

Data Structures Project Report

Lecturer: Dr. Maria Seraphina Astriani, S. Kom., M. T. I.

Class: L2AC

Name: Albertus Santoso
NIM: 2702334885

Name: Gabriel Anderson
NIM: 2702256315

Name: Rafael Anderson
NIM: 2702255981

Table Of Contents

I. Background

1. Background

II. Problem Description

1. Problem Description

III. Solution

1. Data Structures Used and Analysis

2. Results from testing

IV. References

1. References

V. Appendix

1. Program Manual

2. Git Website

3. Presentation Files

I. Background:

At this time of age, many people play video games regardless of their age, making it a source of entertainment in today's digital age. It has been a main source for people nowadays, especially among the younger generations, including ourselves, gaming is part of our everyday's life. One genre in video games that has been gaining popularity lately is gacha games. Gacha games are designed to have people spend virtual currency which can be obtained from tasks or purchased with real money, to receive random in-game items or characters. Gacha games often feature rich storylines, appealing graphics, and frequent updates with new items and events to keep players engaged. However, the random reward system has sparked controversy due to its similarity to gambling and potential to encourage excessive spending and addictive behaviors, particularly among younger players. One of the most popular gacha games right now is Honkai Star Rail. Honkai Star Rail is a role playing gacha video game developed by a company called miHoyo and published worldwide by COGNOSPHERE on April 26 2023. Although many of them are immersed in the storyline of the game, some people don't pay attention to the character's basic information.

II. Problem Description:

Since Honkai Star Rail has been gaining popularity since the release a year ago, it has become one of the main games in the gacha genre. Due to the popularity of the game, the rate of production of new characters has been constantly increasing to keep up with the demand wanted by the players. A challenge that most people face due to the high rate of new characters being introduced to the game is trying to remember their information and not interchange them with other characters. Although it might seem as a simple problem, we often see people struggling in remembering the information of characters due to the game introducing many unfamiliar characters in a short period of time, confusing a character information with another character. Based on our point of view, it is somewhat sad to see people not knowing simple information regarding the game's character as well as their favorite characters, unable to understand the plot of the game and just witnessing unfamiliar characters wandering throughout the game. As we see more and more people interchange characters and character information on a daily basis, we decided to come up with a solution to solve this problem, by coming up with a solution that will hopefully solve or at least decrease the number of people struggling to overcome this problem.

III. Solution:

To solve this problem, we came up with a solution, which is a Honkai Star Rail Character Database System, filled with all the character's information. Users are able to retrieve any information about characters from their path, element, and rarity by entering the character's name. They are also able to print the character's information that consists of the character's path, element, and rarity. An addition to that, we have provided basic features such as sorting and filtering actions. Users are able to sort by the name, rarity, path, element, and the faction of a character which allows the user to retrieve the information they want faster. The filtering action allows the user to filter by the same features stated, it provides the user with characters with the same features as the condition that the user filtered with. This Honkai Star Rail Character Database System main objective is to make users become more familiar with the in-game characters, giving them a better understanding of characters, in hoping that they are able to understand and empathize with the game plot more, allowing an overall better experience as they can feel the plot instead of just witnessing an unfamiliar character wandering throughout the game, which might enhance their gameplay.

1. Data Structures and Analysis

To come up with the most efficient and effective solution, we compared a few data structures and objects. We compared hash tables with different data structures representing the value of the hash table such as array, linked list and lastly a character object. Lastly we are going to compare all of them with a binary search tree, using arrays storing the data.

Note: The analysis of the data structures are only based on theory

Analysis of data structures:

- **Hash Table:** Hashtable operations have a time complexity of $O(1)$ in a regular case. But there is a chance for it to have an $O(n)$ time complexity if there are many items that are hashed into the same index / Collisions. Although that is the case, collisions happen rarely. Thus, we can conclude that hash tables have an average time complexity of $O(1)$ in doing these operations. It has a space complexity of $O(n)$.
- **Array:** Arrays have a time complexity of $O(1)$ in accessing the elements by the index (Random Access) as the arrays are allocated contiguously in the memory, making the random access an arithmetic operation, and as we know, all arithmetic operations have a time complexity of $O(1)$. For searching, it has a time complexity of $O(n)$, as it will continue iterating through the array until it finds the targeted value, and a time complexity of $O(n)$ in adding elements into the array (if the size is not enough). During the initializing of the array, it has a time complexity of $O(1)$ as the number of elements is fixed, and does not change the size of input. For adding elements, it has a time complexity of $O(n)$, as there is a need for resizing the array if it does not fit. It has a space complexity of $O(n)$.
- **Linked List:** In a linked list, it has an average time complexity of $O(n)$ for accessing elements and searching elements because it has to scan through the elements of the list until it finds the target element. Unlike an array, a linked list does not have a fixed size, making the initialization of the linked list take a time complexity of $O(n)$ where n is the amount of elements added into the linked list. The time complexity of adding elements at the beginning or end is $O(1)$ as it is able to use its tail and head properties. Its space complexity is $O(n)$.
- **Binary Search Tree:** For the initialization of the tree data structure, the binary search tree has a worst time complexity of $(O \log n)$ for inserting elements, and a best time complexity of $O(1)$. In addition to that, it has a time complexity of $O(h)$ for searching elements as you have to traverse all of the elements in order, where h is the height of the tree, with a worst time complexity of $O(n)$. The space complexity of a binary search tree is $O(n)$ as well.
- **Character Objects:** As character objects are not data structures, there are limited actions that you are able to perform. The primary functions are the getters and setters. In a getter, it looks up for the targeted property in the object's internal properties, like how a hash table works. Thus it also has a similar time complexity of $O(1)$. Similar to how getters work, setters also have $O(1)$ time complexity.

