# CS 133 Parallel and Distributed Computing
# Winter 2019
# Jason Cong

## Homework #2

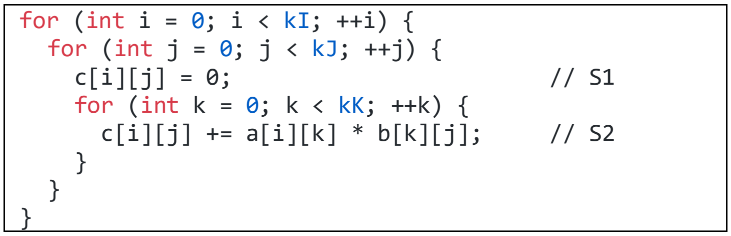
**Due 4pm Monday Feb. 4, 2019**

### Reading assignment:

Lecture notes 4, 5, 6, and 7.

### Homework problems:

1. For the provided matrix multiplication code, (i) please list all data dependencies and their types; and (ii) please exam all loop transformation operations discussed in Lecture 4, and discuss which one can be applied and which one cannot be. For some transformation, it can be applied to some loops, but not others. Please discuss both cases.



```
for (int i = 0; i < kI; ++i) {
  for (int j = 0; j < kJ; ++j) {
    c[i][j] = 0;                         // S1
    for (int k = 0; k < kK; ++k) {
      c[i][j] += a[i][k] * b[k][j];      // S2
    }
  }
}
```

2. Given an integer array *a[ ]* of *N* elements of value between *1* to *m* as the input, please write an efficient OpenMP function to generate the histogram *h* for array *a[ ]* such that *h[i]* is the number of elements in *a* with value *i (1 <= i <= m).* The function header is: `void histogram(int *a, int *h).` In addition, you can use constant variables *N* and *m* in your function directly. Is there a possibility for race condition in your implementation? If so, how do you handle it?

3. Please write an OpenMP program to compute the numerical value of the integration of the function sqrt(x)/(1+$x^3$) between 0 and 1 using 16 threads. Your goal is to make it as efficient as possible. Please present two ways to deal with possible race conditions and compare their efficiency.

4. There is a list of *n* independent tasks with known (but considerably different) runtimes to be performed by *m* processors. We use a simple list scheduling approach to assign each task in the order of the list to the first available idle processor until all tasks are completed. Once a processor finishes a task, it is coming back to ask for more tasks. Alice sorts the list in decreasing order of the task runtimes and then performs list scheduling. Bob sorts the list in increasing order of the task runtimes and then performs list scheduling. Who do you expect to finish first? Please explain why.

5. Given the following OpenMP program (segment) running on four CPU cores using four threads, assuming that the computation of function *f(i, j)* takes one minute on a single CPU core, and we ignore that scheduling overhead. You are asked to experiment with different scheduling methods. Please estimate the completion time under each of the following schedule schemes:

   a. default scheduling
   b. schedule (dynamic, 2)
   c. schedule (guided, 1)

```
#pragma omp parallel for
for (int i = 0; i < 12; i++)
        for (int j = 0; j <= i; j++)
                a[i][j] = f(i, j);
```

### Late submission policy:

We allow one-day delay with 10% penalty. After that, no submission will be accepted and the solutions may be discussed in the discussion sessions.