

CIS3260 (Software Design IV)

Assignment 1 - Tests

Rabia Qureshi (1046427)

Due date: October 4, 2022 @ 11:30am

Table of Contents

Use Case Tests

[Test 1 \(UC1\): Choosing Who Makes the First Move](#)

[Test 2 \(UC2\): Advancing in Spots on a Game Board](#)

[Test 3 \(UC3\): Getting Two Turns in a Row](#)

[Test 4 \(UC4\): Breaking a "Tie"](#)

Untested Method Tests

[Test 5: Coin Methods](#)

[Test 6: Die Methods](#)

[Test 7: Player Methods](#)

[Test 8: Bag Methods](#)

[Test 9: Hand Methods](#)

Use Case Tests

Test 1 (UC1): Choosing Who Makes the First Move

```
Testing Use Case 1: Choosing who makes the first move
Creating 'Player 1', 'Player 2', and 'Player 3'
  Player names EXPECTED: Player 1, Player 2, Player 3; RECEIVED: Player 1, Player 2, Player 3
Creating a quarter for each player
  Coin 1 description EXPECTED: { :sides=>2, :up=>nil, :item=>:coin, :denomination=>0.25 } RECEIVED: { :sides=>2, :up=>nil, :item=>:coin, :denomination=>0.25 }
  Coin 2 description EXPECTED: { :sides=>2, :up=>nil, :item=>:coin, :denomination=>0.25 } RECEIVED: { :sides=>2, :up=>nil, :item=>:coin, :denomination=>0.25 }
  Coin 3 description EXPECTED: { :sides=>2, :up=>nil, :item=>:coin, :denomination=>0.25 } RECEIVED: { :sides=>2, :up=>nil, :item=>:coin, :denomination=>0.25 }
Putting each player's coin in their bag
  Num randomizers in Player 1's bag EXPECTED 1 RECEIVED: 1
  Num randomizers in Player 2's bag EXPECTED 1 RECEIVED: 1
  Num randomizers in Player 3's bag EXPECTED 1 RECEIVED: 1
Transferring each player's coin from their bag to their cup
  Contents in Player 1's bag EXPECTED [] RECEIVED: []
  Num randomizers in Player 1's cup EXPECTED 1 RECEIVED: 1
  Contents in Player 2's bag EXPECTED [] RECEIVED: []
  Num randomizers in Player 2's cup EXPECTED 1 RECEIVED: 1
  Contents in Player 3's bag EXPECTED [] RECEIVED: []
  Num randomizers in Player 3's cup EXPECTED 1 RECEIVED: 1
Flipping each coin three times
  Num flips for coin 1 EXPECTED 3 RECEIVED: 3
  Num flips for coin 2 EXPECTED 3 RECEIVED: 3
  Num flips for coin 3 EXPECTED 3 RECEIVED: 3
  Coin 1 sideup EXPECTED H or T RECEIVED: H
  Coin 2 sideup EXPECTED H or T RECEIVED: H
  Coin 3 sideup EXPECTED H or T RECEIVED: T
Getting results of all coin flips
  Player 1 tally results: [1, 1, 1]
  Player 2 tally results: [1, 0, 1]
  Player 3 tally results: [0, 1, 0]
Clearing all tally results
  Player 1's tally EXPECTED [] RECEIVED: []
  Player 2's tally EXPECTED [] RECEIVED: []
  Player 3's tally EXPECTED [] RECEIVED: []
.
Finished in 0.0095246 seconds.
-----
1 tests, 24 assertions, 0 failures, 0 errors, 0 pendings, 0 omissions, 0 notifications
100% passed
```

Purpose: Test whether API users can create players and coins, store those coins in players' bags, transfer those coins from bags to cups, and "throw" those cups multiple times to retrieve the tally results of those throws based on a given description.

Methods used:

- Player.store()
- RandomizerContainer.count()
- Player.load()
 - Calls Cup.load() and RandomizerContainer.select()
 - Cup.load() makes use of RandomizerContainer.move_all()
- Player.throw()
 - Calls Cup.throw()
 - Makes use of Randomizer.randomize()
- Randomizer.calls()
- Coin.sideup()
 - Inherits from Randomizer.result()
- Player.tally()
 - Calls Results.description()
- Player.clear()

