

# rspec

Ryan Durling

February 3, 2012

## Contents

<b>1</b>	<b>String</b>	<b>1</b>
1.1	#is <sub>i</sub> . . . . .	1
1.1.1	<b>SUCCESS</b> Check to see if string is an Integer . . . . .	1
1.2	#is <sub>s</sub> ? . . . . .	1
1.2.1	<b>SUCCESS</b> Check to see if string is string . . . . .	1
<b>2</b>	<b>Fixnum</b>	<b>1</b>
2.1	#is <sub>i</sub> ? . . . . .	1
2.1.1	<b>SUCCESS</b> Check to see if integer is integer . . . . .	1
<b>3</b>	<b>Array</b>	<b>1</b>
3.1	#s <sub>toi</sub> . . . . .	1
3.1.1	<b>SUCCESS</b> Change the contents of the array with integers and/or strings . . . . .	1
3.2	#add <sub>value</sub> . . . . .	2
3.2.1	<b>SUCCESS</b> “Add value at index ‘num’ by ‘some other num’ . . . . .	2
3.3	#subtract <sub>value</sub> . . . . .	2
3.3.1	<b>SUCCESS</b> “Subtract value at index ‘num’ by ‘some other num’ . . . . .	2
3.4	#shift <sub>right</sub> . . . . .	2
3.4.1	<b>SUCCESS</b> Move elements of the array to the right . . . . .	2
3.5	#shift <sub>left</sub> . . . . .	2
3.5.1	<b>SUCCESS</b> Move elements of the array to the left . . . . .	2
<b>4</b>	<b>Command<sub>File</sub></b>	<b>2</b>
4.1	#new . . . . .	2
4.1.1	<b>SUCCESS</b> Returns a new Command <sub>File</sub> object . . . . .	2
4.1.2	<b>SUCCESS</b> Throws an ArgumentError if passed more than one arguments . . . . .	2
4.1.3	<b>SUCCESS</b> It fills an array with lines of text . . . . .	2
4.1.4	#process <sub>commandfile</sub> . . . . .	2
4.2	#valid <sub>commandstructure</sub> ? . . . . .	2
4.2.1	<b>SUCCESS</b> Return true if commands within the . . . . .	2

4.2.2	#valid <sub>xy</sub> ?	3
4.2.3	#valid <sub>text</sub> ?	3
4.2.4	#valid <sub>commands</sub> ?	3
<b>5</b>	<b>Command</b>	<b>5</b>
5.1	#new	5
5.1.1	<b>SUCCESS</b> Returns a new Command object	5
5.1.2	<b>SUCCESS</b> Throws an ArgumentError if passed less than two arguments	5
5.2	#position	5
5.2.1	<b>SUCCESS</b> Returns an array of the position	5
5.3	#movement	5
5.3.1	<b>SUCCESS</b> Returns an array of the movement	5
<b>6</b>	<b>My<sub>Matrix</sub></b>	<b>5</b>
6.1	#new	5
6.1.1	<b>SUCCESS</b> Returns a new My <sub>Matrix</sub> object	5
6.1.2	<b>SUCCESS</b> Throws an ArgumentError if passed less than two arguments	5
6.1.3	<b>SUCCESS</b> Creates a two dimensional array of nil	5
6.2	#[]	5
6.2.1	<b>SUCCESS</b> Return the xy given within the two dimensional array	5
6.3	#[]=	5
6.3.1	<b>SUCCESS</b> Place the value at the xy given within the two dimensional array	5
6.4	#my <sub>matrix</sub>	5
6.4.1	<b>SUCCESS</b> Returns the two dimensional matrix	5
<b>7</b>	<b>Navigation</b>	<b>5</b>
7.1	#Navigation.VALID <sub>COMMANDS</sub>	5
7.1.1	<b>SUCCESS</b> Returns array of VALID <sub>COMMANDS</sub>	5
7.2	#Navigation.COMPASS	5
7.2.1	<b>SUCCESS</b> Returns array COMPASS	5
7.3	#Navigation.DIRECTION	5
7.3.1	<b>SUCCESS</b> Returns array DIRECTION	5
7.4	#Navigation.execute <sub>move</sub>	5
7.4.1	<b>SUCCESS</b> Returns new coordinate and heading based upon given instruction	5
7.4.2	#Navigation.move	5
7.4.3	#Navigation.change <sub>heading</sub>	6
<b>8</b>	<b>Plateau</b>	<b>6</b>
8.1	#new	6
8.1.1	<b>SUCCESS</b> Returns a new Plateau object	6

