CS5010 - Problem Set 01 - Test Results

pdp-wangzet

September 28, 2014

This test suite tests your implementation of Problem Set 01

1 File: robots.rkt

This week, automated tests will test your question 01
Common Definitions

```
(define INITIAL-ROBOT (initial-robot 100 200))

(define ROBOT-WEST (robot-left INITIAL-ROBOT))

(define ROBOT-EAST (robot-right INITIAL-ROBOT))

(define ROBOT-SOUTH (robot-left ROBOT-WEST))

(define ROBOT-RIGHT-WALL (robot-right (initial-robot 185 200)))

(define ROBOT-LEFT-WALL (robot-left (initial-robot 15 200)))

(define ROBOT-TOP-WALL (initial-robot 100 15))

(define ROBOT-BOTTOM-WALL
(robot-left (robot-left (initial-robot 100 385))))

(define ROBOT-BEYOND-LEFT-FACE-RIGHT
(robot-right (initial-robot -100 200)))

(define ROBOT-BEYOND-RIGHT-FACE-LEFT
(robot-left (initial-robot 300 200)))

(define ROBOT-BEYOND-TOP-FACE-DOWN
(robot-right (robot-right (initial-robot 100 -100))))

(define ROBOT-BEYOND-BOTTOM-FACE-UP (initial-robot 100 500))
```

1.1 Test-Group: initial-robot (1 Points)

1/1

Tests whether the initial robot is correct

1.1.1 Test (equality)

The initial robot should be facing north
Input:

```
(robot-north? (initial-robot 100 200))
```

Expected Output:

```
true
```

Expected Output Value:

```
#t
```

Correct

1.2 Test-Group: robot-left (2 Points)

2/2

Tests basic implementation of robot-left

1.2.1 Test (equality)

A robot facing north will face west when turned left.
Input:

```
(robot-west? (robot-left INITIAL-ROBOT))
```

Expected Output:

```
true
```

Expected Output Value:

```
#t
```

Correct

1.2.2 Test (equality)

A robot facing west will face south when turned left.
Input:

```
(robot-south? (robot-left ROBOT-WEST))
```

Expected Output:

```
true
```

Expected Output Value:

```
#t
```

Correct

1.2.3 Test (equality)

A robot facing south will face east when turned left.
Input:

```
(robot-east? (robot-left ROBOT-SOUTH))
```

Expected Output:

```
true
```

Expected Output Value:

```
#t
```

Correct

1.2.4 Test (equality)

A robot facing east will face north when turned left.
Input:

```
(robot-north? (robot-left ROBOT-EAST))
```

Expected Output:

```
true
```

Expected Output Value:

```
#t
```

Correct

1.3 Test-Group: robot-right (2 Points)

2/2

Tests basic implementation of robot-right

1.3.1 Test (equality)

A robot facing north will face east when turned right.
Input:

```
(robot-east? (robot-right INITIAL-ROBOT))
```

Expected Output:

```
true
```

Expected Output Value:

```
#t
```

Correct

1.3.2 Test (equality)

A robot facing west will face north when turned right.
Input:

```
(robot-north? (robot-right ROBOT-WEST))
```

Expected Output:

```
true
```

Expected Output Value:

```
#t
```

Correct

1.3.3 Test (equality)

A robot facing south will face west when turned right.
Input:

```
(robot-west? (robot-right ROBOT-SOUTH))
```

Expected Output:

```
true
```

Expected Output Value:

```
#t
```

Correct

1.3.4 Test (equality)

A robot facing east will face south when turned right.
Input:

```
(robot-south? (robot-right ROBOT-EAST))
```

Expected Output:

```
true
```

Expected Output Value:

```
#t
```

Correct

1.4 Test-Group: robot-forward (5 Points)

5/5

Tests basic implementation of robot-forward

1.4.1 Test (equality, 1 partial points)

A robot facing north and moving forward should stop at the top wall.
Input:

```
(robot-forward (initial-robot 100 200) 300)
```

Expected Output:

```
ROBOT-TOP-WALL
```

Expected Output Value:

```
#(struct:robot 100 15 "north")
```

Correct

1.4.2 Test (equality, 1 partial points)

A robot facing west and moving forward should stop at the west wall.
Input:

```
(robot-forward ROBOT-WEST 200)
```

Expected Output:

```
ROBOT-LEFT-WALL
```

Expected Output Value:

```
#(struct:robot 15 200 "west")
```

Correct

1.4.3 Test (equality, 1 partial points)

A robot facing east and moving forward should stop at the east wall.
Input:

```
(robot-forward ROBOT-EAST 200)
```

Expected Output:

```
ROBOT-RIGHT-WALL
```

Expected Output Value:

```
#(struct:robot 185 200 "east")
```

Correct

1.4.4 Test (equality, 1 partial points)

A robot facing south and moving forward should stop at the south wall.
Input:

```
(robot-forward ROBOT-SOUTH 300)
```

Expected Output:

```
ROBOT-BOTTOM-WALL
```

Expected Output Value:

```
#(struct:robot 100 385 "south")
```

Correct

1.5 Test-Group: robot-forward (5 Points)

5/5

Tests the one way trap condition for the walls

1.5.1 Test (equality, 0.75 partial points)

A robot beyond the left wall moving forward will enter from outside and stop at right wall.
Input:

```
(robot-forward ROBOT-BEYOND-LEFT-FACE-RIGHT 400)
```

Expected Output:

```
ROBOT-RIGHT-WALL
```

Expected Output Value:

```
#(struct:robot 185 200 "east")
```

Correct

1.5.2 Test (equality, 0.75 partial points)

A robot beyond the right wall moving forward will enter from outside and stop at left wall.
Input:

```
(robot-forward ROBOT-BEYOND-RIGHT-FACE-LEFT 400)
```

Expected Output:

```
ROBOT-LEFT-WALL
```

Expected Output Value:

```
#(struct:robot 15 200 "west")
```

Correct

1.5.3 Test (equality, 0.75 partial points)

A robot beyond the top wall moving forward will enter from outside and stop at bottom wall.
Input:

```
(robot-forward ROBOT-BEYOND-TOP-FACE-DOWN 600)
```

Expected Output:

```
ROBOT-BOTTOM-WALL
```

Expected Output Value:

```
#(struct:robot 100 385 "south")
```

Correct

1.5.4 Test (equality, 0.75 partial points)

A robot beyond the bottom wall moving forward will enter from outside and stop at top wall.
Input:

```
(robot-forward ROBOT-BEYOND-BOTTOM-FACE-UP 600)
```

Expected Output:

```
ROBOT-TOP-WALL
```

Expected Output Value:

```
#(struct:robot 100 15 "north")
```

Correct

1.5.5 Test (equality, 0.75 partial points)

A robot outside the canvas should not stop at the wall boundary.
Input:

```
(robot-forward (robot-right (initial-robot 20 1600)) 1000)
```

Expected Output:

```
(robot-right (initial-robot 1020 1600))
```

Expected Output Value:

```
#(struct:robot 1020 1600 "east")
```

Correct

2 Results

Successes: 18
Wrong Outputs: 0
Errors: 0
Achieved Points: 15
Total Points (rounded): 15/15