**ECE 3822: Engineering Computation II**

# Homework No. 10: Sparse Matrices

**Goal:** The goal of this homework is to demonstrate how you can use a data structure to solve a large-scale math problem. It will also teach you something about how C++ classes are implemented and how to overload operators.

**Description:** There is one deliverable in this project ­– a C++ class implementation. You are to upload the following to your assignment directory: (1) a Makefile; (2) a main program called example.cc that does the things requested below; (3) a header file called Example.h that defines your class; and (4) a source code file example\_00.cc that contains the implementation of your class functions.

Your C++ class must do the following things:

1. Construct a sparse matrix of floats of dimension 1M x 1M using zero-based indexing that contains 3 non-zero elements: a(1,1), a(2,2), a(999,999) all are set equal to 27. Use your linked list class to implement this matrix. Only represent the non-zero elements; assume all unspecified elements are zero.
2. Implement a destructor the cleans up memory properly.
3. Implement a resize function that creates a matrix of a particular dimension with all zero elements.
4. Overload the parentheses operator so that you can write “a(1,1) = 27” for any index in the matrix. Bounds-checking must be performed.
5. Overload the “+” operator so that I can write “a = b + c” and have the matrices added to together.
6. Overload the “=” operator so that I can write “a = b”;
7. Create a debug method that dumps the contents of the matrix to stdout by printing all nonzero values (the indices of the value and its value).

Your main program should do the following:

MySparseMatrix a(1000000,1000000);

MySparseMatrix b, c;

a(1,1) = 27.0; a(2,2) = 27.0; a(3,3) = 27.0;

b = a;

c = a + b;

c.debug(stdout);

c.resize(10,10);

c(1,1) = -1.0;

c(10,10) = -1.0; (should throw an error)

c.debug(stdout)

Note that the resize method in this case needs to clean up memory from its previous state before it creates the new matrix.

Sparse matrices are a very useful data representation for doing large-scale linear algebra problems. In the header of your main program file, comment on the memory efficiency of your implementation. How much more memory do you require per element compared to a C vector or matrix representation?

Submit your solution in this directory: /data/courses/ece\_3822/current/homework/hw\_10/<lastname>.