Test 2 (UC2): Advancing in Spots on a Game Board

```
Testing Use Case 2: Advancing in Spots on a Game Board
Creating 'Player 1' and 'Player 2'
  Player names EXPECTED: Player 1, Player 2; RECEIVED: Player 1, Player 2
Creating two dice for each player
Putting each player's dice in their hand
  Player 1's dice attributes EXPECTED: {:sides=>6, :up=>nil, :item=>:die, :colour=>:yellow}; RECEIVED: {:sides=>6, :up=>nil, :item=>:die, :colour=>:yellow}
  Num randomizers in Player 1's hand EXPECTED: 2; RECEIVED: 2
  Player 2's dice attributes EXPECTED: {:sides=>6, :up=>nil, :item=>:die, :colour=>:blue}; RECEIVED: {:sides=>6, :up=>nil, :item=>:die, :colour=>:blue}
  Num randomizers in Player 2's hand EXPECTED: 2; RECEIVED: 2
Putting each player's dice in their bag
  Num randomizers in Player 1's hand EXPECTED: 0; RECEIVED: 0
  Num randomizers in Player 2's hand EXPECTED: 0; RECEIVED: 0
  Num randomizers in Player 1's bag EXPECTED: 2; RECEIVED: 2
  Num randomizers in Player 2's bag EXPECTED: 2; RECEIVED: 2
  Sideup values of dice in Player 1's bag are nil EXPECTED: true; RECEIVED: true
  Sideup values of dice in Player 2's bag are nil EXPECTED: true; RECEIVED: true
Transferring each player's dice from their bag to their cup
  Num randomizers in Player 1's bag EXPECTED: 0; RECEIVED: 0
  Num randomizers in Player 2's bag EXPECTED: 0; RECEIVED: 0
  Num randomizers in Player 1's cup EXPECTED: 2; RECEIVED: 2
  Num randomizers in Player 2's cup EXPECTED: 2; RECEIVED: 2
Rolling dice
  Num randomizations performed on each die is 1 EXPECTED: true; RECEIVED: true
  Sideup values of Player 1's dice are not out of range EXPECTED: true; RECEIVED: true
  Sideup values of Player 2's dice are not out of range EXPECTED: true; RECEIVED: true
Getting Player 1 and Player 2's dice results
  player1.sum({}) EXPECTED: [7] RECEIVED: [7]
  player2.sum({}) EXPECTED: [9] RECEIVED: [9]
Finished in 0.0078987 seconds.

1 tests, 28 assertions, 0 failures, 0 errors, 0 pendings, 0 omissions, 0 notifications
100% passed
```

Purpose: Test whether API users can create players and dice, store those dice in their hands and transfer them to their bags and finally to their cups, “throw” those cups, and retrieve the sum of the side-up values of their thrown dice after one throw.

Methods used:

- RandomizerContainer.store_all()
- Player.move_all()
 - Bag.move_all()
 - Inherits from RandomizerContainer.move_all() and calls Randomizer.reset()
- RandomizerContainer.count()
- Die.sideup()
 - Inherits from Randomizer.result()
- Player.load()
 - Calls Cup.load() and RandomizerContainer.select()
 - Cup.load() calls RandomizerContainer.move_all()
- Player.throw()
 - Calls Cup.throw()
 - Calls Randomizer.randomize()
- Player.sum()

Test 3 (UC3): Getting Two Turns in a Row

```
Testing Use Case 3: Getting two turns in a row
Creating 'Player 1' and 'Player 2'
  Player names EXPECTED: Player 1, Player 2; RECEIVED: Player 1, Player 2
Creating two dice for Player 1 to store in bag
  Die description EXPECTED: {:sides=>6, :up=>nil, :item=>:die, :colour=>:yellow} RECEIVED: {:sides=>6, :up=>nil, :item=>:die, :colour=>:yellow}
Putting die in Player 1's hand
  Num randomizers in Player 1's hand EXPECTED: 1; RECEIVED: 1
Putting die in Player 1's bag
  Num randomizers in Player 1's hand EXPECTED: 0; RECEIVED: 0
  Num randomizers in Player 1's bag EXPECTED: 1; RECEIVED: 1
Transferring the die from Player 1's bag to their cup
  Num randomizers in Player 1's bag EXPECTED: 0; RECEIVED: 0
  Num randomizers in Player 1's cup EXPECTED: 1; RECEIVED: 1
Rolling die
  Num die rolls EXPECTED: 1; RECEIVED: 1
  Sideup value of die is not out of range EXPECTED: true; RECEIVED: true
  Player 1's roll result: 6 out of 6
Player 1 gets another turn
Rolling die
  Num die rolls EXPECTED: 2; RECEIVED: 2
  Sideup value of die is not out of range EXPECTED: true; RECEIVED: true
  Player 1's roll result: 6 out of 6
Player 1 gets another turn
Rolling die
  Num die rolls EXPECTED: 3; RECEIVED: 3
  Sideup value of die is not out of range EXPECTED: true; RECEIVED: true
  Player 1's roll result: 3 out of 6
Player 1's turn is over
Putting die in Player 2's hand
  Num randomizers in Player 1's cup EXPECTED: 0; RECEIVED: 0
  Num randomizers in Player 2's hand EXPECTED: 1; RECEIVED: 1
*
Finished in 0.007664 seconds.
1 tests, 16 assertions, 0 failures, 0 errors, 0 pendings, 0 omissions, 0 notifications
100% passed
```

