# Toy Robot Simulator

## Problem Description:

The application is a simulation of a toy robot moving on a square tabletop, of dimensions 5 units x 5 units. There are no other obstructions on the table surface. The robot is free to roam around the surface of the table, but must be prevented from falling to destruction. This also includes the initial placement of the toy robot.

## Constraints:

The first valid command to the robot is a PLACE command.
Any move that would cause the robot to fall are ignored.

## High Level Design:

The simulation is on a square top of dimension 5 units x 5 units. Positive co-ordinates of X and Y are considered thus the dimension are 0 to 4 units along X & Y. The Facing Direction could be North, South, East and West.

| | | | | |
|---|---|---|---|---|
| 4 | | | | |
| 3 | | | | |
| 2 | | | | |
| 1 | | | | |
| 0 | 1 | 2 | 3 | 4 |

The following are the commands that the application can read:

- PLACE X,Y,F
- MOVE
- LEFT
- RIGHT
- REPORT

PLACE will put the toy on the table in position X, Y and facing the direction as specified by the user.The first valid command to the robot is a PLACE command, after that, any sequence of commands can be issued, in any order, including another PLACE command.
The application discards all commands in the sequence until a valid PLACE command has been executed.

The validity of 'PLACE' command is ensured by checking the value of X & Y co-ordinates.

Once the Toy Robot has been placed on the Simulator, it can take any commands in any order.

- MOVE - move the toy robot one unit forward in the direction it is currently facing
- LEFT - rotate the robot 90 degrees in the left direction without changing the position of the robot.
- RIGHT - rotate the robot 90 degrees in the right direction without changing the position of the robot.
- REPORT- announce the X,Y and F of the robot

## Detailed Design:

Application is built in JavaScript whose input and output could be viewed on a browser.

The HTML Output Window is designed to take inputs from the user and also report the current position of the Robot.

Input is entered on a text box and on the click of the 'Submit' button, the data entered is verified. If valid command and data have been entered, the robot is placed on the simulator and respond to subsequent commands from the user. The current position of the robot could be seen anytime by issuing the appropriate command.

- Global variables are declared and initialised to hold the initial value of the robot position along X coordinate, Y coordinate and the direction it faces.
- inputCall(str) - gets the input from the text field, converts the entire string to lowercase and checks if it contains 'PLACE'. If it's a valid PLACE command, the validity of position placed is verified and the updated if valid. If invalid, inputs are received until a valid PLACE and position specified.
- initialPos(x,y,f) - updates the global variables upon a valid PLACE command
- updatePos(x,y,f) - updates global variables after every valid command
- currentPos() - returns the current position of the robot.
- check(x, y) - This function checks the validity of the X and Y co-ordinates after every command iteration
- command(cmd) - reads the input from the user and calls the function corresponding to the entered command
- move(x,y,f) - Upon 'MOVE' as the input, the simulator should get the current position of the robot and advance 1 unit along the co-ordinate its currently facing. The updated position would not be printed out to the user.

| Currently Facing Direction along X,Y | Updated position on X coordinate | Updated position on Y coordinate | Facing Direction along updated X,Y |
|---|---|---|---|
| **North** | X | Y+1 | North |
| **South** | X | Y-1 | South |
| **East** | X+1 | Y | East |
| **West** | X-1 | Y | West |

- left(x,y,f) - Upon 'LEFT' as the input, the simulator should get the current position of the robot and turn 90 degrees left of the currently facing direction with no update on its current position. The updated position would not be printed out to the user.

| Currently Facing Direction | Position on X coordinate | Position on Y coordinate | Updated Facing Direction along X,Y |
|---|---|---|---|
| **North** | X | Y | West |
| **South** | X | Y | East |
| **East** | X | Y | North |
| **West** | X | Y | South |

- right(x,y,f) - Upon ‘RIGHT’ as the input, the simulator should get the current position of the robot and turn 90 degrees right of the currently facing direction with no update on its current position. The updated position would not be printed out to the user.
-

| Currently Facing Direction | Position on X coordinate | Position on Y coordinate | Updated Facing Direction along X,Y |
|---|---|---|---|
| **North** | X | Y | East |
| **South** | X | Y | West |
| **East** | X | Y | South |
| **West** | X | Y | North |

- report() - Reports the current position of the robot on the HTML window.

## Test Cases

## Negative Test Case Summary:

| DESCRIPTION OF TEST CASE | iNPUT | OUTPUT | STATUS |
|---|---|---|---|
| INVALID COMMANDS BEFORE A VALID ‘PLACE’ | MOVE | INVALID INPUT | PASS |
| INVALID COMMANDS BEFORE A VALID ‘PLACE’ | LEFT | INVALID INPUT | PASS |
| INVALID COMMANDS BEFORE A VALID ‘PLACE’ | RIGHT | INVALID INPUT | PASS |
| INVALID COMMANDS BEFORE A VALID ‘PLACE’ | REPORT | INVALID INPUT | PASS |
| INVALID COMMANDS BEFORE A VALID ‘PLACE’ | MOVE 1,1,NORTH | INVALID INPUT | PASS |
| INVALID COMMANDS BEFORE A VALID ‘PLACE’ | LEFT 1,1,WEST | INVALID INPUT | PASS |
| INVALID COMMANDS BEFORE A VALID ‘PLACE’ | MOVE 1,-1,SOUTH | INVALID INPUT | PASS |
| INVALID COMMANDS BEFORE A VALID ‘PLACE’ | RIGHT 0,0,EAST | INVALID INPUT | PASS |
| INVALID PLACE COMMAND | PLACE 1,1,FBSJGE | INVALID INPUT | PASS |
| INVALID PLACE COMMAND | PLACE 5.3,NORTH | INVALID INPUT | PASS |
| INVALID PLACE COMMAND | PLACE 0,-1.NORTH | INVALID INPUT | PASS |
| INVALID MOVE AFTER A VALID PLACE | PLACE 4,4,NORTH<br>MOVE | NO OUTPUT | PASS |

## Positive Test Case Summary:

| INPUT | OUTPUT |
|---|---|
| **PLACE 0,0,NORTH<br>MOVE<br>REPORT** | 0,1,NORTH |
| **PLACE 0,0,NORTH<br>LEFT<br>REPORT** | 0,0,WEST |
| **PLACE 0,0,NORTH<br>RIGHT<br>REPORT** | 0,0,EAST |

| INPUT | OUTPUT |
|---|---|
| **PLACE 1,2,EAST**<br>**MOVE**<br>**MOVE**<br>**LEFT**<br>**MOVE**<br>**REPORT** | 3,3,NORTH |
| **PLACE 3,2,EAST**<br>**MOVE**<br>**MOVE**<br>**LEFT**<br>**REPORT** | 4,2,NORTH<br><br>—NO UPDATE ON THE SECOND MOVE COMMAND |