Homework 3
Due Friday October 20
Parallel Counting Sort

- Implement a parallel version of Counting Sort using OpenMP
- Test your program by generating a random sequence of ints in the range 0 through N-1 where N is provided as a command line argument
- The program expects two or three command line arguments. The first argument is the size of array to be sorted. The second argument is the value of N (see above). If a third argument is provided it is the name of a text file to which the sorted results should be written. If a third argument is not provided the sorted results should be written to standard output
- Sort the results in **descending** order
- See the class discussion of Counting Sort
Counting Sort

• Counting Sort is a specialized sort that can be used when the sort keys are of limited range (for example one or two byte ints)

• The sort uses an auxiliary array to store the number of times each key appears in the original array.

• For example if there are 256 possible keys then the auxiliary array has 256 slots
Counting Sort Example

Original Array 7,3,1,2,2,6,5,6,0,1,0,6,3,2,5,7,7

Counting Array 2,2,3,2,0,2,3,3

Sorted Array 0,0,1,1,2,2,2,3,3,5,5,6,6,6,7,7,7
Prefix Sum Pattern

size = 8;
    rounds = 3;
    for ( j = 1, k = 2; j <= rounds; j++, k = k*2) {
        n = size / k;
        #pragma omp parallel default(shared) private(i) num_threads(n)
        {
            i = omp_get_thread_num();
            i = i*k;
            if (x[i] < x[i+k/2])
            x[i] = x[i+k/2];
        }
    }
Prefix Sums of the Counting Array

Counting Array 2,2,3,2,0,2,3,3

Prefix Sums 0,2,4,7,9,9,11,14
Counting Sort

• Possible sources of parallelism
  – Partition the original array to generate count arrays
  – Combine the results of the count arrays
  – Find the prefix sums in the count array
  – Assign values to the original array based in the prefix sums
Homework 3 Submission

• Send only one C file called countingsort.c to tgendreau@uw.lax.edu
• Put the CountingSort code in a function
• The main function should process the command line arguments and call the CountingSort function. The OpenMP code should be in the CountingSort function and any other functions as needed