Conclusion based on the analysis:

Assuming that our hash table experiences a few collisions, we think that it will work more efficiently compared to the binary search tree. Based on the other data structure, we think that the array will be the best. As during the process of populating the hashtable, we add using the Random access (Index), and we know that arrays are more efficient compared to the linkedlist. In addition to that, the initialization of the array also takes a faster time. Although the linkedlist has the upperhand on inserting and deleting elements at the beginning / end, our program does not really need the inserting and deleting properties as the characters do not change as the program is running. In conclusion, based on theory, we think that the hashtable with the array as the pair value will be the most suitable and efficient for our program specifically.

2. Testing Results

Note: The runtime is in milliseconds, while the space is in bytes

Number of Characters = 12

Data Structure (n = 12)	HT (Array)	HT (Linkedlist)	HT (Charobject)	BST
Runtime case A	1,076	1127,7	1140,3	1036,3
Runtime case B	1,023	1049,1	1130,1	1075,2
Runtime case C	1,195	1204,1	1180,3	1053,3
Runtime case D	4,6	2,3	0,5	6,3
Runtime case E	364,1	511	497,2	504,5
Runtime case F	1201,5	1178,1	1,581	1056,8
Runtime case G	202,9	745,1	244,8	293,6
Space A	248,384	461,424	464,456	463,944
Space B	248,384	463,496	461,392	704,536
Space C	248,392	461,640	461,424	711936
Space D	463,440	925,688	704,448	925,688
Space E	1,386,280	923,808	711,864	922,800
Space F	711,880	925856	704,432	923,288
Space G	463,504	704,448	711,848	1,384,712

For the comparison table using 12 characters, Array wins 3 cases out of 7 in time complexity. Binary Search Tree wins 2 out of 7 cases in our comparison for time complexity. For the space complexity, hash tables with the array as its value wins 5 of the 7 cases in our table.

Number of Characters = 24

Data Structure (n = 24)	HT (Array)	HT (Linkedlist)	HT (Charobject)	BST
Runtime case A	1,144	1292,9	1183,2	1078,1
Runtime case B	1201,6	1197,8	1132,7	1091,2
Runtime case C	1212,1	1278,3	1222,1	1,118
Runtime case D	6,1	5,2	1	8,1
Runtime case E	459,1	612,3	565,8	527,8
Runtime case F	1410,2	1245,6	1687,5	1121,9
Runtime case G	237,9	1154,1	308	328,4
Space A	461,448	1,174,264	704,456	706,480
Space B	463,448	711,840	463,472	1,628,856
Space C	462,436	701,870	706,480	1,630,880
Space D	703,520	1,173,744	924,872	1,164,912
Space E	2,098,008	1,386,488	924,880	1,634,688
Space F	925,856	2,558,544	924,912	1,387,248
Space G	924,872	1,386,264	925,880	1,386,264

For the comparison table using 24 characters, the binary search tree wins 4 over 7 cases, with the hashtable with the array value coming in second winning 3 over 7 cases in time complexity. For space complexity, the hashtable with the array value surpasses the other data structures, winning 4 over 7 cases.

Number of Characters = 48

Data Structure (n = 48)	HT (Array)	HT (Linkedlist)	HT (Charobject)	BST
Runtime case A	1239,6	1558,3	1264,3	1124,4
Runtime case B	1221,3	1233,9	1265,5	1218,5
Runtime case C	1230,2	1311,1	1236,5	1441,9
Runtime case D	8,5	8,7	1,4	9,7
Runtime case E	554,8	691,2	587,5	637,5
Runtime case F	1225,6	1407,3	1782,9	1136,7
Runtime case G	270,7	1167,4	365,5	310,2
Space A	461,488	1,633,712	923,028	711,888
Space B	463,464	1,174,312	712,264	2,097,184
Space C	461,408	1,174,248	925,088	2,097,128
Space D	922,824	1,385,088	925,824	1,386,280
Space E	2,558,544	2,098,080	1,386,232	2,097,168
Space F	2,552,160	2,098,536	1,386,488	2,098,144
Space G	1,386,352	1,386,288	1,172,232	1,633,704

For the comparison table using 48 characters, the binary search tree comes first winning 3 over 7 cases, with the hashtable array coming in second winning 2 over 7 cases in time complexity. For space complexity, the hash table array wins 4 over 7 cases.

Number of Characters = 96

Data Structure (n = 96)	HT (Array)	HT (Linkedlist)	HT (Charobject)	BST
Runtime case A	1252,3	1707,5	1,363	1261,9
Runtime case B	1372,7	1354,6	1314,4	1231,5
Runtime case C	1688,8	1343,9	1296,4	1784,1
Runtime case D	9,6	9,59	1,48	11,8
Runtime case E	821,4	765,3	729,1	810,4
Runtime case F	1478,1	1727,7	1943,6	1232,6
Runtime case G	448,5	1,432	378,7	351,8
Space A	1,635,736	2,097,160	1,174,440	1,174,272
Space B	2,097,176	1,635,768	1,174,208	2,558,768
Space C	1,636,872	1,635,848	1,174,376	2,558,128
Space D	1,385,224	1,634,664	1,386,264	1,633,696
Space E	2,558,560	2,558,520	1,635,600	2,559,504
Space F	2,559,560	2,558,544	1,635,624	2,558,696
Space G	1,847,648	2,097,184	1,635,624	1,635,736

For the comparison table with 96 characters, there is a tie between the hash table array and the binary search tree where each data structure won 3 out of 7 cases. On the other hand, the hash table with the character object won 4 out of 7 cases for the space complexity.

