Tracing & Hypothesis

Bug 4: No SPADE is ever rolled or guessed

1   Tracing

From Section 3 above, it is noticed that the DiceValue is determined by the *roll()* method:

```java
public class Dice {

    private DiceValue value;

    public Dice() {
        value =  DiceValue.getRandom();
    }

    public DiceValue getValue() {
        return value;
    }

    public DiceValue roll() {
        // return DiceValue.getRandom();
        value = DiceValue.getRandom();
        return value;
    }

    public String toString() {
        return value.toString();
    }
}
```

It is also noticed that the pick value is also from *getRandom()* method:

```java
while (player.balanceExceedsLimitBy(bet) && player.getBalance() < 200)
{
    turn++;
    DiceValue pick = DiceValue.getRandom();
```

As the method call the *getRandom()* method from **DiceValue.java**, the problem is probably from there:

```java
public static DiceValue getRandom() {
    int random = RANDOM.nextInt(DiceValue.SPADE.ordinal());
    return values()[random];
}
```

It is then noticed that the value of RANDOM is from the Math class Random, but the *nextInt()* calls the value of enum with the ordinal value as SPADE. As from the enum declaration, the *nextInt()* method will call randomly a number from ordinal value 0 to ordinal value of SPADE as 5. So SPADE never appears in any Random method.

```java
public enum DiceValue {
    CROWN, ANCHOR, HEART, DIAMOND, CLUB, SPADE;
```
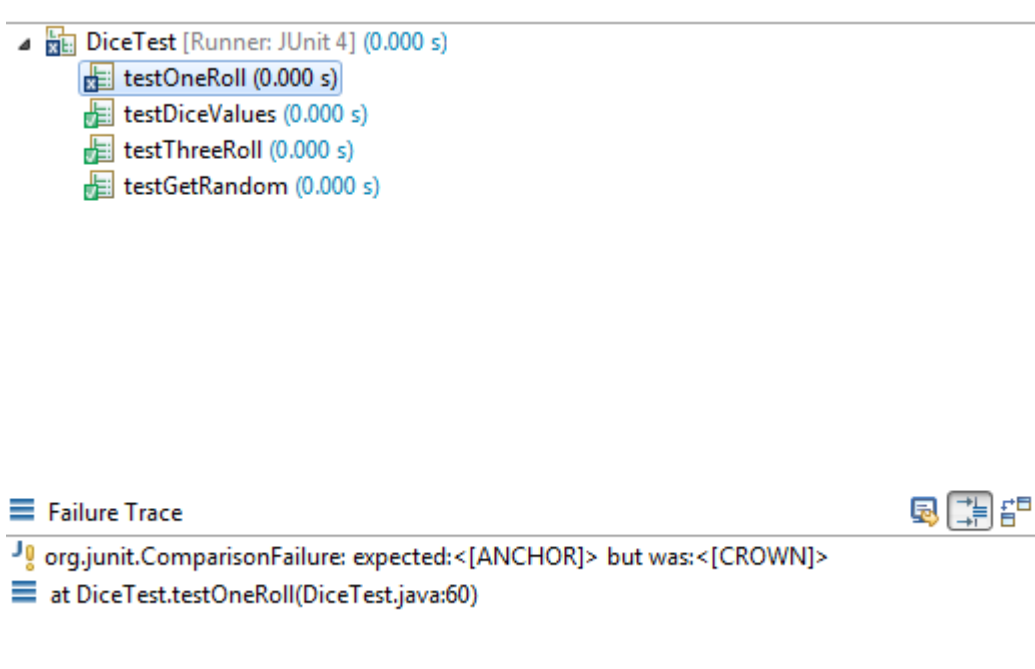
2 Hypotheses

The hypotheses are:

- (1): As both pick and roll are decided randomly, this cause the issue that SPADE never appears in pick and roll.
- (2): SPADE is never called due to the failure of logic in *nextIn()* of *getRandom()* method.

The testing of the hypothesis is done by the result of the DiceTest with one roll:

```
@Test
public void testOneRoll() {
    System.out.println("\nTest Dice roll values ");
    Dice diceTest = new Dice();
    String valueTest = diceTest.roll().toString();
    System.out.println("Expected: " + valueTest + " | Actual result: " + diceTest.getValue().toString());
    assertEquals(valueTest, diceTest.getValue().toString());
}
```

⊿ ⊞ DiceTest [Runner: JUnit 4] (0.000 s)
    ⊞ testOneRoll (0.000 s)
    ⊞ testDiceValues (0.000 s)
    ⊞ testThreeRoll (0.000 s)
    ⊞ testGetRandom (0.000 s)

☰ Failure Trace

org.junit.ComparisonFailure: expected:<[ANCHOR]> but was:<[CROWN]>
☰ at DiceTest.testOneRoll(DiceTest.java:60)

The testing of the hypothesis can also be done by replicating the bugs if changing the codes:

The first is changing the position of CLUB and SPACE in the enum declaration, so the ordinal value of SPACE will be 4. If the hypotheses are correct, CLUB and SPACE will not appear.

```
public enum DiceValue {
    CROWN, ANCHOR, HEART, DIAMOND, SPADE, CLUB;
```

The console shows that no SPADE or CLUB. The unit test also confirms that there are errors.

```
Start Game
Fred starts with balance 100, limit 0
Turn 1: Fred bet 5 on ANCHOR
Rolled HEART, HEART, HEART
Fred lost, balance now 95

Turn 2: Fred bet 5 on HEART
Rolled DIAMOND, CROWN, CROWN
Fred lost, balance now 90

Turn 3: Fred bet 5 on HEART
Rolled CROWN, DIAMOND, HEART
Fred won 5, balance now 95

Turn 4: Fred bet 5 on CROWN
Rolled HEART, CROWN, DIAMOND
Fred won 5, balance now 100

Turn 5: Fred bet 5 on HEART
Rolled ANCHOR, ANCHOR, CROWN
Fred lost, balance now 95

Turn 6: Fred bet 5 on ANCHOR
Rolled DIAMOND, CROWN, DIAMOND
Fred lost, balance now 90
```
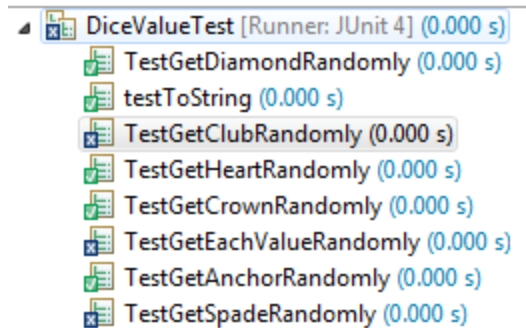
▲ DiceValueTest [Runner: JUnit 4] (0.000 s)
    TestGetDiamondRandomly (0.000 s)
    testToString (0.000 s)
    TestGetClubRandomly (0.000 s)
    TestGetHeartRandomly (0.000 s)
    TestGetCrownRandomly (0.000 s)
    TestGetEachValueRandomly (0.000 s)
    TestGetAnchorRandomly (0.000 s)
    TestGetSpadeRandomly (0.000 s)

The second is to change the SPADE in the *nextInt* into HEART (ordinal value = 2). If the hypotheses are correct, only CROWN or ANCHOR will appear.

```java
public static DiceValue getRandom() {
    int random = RANDOM.nextInt(DiceValue.HEART.ordinal());
    return values()[random];
}
```

The console confirms the error:

```
Start Game
Fred starts with balance 100, limit 0
Turn 1: Fred bet 5 on CROWN
Rolled CROWN, ANCHOR, ANCHOR
Fred won 5, balance now 105

Turn 2: Fred bet 5 on CROWN
Rolled CROWN, CROWN, CROWN
Fred won 15, balance now 120

Turn 3: Fred bet 5 on ANCHOR
Rolled ANCHOR, CROWN, ANCHOR
Fred won 10, balance now 130

Turn 4: Fred bet 5 on ANCHOR
Rolled CROWN, CROWN, CROWN
Fred lost, balance now 125

Turn 5: Fred bet 5 on CROWN
Rolled ANCHOR, ANCHOR, ANCHOR
Fred lost, balance now 120

Turn 6: Fred bet 5 on ANCHOR
Rolled ANCHOR, CROWN, ANCHOR
Fred won 10, balance now 130

Turn 7: Fred bet 5 on ANCHOR
Rolled ANCHOR, ANCHOR, ANCHOR
Fred won 15, balance now 145
```