Data Structures and Algorithms for Engineers

Programming Assignment 2: Sparse Matrix

Task Description: Using any programming language of your choice.

1. Load two sparse matrices from an input file.
2. Perform addition, subtraction, and multiplication on the matrices.

Note: Read the instructions carefully before you attempt to write the solution.

Instructions

1) Download the code and sample data for this assignment from this location.
   . Organize the code and the sample input into the following locations:
   /dsa/sparse_matrix/code/src/
   /dsa/sparse_matrix /sample_inputs/
   Feel free to organize your code or files the way you want.

2) Implement code to:
   a) Read a sparse matrix from a file. The format of the file will be:

      rows=8433
      cols=3180
      (0, 381, -694)
      (0, 128, -838)
      (0, 639, 857)
      (0, 165, -933)
      (0, 1350, -89)

      The first row gives the number of rows. The second row gives the number of columns. From the third row onwards, there is one entry in parenthesis with row, column, and the integer value separated by commas. All other values in the matrix will be zero by default. For example, in the given sample, the number of rows is 8433, the number of columns is 3180. Row 0 and column 381 has the value -694. Row 0 and column 128 has the value -838, and so on.

   b) Your goal is to implement a data structure that optimizes both memory and run time while storing such large matrices. You can have the following functions in your code:

      SparseMatrix(char* matrixFilePath)
      SparseMatrix(int numRows, int numCols)
      getElement(int currRow, int currCol)
      setElement(int currRow, int currCol, int value)

   c) You are free to define other data structures, classes, and helper functions in your code file. **You cannot use std::template packages or built-in Libraries of the programming language you have selected, you have to write your own functions or classes to solve the problem**. Youcan reuse code, or classes that YOU have written. You may also reuse codes that

have been given in class. However, you must document and cite the sources for such codes. Your code will be tested to perform operations like addition, subtraction, and multiplication. Be sure to implement each of these operations as a function in your code.

d) A few samples of input files and result files are given in zip file.
e) Your code must handle following variations in the input file:
    i) If the file has any whitespaces on some lines, these should be ignored.
    ii) If the file has the wrong format i.e., different kind of parenthesis, or floating point values, the code must throw an std::invalid_argument("Input file has wrong format") error and stop.

Note:

- **You are not allowed to use built in libraries for example: regex and other standard built in libraries. You should implement your own custom functions and classes to solve the problem.**
- **When your code is executed, the user should be allowed to select the matrix operation they want to perform and based on this, your code should read/load two sparse matrix the input files and then call the function to perform addition/subtraction/multiplication operation on the files.**
- **You might want to revisit the basic mathematical ruling for matrix multiplication, Addition and Subtraction. For example, to multiply two matrices the number of rows in the second matrix must be equal number of columns in the first matrix. You should consider these rulings your code respectively and use exception handling to handle situation that might lead to error in your code.**

Grading method:

1) If your code runs and generates an output file in the correct format for each input file, you get submission points.
2) If your method generates correct results for each test file, you get points for correctness.
3) We will review your source code to examine the internal documentation and award points for proper use of meaningful internal documentation.