Number of Characters = 192

Data Structure (n = 192)	HT (Array)	HT (Linkedlist)	HT (Charobject)	BST
Runtime case A	1278,7	2056,8	1,448	1,397
Runtime case B	1,801	1515,7	1436,1	1246,4
Runtime case C	2284,6	1499,1	1412,7	1853,3
Runtime case D	13,3	1,6	1,9	16,7
Runtime case E	1,097	882	773,2	872,5
Runtime case F	2032,2	2,057	2067,6	1237,4
Runtime case G	546,9	1674,8	561,4	395,2
Space A	2,097,280	2,531,378	1,635,576	1,174,328
Space B	2,098,168	2,109,680	1,174,360	2,797,136
Space C	2,097,192	2,097,080	1,635,720	2,678,438
Space D	2,097,160	2,413,120	1,633,712	2,097,088
Space E	1,636,816	2,785,128	2,096,264	2,745,984
Space F	2,097,160	2,746,552	2,097,160	2,797,192
Space G	2,097,096	2,756,346	2,097,176	1,636,744

For the comparison table with 192 characters, the binary search tree wins 4 out of 7 cases for the time complexity. While the hash table with the character object wins 4 out of 7 cases for the space complexity.

Conclusion:

For a small amount of data like 12, hash tables with arrays are the fastest, with the binary search trees coming in second. In addition to that, the hash tables with the arrays also wins the majority of the space complexity.

However, for a larger quantity of data like 192, binary search trees are significantly faster than other data structures. But for the space occupied, character objects come in first, whereas arrays come in second.

But for our honkai star rail database, where the amount of characters is 48 characters. The fastest working data structure is the binary search tree winning the most cases. On the other hand, the hash table with the arrays as the value wins the most cases for the space complexity comparison. Thus, if you want to prioritize speed over memory, the binary search tree is the best data structure. On the other hand, the hash table with the arrays is best if you want to prioritize the memory usage.

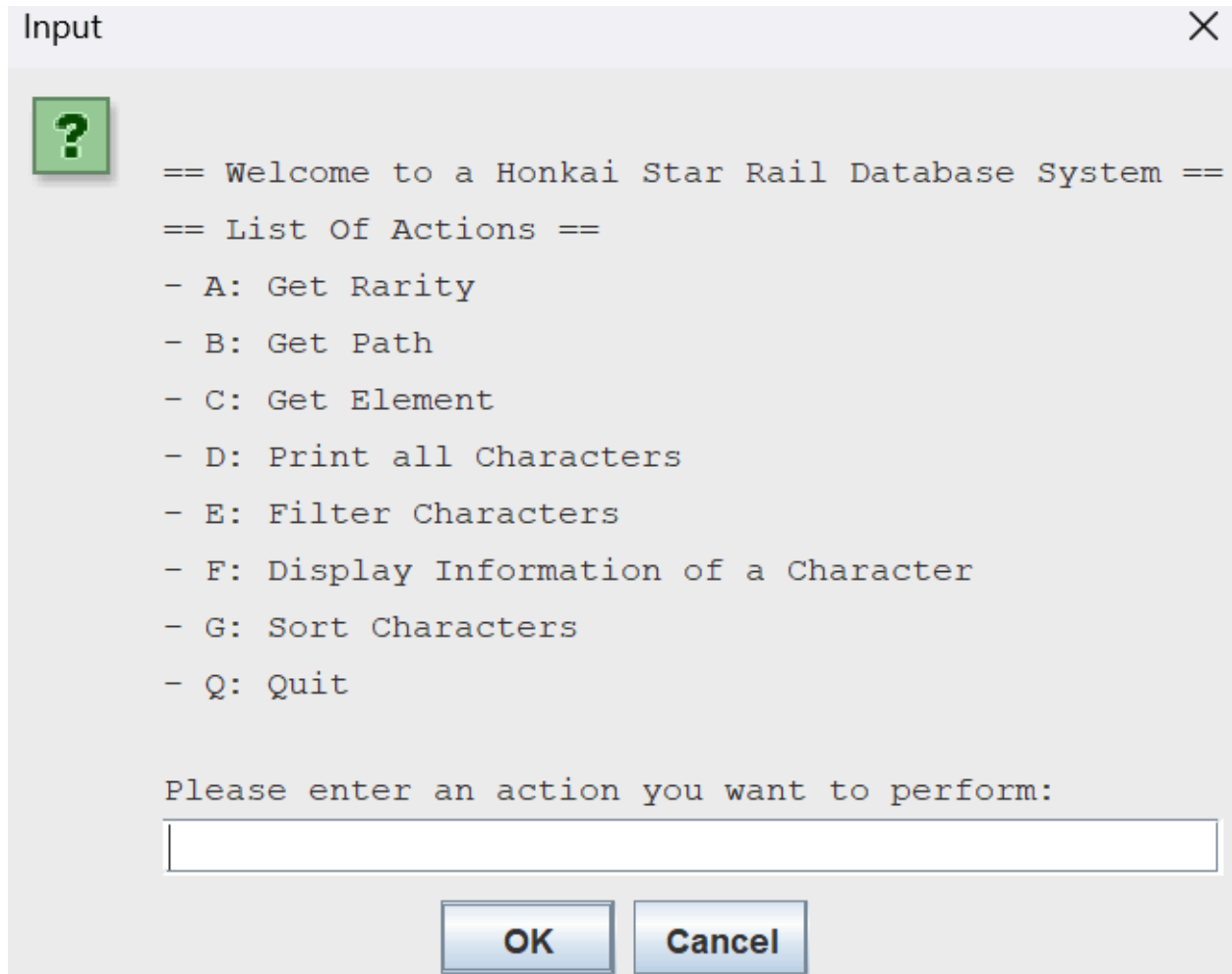
But with the rapid rate of the increase in characters, there might be a different data structure that is the most suitable to act as the storage of the honkai star rail database system.

