```
#include <pthread.h>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>

pthread_mutex_t mutex = PTHREAD_MUTEX_INITIALIZER;

struct sumInfo {
    double *a;
    double sum;
    int N;
    int size;
    long tid;
};

/**
 * @brief Computes partial sum of N according to its tid
 *
 * @param arg
 * @return struct sumInfo
 */
void *compute(void *arg)
{
    struct sumInfo *struct_sum = (struct sumInfo *)arg;
    int myStart, myEnd, myN, i;

    // determine start and end of computation for the current thread
    myN = struct_sum->N / struct_sum->size;
    myStart = struct_sum->tid * myN;
    myEnd = myStart + myN;
    if (struct_sum->tid == (struct_sum->size - 1)) myEnd = struct_sum->N;

    // compute partial sum
    double mysum = 0.0;
    for (i = myStart; i < myEnd; i++)
        mysum += struct_sum->a[i];

    struct_sum->sum = mysum;
    return struct_sum;
}

int main(int argc, char **argv)
{
    struct sumInfo struct_sum;
    struct_sum.a = NULL;
    struct_sum.sum = 0.0;
    long i;
    pthread_t *tid;

    if (argc != 3) {
        printf("Usage: %s <# of elements> <# of threads>\n", argv[0]);
        exit(-1);
    }

    struct_sum.N = atoi(argv[1]);     // no. of elements
    struct_sum.size = atoi(argv[2]);  // no. of threads

    // allocate vector and initialize
    tid = (pthread_t *)malloc(sizeof(pthread_t) * struct_sum.size);
    struct_sum.a = (double *)malloc(sizeof(double) * struct_sum.N);
    for (i = 0; i < struct_sum.N; i++)
        struct_sum.a[i] = (double)(i + 1);

    // create threads
    for (i = 0; i < struct_sum.size; i++) {
        struct sumInfo *new_struct_sum = malloc (sizeof *new_struct_sum);
        *new_struct_sum = struct_sum;
        new_struct_sum->tid = i;

        pthread_create(&tid[i], NULL, compute, new_struct_sum);
    }

    // wait for them to complete
    for (i = 0; i < struct_sum.size; i++) {
        void *thread_return;
        pthread_join(tid[i], &thread_return);
        struct_sum.sum += ((struct sumInfo *)thread_return)->sum;
    }

    printf("The total is %g, it should be equal to %g\n",
           struct_sum.sum, ((double)struct_sum.N * (struct_sum.N + 1)) / 2);

    return 0;
}
```