DATA STRUCTURE PROJECT REPORT

By:
ALLENXAVINZKY ADJIEWIBOWO
NIM: 2802467880
CLASS: L2AC

ANDRES WINSON
NIM: 2802501123
CLASS: L2AC

MAJOR: COMPUTER SCIENCE

BINUS INTERNATIONAL UNIVERSITY
JAKARTA
BATCH 2028

TABLE OF CONTENTS

CHAPTER 1 BACKGROUND 1
1.1 Poker 1
1.1.1 Betting 1
1.1.2 Raise 1
1.1.3 Check 1
1.1.4 Call 1
1.1.5 Fold 2
1.1.6 Game Stages 2
1.1.6.1 Prelop 2
1.1.6.2 Flop 2
1.1.6.3 Turn 2
1.1.6.4 River 2
1.1.6.5 Showdown 3
1.2 LinkedList 3
1.2.1 Types 3
1.2.1.1 Characteristics 3
1.3 ArrayList 3
CHAPTER 2 PROBLEM DESCRIPTION 4
2.1 Problem Description 4
CHAPTER 3 PROPOSED SOLUTION 4
3.1 Proposed Solution 4
3.2 Class Diagram 5
3.2.1 ArrayList Class Diagram 5
3.2 LinkedList Class Diagram 6
CHAPTER 4 TESTING AND CONCLUSION 7
4.1 Testing Plan 7
4.2 Results 7
4.2.1 ArrayList Results 7
4.2.2 LinkedList Results 10
4.3 Memory Usage Graphs 13
4.3.1 ArrayList Memory Usage Graph 13
4.3.2 LinkedList Memory Usage Graph 14
4.4 Conclusion 14
CHAPTER 5 TEAM WORKLOAD AND TASK DIVISION 15
REFERENCES 16
APPENDIX 17

CHAPTER 1 BACKGROUND

1.1 Poker

Traditionally, Poker is a gambling game, where 2 or more players are dealt 2 cards each at the start, the goal of the game is to bet your money on your cards in hopes that they beat the opponents card by value by comparing the value of your card and the opponent's cards as well as comparing the value if combined with the 5 cards that were dealt by the dealer on the table (Hellmuth, 2003 & Bicycle Cards).

1.1.1 Betting

The essence of Poker is betting, each round the players will be given a chance to bet their money onto their cards, each player is given 5 rounds of betting in a single game. Betting can be split into two, normal bet, which happens on all rounds except the Flop, and mandatory bet which happens only in the Flop stage, how big the mandatory bet be depends on the currency, but is generally a small amount to the total money you have (Hellmuth, 2003).

1.1.2 Raise

Raise is one of the actions that players can do during each betting session. What players when they raise is that they increase their bet by the desired amount, and what other players can do is to either call that bet or to raise it even higher with their bet or they could fold if they wished (Hellmuth, 2003).

1.1.3 Check

Check is one of the actions that players can choose to do during a betting session. When a player checks, they don't bet anything and wish to continue to the next round. A check can only be done if every other player checks, if one player raises their bet, the player who checked can't go to the next round and must either call, raise their bet higher, or fold to the bet (Hellmuth, 2003).

1.1.4 Call

Call is one of the options the players can choose to do in a betting session, though this option only appears if one of the players raises their bet. If a player calls the bet of another player, that means that the player that called raised their bet to match the bet of the player that raised (Hellmuth, 2003).

1.1.5 Fold

Fold is one of the options that players can choose to do in a betting session. When a player folds, they essentially surrender and go out of the game until the winner is decided and a new game is started (Hellmuth, 2003).

1.1.6 Game Stages

In Poker there are 5 recognized game rounds in a session, that being Preflop, Flop, Turn, River, and Showdown (Hellmuth, 2003).

1.1.6.1 Prelop

Preflop is the stage of the game where the dealer gives the players their two cards while the table cards aren't dealt yet. In the Preflop stage, the players are allowed to see their cards as well as raise their bet if they wish to, the betting session will stop once everyone calls on the current bet or everyone agrees to all in or fold (Hellmuth, 2003).

