Ruby 初級者向けレッスン第 15 回（解答例）

okkez @ Ruby 関西, サカイ@小波ゼミ, チカホリ@小波ゼミ

2007 年 10 月 27 日

Test::Unit を使ったバージョン

Step 1

テストコード (test_stack.rb)

```
01: require 'stack'
02: require 'test/unit'
03: class TestStack < Test::Unit::TestCase
04:   def setup
05:     @stack = Stack.new
06:   end
07:
08:   def test_empty?
09:     assert(@stack.empty?, 'a new stack is empty.')
10:   end
11: end
```

スタッククラスのコード (stack.rb)

```
01: class Stack
02:   def empty?
03:     true
04:   end
05: end
```

Step 2

テストコード (test_stack.rb)

```
01: require 'stack'
02: require 'test/unit'
03: class TestStack < Test::Unit::TestCase
04:   def setup
```

```
05:     @stack = Stack.new
06:   end
07:
08:   def test_empty?
09:     assert(@stack.empty?, 'a new stack is empty.')
10:   end
11:
12:   def test_push_and_pop
13:     @stack.push(3)
14:     assert_equal(3, @stack.pop, 'pop returns the last value.')
15:   end
16: end
```

スタッククラスのコード (stack.rb)

```
01: class Stack
02:   def empty?
03:     true
04:   end
05:
06:   def push(val)
07:   end
08:
09:   def pop
10:     return 3
11:   end
12: end
```

Step 3

テストコード (test_stack.rb)

```
01: require 'stack'
02: require 'test/unit'
03: class TestStack < Test::Unit::TestCase
04:   def setup
05:     @stack = Stack.new
06:   end
07:
08:   def test_empty?
09:     assert(@stack.empty?, 'a new stack is empty.')
10:   end
11:
```

```
12:   def test_push_and_pop
13:     @stack.push(3)
14:     assert_equal(3, @stack.pop, 'pop returns the last value.')
15:   end
16:
17:   def test_push_and_size
18:     @stack.push(3)
19:     assert_equal(1, @stack.size, 'push increments the size.')
20:   end
21: end
```

スタッククラスのコード (stack.rb)

```
01: class Stack
02:   def empty?
03:     true
04:   end
05:
06:   def push(val)
07:   end
08:
09:   def pop
10:     return 3
11:   end
12:
13:   def size
14:     1
15:   end
16: end
```

Step 4

テストコード (test_stack.rb)

```
01: require 'stack'
02: require 'test/unit'
03: class TestStack < Test::Unit::TestCase
04:   def setup
05:     @stack = Stack.new
06:   end
07:
08:   def test_empty?
09:     assert(@stack.empty?, 'a new stack is empty.')
```

```
10:   end
11:
12:   def test_push_and_pop
13:     @stack.push(3)
14:     assert_equal(3, @stack.pop, 'pop returns the last value.')
15:   end
16:
17:   def test_push_and_size
18:     @stack.push(3)
19:     assert_equal(1, @stack.size, 'push increments the size.')
20:     @stack.push(5)
21:     assert_equal(2, @stack.size, 'push increments the size.')
22:   end
23: end
```

スタッククラスのコード (stack.rb)

```
01: class Stack
02:   def initialize
03:     @size = 0
04:   end
05:
06:   def empty?
07:     true
08:   end
09:
10:   def push(val)
11:     @size += 1
12:   end
13:
14:   def pop
15:     return 3
16:   end
17:
18:   def size
19:     return @size
20:   end
21: end
```

Step 5

テストコード (test_stack.rb)

```
01: require 'stack'
02: require 'test/unit'
03: class TestStack < Test::Unit::TestCase
04:   def setup
05:     @stack = Stack.new
06:   end
07:
08:   def test_empty?
09:     assert(@stack.empty?, 'a new stack is empty.')
10:   end
11:
12:   def test_push_and_pop
13:     @stack.push(3)
14:     assert_equal(3, @stack.pop, 'pop returns the last value.')
15:   end
16:
17:   def test_push_and_size
18:     @stack.push(3)
19:     assert_equal(1, @stack.size, 'push increments the size.')
20:     @stack.push(5)
21:     assert_equal(2, @stack.size, 'push increments the size.')
22:   end
23:
24:   def test_push_and_empty?
25:     @stack.push(3)
26:     assert_equal(false, @stack.empty?, 'a stack with data is not empty.')
27:   end
28: end
```

