**Freightliner Developer Coding Challenge**

Create a console application with C# to implement a simple robot game:

Simulate a robot moving across a board with 3x3 dimension. User can input some defined instructions to move the robot across the chess board, but the moves are constrained with several rules:

Allowed instructions:

* PLACE <X> <Y> <DIR>: To put the robot into a specific position and face to a specific direction, X means the row index, Y means the column index and DIR is the facing direction. Example: “PLACE 2 0 NORTH” means placing the robot to the cell in the first column and third row, and facing to North
* TURN <DIR>: To turn the facing direction of the robot. Example: “TURN SOUTH” means changing the robot to face to South.
* MOVE: Move the robot one step forward to the facing direction.
* PRINT: Tell the position of the robot. Example: it will print out “2 0 NORTH” if the robot is standing at the cell in the first column and third row, and facing to North.

Notes:

* Row and Column index start with zero (0)

|  |  |  |
| --- | --- | --- |
| (2,0) | (2,1) | (2,2) |
| (1,0) | (1,1) | (1,2) |
| (0,0) | (0,1) | (0,2) |

* Valid Directions: NORTH, WEST, SOUTH, EAST
* The first instruction must be a “PLACE” instruction to put the robot into an initial position and direction.

Constraints:

* The robot cannot fall over the board. Example: when it is standing at the cell of 2,2 and facing to North, if the user gives a “MOVE” instruction, it will print out “Stop! Going to fall!” and stay still at the original position.
* If an invalid instruction was given, print out “invalid instruction” and do nothing

Bonus points:

* Make use of coding principle (SOLID/DRY etc) to make it maintainable and extendable
* A easy configurable dimension of the board
* Comments in code

Average hours for this assignment: 4 hours

Submission: Git or zip file

Full Example:

* (program response) Please enter instruction:
* (user input) MOVE
* (program response) Error: First instruction must be PLACE
* (user input) PLACE 1 1 WEST
* (user input) TURN NORTH
* (user input) MOVE
* (user input) PRINT
* (program response) MO
* (user input) MOVE
* (program response) Stop! Going to fall!
* (user input) JUMP
* (program response) Invalid instruction