**TEST STRATEGY – Buffer Class**

# CLASS DIAGRAM

|  |
| --- |
| BUFFER |
| list : ArrayList  maxElements : int |
| Buffer()  Buffer( ArrayList<Multiple>, int)  getList() : ArrayList<Multiple  getMaxElements() : int  setList(ArrayList<Multiple)  setMaxElements(int) |

1. Create a buffer object with the default constructor

1.1 TEST DATA: maxElements = 5

list = new ArrayList

1.2 EXPECTED RESULT: maxElements = 5

list = new ArrayList

1.3 ACTUAL RESULT:

Actual result matches Expected result



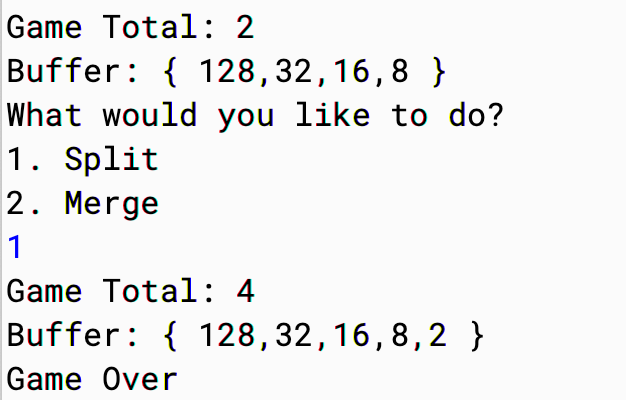
1. Functional Testing:

2.1.1 Definition of Case & Objective:

Game should stop and end if buffer size has reached its maximum value of 5. No more merge should be possible. The game should not allow to proceed further.

2.1.2 Actual Test: Manually tested the test case with the game.

2.1.3 Outcome:



Since 4 doesn’t match with any value in the buffer and the buffer already has 5 elements, meaning the new random number 4 will be added to the buffer which makes maxElement count as 6. Hence the game ends.

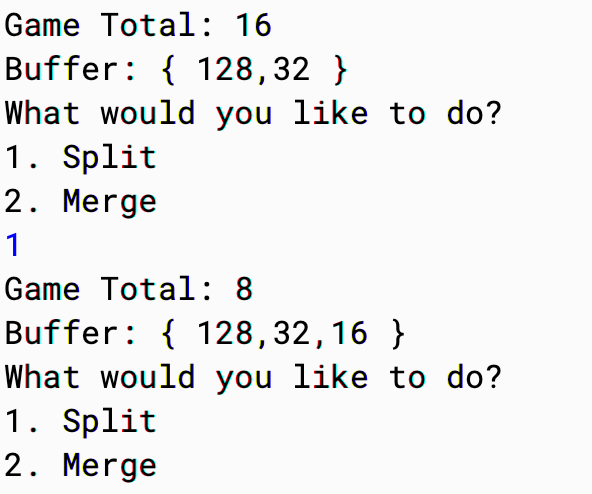
Hence test successful.

2.2.1 Definition of Case & Objective:

If the player plays split in a particular turn, the number should be added to the buffer.

2.2.2 Actual Test: Manually tested the test case with the game.

2.2.3 Outcome:



When split was played, 16 got added to the buffer before the next chance.

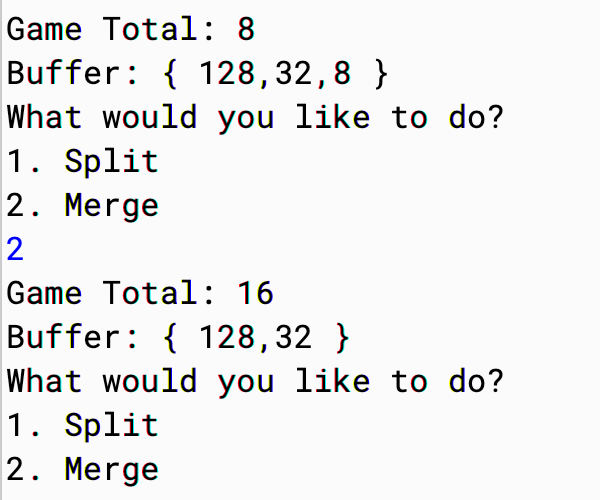
Hence test successful.

2.3.1 Definition of Case & Objective:

If the player plays merge in a particular turn, the number should be removed from the buffer and added to the game total.

2.3.2 Actual Test: Manually tested the test case with the game.

2.3.3 Outcome:



When merge was played, 8 got added to the game total and was subsequently removed from the buffer, before the next chance.

