Lab #6 Pthreads Programming I

Purpose: to learn how to write parallel programs using Pthreads.

1. (5 points) Compile and run your first Pthreads program.
   a. Create a file (e.g. lab6a.c), and enter the following code.

```
/*pthreadhello.c program*/

#include <pthread.h>
#include <stdlib.h>
#include <stdio.h>

#define NUM_THREADS     5

void *PrintHello(void *threadid)
{
     printf("\nHello World! from thread %d\n", threadid);
     pthread_exit(NULL);
}

int main(int argc, char *argv[])
{
          pthread_t threads[NUM_THREADS];
          int rc, t;

          for(t=0; t<NUM_THREADS; t++){

                    rc = pthread_create(&threads[t], NULL, PrintHello, (void *)t);
                    if (rc){
                              printf("ERROR: return error from pthread_create() is %d\n", rc);
                              exit(-1);
                    }
          }

          for(t=0; t<NUM_THREADS; t++){
                    rc = pthread_join(threads[t], NULL);
                    if (rc){
                              printf("ERROR: return error from pthread_join() is %d\n", rc);
                              exit(-1);
                    }
  }

  return 0;
}
```

   b. Compile your code using ***gcc***. E.g. gcc –o lab6a lab6a.c -lpthread.
   c. Run your program.

2. (25 points) Write a Pthreads program to perform matrix-matrix multiplication (A x B = C). A, B and C are *nxn* matrices.

   a. All threads share matrix A, B and C. The main program initializes the matrixes.

   b. The matrix size *n* and the number of threads *nthreads* should be passed in from command line, such as

      Lab6b n nthreads

   c. Each thread computes *n/nthreads* number of rows of matrix C. If *n* cannot be perfectly divided by *nthreads*, one of thread should be able to take care of the extra rows.

   d. If *n<= 40*, print out matrix C.

   e. Set timers before creating the threads and after terminating all the threads. Run your program using different number of threads and fill out the following table.

	1 thread	2 threads	4 threads	8 threads
n=256				
n=512				

   f. Analyze and explain the data in the table.