スタッククラスのコード (stack.rb)

```
01: class Stack
02:   def initialize
03:     @size = 0
04:   end
05:
06:   def empty?
07:     return @size == 0
08:   end
09:
```

```
10:   def push(val)
11:     @size += 1
12:   end
13:
14:   def pop
15:     return 3
16:   end
17:
18:   def size
19:     return @size
20:   end
21: end
```

Step 6

テストコード (test_stack.rb)

```
01: require 'stack'
02: require 'test/unit'
03: class TestStack < Test::Unit::TestCase
04:   def setup
05:     @stack = Stack.new
06:   end
07:
08:   def test_empty?
09:     assert(@stack.empty?, 'a new stack is empty.')
10:   end
11:
12:   def test_push_and_pop
13:     @stack.push(3)
14:     assert_equal(3, @stack.pop, 'pop returns the last value.')
15:   end
16:
17:   def test_push_and_size
18:     @stack.push(3)
19:     assert_equal(1, @stack.size, 'push increments the size.')
20:     @stack.push(5)
21:     assert_equal(2, @stack.size, 'push increments the size.')
22:   end
23:
24:   def test_push_and_empty?
25:     @stack.push(3)
26:     assert_equal(false, @stack.empty?, 'a stack with data is not empty.')
```

```
27:   end
28:
29:   def test_empty_pop
30:     assert_raise(Stack::EmptyStackError,
31:                  'to pop a empty stack raise an error.') {@stack.pop}
32:   end
33: end
```

スタッククラスのコード (stack.rb)

```
01: class Stack
02:   class EmptyStackError < StandardError; end
03:
04:   def initialize
05:     @size = 0
06:   end
07:
08:   def empty?
09:     return @size == 0
10:   end
11:
12:   def push(val)
13:     @size += 1
14:   end
15:
16:   def pop
17:     raise EmptyStackError if empty?
18:     return 3
19:   end
20:
21:   def size
22:     return @size
23:   end
24: end
```

Step 7

テストコード (test_stack.rb)

```
01: require 'stack'
02: require 'test/unit'
03: class TestStack < Test::Unit::TestCase
04:   def setup
```

```
05:     @stack = Stack.new
06:   end
07:
08:   def test_empty?
09:     assert(@stack.empty?, 'a new stack is empty.')
10:   end
11:
12:   def test_push_and_pop
13:     @stack.push(3)
14:     assert_equal(3, @stack.pop, 'pop returns the last value.')
15:   end
16:
17:   def test_push_and_size
18:     @stack.push(3)
19:     assert_equal(1, @stack.size, 'push increments the size.')
20:     @stack.push(5)
21:     assert_equal(2, @stack.size, 'push increments the size.')
22:   end
23:
24:   def test_push_and_empty?
25:     @stack.push(3)
26:     assert_equal(false, @stack.empty?, 'a stack with data is not empty.')
27:   end
28:
29:   def test_empty_pop
30:     assert_raise(Stack::EmptyStackError,
31:                  'to pop a empty stack raise an error.') {@stack.pop}
32:   end
33:
34:   def test_push_push_pop_and_size
35:     @stack.push(3)
36:     @stack.push(5)
37:     @stack.pop
38:     assert_equal(1, @stack.size, 'pop decrements the size.')
39:   end
40: end
```

スタッククラスのコード (stack.rb)

```
01: class Stack
02:   class EmptyStackError < StandardError; end
03:
04:   def initialize
```

```
05:     @size = 0
06:   end
07:
08:   def empty?
09:     return @size == 0
10:   end
11:
12:   def push(val)
13:     @size += 1
14:   end
15:
16:   def pop
17:     raise EmptyStackError if empty?
18:     @size -= 1
19:     return 3
20:   end
21:
22:   def size
23:     return @size
24:   end
25: end
```

