Slide 1

Administrivia

- Homework 6 writeup on the Web (finally!) Design due Tuesday, code Thursday.
- Sample code for sorted list, binary search tree, using recursion, on "sample programs" page.

Slide 2

Homework 6 — GUI Features

- Overall layout of game is `BorderLayout`, with screen in middle and "game status panels" on four sides — returned by `getGameStatusPanel` (in player), usually a `JPanel`.
- Menu bar is in `GameSetup`, can be modified.
- Screen editor program has support for "editing properties" (of screens, blocks, entities) — `getEditPropertiesPanel`. Could use this to give slightly different properties to different instances (e.g., walls of different colors, enemies with different speeds).
- Homework 6 asks you to use these features to (1) display something, and (2) get input from the user (either in the game or in the screen editor).

Slide 3

Priority Queues, Revisited

- Several data structures we could use to implement priority queue ADT:
  - Unsorted linked list.
  - Sorted linked list.
  - Sorted binary tree.

  Compare how much work to add/remove if N elements. Can we do better? Maybe!

Slide 4

Heaps

- Heap is another tree-based data structure, with two properties:
  - A node is always "bigger than" both its children.
  - Tree is "complete".
- For a priority queue, we want to retrieve the "biggest" thing (for game problem, smallest update time). Does this seem useful?
- Note also that we can store a complete binary tree in an array.
- How to insert and remove? Compare running times.

Slide 5

Minute Essay

- None — quiz.