```
/*****************************************************************************
Authors  : Albert Owusu-Asare , Box 4497, <owusuasa@grinnell.edu>
         : Ezra Edgerton      , Box 3503, <edgerton@grinnell.edu>

This document contains answers to questions from CS213 Lab: Threads


Questions:
http://www.cs.grinnell.edu/~weinman/courses/CSC213/2014F/labs/threads.html

 modified code has the MODIFIED keyword

 * Created by Jerod Weinman, 21 May 2008



*****************************************************************************/

#ifndef __MATRIXOP_H__
#define __MATRIXOP_H__

#include "matrix.h"

/* Matrix addition C = A + B
 *
 * Preconditions:
 *   Matrix parameters a,b and c all have the same dimension
 *
 * Postconditions:
 *   The resulting matrix sum is stored in parameter c
 *   Return value of 0 indicates successful completion of the addition
 */
int mtxAdd( const struct matrix_t *a, const struct matrix_t *b,
                  struct matrix_t *c );



/*
 * MODIFIED
 * Matrix multiplication C = AB
 *
 * Preconditions:
 *   Matrix parameters a,b,c have the required dimenetion.
    mtxCheckMultiplyDim(a,b,c) returns >0
 *
 * Postconditions:
 *   The resulting matrix sum is stored in parameter c
 *   Return value of 0 indicates successful completion of the multiplication
 */

int mtxMultiplyMax( const struct matrix_t *a, const struct matrix_t *b,
                 struct matrix_t *c);

int mtxMultiplyMin( const struct matrix_t *a, const struct matrix_t *b,
                 struct matrix_t *c);


/*
 *MODIFIED
 * struct code taken from Jerod Weinman's CSC213 Threads lab:
 * http://www.cs.grinnell.edu/~weinman/courses/CSC213/2014F/labs/threads.html

 * matrixThreadParam_t contains the parameters needed for a threaded matrix
 * product.
 */
struct matrixThreadParam_t {
  const struct matrix_t *a;
  const struct matrix_t *b;
  struct matrix_t *c;
  int numThreads;
  int threadId;
};

/*
 * MODIFIED
 * Computes the threaded concurrent matrix product.
 * preconditions:  none
 * postconditions : matrix c is modified such that each of its values agrees with m
atrix
 * multiplication of c = a * b.
 *
 * much of the code involving the creation and running of threads was taken from th
e POSIX
 * Threads Programming tutorial:
 * https://chttps://computing.llnl.gov/tutorials/pthreads/#Joining
 * computing.llnl.gov/tutorials/pthreads/#PassingArguments
 */

int parMtxMultiply( const struct matrix_t *a,
                    const struct matrix_t *b,
                    struct matrix_t *c,
                    int numThreads);

/*
 * MODIFIED
 * Does the matrix multiplication work for one thread
 * preconditions:  none
 * postconditions : matrix c is modified such that each of its values agrees with m
atrix
 * multiplication of c = a * b.
 *
 * much of the code involving the creation and running of threads was taken from th
e POSIX
 * Threads Programming tutorial:
 * https://chttps://computing.llnl.gov/tutorials/pthreads/#Joining
 * computing.llnl.gov/tutorials/pthreads/#PassingArguments
 */

  void* threadMtxMultiplyMin(void *multParam);

#endif
```