IV. References

1. GeeksforGeeks, “Implementing a Linked List in Java using Class,” *GeeksforGeeks*, Sep. 2023. [Online]. Available: <https://www.geeksforgeeks.org/implementing-a-linked-list-in-java-using-class/>
2. “Linking the GUI to a database,” in *Apress eBooks*, 2007, pp. 123–152. [Online]. Available: https://link.springer.com/chapter/10.1007/978-1-4302-0440-4_7
3. B. F. Burton and V. W. Marek, “Applications of JAVA programming language to database management,” *SIGMOD Record*, vol. 27, no. 1, pp. 27–34, Mar. 1998. [Online]. Available: <https://doi.org/10.1145/273244.273254>.
4. “Hash Table Data structure.” [Online]. Available: https://www.tutorialspoint.com/data_structures_algorithms/hash_data_structure.htm#:~:text=Hash%20Table%20is%20a%20data,index%20of%20the%20%20desired%20data.
5. GeeksforGeeks, “Hashtable in Java,” *GeeksforGeeks*, Jun. 2023. [Online]. Available: <https://www.geeksforgeeks.org/hashtable-in-java/>
6. GeeksforGeeks, “Searching in Binary Search Tree (BST),” *GeeksforGeeks*, Nov. 2023. [Online]. Available: <https://www.geeksforgeeks.org/binary-search-tree-set-1-search-and-insertion/>
7. Bro Code, “Learn Hash Tables in 13 minutes #,” *YouTube*. Oct. 2021. [Online]. Available: <https://www.youtube.com/watch?v=FsfRsGFHuv4>
8. “Java Swing Tutorial - javatpoint,” *www.javatpoint.com*. [Online]. Available: <https://www.javatpoint.com/java-swing>
9. GeeksforGeeks, “LinkedList in java,” *GeeksforGeeks*, Jun. 2023. [Online]. Available: <https://www.geeksforgeeks.org/linked-list-in-java/>
10. Derek Banas, “Java Binary Search Tree,” *YouTube*. Mar. 2013. [Online]. Available: <https://www.youtube.com/watch?v=M6lYob8STMI>
11. GeeksforGeeks, “Binary search tree,” *GeeksforGeeks*, Feb. 2024. [Online]. Available: <https://www.geeksforgeeks.org/binary-search-tree-data-structure/>
12. P. Plyenko, “Java Hashtable,” *CodeGym*, Oct. 2023. [Online]. Available: <https://codegym.cc/groups/posts/218-what-is-hashtable-java-hashtable-with-examples>
13. Dr. Maria Seraphina Astriani, S. Kom., M. T. I.

V. Appendix

1. Program Manual

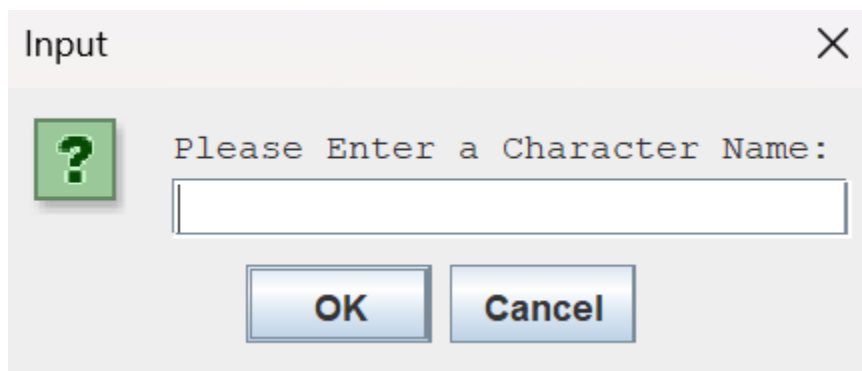


Explanation: When users initially execute the program, they are greeted with an interface programmed using JOptionPane. It welcomes the users to the program and lists out the list of actions the users are able to perform. Which are Get Rarity, Get Path, Get Element, Print All Characters, Filter Characters, Display Information of a Character, Sort Characters and Quit. Users are prompted to enter one of the listed actions.