1.1.6.2 Flop

Flop is the stage where the second round of betting happens. In this stage, the dealer will deal 3 cards while the player may choose to either fold, continue to the next round without betting via checking, or raise the bet until everyone agrees to stop raising the bet via call (Hellmuth, 2003).

1.1.6.3 Turn

Turn is the third stage where the third round of betting happens. In this stage, the dealer will only deal one card, while the player may choose to either fold, continue to the next round without betting via checking, or raise the bet until everyone agrees to stop raising the bet via call (Hellmuth, 2003).

1.1.6.4 River

River is the fourth stage where the fourth round of betting happens. Same like in the Turn stage, the dealer will only deal one card, while the player may choose to either fold, continue to the next round without betting via checking, or raise the bet until everyone agrees to stop raising the bet via call (Hellmuth, 2003).

1.1.6.5 Showdown

Showdown is the final stage of a poker game. In this stage, the dealer will no longer deal any cards as all 5 cards had been revealed on the table, and the players will get a chance to either raise their bet, check, call, or fold, once everyone had done their choice, all of the player's cards will be revealed and the winner gets all the total money everyone had betted during the game. After this, the game will either continue with the surviving players that still have some money, or end when there are no more players that can continue (Hellmuth, 2003).

1.2 LinkedList

LinkedList is a data structure that helps store data in the actual memory of a program. How LinkedList works is that every data is contained in their own node, and each node has a second value which is either the address of the next node or a null value which means the node ends there (Jain, 2017; Cormen, 2009; Manber, 1989).

1.2.1 Types

LinkedLists is a data structure that has 3 variants which can be seen by what types of navigation are possible in it which are single LinkedList, Doubly LinkedList, and Circular LinkedList (Koffman, 2021 & Neso Academy, 2022).

1.2.1.1 Characteristics

A single LinkedList can be identified by its forward only navigation, which means that the iterator cannot go backwards in the list. However, a doubly LinkedList has both forward and backward navigation. Finally, a circular LinkedList has a node connecting the last element of the list with the first element of the list (Koffman, 2021 & Neso Academy, 2022).

1.3 ArrayList

Arraylist is a data structure that allows a program to store data in the memory. ArrayList is similar to an Array but unlike an Array, Arraylist allows for expandable storage. ArrayList also allows for easy modification in the form of insertion and deletion which Array doesn't allow (Devi, 2019; Geeks for Geeks, 2025; Rahman, 2016).

CHAPTER 2 PROBLEM DESCRIPTION

2.1 Problem Description

Poker has simple rules, there are players and there is the dealers, the players will bet their money on their cards while comparing their cards to the cards on the table dealt by the dealer (Hellmuth, 2003 & Bicycle Cards), but despite that, it is very different when porting the game into code, so we were curious on how to implement storing the cards, and later on curious on what would be the best Data Structure for the operations that will be required in the game, like discarding cards, inserting cards, and clearing decks. To code a poker game, there are 4 main functions that is needed to be implemented, those 4 being a Randomizer, Insertion, Deletion, and Storage, and thus we needed a data structure that supports a Randomizer, a method for Deletion and Insertion, as well as Storage, which narrows it down to LinkedList and ArrayList.

The testing of the most efficient data structure will be dictated by tracking the runtime speed and every instance of the memory usage in the “Table class”, which is a pseudo-facade class that filters all the needed methods from the other classes that handles the deck of the cards, the players’ attributes and methods, the hierarchy of the cards as well as its comparing methods, and the hand class which takes care of the players’ cards.

