an argument, to retain the number of decimals in the argument and return a number with the same number of decimals).

Finally, the writeFile method uses a path similar to the one presented as an argument, and uses a Stringbuilder to take the information from each row and store it, then a bufferwriter to save the information to a file.