Slide 1

Administrivia

- Homework 3 on Web. Due next Thursday. Homework 4 and project information to be on Web soon.

Slide 2

Some Unsolicited Advice

- Fighting with one's tools is no fun. If you don't like whatever text editor you're using for C, either spend some time trying to learn more about it (e.g., `vimtutor`) or choose another one! You may like `gedit`. A student recommends `KDevelop`.
- If you don't remember syntax for separating C's `stdout` output stream from its `stderr` output stream, this might be a good time to review. Examples of redirecting output:

  `foo 1>foo.out` to put output to `stdout` in `foo.out`. (`stderr` still goes to the terminal.)

  `foo 2>foo.err` to put output to `stderr` in `foo.out`. (`stdout` still goes to the terminal.)

  `foo 1>foo.out 2>foo.err` to redirect both output streams.

Slide 3

Example Application — Heat-Diffusion Problem, Contiued

- (Finish reviewing code.)

Slide 4

Homework 3 Background

- (Look at homework writeup briefly.)

Slide 5

Minute Essay

- The three most-common *Algorithm Structure* patterns are
  - *Task Parallelism* (e.g., our numerical-integration example, or the Monte Carlo pi homework)
  - *Geometric Decomposition* (e.g., our heat-diffusion equation example)
  - *Divide And Conquer* (e.g., mergesort)

  Which of these seems like the best fit for a program to “play” Conway’s game of life?

Slide 6

Minute Essay Answer

- It should be reasonably apparent that *Divide And Conquer* is not a good fit. The line between the other two is admittedly blurry, but in my thinking the better choice is *Geometric Decomposition*, because it explicitly addresses what to do when updates to one part of a large data structure depend on other parts of the same or a related data structure.