Action A, B, and C:

- A: Get Rarity
- B: Get Path
- C: Get Element

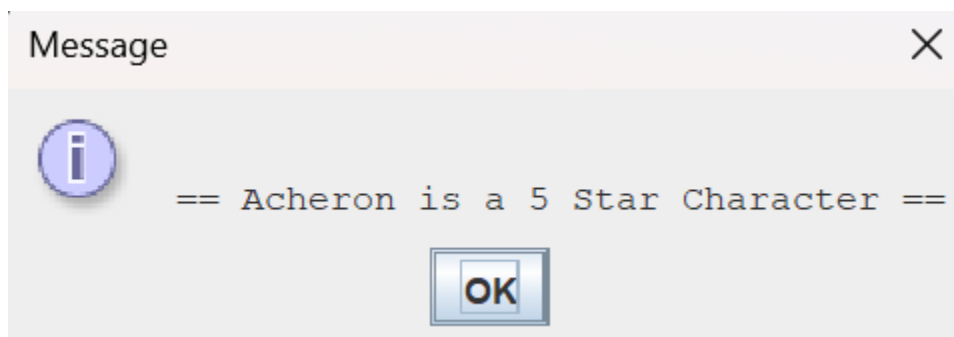
Explanation: For the first three actions listed, which are get rarity, get path, and get element. All 3 actions have similar ways of actions so it will be explained together.



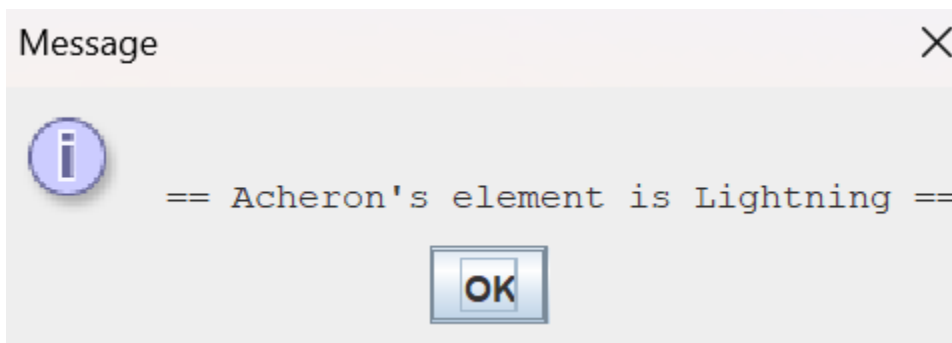
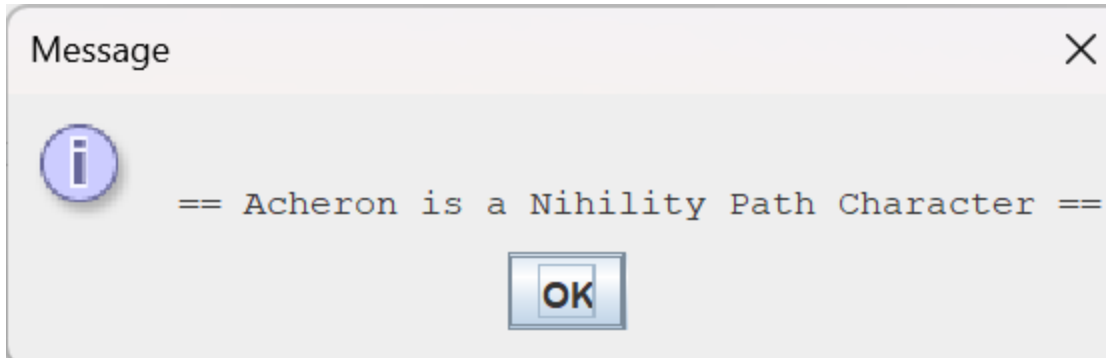
A screenshot of an 'Input' dialog box. The title bar says 'Input' with a close button (X) on the right. Inside the dialog, there is a green square icon with a white question mark. To the right of the icon is the text 'Please Enter a Character Name:'. Below this text is a text input field. At the bottom of the dialog are two buttons: 'OK' and 'Cancel'.

Explanation: After entering one of the 3 actions, users are prompted to enter the character's name. And based on the alphabet, the system will behave differently.

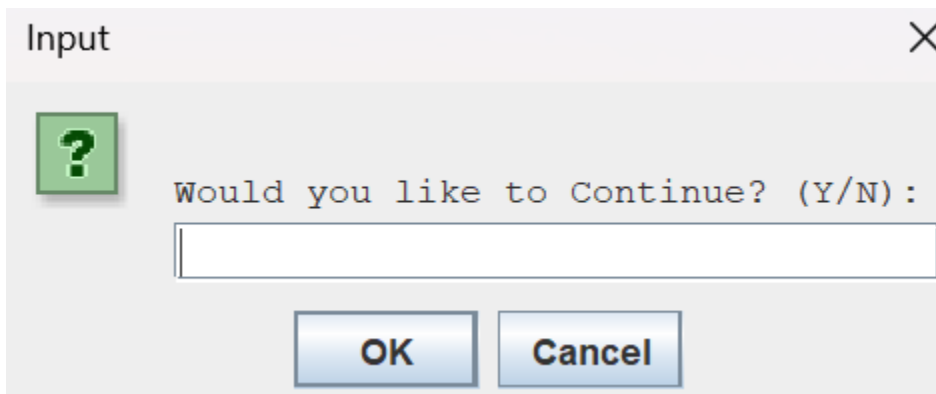
To Show what the outputs are, the input for each case (character name) will be Acheron.



A screenshot of a 'Message' dialog box. The title bar says 'Message' with a close button (X) on the right. Inside the dialog, there is a blue circular icon with a white lowercase 'i'. To the right of the icon is the text '== Acheron is a 5 Star Character =='. Below this text is an 'OK' button.







Explanation: As shown by the pictures above, the rarity, path, and element is given by taking a character's name as an input for the program.



Explanation: After each action that is done by the user, the program will prompt the user with a yes or no question. It asks the user whether the user would like to continue the program or not. If the user inputs N, the program will stop. On the other hand, if the user chooses to continue the program, the interface that shows the list of actions in the beginning of the program will pop up again.