Step 8

テストコード (test_stack.rb)

```
01: require 'stack'
02: require 'test/unit'
03: class TestStack < Test::Unit::TestCase
04:   def setup
05:     @stack = Stack.new
06:   end
07:
08:   def test_empty?
09:     assert(@stack.empty?, 'a new stack is empty.')
10:   end
11:
12:   def test_push_and_pop
13:     @stack.push(3)
14:     assert_equal(3, @stack.pop, 'pop returns the last value.')
15:   end
16:
17:   def test_push_and_size
```

```
18:     @stack.push(3)
19:     assert_equal(1, @stack.size, 'push increments the size.')
20:     @stack.push(5)
21:     assert_equal(2, @stack.size, 'push increments the size.')
22:   end
23:
24:   def test_push_and_empty?
25:     @stack.push(3)
26:     assert_equal(false, @stack.empty?, 'a stack with data is not empty.')
27:   end
28:
29:   def test_empty_pop
30:     assert_raise(Stack::EmptyStackError,
31:                  'to pop a empty stack raise an error.') {@stack.pop}
32:   end
33:
34:   def test_push_push_pop_and_size
35:     @stack.push(3)
36:     @stack.push(5)
37:     @stack.pop
38:     assert_equal(1, @stack.size, 'pop decrements the size.')
39:   end
40:
41:   def test_push_push_and_pop
42:     @stack.push(3)
43:     @stack.push(5)
44:     assert_equal(5, @stack.pop, 'pop returns the last value.')
45:   end
46: end
```

スタッククラスのコード (stack.rb)

```
01: class Stack
02:   class EmptyStackError < StandardError; end
03:
04:   def initialize
05:     @size = 0
06:     @values = Array.new
07:   end
08:
09:   def empty?
10:     return @size == 0
11:   end
```

```
12:
13:   def push(val)
14:     @size += 1
15:     @values[@size - 1] = val
16:   end
17:
18:   def pop
19:     raise EmptyStackError if empty?
20:     val = @values[@size - 1]
21:     @size -= 1
22:     return val
23:   end
24:
25:   def size
26:     return @size
27:   end
28: end
```

RSpec を使用したバージョン

Test::Unit 版とは実装コードも少し異なっていますが、それは書かれた時期の違いということで。

Step 1

テストコード (stack_spec.rb)

```
01: require 'stack'
02:
03: describe Stack, "when empty" do
04:   before do
05:     @empty_stack = Stack.new
06:   end
07:
08:   it "should be empty" do
09:     @empty_stack.should be_empty
10:   end
11:
12:   after do
13:     @empty_stack = nil
14:   end
15: end
16:
```

スタッククラスのコード (stack.rb)

```
01:
02: class Stack
03:   def empty?
04:     true
05:   end
06: end
```

Step 2

テストコード (stack_spec.rb)

```
01: require 'stack'
02:
03: describe Stack, "when empty" do
04:   before do
05:     @empty_stack = Stack.new
06:   end
07:
08:   it "should be empty" do
09:     @empty_stack.should be_empty
10:   end
11:
12:   after do
13:     @empty_stack = nil
14:   end
15: end
16:
17: describe Stack, "when push 3" do
18:   before do
19:     @stack = Stack.new
20:     @stack.push(3)
21:   end
22:
23:   it "should pop 3" do
24:     @stack.pop.should == 3
25:   end
26:
27:   after do
28:     @stack = nil
29:   end
30: end
```

スタッククラスのコード (stack.rb)

```
01:
02: class Stack
03:   def empty?
04:     true
05:   end
06:
07:   def push(val)
08:   end
09:
10:   def pop
11:     3
12:   end
13: end
```

Step 3

テストコード (stack_spec.rb)

```
01: require 'stack'
02:
03: describe Stack, "when empty" do
04:   before do
05:     @empty_stack = Stack.new
06:   end
07:
08:   it "should be empty" do
09:     @empty_stack.should be_empty
10:   end
11:
12:   after do
13:     @empty_stack = nil
14:   end
15: end
16:
17: describe Stack, "when push 3" do
18:   before do
19:     @stack = Stack.new
20:     @stack.push(3)
21:   end
22:
23:   it "should pop 3" do
24:     @stack.pop.should == 3
```

```
25:   end
26:
27:   it "should size 1" do
28:     @stack.size.should == 1
29:   end
30:
31:   after do
32:     @stack = nil
33:   end
34: end
```

スタッククラスのコード (stack.rb)

```
01:
02: class Stack
03:   def empty?
04:     true
05:   end
06:
07:   def push(val)
08:   end
09:
10:   def pop
11:     3
12:   end
13:
14:   def size
15:     1
16:   end
17: end
```

