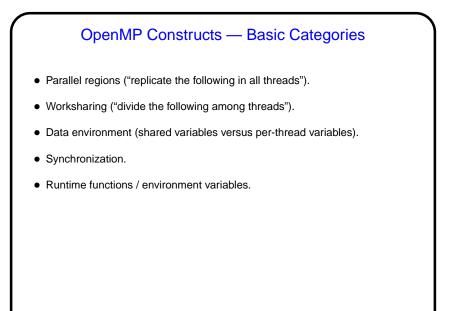


OpenMP — Overview (Review)
Parallel programming environment for shared-memory programming, possibly emerging as de facto standard.
Set of extensions to selected sequential programming languages — compiler directives, library functions.

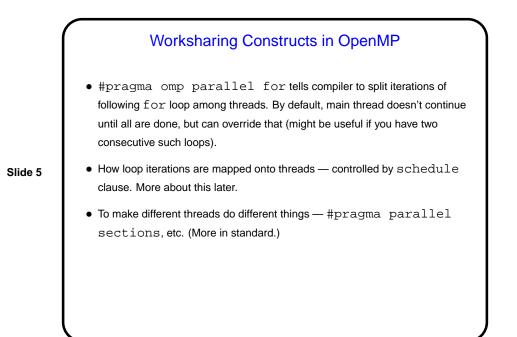
Slide 1

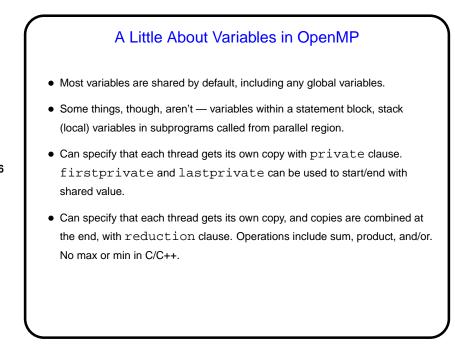
Slide 3

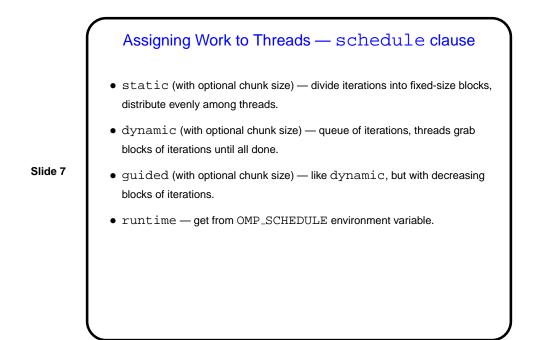


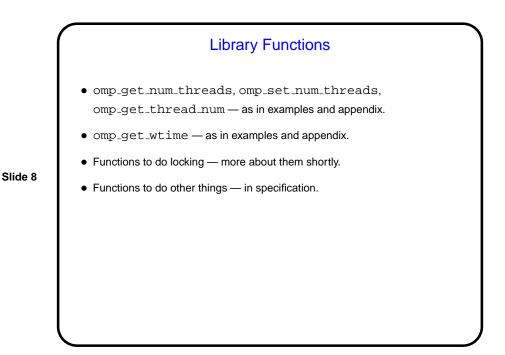
## Parallel Regions in OpenMP

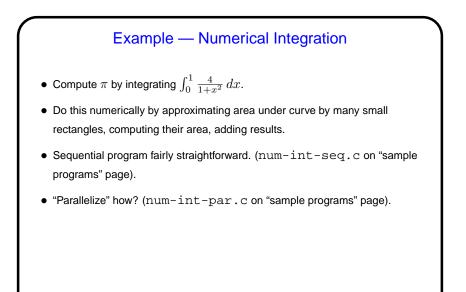
- #pragma omp parallel tells compiler to do following block in all threads (starting team of threads if necessary). Execution doesn't proceed in main thread until all are done. Example — "hello world" shown earlier.
- Block must be a "structured block" block with one point of entry (at top) and one point of exit (at bottom). In C/C++, this is a statement or statements enclosed in brackets (with no gotos into / out of block).



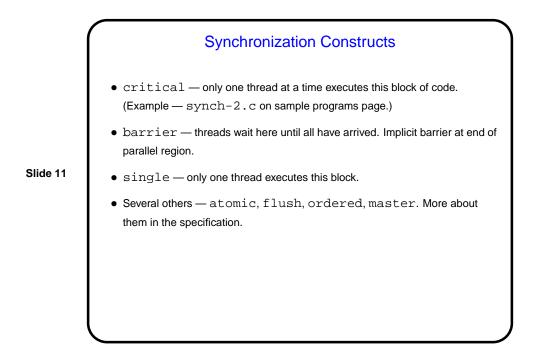


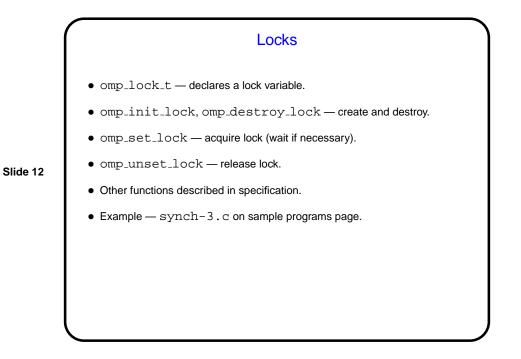


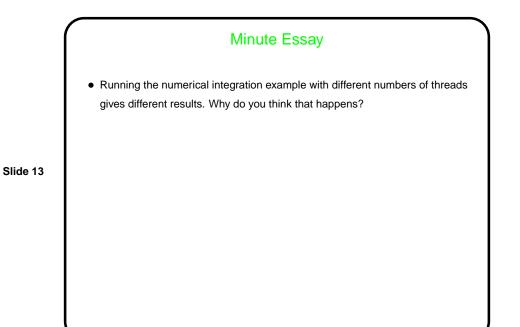




## Homework 1 Background In Homework 1, you will make a first pass at writing a set of programs (one using OpenMP, one using MPI, and one using Java) to solve the following problem. (We'll talk more about it in class after you've tried it.) We talked about computing π using numerical integration. Another interesting (surprising?) approach uses a "Monte Carlo" method: Consider a square with sides of length 2 (any unit you like), enclosing a circle of radius 1. Approximate the area of the circle by "throwing darts" at the square, counting how many fall within the circle, and calculating the ratio of those within the circle to the total number. Model "throwing darts" by using pseudorandom number generator to generate coordinates of a point.







## Minute Essay Answer • The order in which the partial results (produced by the iterations of the loop to compute areas of rectangles) are added together depends on the number of threads and the scheduling — and floating-point arithmetic is not associative (!).