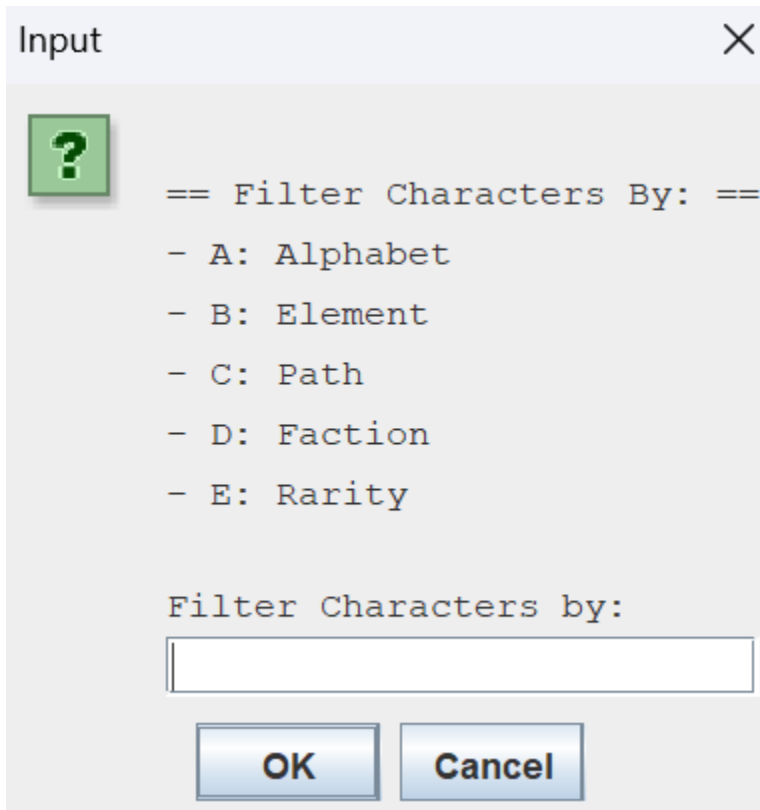
Action D:

Message ×					
	Character:	Rarity:	Path:	Element:	Faction:
	1. Acheron	5 Star	Nihility	Lightning	Self Annihilator
	2. Argenti	5 Star	Erudition	Physical	Knights of Beauty
	3. Arlan	4 Star	Destruction	Lightning	Herta Space Station
	4. Asta	4 Star	Harmony	Fire	Herta Space Station
	5. Aventurine	5 Star	Preservation	Imaginary	IPC
	6. Bailu	5 Star	Abundance	Lightning	The Xianzhou Loufu
	7. Blackswan	5 Star	Nihility	Wind	Garden of Recollection
	8. Blade	5 Star	Destruction	Wind	Stellaron Hunter
	9. Bronya	5 Star	Harmony	Wind	Belobog
	10. Clara	5 Star	Destruction	Physical	Belobog
	11. Danheng	4 Star	The Hunt	Wind	Astral Express
	12. Dr.ratio	5 Star	The Hunt	Imaginary	Intelligentsia Guild
	13. Fuxuan	5 Star	Preservation	Quantum	The Xianzhou Loufu
	14. Gallagher	4 Star	Abundance	Fire	Penacony
	15. Gepard	5 Star	Preservation	Ice	Belobog
	16. Guinaifen	4 Star	Nihility	Fire	The Xianzhou Loufu
	17. Hanya	4 Star	Harmony	Physical	Xianzhou Loufu
	18. Herta	4 Star	Erudition	Ice	Herta Space Station
	19. Himeko	5 Star	Erudition	Fire	Astral Express
	20. Hook	4 Star	Destruction	Fire	Belobog
	21. Huohuo	5 Star	Abundance	Wind	The Xianzhou Loufu
	22. Imbibitorlunae	5 Star	Destruction	Imaginary	Astral Express
	23. Jingyuan	5 Star	Erudition	Lightning	The Xianzhou Loufu
	24. Jingliu	5 Star	Destruction	Ice	The Xianzhou Loufu
	25. Kafka	5 Star	Nihility	Lightning	Stellaron hunter
	26. Luka	4 Star	Nihility	Physical	Belobog
	27. Luocha	5 Star	Abundance	Imaginary	The Xianzhou Loufu
	28. Lynx	4 Star	Abundance	Quantum	Belobog
					

Message ×					
	Character:	Rarity:	Path:	Element:	Faction:
	29. March7th	4 Star	Preservation	Ice	Astral Express
	30. Misha	4 Star	Destruction	Ice	Penacony
	31. Natasha	4 Star	Abundance	Physical	Belobog
	32. Pela	4 Star	Nihility	Ice	Belobog
	33. Qingque	4 Star	Erudition	Quantum	The Xianzhou Loufu
	34. Robin	5 Star	Harmony	Physical	Penacony
	35. Ruanmei	5 Star	Harmony	Ice	Herta Space Station
	36. Sampo	4 Star	Nihility	Wind	Belobog
	37. Seele	5 Star	The Hunt	Quantum	Belobog
	38. Serval	4 Star	Erudition	Lightning	Belobog
	39. Silverwolf	5 Star	Nihility	Quantum	Stellaron Hunter
	40. Sparkle	5 Star	Harmony	Quantum	Masked Fools
	41. Sushang	4 Star	The Hunt	Physical	The Xianzhou Loufu
	42. Tingyun	4 Star	Harmony	Lightning	The Xianzhou Loufu
	43. Topaz	5 Star	The Hunt	Fire	IPC
	44. Trailblazer	5 Star	Adaptive	Adaptive	Astral Express
	45. Welt	5 Star	Nihility	Imaginary	Astral Express
	46. Xueyi	4 Star	Destruction	Quantum	The Xianzhou Loufu
	47. Yanqing	5 Star	The Hunt	Ice	The Xianzhou Loufu
	48. Yukong	4 Star	Harmony	Imaginary	The Xianzhou Loufu
					

Explanation: If the user chooses to perform the action D, the system will print out all of the characters along with their information in an ascending alphabetical order that is separated into two sections, as it would not fit in one.

