# Bug reporting and investigation

Refer to program output: Sample1.txt (“sample 1”), Sample2.txt (“sample 2”), and Sample3.txt (“sample3.txt”).

## One match pays out 0 instead of the expected equal to the bet

* Initially found through unit testing the Game class (GameTest: TestPlayRoundOneMatch())
* Also Bug 1 in initial bug report provided with assignment
* Also displayed in sample 1 on turn 1:

*Fred starts with balance 100, limit 0*

*Turn 1: Fred bet 5 on CROWN*

*Rolled HEART, HEART, CROWN*

*Fred won 5, balance now 100*

Fred got one match, so should win $5. He started with $100, so should have $105. But he only has $100 still.

## Two matches pays out 1X bet instead of the expected 2X bet

* Initially found through unit testing the Game class (GameTest: TestPlayRoundTwoMatches())
* Also displayed in the sample 1 on turn 3:

*Fred lost, balance now 95*

*Turn 3: Fred bet 5 on HEART*

*Rolled HEART, HEART, CROWN*

*Fred won 10, balance now 100*

Since Fred got two matches, he wins $10. He had $95 before, so, he should have $105 now. But he only has $100.

## Three matches pays out 2X bet instead of the expected 3X bet

* Initially found through unit testing the Game class (GameTest: TestPlayRoundThreeMatches())

## All rolls are the same in each run through

* In sample 1, all the rolls are Heart, Heart, Crown. In sample 2, all the rolls are Diamond, Diamond, Club. In sample 3, the rolls are all Anchor, Heart, Anchor.
* Hypothesis: when ‘roll’ is called on each dice, the result is not saved anywhere, so they stay at their initial values.
* How it should work:
  1. When the Dice is created, getRandom is called and the result saved in ‘value’, a DiceValue enum. This value would probably never get used, since ‘roll’ would be called before ever using it.
  2. During the playRound function, each die is “roll”ed which should call getRandom again and put a new DiceValue enum into value, which is then checked with getValue and compared to the DiceValue that was picked.

We are guessing that number 1 happens, but number 2 does not.

## Spades are never rolled

* Initially found during unit testing of DiceValue (after 100 rolls, Spades are not produced using the function getRandom).
* Hypothesis: As Spades are the highest ordinal in the enum DiceValue, this is probably an “out by 1” programming error. I.e. the getRandom function gets a random int up to, but not including Spades. The fix will probably be to add “+ 1” in the function.
  + Idea for test: change order of enum values (it would be poor practice to hardcode these values anywhere, so if this causes something else to break, can address the hardcoding then). If the value at the end of the list now doesn’t get rolled, then this would confirm the hypothesis (will repeat a large number of times to be certain).