CHAPTER 3 PROPOSED SOLUTION

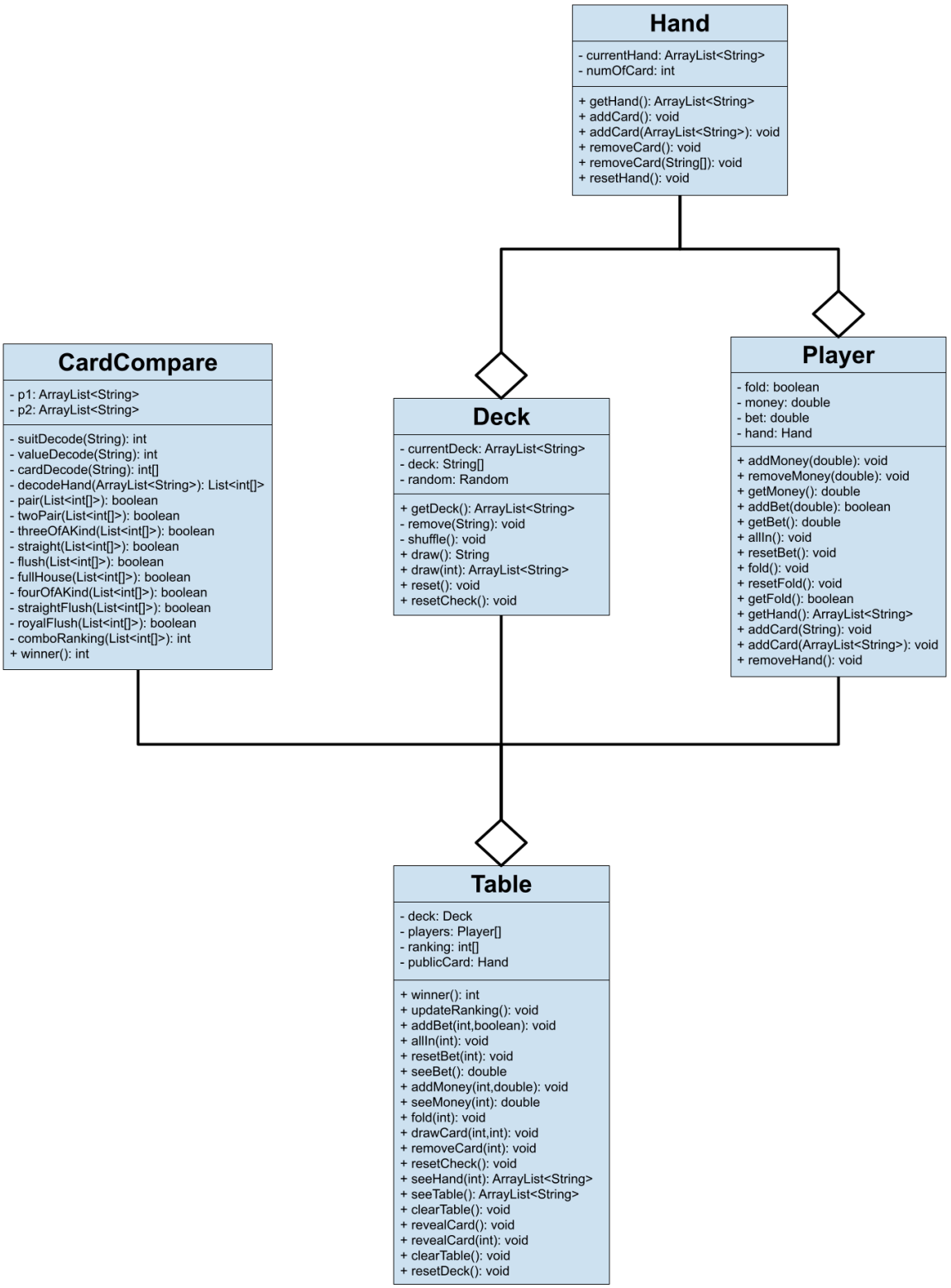
3.1 Proposed Solution

To solve the problem, we have decided to use 2 data structures which are LinkedList and ArrayList and to make it easier to compare we made to versions of the program, one using LinkedList and one using ArrayList, considering that both data structures have identical syntax, we were able to convert the ArrayList version to a LinkedList version without much alteration apart from changing every instance of ArrayList into LinkedList. The LinkedList and Arraylist data structures are used as follows:

