Opening Discussion

- PQs in projects: Actions that take time.
- Minute essays:
  - Nodes use more memory. Might be slower.
Let's write a priority queue that uses a doubly-linked list with a sentinel.

We'll also make a trait because we will implement other versions later.
Recursion

- You should have learned about recursive functions in 1320. A recursive function is simply a function that calls itself.

- You can use recursion to imitate loops, but we won't do that very often in C/Java/Scala. Where recursion comes in really handy is when a function needs to test more than one alternative at a time.

- This works nicely because the call stack remembers where you are in a given function so when you return back, you can take off from that point again.
One of my favorite recursive algorithms is maze solving. This is a special case of graph traversals which are common problems in CS.

We'll use a 2D array of Ints as our maze and we can even put this into our drawing program.

I want to write code to find the shortest path through a maze or count all paths through a maze.

We can try to make this nice and graphical as well so it fits properly into our drawing program.
Another one of my favorite recursive algorithms is formula parsing. This allows us to have the user type in a function and our code can evaluate it.

We do this through “divide and conquer”. We split the formula in two across the lowest precedence operator then recursively evaluate the two halves.

We can use this to put function plotting into our program if we give it the ability to handle a variable.
Can you think of a fun problem for recursion?

The midterm is Wednesday. Review session will be 6:30 on Tuesday.