# Scenario – Bug Reports

## Scenario Description

* This set of tests aims to replicate the reported bugs for the ‘Crown and Anchor’ game
* A single test will reliably replicate a single reported bug
* For each script a set of pre-conditions, post conditions, required data, required actions and expected results will be included, as well as any necessary tear-down steps.
* As much as possible acceptance criteria will be clear, unambiguous and quantifiable.

## Version Control

|  |  |  |  |
| --- | --- | --- | --- |
| Version # | Date | Author | Description |
| 0.1 | 20/09/2014 | Sam Stow | Initial Draft |

## Test Scripts

The following scripts will cover this scenario:

1. Bug – Game does not pay out at correct level
2. Bug – Player cannot reach betting limit
3. Bug - Odds in the game do not appear to be correct.
4. Bug – Dice seem to always roll the same after first roll.

## Use Case

* Gameplay

## Test Components/Requirements

This test scenario covers the following high-level test requirements (see scripts below for specific requirements covered by each test script):

* User wishes to play the Crown and Anchor game

## User Groups

* Players of the game

## Script 1: Bug – Game does not pay out at correct level

### Script Description

* This script covers the bug which occurs when a player wins on 1 match, the balance does not increase.

### Testing Requirements

This test concerns the following rules of game pay:

* A wager on a particular symbol shall win if the symbol appears on one or more of the uppermost face of the three dice and shall lose if the symbol does not appear.

|  |  |
| --- | --- |
| if the symbol appears on the uppermost face of 3 dice | 3 to 1 |
| if the symbol appears on the uppermost face of 2 dice | 2 to 1 |
| if the symbol appears on the uppermost face of 1 dice | 1 to 1 |

* Winning wagers shall be paid at odds not less than:-
* If a user therefore bets $5 on a particular symbol and it appears on only 1 matching dice, the balance of the user should increase by $5.

### Pre-conditions

* A user must have registered for gameplay and have a positive balance greater than the minimum bet. .
* A game must be initialized with 3 dice.

### Required Data

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test Data Table** | | | | | |
|  | **1** | **2** | **3** | **4** | **5** |
| A valid user | Name “Sam” | Starting balance 100 | Minimum balance limit  0 |  |  |
| Three new dice | No initialization values | Variable names d1, d2, d3 |  |  |  |
| Game | Initialised with dice d1, d2 & d3 |  |  |  |  |
| A bet | Pick “Crown” | Bet $5 |  |  |  |
|  |  |  |  |  |  |

### Post-conditions

* Post conditions are dependant on dice values returned. Require multiple runs to verify different post conditions.
* Cases
  + 0 matches (No ‘crown’ rolled) – result = -5, player balance decreases by bet amount ($5) – final balance $95
  + 1 match (1 ‘crown’ rolled) – result = 5, player balance increases by 1 x bet amount ($5) – final balance $105
  + 2 matches (2 ‘crowns’ rolled) – result = 10, player balance increases by 2 x bet amount ($10) – final balance $110
  + 3 matches (3 ‘crowns’ rolled) – result = 15, player balance increases by 3 x bet amount ($15) – final balance $120

### Script Steps

| **Step #** | **Test Action** | **Expected Results** | **Pass/ Fail** |
| --- | --- | --- | --- |
| 1 | Create player | Player exists with name “Sam”, balance 100, and limit 0 |  |
| 2 | Create dice | 3 new dice exist – d1, d2, d3 |  |
| 3 | Create game with dice | A new valid game exists |  |
| 4 | Play round (Pick ‘crown’, bet 5) | A result (winnings amount) and 3 new dice values |  |
| 5 | Check result (dice values and winnings) | Confirm winnings amount is correct based on dice values as follows:   * 0 crowns – Winnings = -5 * 1 crown – Winnings = 5 * 2 crowns – Winnings = 10 * 3 crowns – Winnings = 15 |  |
| 6 | Check player balance | Confirm player balance is adjusted by winnings amount |  |
| 7 | Repeat steps 4 & 5 until the case where only 1 match occurs and verify result | Confirm winnings for 1 match = 5 and player balance is increased by 5. |  |
| 8 |  |  |  |
| 9 |  |  |  |

### Test Execution

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Date/Time | Tester | Test ID | Test Phase | Status |
|  |  |  |  |  |

## Script 2: Bug – Player cannot reach betting limit

### Script Description

* This script covers the bug which occurs when a betting limit is set to 0, but the game ends with player still with 5 (dollars) remaining.

### Testing Requirements

This test concerns the rules where a player man continue to play until his balance is no longer above the betting limit (ie he/she has used all available funds above the betting limit)

### Pre-conditions

* A user must have registered for gameplay and have a positive balance greater than the minimum bet.
* A game must be initialized with 3 dice.

### Required Data

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test Data Table** | | | | | |
|  | **1** | **2** | **3** | **4** | **5** |
| A valid user | Name “Sam” | Starting balance 10 | Minimum balance limit  0 |  |  |
| Three new dice | No initialization values | Variable names d1, d2, d3 |  |  |  |
| Game | Initialised with dice d1, d2 & d3 |  |  |  |  |
| A bet | Pick “Crown” | Bet $5 |  |  |  |
|  |  |  |  |  |  |

### Post-conditions

* We expect gameplay to end when the players balance is the same as the betting limit, such that the player cannot place a bet without taking them under the betting limit.
* We therefore expect that the player balance is equal to zero in this case.