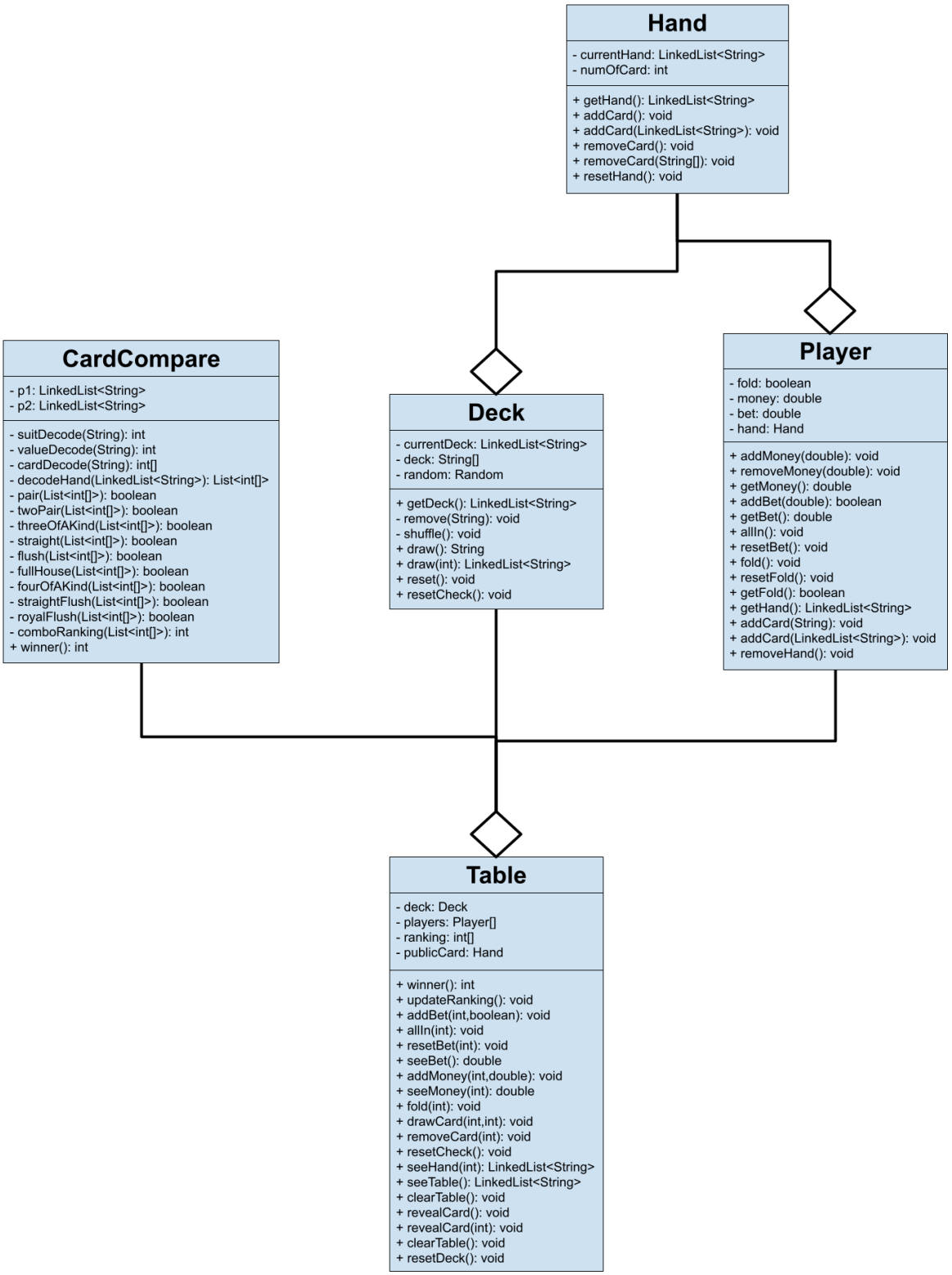
1. Deck storage
2. Insertion of cards into the player hand
3. Deletion of cards from the player hand.

