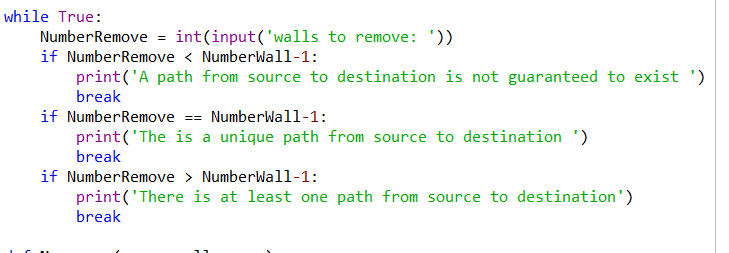
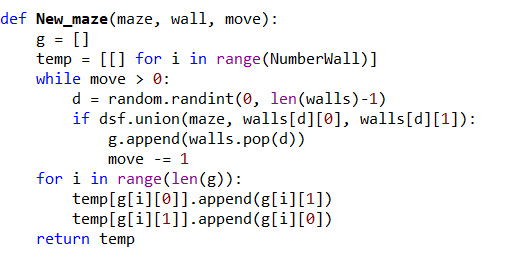
1. Modify your maze-building program to allow for both cases mentioned above. Your program should display n, the number of cells, and ask the user for m, the number of walls to remove, then display a message indicating one of the following: (a) A path from source to destination is not guaranteed to exist (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)

For this part I just to ask the number of walls want to remove, and use three if statements to check all the situation.



Running time is 

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.



For this part, to move the walls is like last lab, just have to change the random number to the number we want.

For the adjacency list, we need to create two list, one is empty, and one is with total number of wall of[]. Then put the data of walls in to this list, the opposite way.

Running time is 