### Script Steps

| **Step #** | **Test Action** | **Expected Results** | **Pass/ Fail** |
| --- | --- | --- | --- |
| 1 | Create player | Player exists with name “Sam”, balance 10, and limit 0 |  |
| 2 | Create dice | 3 new dice exist – d1, d2, d3 |  |
| 3 | Create game with dice | A new valid game exists |  |
| 4 | Play round (Pick ‘crown’, bet 5) | A result (winnings amount) and 3 new dice values |  |
| 5 | Check result | Confirm winnings amount is correct based on dice values as follows:   * 0 crowns – Winnings = -5 * 1 crown – Winnings = 5 * 2 crowns – Winnings = 10 * 3 crowns – Winnings = 15 |  |
| 6 | Check player balance | Confirm player balance is adjusted by winnings amount |  |
| 7 | Repeat steps 4 & 5 until game play is ended | Confirm player balance is zero |  |
| 8 |  |  |  |
| 9 |  |  |  |

### Test Execution

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Date/Time | Tester | Test ID | Test Phase | Status |
|  |  |  |  |  |

## Script 3: Bug – Odds in the game do not appear to be correct.

### Script Description

* This script covers the bug which occurs at the end of gameplay when reviewing the win loss ratio. 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. Win rates appear to be either ~40% or ~60%.

### Testing Requirements

This test concerns the overall outcome of the game which should be 8% in favour of the house. Therefore the overall win/loss ratio after a sufficient number of rounds played should approximate 42%, or the player should only win 42 games out of every 100. Obviously a margin of error is allowable but this reduces as the number of games played increases.

### Pre-conditions

* A user must have registered for gameplay and have a positive balance greater than the minimum bet.
* A game must be initialized with 3 dice.

### Required Data

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test Data Table** | | | | | |
|  | **1** | **2** | **3** | **4** | **5** |
| A valid user | Name “Sam” | Starting balance 100 | Minimum balance limit  0 |  |  |
| Three new dice | No initialization values | Variable names d1, d2, d3 |  |  |  |
| Game | Initialised with dice d1, d2 & d3 |  |  |  |  |
| A bet | Pick “Crown” | Bet $5 |  |  |  |
| Number of games to play | Number of games = 100 |  |  |  |  |

### Post-conditions

* We expect that after 100 games played the overall win/loss ratio should approximate 42% with a small margin of error (~1%).
* We expect to obtain consistent results after running the test 5 times.

### Script Steps

| **Step #** | **Test Action** | **Expected Results** | **Pass/ Fail** |
| --- | --- | --- | --- |
| 1 | Create dice | 3 new dice exist – d1, d2, d3 |  |
| 2 | Create game with dice | A new valid game exists |  |
| 3 | Create player | Player exists with name “Sam”, balance 10, and limit 0 |  |
| 4 | Play round with specified player, bet and pick | A win or loss and associated balance adjustment as per game rules |  |
| 5 | Repeat step 4 until gameplay ends | Games ends |  |
| 6 | Repeat steps 3-5 x number of games (100) |  |  |
| 7 | Check win loss ratio | Win loss ratio ~42% +/- 1% |  |
| 8 | Repeat steps 1 – 7 times fives and verify consistent result. |  |  |
| 9 |  |  |  |

### Test Execution

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Date/Time | Tester | Test ID | Test Phase | Status |
|  |  |  |  |  |

## Script 4: Bug – Dice seem to always roll the same after first roll

### Script Description

* This script covers the bug where the values of dice rolled appear to remain the same after each round is played.

### Testing Requirements

This test concerns the rules where dice values are retrieved after playing a round of the game. After the first round, for each subsequent round the dice values retrieved should be random.

### Pre-conditions

* A user must have registered for gameplay and have a positive balance greater than the minimum bet.
* A game must be initialized with 3 dice.

### Required Data

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test Data Table** | | | | | |
|  | **1** | **2** | **3** | **4** | **5** |
| A valid user | Name “Sam” | Starting balance 10 | Minimum balance limit  0 |  |  |
| Three new dice | No initialization values | Variable names d1, d2, d3 |  |  |  |
| Game | Initialised with dice d1, d2 & d3 |  |  |  |  |
| A bet | Pick “Crown” | Bet $5 |  |  |  |
|  |  |  |  |  |  |

### Post-conditions

* We expect to be able to play through a game until the end.
* We expect the dice values after each round to vary randomly.

### Script Steps

| **Step #** | **Test Action** | **Expected Results** | **Pass/ Fail** |
| --- | --- | --- | --- |
| 1 | Create player | Player exists with name “Sam”, balance 10, and limit 0 |  |
| 2 | Create dice | 3 new dice exist – d1, d2, d3 |  |
| 1 | Create game with dice | A new valid game exists |  |
| 2 | Play round (Pick ‘crown’, bet 5) | A result (winnings amount) and 3 new dice values |  |
| 3 | Repeat step 2 until game play ends | Gameplay ends when user balance equals betting limit (zero) |  |
| 4 | Check dice values for rounds played | Dice values should vary randomly from round to round. |  |
| 5 | Check win/loss ratio | Win/loss ratio should be ~42%, with an acceptable margin of error (+/-10%) for a smaller number of rounds. |  |
| 6 |  |  |  |
| 7 |  |  |  |

### Test Execution

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Date/Time | Tester | Test ID | Test Phase | Status |
|  |  |  |  |  |