3.2 Class Diagram

3.2.1 ArrayList Class Diagram



3.2 LinkedList Class Diagram



CHAPTER 4 TESTING AND CONCLUSION

4.1 Testing Plan

The testing of the programs will be done via running a special testing program which tests the memory usage and running time of the program. The independent variable will be the choices which you can input numbered from 1 to 3, where there is 3 types of choices, that is the main menu which consists of “Play”, numbered 1, and “Exit”, numbered 2, the round choices menu which consists of “Raise”, numbered 1, “Check”, numbered 2, and Fold, numbered 3, and finally the retry menu which consists of “Play Again”, numbered 1, and “Quit”, numbered 2

The tests will be conducted 5 times with 10 differing inputs for each data structure and the results will be averaged and compared to each of the data structure’s results. After everything is compiled, the data structure with the least amount of memory usage and running time will be dubbed the most efficient for the problem that it is being used for in the poker game.

4.2 Results

4.2.1 ArrayList Results

input	Trial 1	
	Total memory usage (B)	Time (Sec)
1,3,1	43896	2.930
1,2,2,2,2,1	543776	5.569
1,2,2,2,2,2,3,1	41040	6.738
1,2,2,2,2,2,2,2,2,2,1	551192	7.584
1,2,2,2,2,2,2,2,2,2,2,3,1	547144	7.560
1,2,2,2,2,2,2,2,2,2,2,2,2,2,1	42640	9.039
1,2,2,2,2,2,2,2,2,2,2,2,2,2,2,3,1	544552	10.381
1,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,1	544488	11.125
1,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,3,1	547248	10.175
1,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,1	43544	11.314

input	Trial 2	
	Total memory usage (B)	Time (Sec)
1,3,1	36992	2.648
1,2,2,2,2,1	41536	4.206
1,2,2,2,2,2,3,1	544184	5.399
1,2,2,2,2,2,2,2,2,2,1	42048	6.229
1,2,2,2,2,2,2,2,2,2,2,3,1	551272	6.635
1,2,2,2,2,2,2,2,2,2,2,2,2,2,1	41096	8.444
1,2,2,2,2,2,2,2,2,2,2,2,2,2,2,3,1	544872	8.678
1,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,1	41160	8.428
1,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,3,1	545704	9.584
1,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,1	42520	9.750

input	Trial 3	
	Total memory usage (B)	Time (Sec)
1,3,1	39056	2.196
1,2,2,2,2,1	40336	3.608
1,2,2,2,2,2,3,1	47832	5.014
1,2,2,2,2,2,2,2,2,2,1	40824	6.720
1,2,2,2,2,2,2,2,2,2,3,1	41144	7.595
1,2,2,2,2,2,2,2,2,2,2,2,2,1	544752	8.338
1,2,2,2,2,2,2,2,2,2,2,2,2,3,1	544488	10.181
1,2,2,2,2,2,2,2,2,2,2,2,2,2,2,1	42552	10.317
1,2,2,2,2,2,2,2,2,2,2,2,2,2,2,3,1	42344	11.207
1,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,1	42192	10.269

input	Trial 4	
	Total memory usage (B)	Time (Sec)
1,3,1	36664	2.252
1,2,2,2,2,1	40848	3.504
1,2,2,2,2,2,3,1	43192	5.094
1,2,2,2,2,2,2,2,2,2,1	41400	5.271
1,2,2,2,2,2,2,2,2,2,2,3,1	48968	6.847
1,2,2,2,2,2,2,2,2,2,2,2,2,2,2,1	551488	8.037
1,2,2,2,2,2,2,2,2,2,2,2,2,2,2,3,1	41896	9.003
1,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,1	41152	10.787
1,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,3,1	547840	9.516
1,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,1	546144	9.790

input	Trial 5	
	Total memory usage (B)	Time (Sec)
1,3,1	36688	2.395
1,2,2,2,2,1	550984	3.978
1,2,2,2,2,2,3,1	52024	4.469
1,2,2,2,2,2,2,2,2,2,1	551000	6.445
1,2,2,2,2,2,2,2,2,2,2,3,1	544416	6.391
1,2,2,2,2,2,2,2,2,2,2,2,2,2,1	41088	8.562
1,2,2,2,2,2,2,2,2,2,2,2,2,2,2,3,1	42400	8.104
1,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,1	42344	9.307
1,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,3,1	42880	9.102
1,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,1	42480	10.004

