BUG03 Debug Log

# Bug:

**Bug 3:** Odds in the game do not appear to be correct.  
Crown and Anchor games have an approximate 8% bias to the house. So the win : (win+lose) ratio  
should approximately equal 0.42. This does not appear to be the case.

# Assumptions:

* None

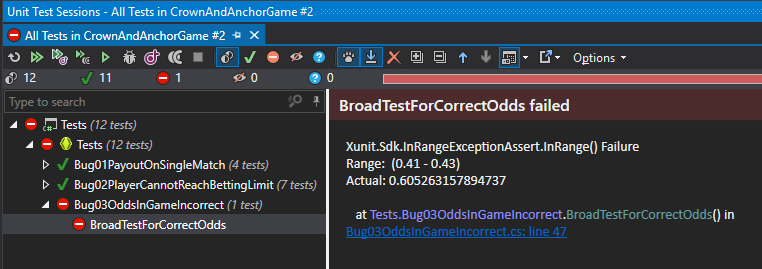
# Produce UAT Tests for each Bug

Done

# Changes to Base Code

A slight change to the ReadLine in the Program.cs so it isn’t inside the Play100 for testing purposes.

# Produce a broad Unit Test for each bug

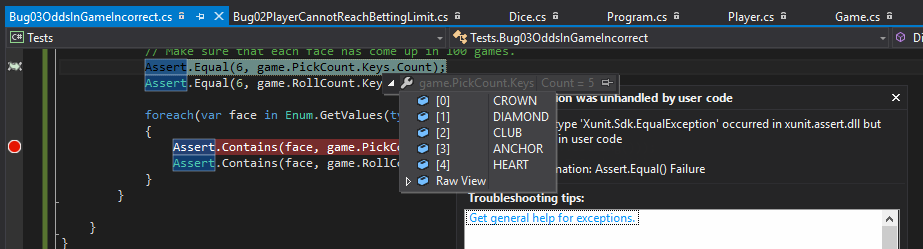


Unit test created – runs through 100 games and keeps track of the ratio, checks if it is within a range.

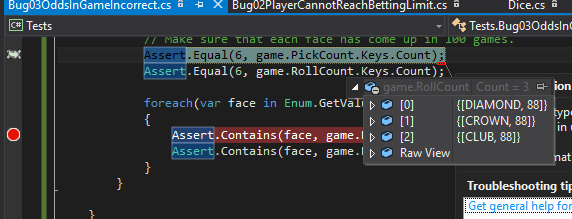
# See if there are any obvious additional Tests that I can introduce to cover side-effects

We already noted earlier in the discoveries of Bug 1 that there was a lack of randomization at play, so we will test that there is a solid randomizing in the game.

Testing results in the following obvious issues on the test created:



Consistently, the SPADE face isn’t showing on the player’s picks.



The RollCount keys are only showing the same three faces rolled every time, and therefore need to be randomized.

# Finding the cause

## Run look at where the Unit Tests fail and trace that line of code and check object states at those times

## Manually step through the code, the stacktrace, and the object windows to see what is being set and where the bug is occurring

First we’ll look at the Pick randomization – we’re missing SPADE consistently when running the test.

We’re using the Play100Games, but it doesn’t actively affect the dice rolling for the player, and instead passes a reference to PlayGame.

while (player.balanceExceedsLimitBy(bet) && player.Balance < 200)

{

try

{

PlayRound(bet, game, player, pick, ref winCount, ref loseCount);

}

catch (ArgumentException e)

{

Console.WriteLine("{0}\n\n", e.Message);

}

pick = Dice.RandomValue;

PlayGame passes the pick down to the PlayRound method, but doesn’t accept anything back, and passes it as a value type and not a reference type, so it can’t be replaced there.

This leaves the one line in PlayGame: pick = Dice.RandomValue;

If we trace into the Dice.RandomValue field we see it is a getter that returns a new random face.

public static DiceValue RandomValue

{

get

{

return (DiceValue)VALUES.GetValue(RANDOM.Next(VALUES.Length-1));

}

}

The issue here is that it returns .Length – 1 which means that it won’t return all 6 values of the array.

I’ve added a test that runs the randomizer 100 times and tests if it returns all six faces:

[Fact]

public void DiceReturnsAllSixFaces()

{

var faces = new HashSet<DiceValue>();

for (int i = 0; i < 100; i++)

{

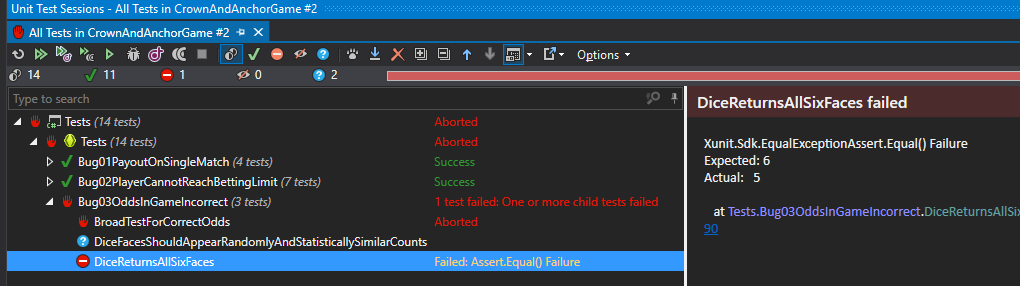
var face = Dice.RandomValue;

if (!faces.Contains(face)) faces.Add(face);

}

Assert.Equal(6, faces.Count);

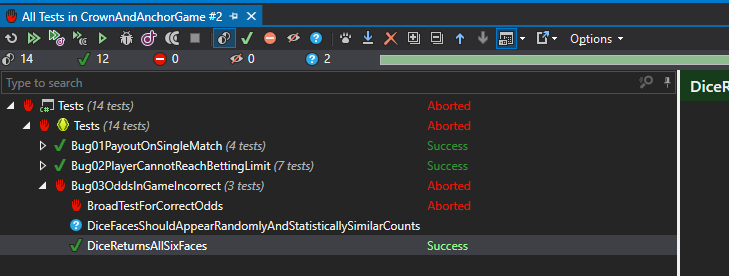
}



# Fix the Bug

Attacking first the fact the randomizer only picks between 5 faces, I am fixing that bug initially.

return (DiceValue)VALUES.GetValue(RANDOM.Next(VALUES.Length));



# Test for Side-Effects

# Additional Discoveries

# UAT Run