Purpose: Test whether API users can create players and a shared die that is rolled and passed onto the opponent if the player does not get the highest possible side-up value.

Methods used:

- RandomizerContainer.store()
- Player.move_all()
 - Bag.move_all()
 - Inherits from RandomizerContainer.move_all() and calls Randomizer.reset()
- RandomizerContainer.count()
- Player.load()
 - Calls Cup.load() and RandomizerContainer.select()
 - Cup.load() calls RandomizerContainer.move_all()
- Player.throw()
 - Calls Cup.throw()
 - Calls Randomizer.randomize()
- Randomizer.calls()
- Die.sideup()
 - Inherits from Randomizer.result()
- Cup.empty()
- Hand.empty()
- Player.store()

Test 4 (UC4): Breaking a "Tie"

```
Testing Use Case 4: Breaking a tie
Creating 'Player 1' and 'Player 2'
  Player names EXPECTED: Player 1, Player 2; RECEIVED: Player 1, Player 2
Creating two dice for Player 1 to store in bag
  Die 1 description EXPECTED: {:sides=>5, :up=>nil, :item=>:die, :colour=>:green} RECEIVED: {:sides=>5, :up=>nil, :item=>:die, :colour=>:green}
  Die 2 description EXPECTED: {:sides=>5, :up=>nil, :item=>:die, :colour=>:blue} RECEIVED: {:sides=>5, :up=>nil, :item=>:die, :colour=>:blue}
  Die 3 description EXPECTED: {:sides=>5, :up=>nil, :item=>:die, :colour=>:red} RECEIVED: {:sides=>5, :up=>nil, :item=>:die, :colour=>:red}
  Die 4 description EXPECTED: {:sides=>5, :up=>nil, :item=>:die, :colour=>:yellow} RECEIVED: {:sides=>5, :up=>nil, :item=>:die, :colour=>:yellow}
Putting dice in Player 1's hand
  Num randomizers in Player 1's hand EXPECTED: 4; RECEIVED: 4
Placing dice in Player 1's bag
  Num randomizers in Player 1's hand EXPECTED: 0; RECEIVED: 0
  Num randomizers in Player 1's bag EXPECTED: 4; RECEIVED: 4
Randomly picking and placing one die in Player 1's cup
  Num randomizers in Player 1's bag EXPECTED: 3; RECEIVED: 3
  Num randomizers in Player 1's cup EXPECTED: 1; RECEIVED: 1
Desired die is yellow and lands on 3
  Player 1's die is desired colour
Rolling die
  Num die rolls EXPECTED: 1; RECEIVED: 1
  Sideup value of Player 1's die is not out of range EXPECTED: true; RECEIVED: true
  Player 1's roll result: 2
Player 1's die did not match desired description
  Player 1's die: {:sides=>5, :up=>2, :item=>:die, :colour=>:yellow}
Player 1's turn is over
Player 1 places die back in shared bag for Player 2's turn
  Num randomizers in Player 1's cup EXPECTED: 0; RECEIVED: 0
  Player 1 and Player 2 are sharing a bag EXPECTED: true; RECEIVED: true
  Num randomizers in shared bag EXPECTED: 4; RECEIVED: 4
Finished in 0.0078376 seconds.
-----
1 tests, 17 assertions, 0 failures, 0 errors, 0 pendings, 0 omissions, 0 notifications
100% passed
```

Purpose: Test whether API users can create players with a shared bag that contains dice, one of which is rolled by a player in hopes of meeting certain criteria to break a tie and gain an advantage over the opponent.