input	Average	
	Total memory usage (B)	Time (Sec)
1,3,1	38659.2	2.484
1,2,2,2,2,1	243496.0	4.173
1,2,2,2,2,2,3,1	145654.4	5.343
1,2,2,2,2,2,2,2,2,2,1	245292.8	6.450
1,2,2,2,2,2,2,2,2,2,2,3,1	346588.8	7.006
1,2,2,2,2,2,2,2,2,2,2,2,2,2,1	244212.8	8.484
1,2,2,2,2,2,2,2,2,2,2,2,2,2,2,3,1	343641.6	9.269
1,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,1	142339.2	9.993
1,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,3,1	345203.2	9.917
1,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,1	143376.0	10.225

4.2.2 LinkedList Results

input	Trial 1	
	Total memory usage (B)	Time (Sec)
1,3,1	37840	3.163
1,2,2,2,2,1	42888	4.276
1,2,2,2,2,2,3,1	552168	6.811
1,2,2,2,2,2,2,2,2,2,1	43040	6.785
1,2,2,2,2,2,2,2,2,2,2,3,1	545656	7.858
1,2,2,2,2,2,2,2,2,2,2,2,2,2,1	546944	8.666
1,2,2,2,2,2,2,2,2,2,2,2,2,2,2,3,1	43008	9.246
1,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,1	547624	8.698
1,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,3,1	44904	11.675
1,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,1	547328	10.403

input	Trial 2	
	Total memory usage (B)	Time (Sec)
1,3,1	541200	2.545
1,2,2,2,2,1	42472	4.794
1,2,2,2,2,2,3,1	545216	5.371
1,2,2,2,2,2,2,2,2,2,1	551472	6.056
1,2,2,2,2,2,2,2,2,2,2,3,1	545792	7.336
1,2,2,2,2,2,2,2,2,2,2,2,2,2,1	42160	6.842
1,2,2,2,2,2,2,2,2,2,2,2,2,2,2,3,1	43368	8.596
1,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,1	43024	8.892
1,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,3,1	43736	10.033
1,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,1	44480	10.496

input	Trial 3	
	Total memory usage (B)	Time (Sec)
1,3,1	37864	3.45
1,2,2,2,2,1	545768	4.613
1,2,2,2,2,2,3,1	545112	5.433
1,2,2,2,2,2,2,2,2,2,1	50696	6.57
1,2,2,2,2,2,2,2,2,2,3,1	45080	7.415
1,2,2,2,2,2,2,2,2,2,2,2,2,1	42496	8.067
1,2,2,2,2,2,2,2,2,2,2,2,2,2,3,1	545944	9.491
1,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,1	548688	11.128
1,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,3,1	45416	10.149
1,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,1	547400	10.741

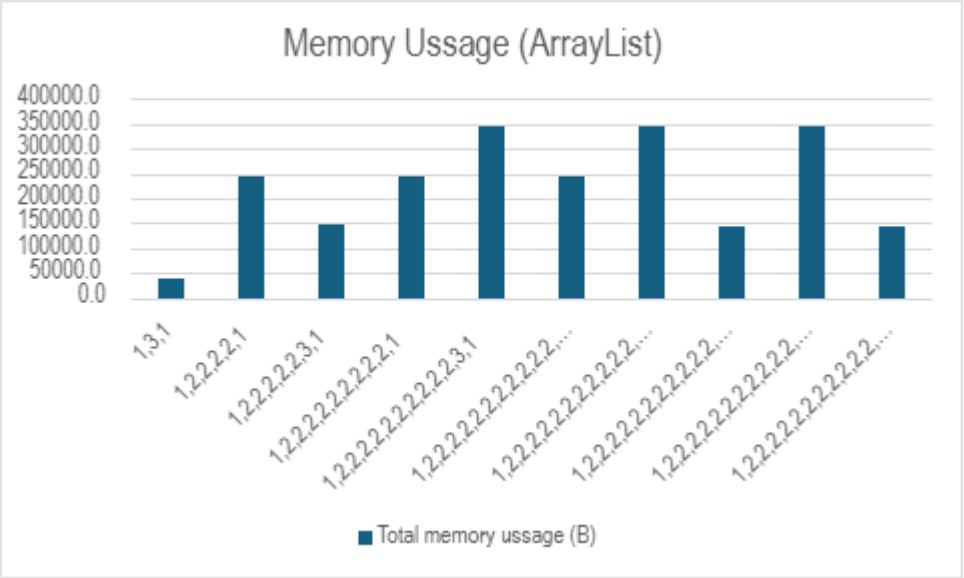
input	Trial 4	
	Total memory usage (B)	Time (Sec)
1,3,1	38184	2.921
1,2,2,2,2,1	42432	4.144
1,2,2,2,2,2,3,1	41976	5.738
1,2,2,2,2,2,2,2,2,2,1	43688	6.248
1,2,2,2,2,2,2,2,2,2,2,3,1	49032	7.59
1,2,2,2,2,2,2,2,2,2,2,2,2,2,1	547360	8.241
1,2,2,2,2,2,2,2,2,2,2,2,2,2,2,3,1	42608	9.125
1,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,1	547816	11.243
1,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,3,1	43504	8.932
1,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,1	546992	11.259

