Completing Path Finding

11/9/2009

Opening Discussion

- What did we do last class? What was our objective for adding to the code? What new technique did we learn to help solve this?

Where We Stand

- We currently have a recursive method that will calculate the minimum number of steps we have to take to get somewhere.
- This code works by calling itself in each of the four directions. The distance from where we are will be one greater than the shortest of those four distances.
- Let's look at the code.

Cycles

- Right now our code has one major flaw. It doesn't remember where we have been so it will go back over itself repeatedly, leading to infinite recursion.
- To prevent this, we have to leave “bread crumbs” so that we don't go back over where we have been.
- We will do this with a 2-D array that keeps track of where we have been.

All Paths

- The simplest implementation of this only says if we have been to a location. We sweep up those breadcrumbs as we come back through rooms.
- This method tests all paths through the world and tells us the length of the shortest one.
- This is fine for fairly complete mazes, but the number of paths through open spaces is huge so we need to be smarter.

Optimization

- The easiest optimization for this is to store integers for each room telling how many steps we took to get there.
- If we ever reach a room in a larger number of steps, we go back because we know this path won't be the solution.
- This is sufficiently fast for most mazes.

Minute Essay

- Do you have any questions about recursion?
- I will send an e-mail as soon as I have posted revised IcPs.