Rspec Cheat Sheet

```
require "thing"

describe Thing do

 before do
   @thing = Thing.new
 end

  it "should check universal stability" do
    @thing.answer.should == 42
  end

 it "should enforce universal stability" do
    @thing.val = 3
    @thing.enforce_stability
    @thing.val.should == 42
  end

end
```

loads the file tested by the spec

describe [ClassName] do

- **describe** declares what is being tested and defines a context
- Optionally, you can specify some code to run before each example.
- The beginning of each example is marked by "**it**" followed by a description of expected behavior in quotes
- the first lines of each spec set up the conditions required for the test to succeed
- if any line of the example fails to execute, the spec fails.
- **object.should** and
- **object.should_not** are used in rspec to compare actual to expected values (see the next page for syntax for the different types of comparisons)

Read more about RSpec

http://rspec.info/

http://www.pragprog.com/titles/achbd/the-rspec-book

```
target.should equal <value>
target.should not_equal <value>

target.should be_close <value>, <tolerance>
target.should_not be_close <value>, <tolerance>

target.should be <value>
target.should_not be <value>

target.should be_predicate [optional args]
target.should_not be_predicate [optional args]

target.should be < 6
target.should == 5
target.should_not == 'Samantha'

target.should match <regex>
target.should_not match <regex>

target.should be_an_instance_of <class>
target.should_not be_an_instance_of <class>

target.should be_a_kind_of <class>
target.should_not be_a_kind_of <class>

target.should respond_to <symbol>
target.should_not respond_to <symbol>

lambda {a_call}.should raise_error
lambda {a_call}.should raise_error(<exception> [, message])
lambda {a_call}.should_not raise_error
lambda {a_call}.should_not raise_error(<exception> [, message])

target.should include <object>
target.should_not include <object>

target.should have(<number>).things
target.should have_at_least(<number>).things
target.should have_at_most(<number>).things

target.should have(<number>).errors_on(:field)

lambda { thing.destroy }.should change(Thing, :count).by(-1)
```