CSE 3100 Systems Programming	Fall 2025
Lab #7	

Welcome to Lab 7! Read each question carefully before beginning work; Submit required files on gradescope. A Makefile is provided.
In this assignment, we will work on a project that has multiple source files. Study the Makefile and learn how to write rules for such targets.

Only `madd.c` needs to be changed. Do not change other files.

Part 1. Matrix addition (100 Points).

An implementation of matrix ADT (abstract data type) is given in `matrix.c`. The API (application programming interface) functions that operates on the matrices are listed in `matrix.h`. One of the functions, `addMatrix()`, performs matrix addition, which is implemented in `matrix.c`.

In this lab, we implement `addMatrix_thread()` in `madd.c`. The function performs matrix addition with two threads. It has the same interface as `addMatrix()`, but performs matrix addition with two threads. We only need to change `madd.c`.

`test-madd.c` is provided to test our implementation. The program takes the number of rows and columns from the command line, fills two matrices with random numbers, and compares the result of `addMatrix()` and `addMatrix_thread()`. If no argument is specified, the program works on matrices of size 6×6. In addition, if a command line option `-t<n>` is present, `test-madd` prints the time (in seconds) spent on matrix additions, using the average of `<n>` calls.

Here are some sample sessions of running `test-madd`. In the second session, the program call the functions 20 times to collect timing information. `time1` is the average time on `addMatrix()` and `time2` is the average time on `addMatrix_thread()`. The numbers are likely to change in different runs.

```
$./test-madd
Good work!
$./test-madd 1000 2000 -t20
Good work!
num_runs=20 time1=0.0107 time2=0.0058 speedup=1.8545
```

Debugging. gdb supports multithreading. Run your code in gdb until it stops at a breakpoint or appears to stop making progress. If threads are not making progress, interrupt the execution with `Ctrl-C` to get to the gdb prompt. Here are some commonly used thread commands.

- `info threads` See what threads are running.
- `thread n` Switch to thread `n`, where `n` is a thread number.
- `thread apply [threadno] [all] args` Apply commands to one or more threads.