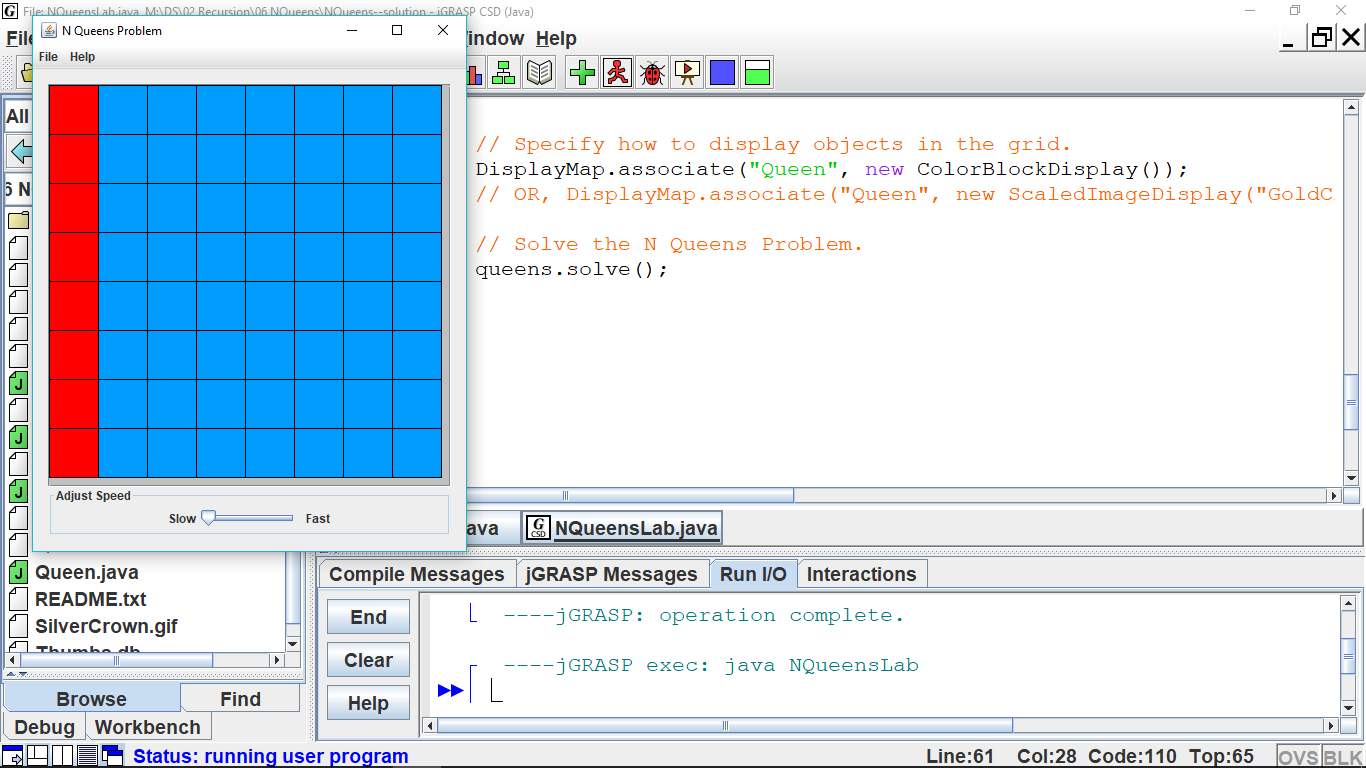
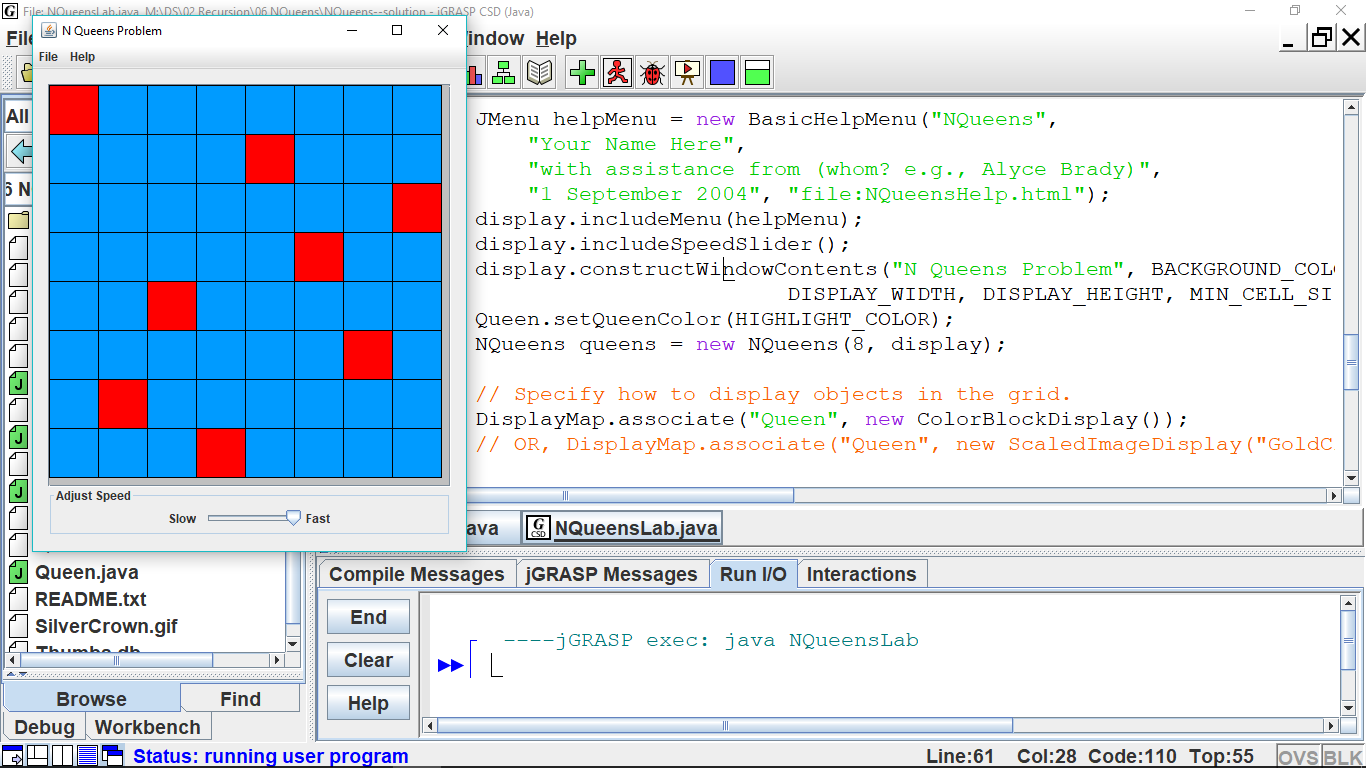
Extra Help on NQueens



1. In the solve method, type in the code that is given to you on the handout (in step #4).
2. You must be able to write the recursive code (in isPlaced) that puts one queen on a row, going vertically down a column, and then removes each queen (in reverse) as the recursion unwinds. (This is step #4 on the handout.) You must be able to explain, to yourself and others, how and why recursion works this way.



1. Now it’s time to think about actually solving the 8-Queens problem (in isPlaced). We know the base case will be when all 8 queens have been successfully placed (the base case has been written for you). Let’s just place queen-0 at the top left. We know queen-1 will be on the next row down, but we must (let’s do it *iteratively*) try each location on that row. If that location is not OK, then we try the next location on that same row. If the location is OK, we add that queen and recur to try to place queen-2 somewhere on its row. If we can’t place queen-2, we remove queen-1 and try the next location on queen-1’s row. After exhaustively trying and recurring, we hope that our code will have successfully placed each queen on her row and column.
2. Since checking whether the location is OK has to do a lot work, the programmer decided to make a separate method locationIsOK. Fortunately, because we decided that we are placing each queen starting at the top and going down the grid, we know we only have to check straight up, diagonally up to the left, and diagonally up to right. We can do each of those iteratively, returning either false or true as may be.