Methods used:

- RandomizerContainer.store_all()
- RandomizerContainer.count()
- Player.move_all()
 - Bag.move_all()
 - Inherits from RandomizerContainer.move_all() and calls Randomizer.reset()
- Player.load()
 - Calls Cup.load() and RandomizerContainer.select()
 - Cup.load() calls RandomizerContainer.move_all()
- Player.throw()
 - Calls Cup.throw()
 - Calls Randomizer.randomize()
- Randomizer.calls()
- Die.sideup()
- Die.colour()
- Player.replace()

Untested Method Tests

Test 5: Coin Methods

```
Testing unused Coin class methods
  Before flipping coin...
    coin.sideup() EXPECTED: nil; RECEIVED: nil
    coin.calls() EXPECTED: 0; RECEIVED: 0
    coin.denomination() EXPECTED: 0.25; RECEIVED: 0.25
    {:sides=>2, :up=>nil, :item=>:coin, :denomination=>0.25}
  Flipping coin...
    coin.sideup() EXPECTED: H or T; RECEIVED: T
    coin.calls() EXPECTED: 1; RECEIVED: 1
    {:sides=>2, :up=>:T, :item=>:coin, :denomination=>0.25}
.
Finished in 0.0019713 seconds.
-----
1 tests, 5 assertions, 0 failures, 0 errors, 0 pendings, 0 omissions, 0 notifications
100% passed
-----
```

Untested methods used: Coin.denomination(), Coin.flip()

Test 6: Die Methods

```
Testing unused Die class methods
  Before rolling die...
    die.sideup() EXPECTED: nil; RECEIVED: nil
    die.sides() EXPECTED: 6; RECEIVED: 6
    die.colour() EXPECTED: :red; RECEIVED: red
    die.calls() EXPECTED: 0; RECEIVED: 0
    {:sides=>6, :up=>nil, :item=>:die, :colour=>:red}
  Rolling die...
    die.sideup() EXPECTED: 1-6; RECEIVED: 3
    die.calls() EXPECTED: 1; RECEIVED: 1
    {:sides=>6, :up=>3, :item=>:die, :colour=>:red}
.
Finished in 0.0031464 seconds.
-----
1 tests, 6 assertions, 0 failures, 0 errors, 0 pendings, 0 omissions, 0 notifications
100% passed
-----
```

Untested methods used: Die.sides(), Die.roll()

Test 7: Player Methods

```
Testing Player class methods
Creating a player
Creating two quarters for player
Putting coins in Player's bag
Transferring Player's coins from their bag to their cup
Flipping each coin three times
  Flipping coins...
    {:sides=>2, :up=>:H, :item=>:coin, :denomination=>0.25}
    {:sides=>2, :up=>:T, :item=>:coin, :denomination=>0.25}
  Flipping coins...
    {:sides=>2, :up=>:T, :item=>:coin, :denomination=>0.25}
    {:sides=>2, :up=>:T, :item=>:coin, :denomination=>0.25}
  Flipping coins...
    {:sides=>2, :up=>:T, :item=>:coin, :denomination=>0.25}
    {:sides=>2, :up=>:T, :item=>:coin, :denomination=>0.25}
Getting results of all coin flips
  player.results({}, 0) EXPECTED: [:T, :T] RECEIVED: [:T, :T]
.
Finished in 0.0042931 seconds.
-----
1 tests, 1 assertions, 0 failures, 0 errors, 0 pendings, 0 omissions, 0 notifications
100% passed
-----
```

Untested methods used: Player.results()

Test 8: Bag Methods

```
Testing Bag class methods
Creating a player
Creating two quarters for player and storing in 'coins' array
Putting coins in Player's bag
  Num randomizers in bag EXPECTED: 2 RECEIVED: 2
  Num randomizers in 'coins' array EXPECTED: 2 RECEIVED: 2
Emptying Player's bag
  Num randomizers in bag EXPECTED: 0 RECEIVED: 0
.
Finished in 0.0029274 seconds.
-----
1 tests, 3 assertions, 0 failures, 0 errors, 0 pendings, 0 omissions, 0 notifications
100% passed
-----
```

Untested methods used: Bag.store_all(), Bag.empty()

Test 9: Hand Methods

```
Testing Hand class methods
  Creating a player
  Creating two quarters for player and storing in 'coins' array
  Creating a hand
  Storing coins in hand
    Num randomizers in hand EXPECTED: 2 RECEIVED: 2
  Removing last added coin from hand
    Num randomizers in hand EXPECTED: 1 RECEIVED: 1
    Removed coin description EXPECTED: {:sides=>2, :up=>nil, :item=>:coin, :denomination=>2} RECEIVED: {:sides=>2, :up=>nil, :item=>:coin, :denomination=>2}
.
Finished in 0.0029935 seconds.
-----
1 tests, 3 assertions, 0 failures, 0 errors, 0 pendings, 0 omissions, 0 notifications
100% passed
-----
```

Untested methods used: Hand.next()