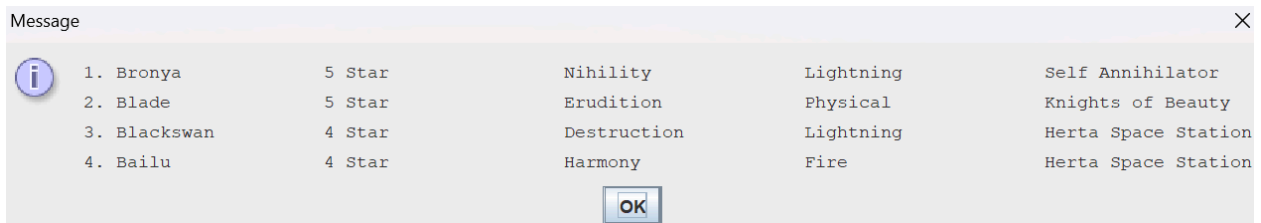
Action E:



Input dialog box titled "Input" with a close button (X). It contains a green question mark icon and the text "Filter Characters By: ==". Below this, there is a list of options: A: Alphabet, B: Element, C: Path, D: Faction, and E: Rarity. At the bottom, there is a text input field labeled "Filter Characters by:" and two buttons: "OK" and "Cancel".

Explanation: For the filter action, there will be an interface that shows what features that the user is able to filter the characters by which are by alphabet, element, path, faction, or rarity.

For Example, the user enters the letter A, and filters the characters by the alphabet of B.





Message dialog box titled "Message" with a close button (X). It contains an information icon (i) and a table of filtered characters. The table has five columns: Name, Stars, Path, Element, and Faction. Below the table is an "OK" button.

1. Bronya	5 Star	Nihility	Lightning	Self Annihilator
2. Blade	5 Star	Erudition	Physical	Knights of Beauty
3. Blackswan	4 Star	Destruction	Lightning	Herta Space Station
4. Bailu	4 Star	Harmony	Fire	Herta Space Station



Explanation: As a result, the characters starting with the alphabet B will be printed along with their information.

More examples of filter by other features as follows:



Filter by the element fire:

Message ×					
	1. Asta	4 Star	Harmony	Fire	Herta Space Station
	2. Gallagher	4 Star	Abundance	Fire	Penacony
	3. Guinaifen	4 Star	Nihility	Fire	The Xianzhou Loufu
	4. Himeko	5 Star	Erudition	Fire	Astral Express
	5. Hook	4 Star	Destruction	Fire	Belobog
	6. Topaz	5 Star	The Hunt	Fire	IPC
					



Filter by the nihility path:

Message ×					
	1. Acheron	5 Star	Nihility	Lightning	Self Annihilator
	2. Blackswan	5 Star	Nihility	Wind	Garden of Recollection
	3. Guinaifen	4 Star	Nihility	Fire	The Xianzhou Loufu
	4. Kafka	5 Star	Nihility	Lightning	Stellaron hunter
	5. Luka	4 Star	Nihility	Physical	Belobog
	6. Pela	4 Star	Nihility	Ice	Belobog
	7. Sampo	4 Star	Nihility	Wind	Belobog
	8. Silverwolf	5 Star	Nihility	Quantum	Stellaron Hunter
	9. Welt	5 Star	Nihility	Imaginary	Astral Express
					



Filter by the Astral Express faction:

Message ×					
	1. Danheng	4 Star	The Hunt	Wind	Astral Express
	2. Himeko	5 Star	Erudition	Fire	Astral Express
	3. Imbibitorlunae	5 Star	Destruction	Imaginary	Astral Express
	4. March7th	4 Star	Preservation	Ice	Astral Express
	5. Trailblazer	5 Star	Adaptive	Adaptive	Astral Express
	6. Welt	5 Star	Nihility	Imaginary	Astral Express
					

