## BIT 115, ASSIGNMENT: MAZE NAVIGATION

Navigating A Robot Through A Maze

Download the file **Maze.java** from website. You should add code to the NavigateMaze command to make the robot go from the starting point to the end point. The only things which you may assume about the maze are the following: with the exception of the entry point and exit point, the outer wall of the maze will be unbroken (so you don't have to worry about the robot wandering out of the maze by going through the 'wrong door out'). You may also assume that the maze will be configured so that the MazeBot will start in between 2 walls, and will be pointing into the maze. I.e., there will be a wall on it's left, and a wall on it's right. You're also guaranteed that there actually is a way to get through the maze to the exit point. You're not allowed to leave the maze (by turning 180 degrees around, and leaving via the entry point), walk around the maze to the exit point, and pronounce yourself done. You must write a program so that the robot will guide itself through the maze; you cannot depend on the user (say) to 'steer' the robot for you.

Thus, the Maze.java file gives a single maze that your robot should be able to navigate through, but this isn't the only maze – any maze that satisfies the limitations set forth in the prior paragraph should be something that your robot can handle. As such, you can't just give it a set of directions like "move forwards 9 times, turn right once, move twice, etc" A picture of *one possible maze* is given here:
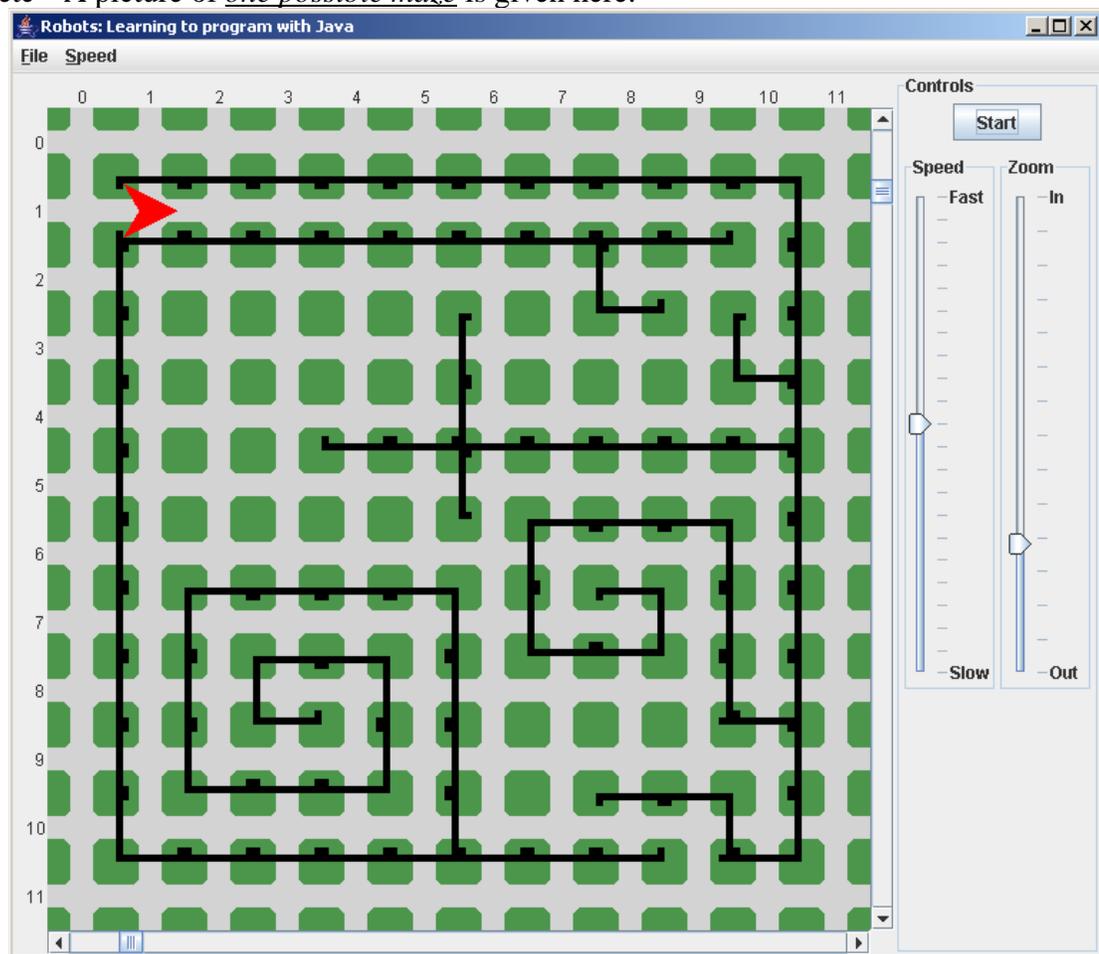
Additionally, each MazeBot should also start with exactly 1,000 Things in it's backpack. Each time the MazeBot moves forwards, it should leave a Thing in the current intersection, but <u>only if there isn't a Thing there already</u>. Obviously, the robot shouldn't break if it runs out of Things (regardless of whether there's something in the intersection or not)

The MazeBot also needs to be able to return the total number of spaces it has moved, via the command (method) named `printTotalNumberOfSpacesMoved`. This will print the total number of spaces that the MazeBot has moved since the NavigateMaze method was called. In other words, if someone calls NavigateMaze on the MazeBot, and it takes 15 moves to move through the maze, then `printTotalNumberOfSpacesMoved` should return 15. An outline of this method is provided in the Maze.java file.

Likewise, you will need to add methods that will tell someone how many spaces the robot has moved while facing east (name this `printTotalNumberOfSpacesMovedEast`), another for the number of spaces moved while facing west (name this `printTotalNumberOfSpacesMovedWest`), another for the number of spaces moved while facing north, and one last for the number of spaces moved while facing south. Much like the plain-vanilla `printTotalNumberOfSpacesMoved,` these methods only record the total number of spaces moved (in their respective directions) since the last time the MazeBot was told to NavigateMaze.

You should create a command, named something like `printAllDistances`, which will use the above method to print out all of that above information, to the human user. Make sure that whenever NavigateMaze finishes navigating it's way through a maze, this new command is called.

Obviously, the MazeBot should never cause itself to break.

You should put your answer to this part of the homework assignment in a file named **Maze.java**.