8.1.2	<b>SUCCESS</b> Throws an <code>ArgumentError</code> if passed the wrong number of arguments . . . . .	6
<b>9</b>	<b>Rover<sub>Controller</sub></b>	<b>6</b>
9.1	<code>#new</code> . . . . .	6
9.1.1	<b>SUCCESS</b> Returns a new instance of a <code>Rover<sub>Controller</sub></code> object . . . . .	6
9.1.2	<b>SUCCESS</b> It takes the first element of the <code>command<sub>structure</sub></code> array . . . . .	6
9.1.3	<code>#create<sub>rover</sub></code> . . . . .	6
9.1.4	<code>#initial<sub>placement</sub></code> . . . . .	6
9.2	<code>#explore</code> . . . . .	7
9.2.1	<b>SUCCESS</b> Each rover moves its full path one at a time .	7
9.2.2	<code>#pass<sub>pathcheck</sub>?</code> . . . . .	7
9.2.3	<code>#move</code> . . . . .	7
<b>10</b>	<b>Rover</b>	<b>7</b>
10.1	<code>#new</code> . . . . .	7
10.1.1	<b>SUCCESS</b> Return a new Rover object . . . . .	7
10.1.2	<b>SUCCESS</b> Throws an <code>ArgumentError</code> if wrong number of arguments . . . . .	7
10.2	<code>#move</code> . . . . .	7
10.2.1	<b>SUCCESS</b> Logs the place the rover was in and . . . . .	7
10.3	<code>#home<sub>coordinate</sub></code> . . . . .	8
10.3.1	<b>SUCCESS</b> Returns the coordinates of where the rover started . . . . .	8
10.4	<code>#tag</code> . . . . .	8
10.4.1	<b>SUCCESS</b> Return the tag number of the rover . . . . .	8
10.5	<code>#command</code> . . . . .	8
10.5.1	<b>SUCCESS</b> Return the list of instructions for the rover .	8
10.6	<code>#name</code> . . . . .	8
10.6.1	<b>SUCCESS</b> Returns the name of the rover . . . . .	8
10.7	<code>#coordinate</code> . . . . .	8
10.7.1	<b>SUCCESS</b> Returns the coordinates of rover . . . . .	8
10.8	<code>#coordinate=</code> . . . . .	8
10.8.1	<b>SUCCESS</b> Sets the coordinate of the rover . . . . .	8
<b>11</b>	<b>Summary</b>	<b>8</b>

## 1 String

### 1.1 #is<sub>i</sub>

1.1.1 SUCCESS Check to see if string is an Integer

### 1.2 #is<sub>s</sub>?

1.2.1 SUCCESS Check to see if string is string

## 2 Fixnum

### 2.1 #is<sub>i</sub>?

2.1.1 SUCCESS Check to see if integer is integer

## 3 Array

### 3.1 #s<sub>toi</sub>

3.1.1 SUCCESS Change the contents of the array with integers and/or strings

and converts the integers to integers

### 3.2 #add<sub>value</sub>

3.2.1 SUCCESS “Add value at index ‘num’ by ‘some other num’

### 3.3 #subtract<sub>value</sub>

3.3.1 SUCCESS “Subtract value at index ‘num’ by ‘some other num’

### 3.4 #shift<sub>right</sub>

3.4.1 SUCCESS Move elements of the array to the right

by removing the last element and putting it at the beginning

### 3.5 #shift<sub>left</sub>

3.5.1 SUCCESS Move elements of the array to the left

by removing the first element and putting it at the end

## 4 Command<sub>File</sub>

### 4.1 #new

#### 4.1.1 SUCCESS Returns a new Command<sub>File</sub> object

#### 4.1.2 SUCCESS Throws an ArgumentError if passed more than one arguments

#### 4.1.3 SUCCESS It fills an array with lines of text

#### 4.1.4 #process<sub>commandfile</sub>

- **SUCCESS** Processes the lines of text into a command structure by gathering the commands and creating a command structure
- #gather<sub>command</sub>
  - **SUCCESS** Gather the lines and splits them into a command array checking for integers and converting them
- #create<sub>commandstructre</sub>
  - **SUCCESS** Takes the the first element, which is the size of the field, and then takes every odd index putting every two commands into a command object and pushing them into the command<sub>structure</sub> array

### 4.2 #valid<sub>commandstructure</sub>?

#### 4.2.1 SUCCESS Return true if commands within the

command<sub>structure</sub> array are valid by checking the xy coordinates, text within the file and the command objects

#### 4.2.2 #valid<sub>xy</sub>?

- **SUCCESS** Return true if xy coordinates given are valid

#### 4.2.3 #valid<sub>text</sub>?

- **SUCCESS** Return true if the text in the file is correct

#### 4.2.4 #valid<sub>commands</sub>?

- **SUCCESS** Return true if the command objects within the command<sub>structure</sub> array are valid by checking the position and movent
- #valid<sub>position</sub>?

