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

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 ().

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).

1.4 Array

Array is a data structure that can store data in memory, however Array cannot be expanded as it is fixed and static, and it cannot add new data that makes the Array longer than its fixed length (Geeks for Geeks, 2025).

1.5 Map

Map is a data structure that stores data with a key.

1.6 Enum

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.\

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

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
1,2,2,2,2,2,2,2,2,2,2,2,2,2,2,1	546944	8.666
1,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,3,1	43008	9.246
1,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,1	547624	8.698
1,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,3,1	44904	11.675
1,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,1	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
1,2,2,2,2,2,2,2,2,2,2,2,2,2,2,1	42160	6.842
1,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,3,1	43368	8.596
1,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,1	43024	8.892
1,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,3,1	43736	10.033
1,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,1	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
1,2,2,2,2,2,2,2,2,2,2,3,1	45080	7.415
1,2,2,2,2,2,2,2,2,2,2,2,2,2,2,1	42496	8.067
1,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,3,1	545944	9.491
1,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,1	548688	11.128
1,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,3,1	45416	10.149
1,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,1	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
1,2,2,2,2,2,2,2,2,2,2,2,2,2,2,1	547360	8.241
1,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,3,1	42608	9.125
1,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,1	547816	11.243
1,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,3,1	43504	8.932
1,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,1	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
1,2,2,2,2,2,2,2,2,2,2,3,1	553424	7.326
1,2,2,2,2,2,2,2,2,2,2,2,2,2,2,1	553576	8.349
1,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,3,1	551120	9.254
1,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,1	44808	11.606
1,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,3,1	547056	8.124
1,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,1	548400	10.647

input	Average	
	Total memory ussage (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
1,2,2,2,2,2,2,2,2,2,2,2,2,2,2,1	346507.2	8.033
1,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,3,1	245209.6	9.142
1,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,1	346392	10.313
1,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,3,1	245408	9.783
1,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,1	446920	10.709

4.3 Conclusion

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

5.1 Workload Rationale

Due to the fact that Andres did all the harder parts of this whole project, Allen must do the rest of the tasks required by the project to balance the workload so it is fair and not all skewed to one person only.

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/

Goodrich, M., Tamassia, R., & Goldwasser, M. H. (2014). *Data structures and algorithms in Java* (6th ed.). John Wiley & Sons.

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.

Martinez, J. (2025). Collections. In *COMP 6699: Object Oriented Programming* (Chapter 12, p. 12.2). Zybooks. https://learn.zybooks.com/zybook/COMP6699001Spring2025_16

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.

Shaffer, C. A. (2012). *Data structures and algorithm analysis in Java* (3rd ed.). Dover Publications.

Xinogalos, S. (2010). Difficulties with collection classes in Java: The case of the ArrayList collection. *Proceedings of the 2nd International Conference on Computer Supported Education*, 120–125. https://doi.org/10.5220/0002796201200125

APPENDIX

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