Step 4

テストコード (stack_spec.rb)

```
01: require 'stack'
02:
03: describe Stack, "when empty" do
04:   before do
05:     @empty_stack = Stack.new
06:   end
07:
08:   it "should be empty" do
```

```
09:     @empty_stack.should be_empty
10:   end
11:
12:   after do
13:     @empty_stack = nil
14:   end
15: end
16:
17: describe Stack, "when push 3" do
18:   before do
19:     @stack = Stack.new
20:     @stack.push(3)
21:   end
22:
23:   it "should pop 3" do
24:     @stack.pop.should == 3
25:   end
26:
27:   it "should size 1" do
28:     @stack.size.should == 1
29:   end
30:
31:   after do
32:     @stack = nil
33:   end
34: end
35:
36: describe Stack, "when push 3 and 5" do
37:   before do
38:     @stack = Stack.new
39:     @stack.push(3)
40:     @stack.push(5)
41:   end
42:
43:   it "should size 2" do
44:     @stack.size.should == 2
45:   end
46: end
```

スタッククラスのコード (stack.rb)

```
01:
02: class Stack
```

```
03:   def initialize
04:     @size = 0
05:   end
06:
07:   def empty?
08:     true
09:   end
10:
11:   def push(val)
12:     @size += 1
13:   end
14:
15:   def pop
16:     3
17:   end
18:
19:   def size
20:     @size
21:   end
22: end
```

Step 5

テストコード (stack_spec.rb)

```
01: require 'stack'
02:
03: describe Stack, "when empty" do
04:   before do
05:     @empty_stack = Stack.new
06:   end
07:
08:   it "should be empty" do
09:     @empty_stack.should be_empty
10:   end
11:
12:   after do
13:     @empty_stack = nil
14:   end
15: end
16:
17: describe Stack, "when push 3" do
18:   before do
```

```
19:     @stack = Stack.new
20:     @stack.push(3)
21:   end
22:
23:   it "should pop 3" do
24:     @stack.pop.should == 3
25:   end
26:
27:   it "should size 1" do
28:     @stack.size.should == 1
29:   end
30:
31:   it "should not be enpty" do
32:     @stack.should_not be_empty
33:   end
34:
35:   after do
36:     @stack = nil
37:   end
38: end
39:
40: describe Stack, "when push 3 and 5" do
41:   before do
42:     @stack = Stack.new
43:     @stack.push(3)
44:     @stack.push(5)
45:   end
46:
47:   it "should size 2" do
48:     @stack.size.should == 2
49:   end
50: end
```

スタッククラスのコード (stack.rb)

```
01:
02: class Stack
03:   def initialize
04:     @size = 0
05:   end
06:
07:   def empty?
08:     size == 0
```

```
09:   end
10:
11:   def push(val)
12:     @size += 1
13:   end
14:
15:   def pop
16:     3
17:   end
18:
19:   def size
20:     @size
21:   end
22: end
```

Step 6

テストコード (stack_spec.rb)

```
01: require 'stack'
02:
03: describe Stack, "when empty" do
04:   before do
05:     @empty_stack = Stack.new
06:   end
07:
08:   it "should be empty" do
09:     @empty_stack.should be_empty
10:   end
11:
12:   it "should raise Stack::EmptyStackError" do
13:     lambda{
14:       @empty_stack.pop
15:     }.should raise_error Stack::EmptyStackError
16:   end
17:
18:   after do
19:     @empty_stack = nil
20:   end
21: end
22:
23: describe Stack, "when push 3" do
24:   before do
```

```
25:     @stack = Stack.new
26:     @stack.push(3)
27:   end
28:
29:   it "should pop 3" do
30:     @stack.pop.should == 3
31:   end
32:
33:   it "should size 1" do
34:     @stack.size.should == 1
35:   end
36:
37:   it "should not be enpty" do
38:     @stack.should_not be_empty
39:   end
40:
41:   after do
42:     @stack = nil
43:   end
44: end
45:
46: describe Stack, "when push 3 and 5" do
47:   before do
48:     @stack = Stack.new
49:     @stack.push(3)
50:     @stack.push(5)
51:   end
52:
53:   it "should size 2" do
54:     @stack.size.should == 2
55:   end
56: end
```

スタッククラスのコード (stack.rb)

```
01:
02: class Stack
03:   class EmptyStackError < StandardError
04:   end
05:   def initialize
06:     @size = 0
07:   end
08:
```

```
09:   def empty?
10:     size == 0
11:   end
12:
13:   def push(val)
14:     @size += 1
15:   end
16:
17:   def pop
18:     raise EmptyStackError if empty?
19:     3
20:   end
21:
22:   def size
23:     @size
24:   end
25: end
```