- **SUCCESS** Return true if the positin in the command object is valid
- `#validmovement?`
  - **SUCCESS** Return true if the movement in the command object is valid



## 5 Command

### 5.1 #new

5.1.1 SUCCESS Returns a new Command object

5.1.2 SUCCESS Throws an ArgumentError if passed less than two arguments

### 5.2 #position

5.2.1 SUCCESS Returns an array of the position

### 5.3 #movement

5.3.1 SUCCESS Returns an array of the movement

## 6 My<sub>Matrix</sub>

### 6.1 #new

6.1.1 SUCCESS Returns a new My<sub>Matrix</sub> object

6.1.2 SUCCESS Throws an ArgumentError if passed less than two arguments

6.1.3 SUCCESS Creates a two dimensional array of nil

### 6.2 #[]

6.2.1 SUCCESS Return the xy given within the two dimensional array

### 6.3 #[]=

6.3.1 SUCCESS Place the value at the xy given within the two dimensional array

### 6.4 #my<sub>matrix</sub>

6.4.1 SUCCESS Returns the two dimensional matrix

## 7 Navigation

### 7.1 #Navigation.VALID<sub>COMMANDS</sub>

7.1.1 SUCCESS Returns array of VALID<sub>COMMANDS</sub>

### 7.2 #Navigation.COMPASS

7.2.1 SUCCESS Returns array COMPASS

### 7.3 #Navigation.DIRECTION

7.3.1 SUCCESS Returns array DIRECTION

### 7.4 #Navigation.execute<sub>move</sub><sup>8</sup>

7.4.1 SUCCESS Returns new coordinate and heading based upon given instruction

#### 7.4.2 #Navigation.move

- SUCCESS Moves coordinate based on heading



#### 7.4.3 `#Navigation.changeheading`

- **SUCCESS** Turns the coordinate left or right based on current heading and instruction
- `#Navigation.calibratecompass`
  - **SUCCESS** Move the compass heading so it matches the heading of the current coordinate

## 8 Plateau

### 8.1 `#new`

#### 8.1.1 **SUCCESS** Returns a new Plateau object

and adds one to the x,y

#### 8.1.2 **SUCCESS** Throws an `ArgumentError` if passed the wrong number of arguments

## 9 `RoverController`

### 9.1 `#new`

#### 9.1.1 **SUCCESS** Returns a new instance of a `RoverController` object

#### 9.1.2 **SUCCESS** It takes the first element of the `commandstructure` array

and creates a new two dimensional array

#### 9.1.3 `#createrover`

- **SUCCESS** Create the rovers based on the number of command objects given

#### 9.1.4 `#initialplacement`

- **SUCCESS** Place the rovers based on the coordinates provided by the command object
- `#passinitialplacementcheck`
  - **SUCCESS** Return true if not placing one rover atop another
  - **SUCCESS** Throw error if there is a collision in placement

- `#passcollisioncheck`
  - \* **SUCCESS** Return true if there is no collision

## 9.2 `#explore`

### 9.2.1 **SUCCESS** Each rover moves its full path one at a time

#### 9.2.2 `#passpathcheck?`

- **SUCCESS** Check to see if the path is valid
- **SUCCESS** Raise an error if path is not within the boundary or if there is a collision along the path
- `#passboundarycheck?`
  - **SUCCESS** Return true if the coordinate is within the prescribed boundary

#### 9.2.3 `#move`

- **SUCCESS** Move a rover based on the command list given from the file by removing it from the map and placing it in its new spot
- `#removerover`
  - **SUCCESS** Remove the rover from the map coordinate it inhabits
- `#placerover`
  - **SUCCESS** Place the rover in its new coordinates

## 10 Rover

### 10.1 `#new`

#### 10.1.1 **SUCCESS** Return a new Rover object

#### 10.1.2 **SUCCESS** Throws an `ArgumentError` if wrong number of arguments

### 10.2 `#move`

#### 10.2.1 **SUCCESS** Logs the place the rover was in and

Return new coordinates based on instruction

**10.3** `#home`<sub>coordinate</sub>

**10.3.1** **SUCCESS** Returns the coordinates of where the rover started

**10.4** `#tag`

**10.4.1** **SUCCESS** Return the tag number of the rover

**10.5** `#command`

**10.5.1** **SUCCESS** Return the list of instructions for the rover

**10.6** `#name`

**10.6.1** **SUCCESS** Returns the name of the rover

**10.7** `#coordinate`

**10.7.1** **SUCCESS** Returns the coordinates of rover

**10.8** `#coordinate=`

**10.8.1** **SUCCESS** Sets the coordinate of the rover

## **11 Summary**

Finished in **0.015143 seconds** 62 examples, 0 failures