input	Trial 5	
	Total memory usage (B)	Time (Sec)
1,3,1	543504	2.725
1,2,2,2,2,1	43224	4.297
1,2,2,2,2,2,3,1	48672	5.139
1,2,2,2,2,2,2,2,2,2,1	553504	5.664
1,2,2,2,2,2,2,2,2,2,3,1	553424	7.326
1,2,2,2,2,2,2,2,2,2,2,2,2,1	553576	8.349
1,2,2,2,2,2,2,2,2,2,2,2,2,3,1	551120	9.254
1,2,2,2,2,2,2,2,2,2,2,2,2,2,2,1	44808	11.606
1,2,2,2,2,2,2,2,2,2,2,2,2,2,2,3,1	547056	8.124
1,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,1	548400	10.647

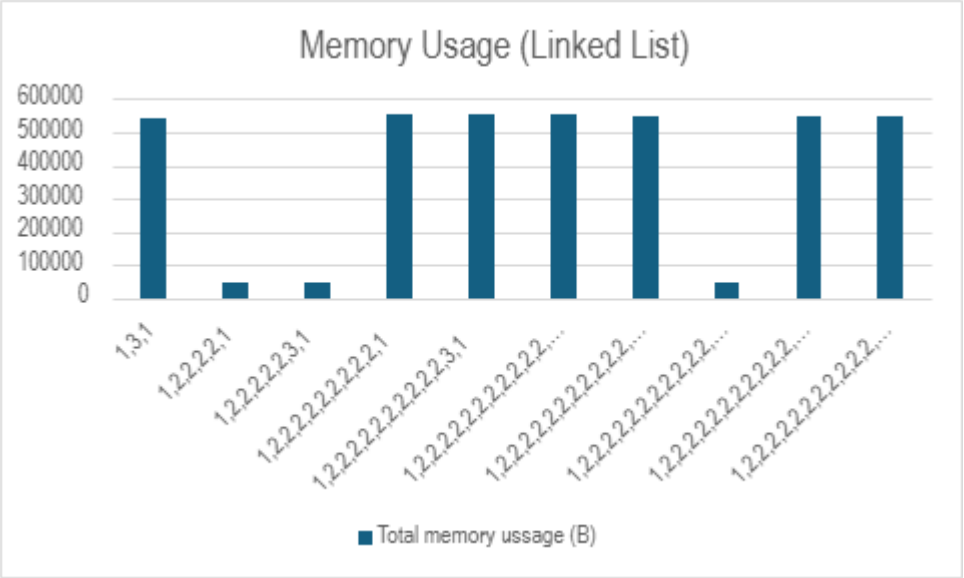
input	Average	
	Total memory usage (B)	Time (Sec)
1,3,1	239718.4	2.961
1,2,2,2,2,1	143356.8	4.425
1,2,2,2,2,2,3,1	346628.8	5.698
1,2,2,2,2,2,2,2,2,2,1	248480	6.265
1,2,2,2,2,2,2,2,2,2,2,3,1	347796.8	7.505
1,2,2,2,2,2,2,2,2,2,2,2,2,2,1	346507.2	8.033
1,2,2,2,2,2,2,2,2,2,2,2,2,2,3,1	245209.6	9.142
1,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,1	346392	10.313
1,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,3,1	245408	9.783
1,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,1	446920	10.709