Step 7

テストコード (stack_spec.rb)

```
01: require 'stack'
02:
03: describe Stack, "when empty" do
04:   before do
05:     @empty_stack = Stack.new
06:   end
07:
08:   it "should be empty" do
09:     @empty_stack.should be_empty
10:   end
11:
12:   it "should raise Stack::EmptyStackError" do
13:     lambda{
14:       @empty_stack.pop
15:     }.should raise_error Stack::EmptyStackError
16:   end
17:
18:   after do
19:     @empty_stack = nil
20:   end
21: end
```

```
22:
23: describe Stack, "when push 3" do
24:   before do
25:     @stack = Stack.new
26:     @stack.push(3)
27:   end
28:
29:   it "should pop 3" do
30:     @stack.pop.should == 3
31:   end
32:
33:   it "should size 1" do
34:     @stack.size.should == 1
35:   end
36:
37:   it "should not be enpty" do
38:     @stack.should_not be_empty
39:   end
40:
41:   after do
42:     @stack = nil
43:   end
44: end
45:
46: describe Stack, "when push 3 and 5" do
47:   before do
48:     @stack = Stack.new
49:     @stack.push(3)
50:     @stack.push(5)
51:   end
52:
53:   it "should size 2" do
54:     @stack.size.should == 2
55:   end
56:
57:   after do
58:     @stack = nil
59:   end
60: end
61:
62: describe Stack, "when push 3 and 5, then pop" do
63:   before do
64:     @stack = Stack.new
```

```
65:     @stack.push(3)
66:     @stack.push(5)
67:     @stack.pop
68:   end
69:
70:   it "should size 1" do
71:     @stack.size.should == 1
72:   end
73:
74:   after do
75:     @stack = nil
76:   end
77: end
78:
```

スタッククラスのコード (stack.rb)

```
01:
02: class Stack
03:   class EmptyStackError < StandardError
04:   end
05:   def initialize
06:     @size = 0
07:   end
08:
09:   def empty?
10:     size == 0
11:   end
12:
13:   def push(val)
14:     @size += 1
15:   end
16:
17:   def pop
18:     raise EmptyStackError if empty?
19:     @size -= 1
20:     3
21:   end
22:
23:   def size
24:     @size
25:   end
26: end
```

Step 8

テストコード (stack_spec.rb)

```
01: require 'stack'
02:
03: describe Stack, "when empty" do
04:   before do
05:     @empty_stack = Stack.new
06:   end
07:
08:   it "should be empty" do
09:     @empty_stack.should be_empty
10:   end
11:
12:   it "should raise Stack::EmptyStackError" do
13:     lambda{
14:       @empty_stack.pop
15:     }.should raise_error Stack::EmptyStackError
16:   end
17:
18:   after do
19:     @empty_stack = nil
20:   end
21: end
22:
23: describe Stack, "when push 3" do
24:   before do
25:     @stack = Stack.new
26:     @stack.push(3)
27:   end
28:
29:   it "should pop 3" do
30:     @stack.pop.should == 3
31:   end
32:
33:   it "should size 1" do
34:     @stack.size.should == 1
35:   end
36:
37:   it "should not be enpty" do
38:     @stack.should_not be_empty
39:   end
40:
```

```
41:   after do
42:     @stack = nil
43:   end
44: end
45:
46: describe Stack, "when push 3 and 5" do
47:   before do
48:     @stack = Stack.new
49:     @stack.push(3)
50:     @stack.push(5)
51:   end
52:
53:   it "should size 2" do
54:     @stack.size.should == 2
55:   end
56:
57:   it "should pop 5" do
58:     @stack.pop.should == 5
59:   end
60:
61:   after do
62:     @stack = nil
63:   end
64: end
65:
66: describe Stack, "when push 3 and 5, then pop" do
67:   before do
68:     @stack = Stack.new
69:     @stack.push(3)
70:     @stack.push(5)
71:     @stack.pop
72:   end
73:
74:   it "should size 1" do
75:     @stack.size.should == 1
76:   end
77:
78:   after do
79:     @stack = nil
80:   end
81: end
82:
```

スタッククラスのコード (stack.rb)

```
01:
02: class Stack
03:   class EmptyStackError < StandardError
04:   end
05:   def initialize
06:     @size = 0
07:     @stack = []
08:   end
09:
10:   def empty?
11:     size == 0
12:   end
13:
14:   def push(val)
15:     @size += 1
16:     @stack.push(val)
17:   end
18:
19:   def pop
20:     raise EmptyStackError if empty?
21:     @size -= 1
22:     @stack.pop
23:   end
24:
25:   def size
26:     @size
27:   end
28: end
```