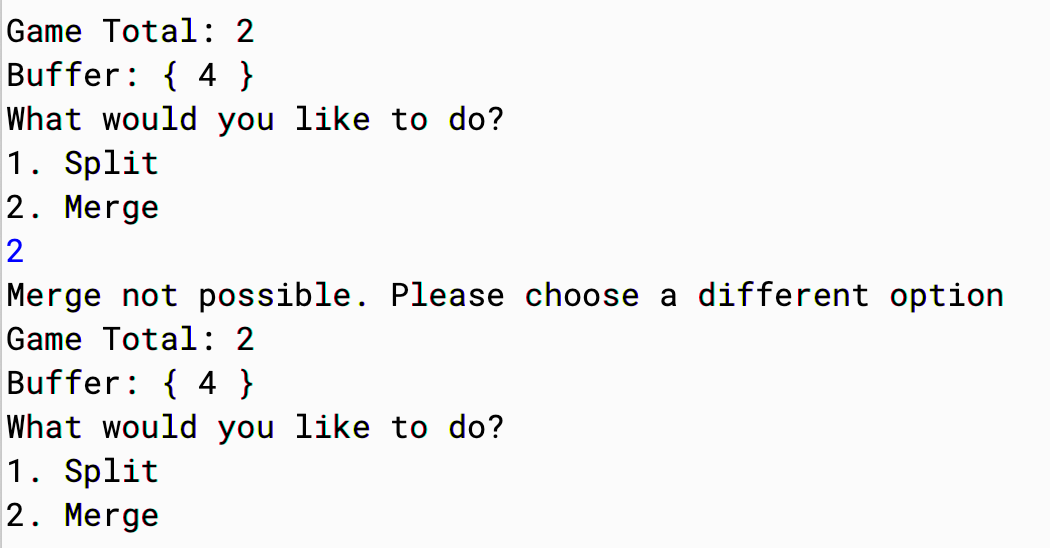
Hence test successful.

2.4.1 Definition of Case & Objective:

If the player plays merge in a particular turn, but the conditions to play merge doesn’t get satisfied (ie : Buffer doesn’t have a value equal to the game total), then game should not let it play that step.

2.4.2 Actual Test: Manually tested the test case with the game.

2.4.3 Outcome:



When merge was played even when condition wasn’t valid, the game showed a message and continued the game, giving the player instructions to choose another option.

Hence test successful.

2.5.1 Definition of Case & Objective:

If the player plays split in a particular turn, but the conditions to play merge doesn’t get satisfied, then game should not let it play that step.

2.5.2 Actual Test: Manually tested the test case with the game.

2.5.3 Outcome:



When split was played even when condition wasn’t valid, the game showed a message and continued the game, giving the player instructions to choose another option.

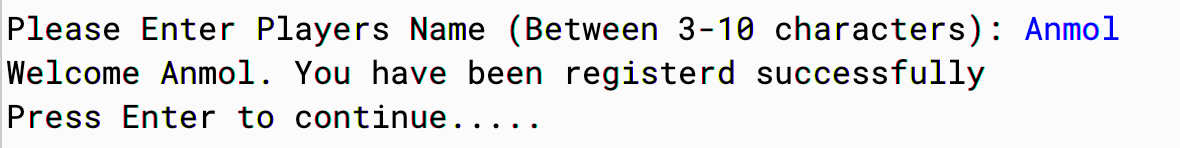
Hence test successful.

2.6.1 Definition of Case & Objective:

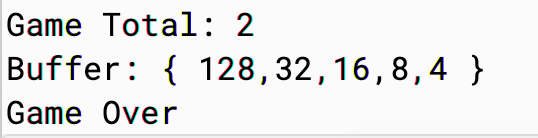
At the end of the game, the players name and the highest number in the buffer is displayed as the result of game in the output file.

2.6.2 Actual Test: Manually tested the test case with the game.

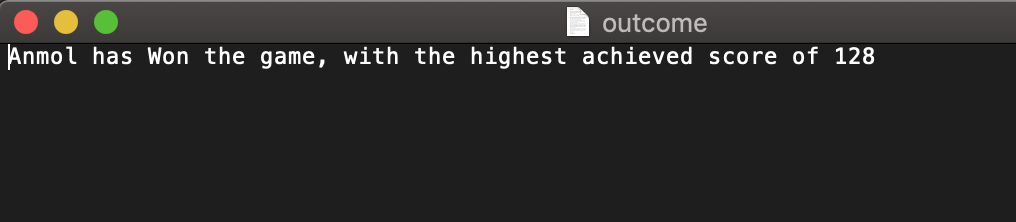
2.6.3 Outcome:



Entered Anmol as player name.



Values in the buffer and the game total variable at the end of the game indicated by the Game over printed at the end.



Output.txt file shows 128 as the highest score and the name of the player.

Hence test successful.

1. Mutator validation

-Set method for maxElemets is not needed as the value is hard coded as 5 and initiated in the constructor in buffer class

-Get method for maxElement is validated as the game doesn’t allow to proceed further when the buffer array has more than 5 elements. That validation is done in the code by getMaxElements() getter method.

TEST RESULT: