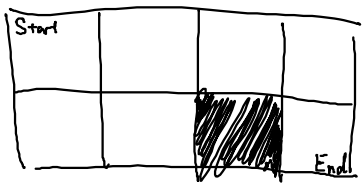
Solving Mazes

- A maze is a grid
- Start in upper-left
- End in lower-right



Function search(frontier)

Add (start, null) to frontier

While there are options in frontier collection:

Remove (current, previous) from frontier

Mark current as visited

Set current's prev to previous

If current is exit?

Build a list containing path by starting at end and retracing our steps

End If

For each neighbor of current:

If neighbor is not visited:

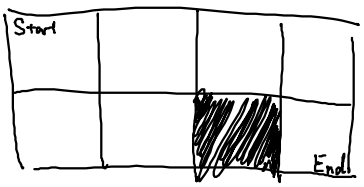
Insert (neighbor, current) to frontier

End If

End For

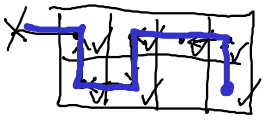
End While

▷, ▽, ◁, △

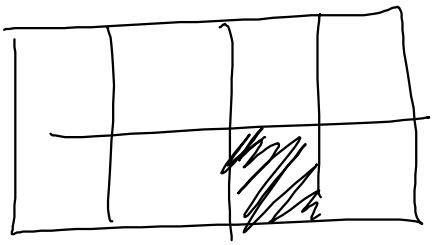


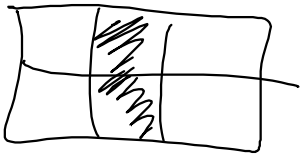
~~3,1 from 3,0~~
~~3,0 from 2,0~~
~~2,0 from 1,0~~
~~1,0 from 1,1~~
~~1,1 from 0,1~~
~~0,1 from 0,0~~
1,0 from 0,0
~~0,0 from nowhere~~

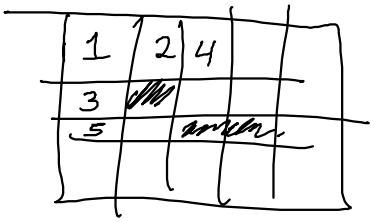
Frontier

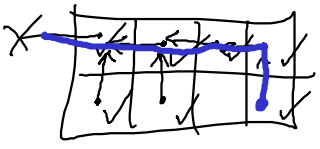


Visited





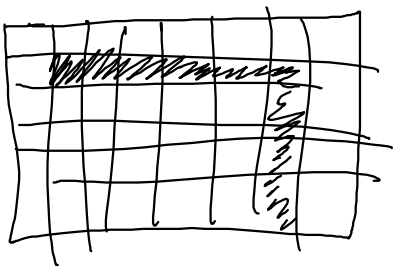


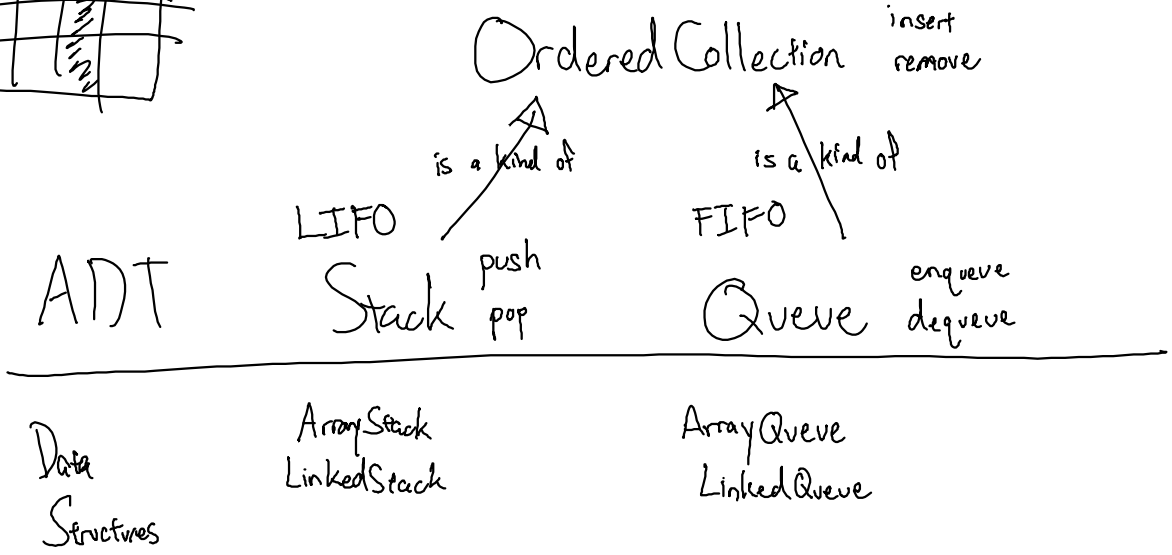


visited

~~0,0 from nowhere~~
~~1,0 from 0,0~~
~~0,1 from 0,0~~
~~2,0 from 1,0~~
~~1,1 from 1,0~~
~~1,1 from 0,1~~
~~3,0 from 2,0~~
~~3,1 from 3,0~~

frontier





Search:

Depth-First Search (DFS) uses a stack

- Tends to use less memory in any given moment

Breadth-First Search (BFS) uses a queue

- Always find shortest path