CS2302 - Data Structures
Spring 2019
Lab # 7
Graphs

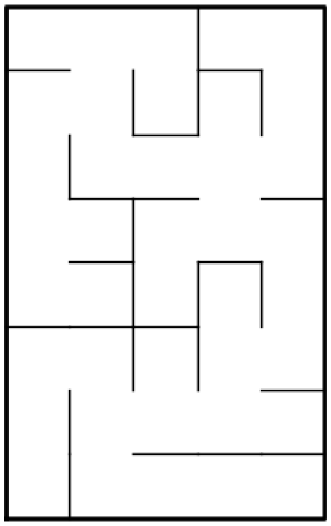
Deadline: Monday, April 29, 2019

For lab 6 you wrote a program that created a maze where each cell was reachable from any other cell and there was a unique path from the start to the destination. Your program works by removing one wall at a time (making sure that the cells separated by that wall were not reachable from each other) until the disjoint set forest representing the maze has exactly one tree. If the maze has n cells, your program would remove exactly $n-1$ walls to reach this situation.

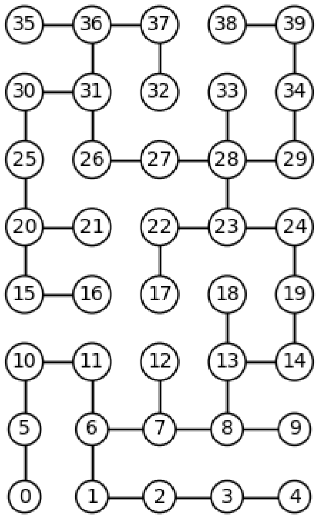
If you remove less than $n-1$ walls, some cells will not be reachable from the start cell. If you remove more than $n-1$ walls (notice that after removing $n-1$ walls, all remaining walls separate cells that are reachable from each other, and thus belong to the same tree in the disjoint set forest), you would have multiple paths from the source to the destination.

Your task for this lab consists of the following:

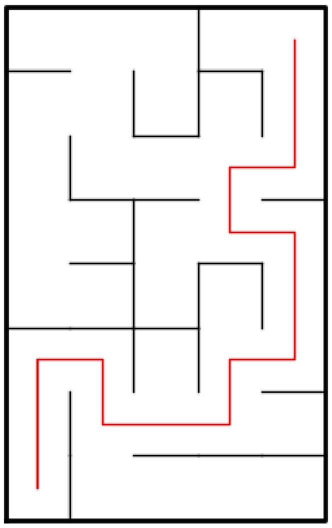
1. Modify your maze-building program to allow for both cases mentioned above. Your program should display n, the number of cells, and ask the user for m, the number of walls to remove, then display a message indicating one of the following:
   (a) A path from source to destination is not guaranteed to exist (when $m < n-1$)
   (b) The is a unique path from source to destination (when $m = n-1$)
   (c) There is at least one path from source to destination (when $m > n-1$)
2. Write a method to build the adjacency list representation of your maze. Cells in the maze should be represented by vertices in the graph. If two cells u and v are contiguous and there is no wall separating them, then there must be an edge from u to v in the graph. The example below shows a maze and the corresponding graph representation.
3. Implement the following algorithms to solve the maze you created, assuming the starting position is bottom-left corner and the goal position is the top-right corner.
   (a) Breadth-first search.
   (b) Depth-first search using a stack. This is identical to breadth-first search but the queue is replaced by a stack.
   (c) Depth-first search using recursion.
4. As usual, write a report describing your results. Display the paths found by your algorithms and compare their running times for different maze sizes.



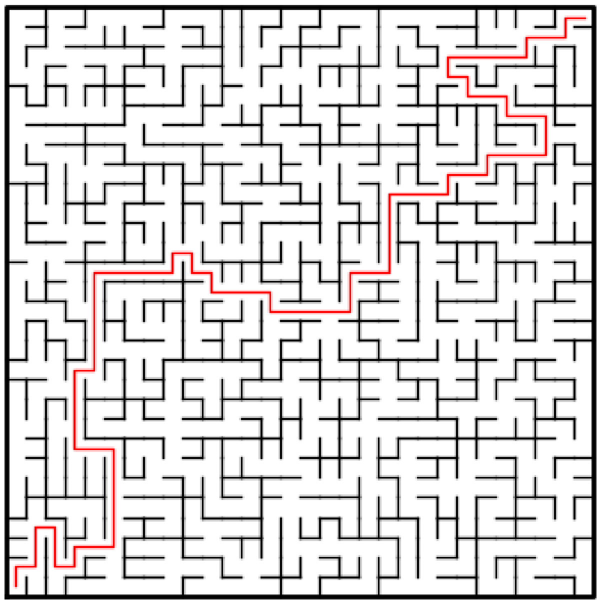
Maze



Corresponding graph representation



Path from start to goal



Solution of a larger maze