4.3 Memory Usage Graphs

4.3.1 ArrayList Memory Usage Graph



4.3.2 LinkedList Memory Usage Graph



4.4 Conclusion

Based on the testing, it can be concluded that the best and most efficient data structure to solve the problem is ArrayList as compared to LinkedList, the amount of memory used by ArrayList is smaller on average compared to LinkedList with ArrayList using 223846.4 Bytes of memory while LinkedList using 295641.760 Bytes of memory. Meanwhile, Comparing the average runtime of both the LinkedList version and the ArrayList version, ArrayList wins with an average time of 7.334 seconds while LinkedList takes a bit longer with 7.483 seconds. However, since the test has a human factor, there would be slight millisecond differences each trial due to the constraints of the human reaction time and typing speed.

CHAPTER 5 TEAM WORKLOAD AND TASK DIVISION

Task	Task Description	Person in charge
Basic/template class system	Make the entire class system for the whole program	Andres Winson
ArrayList version of the code	Code the class system with ArrayList in mind	Andres Winson
LinkedList version of the code	Convert every ArrayList on any class into LinkedList and make it work	Allenxavinzky Adjiewibowo
Main class	Make the Main class for the program	Allenxavinzky Adjiewibowo
Program stress testing	Stress test the program to find out bugs and glitches	Allenxavinzky Adjiewibowo
Documentation	Document everything about the code	Allenxavinzky Adjiewibowo
Project Report	Make the project report	Allenxavinzky Adjiewibowo and Andres Winson
Presentation slides	Make the presentation slides for presentation day	Allenxavinzky Adjiewibowo and Andres Winson
Data structure efficiency testing	Data structure testing and recording the testing results from both data structures	Allenxavinzky and Andres Winson

REFERENCES

Bicycle Cards. (n.d.). *Basics of poker*. Bicycle Cards. https://bicyclecards.com/how-to-play/basics-of-poker

Cormen, T. H., & Leiserson, C. E. (2009). *Introduction to algorithms* (3rd ed.). MIT Press.

Devi, K. R. (2019). Analysis of ArrayList and linked list. *International Journal of Computer Sciences and Engineering, 7*(5), 1566–1570. https://doi.org/10.26438/ijcse/v7i5.15661570

Geeks for Geeks. (2025). *Array vs ArrayList in Java*. Geeks for Geeks. https://www.geeksforgeeks.org/java/array-vs-arraylist-in-java/

Hellmuth, P. (2003). *Let's play poker*. HarperResource.

Jain, H. (2017). *Problem solving in data structures & algorithms using Java*. CreateSpace Independent Publishing Platform.

Koffman, E. B., & Wolfgang, P. A. T. (2021). *Data structures: Abstraction and design using Java*. John Wiley & Sons.

Manber, U. (1989). *Introduction to algorithms: A creative approach*. Addison-Wesley.

Neso Academy. (2022). *Introduction to Linked List*. Neso Academy. https://www.youtube.com/watch?v=R9PTBwOzceo

Rahman, A. (2016). Performance analysis of ArrayList and HashMap. *The 16th Winona Computer Science Undergraduate Research Symposium*, 33–40.

APPENDIX

Github Link:
https://github.com/Simayawan/OOP-DS-FP.git

Documentation (Poster, Video demo, Data results, etc):
https://drive.google.com/drive/folders/1L9eTDZD8ez3Uo5eDBfdaWBdBec_vzsNr?usp=sharing

Manual:
https://docs.google.com/document/d/1VufMGNh_3DLH9V60CVlVCwZ4JOKqtk-3xdm0lSP5BLw/edit?usp=sharing