Chapter 1 - Algorithms and Abstractions\Robot Mazes\Operators\Operators HW (STUDENT)\Operators HW.html

```
<!DOCTYPE html>

<!--
 *
 * Maze Simulator (c) by Christopher Grattoni
 * Maze Simulator is licensed under a
 * Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License.
 * You should have received a copy of the license along with this work.
 * If not, see <http://creativecommons.org/licenses/by-nc-sa/4.0/>.
 *
 * Last Edited: Aug 9, 2017
 *
 -->

<html>
<head>
    <title>
        Robot Maze Simulator
    </title>

    <style>
        canvas{
            background: #000000;
        }
    </style>
    <script type="text/javascript" src="maze.js"></script>
    <script type="text/javascript" src="speed.js"></script>
    <script type="text/javascript" src="security.js"></script>
    <script type="text/javascript" src="movementfunctions.js"></script>
    <script type="text/javascript" src="engine.js"></script>
    <script>

        /**
         *
         * 1) READ THIS ENTIRE COMMENT
         * 2) AFTER READING, YOU MAY DELETE THIS COMMENT
         * 3) INSERT YOUR OWN CODE HERE TO CONTROL YOUR ROBOT.
         * 4) YOUR GOAL IS TO GET YOUR ROBOT TO THE GRAY SQUARE.
         *
         * Functions you can use:
         *      moveForward(): The robot will move forward
         *          by one square relative to the direction
         *          it is currently facing. If you move into
         *          a white square, the game continues.
         *          If you move into a gray square, you win.
         *          If you try to move into a black square,
         *          you lose the game.
         *
         *      rotateRight(): The robot will rotate to the
         *          right relative to its current orientation.
         *
```

```
 *      rotateLeft(): The robot will rotate to the
 *          left relative to its current orientation.
 *
 *      goalReached(): The function returns true if
 *          you have reached the end of the maze. It
 *          returns false if you are still in a white
 *          square. This function can only be called
 *          100 times per maze to try to prevent the
 *          game from crashing.
 *
 *      canMove(direction): This function returns true
 *          if the robot can move in the specified direction
 *          relative to its current orientation. Otherwise,
 *          it returns false. You must replace the parameter
 *          'direction' with one of the following arguments:
 *              'forward'
 *              'backward'
 *              'left'
 *              'right'
 *          Note: you need to include the quotes since this function
 *          only accepts a string as its argument.
 *
 *  Other programming techniques you can use:
 *      -You can use iteration, such as 'for loops' and 'while loops'.
 *      -You can define your own functions.
 *      -You can define your own variables.
 *
 */

 /**
  *     *** INSTRUCTIONS ***
  *     Code the situation below correctly in order to solve the puzzle!
  *     Use the || and && operators to achieve the goal.
  *
  *     The procedure should run until the goal has been reached
  *
  *     Looking at the maze there is a pattern that can be followed to solve it!
  *     HINT: What should happen at each different type of intersection?
  */

function robotInstructions()
{
    // Code goes here
    while (!goalReached())
    {
        moveForward();
        if(!canMove("left") && canMove("forward") && canMove("right"))
        {
            rotateRight();
        }
        else if(canMove("left") && canMove("forward") && !canMove("right"))
        {
            rotateLeft();
        }
        else if(canMove("left") && canMove("forward") && canMove("right"))
        {
```

```
                    rotateLeft();
                }
                if(!canMove("left") && !canMove("forward") && !canMove("right"))
                {
                    rotateLeft();
                    rotateLeft();
                    moveForward();
                    rotateRight();
                }
            }
        }
    </script>

</head>


<body onload="gameFrameworkInit()">

    <canvas id="myCanvas" width="400" height="500"></canvas>

</body>


</html>

```