Data Structures Project 1

Due date: Friday, Oct. 16, 2015

Implement a Sparse Matrix ADT (SMatrix) as outlined below.

Task:

1. Implement the SMatrix ADT. You may use the given sample main program to test your implementation. But you should do more tests to ensure all possible scenarios are covered.
2. Submit your ADT as a library (SMaxtrixADT.h, SMatrixADT.c) to course website.
3. Help on creating and link C-Library at http://www.cs.swarthmore.edu/~newhall/unixhelp/howto_C_libraries.html

Note

1. Please follow "good" programming practices. For example, use meaningful variable names, properly formatting your program, insert appropriate comments.
2. Test your program thoroughly before submitting for grading.
3. Discuss your solutions with others, but DO YOUR OWN WORK! Copying other's work or let others copy your work are consider academic dishonesty. It will result in 0 for both.

A sample main program

```
#include <stdio.h>
#include "SMatrixADT.h"

#define DONE 'X'            /* quit the program */
#define ADDITION '+'
#define SUBTRACTION '-'
#define TRANSPOSE 'T'
#define MULTIPLY '*'

int main (){
   MatrixType A, B, C, D;
   int  w1, w2;
   char op; /* operation to be performed */

   /* read and print matrices */
   MTX_read (&A);
   MTX_read (&B);
   MTX_read (&C);

   /* Perform matrix operations */
   while (getchar() != '\n');
   scanf ("%d", &op);
   while (op != DONE) {
      switch (op) {
         case ADDITION:
            printf ("Addition\n");
            if (MTX_add (&A, &B, &C)!= MAT_ERROR)
               MTX_print (&C);
            else
               printf ("Matrix ADDITION error!\n");
            break;
         case SUBTRACTION:
             . . .
         case TRANSPOSE:
             . . .
         case MULTIPLY:
```

```
                . . .
            default :
                  printf ("Invalid OPERATION error!\n");
            } /* switch */
        while (getchar() != '\n');
        scanf ("%d", &op);
    } /* while */

     /** you should do more testing than the above **/
}
```

ADT SMatrix is

Objects:

Sparse Matrix (with dimension *rows x cols*) of integers.

Functions:

`int SMTX_read (SMatrixType A)`

::= read in a matrix from *stdin* and stores the matrix in A.

::= returns `SMTX_ERROR` if something went wrong

/* For this project, the input format will be

Line 1: two integers, rows & cols, dimension of the matrix

Line 2 to rows+1: contains cols number of integers

Should take the input and convert to your proper ADT format */

`int SMTX_print(SMatrixType A)`

::= print the sparse matrix A in the following format

/* Line 1: print "Rows = ??, Cols = ??, # of non-zero entries = ??"

Line 2 ~ ??: print "< Ri, Ci, entry-value>," one 3-tuple per line */

`int SMTX_add (SMatrixType A, B, C)`

::= C <= A + B

::= returns `SMTX_ERROR` if something went wrong

`int SMTX_subtract (SMatrixType A, B, C)`

::= C <= A - B

::= returns `SMTX_ERROR` if something went wrong

`int SMTX_transpose (SMatrixType A, B)`

::= $B <= A^T$

::= returns `SMTX_ERROR` if something went wrong

`int SMTX_multiply (SMatrixType A, B, C)`

::= C <= A x B

::= returns `SMTX_ERROR` if something went wrong

/* please use the quick algorithm given in lecture */

Data typing:

```
#define SMTX_ERROR -1
#define MAX_SMTX_SIZE 100
typedef struct SMatrix {
   . . .
   } SMatrixType
```