18.10 Recursive Backtracking

Our recursive methods all have a similar architecture—if the base case is reached, return a result; if not, make one or more recursive calls. This section explores a more complex recursive technique that finds a path through a maze, returning true if there's a possible solution. The solution involves moving through the maze one step at a time, where moves can be made by going down, right, up or left (diagonal moves are not permitted). From the current location in the maze (starting with the entry point), the following steps are taken: For each possible direction, the move is made in that direction and a recursive call is made to solve the remainder of the maze from the new location. When a dead end is reached (i.e., we cannot take any more steps forward without hitting a wall), we *back up* to the previous location and try to go in a different direction. If no other direction can be taken, we back up again. This process continues until we find a point in the maze where a move *can* be made in another direction. Once such a location is found, we move in the new direction and continue with another recursive call to solve the rest of the maze.

To back up to the previous location in the maze, our recursive method simply returns false, moving up the method-call chain to the previous recursive call (which references the previous location in the maze). Using recursion to return to an earlier decision point is known as **recursive backtracking**. If one set of recursive calls does not result in a solution to the problem, the program *backs up* to the previous decision point and makes a different decision, often resulting in another set of recursive calls. In this example, the previous decision point is the previous location in the maze, and the decision to be made is the direction that the next move should take. One direction has led to a *dead end*, so the search continues with a *different* direction. The recursive backtracking solution to the maze problem uses recursion to return only *partway* up the method-call chain, then try a different direction. If the backtracking reaches the entry location of the maze and the paths in all directions have been attempted, the maze does not have a solution.

The exercises ask you to implement recursive backtracking solutions to the maze problem (Exercises 18.20–18.22) and the Eight Queens problem (Exercise 18.15), which attempts to find a way to place eight queens on an empty chessboard so that no queen is "attacking" any other (i.e., no two queens are in the same row, in the same column or along the same diagonal).