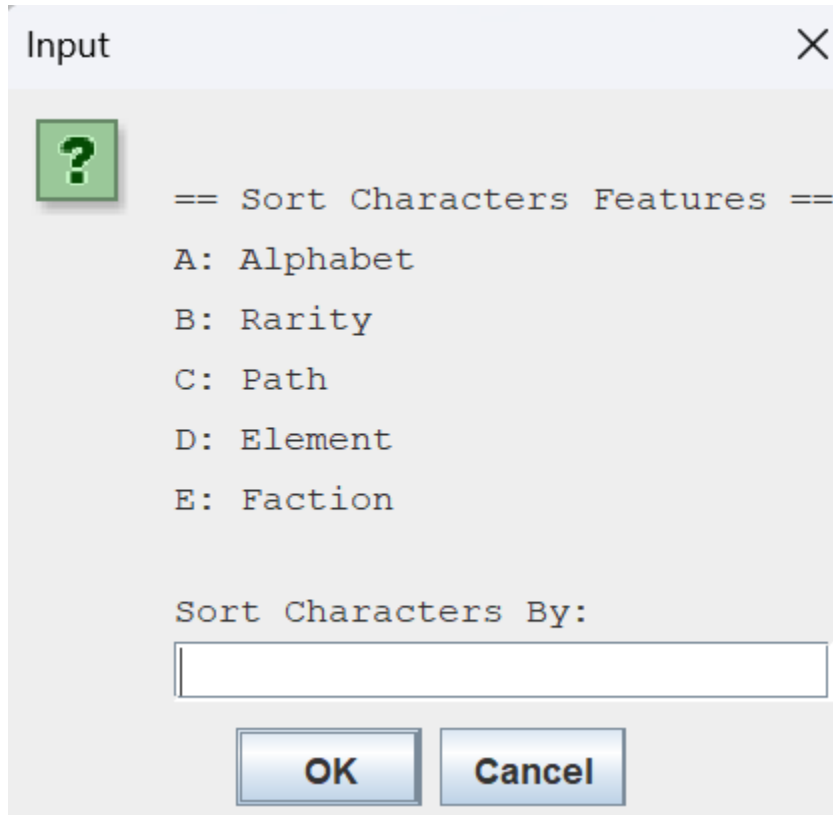
Filter by the the 4 star Rarity:

Message ×					
	1. Arlan	4 Star	Destruction	Lightning	Herta Space Station
	2. Asta	4 Star	Harmony	Fire	Herta Space Station
	3. Danheng	4 Star	The Hunt	Wind	Astral Express
	4. Gallagher	4 Star	Abundance	Fire	Penacony
	5. Guinaifen	4 Star	Nihility	Fire	The Xianzhou Loufu
	6. Hanya	4 Star	Harmony	Physical	Xianzhou Loufu
	7. Herta	4 Star	Erudition	Ice	Herta Space Station
	8. Hook	4 Star	Destruction	Fire	Belobog
	9. Luka	4 Star	Nihility	Physical	Belobog
	10. Lynx	4 Star	Abundance	Quantum	Belobog
	11. March7th	4 Star	Preservation	Ice	Astral Express
	12. Misha	4 Star	Destruction	Ice	Penacony
	13. Natasha	4 Star	Abundance	Physical	Belobog
	14. Pela	4 Star	Nihility	Ice	Belobog
	15. Qingque	4 Star	Erudition	Quantum	The Xianzhou Loufu
	16. Sampo	4 Star	Nihility	Wind	Belobog
	17. Serval	4 Star	Erudition	Lightning	Belobog
	18. Sushang	4 Star	The Hunt	Physical	The Xianzhou Loufu
	19. Tingyun	4 Star	Harmony	Lightning	The Xianzhou Loufu
	20. Xueyi	4 Star	Destruction	Quantum	The Xianzhou Loufu
	21. Yukong	4 Star	Harmony	Imaginary	The Xianzhou Loufu
					

Action F:

Message ×					
	Character:	Rarity:	Path:	Element:	Faction:
	1. Acheron	5 Star	Nihility	Lightning	Self Annihilator
					



Explanation: There is another action that is similar to the first three actions, which is the print character action. Similar to the previous actions, this action prints the character information by taking a character name as an input, but it prints all of the attributes at the same time.



Action G:

The image shows a standard Windows-style dialog box titled "Input" with a close button (X) in the top right corner. On the left side of the dialog is a green square icon containing a white question mark. The main text area contains the prompt "==" Sort Characters Features ==" followed by a list of five options: "A: Alphabet", "B: Rarity", "C: Path", "D: Element", and "E: Faction". Below this list is the label "Sort Characters By:" followed by an empty rectangular text input field. At the bottom of the dialog are two buttons: "OK" and "Cancel".

Explanation: Users are prompted to enter an option to enter a feature to sort characters by alphabetically in that particular feature.

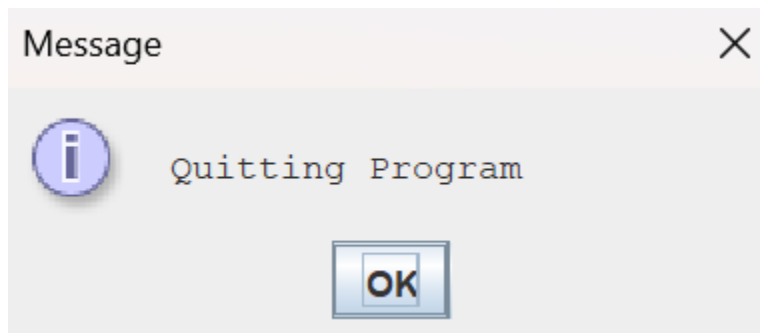
For example, the user enters the letter D which refers to the element of the characters:

Message ×					
	1. Trailblazer	5 Star	Adaptive	Adaptive	Astral Express
	2. Asta	4 Star	Harmony	Fire	Herta Space Station
	3. Gallagher	4 Star	Abundance	Fire	Penacony
	4. Guinaifen	4 Star	Nihility	Fire	The Xianzhou Loufu
	5. Himeko	5 Star	Erudition	Fire	Astral Express
	6. Hook	4 Star	Destruction	Fire	Belobog
	7. Topaz	5 Star	The Hunt	Fire	IPC
	8. Gepard	5 Star	Preservation	Ice	Belobog
	9. Herta	4 Star	Erudition	Ice	Herta Space Station
	10. Jingliu	5 Star	Destruction	Ice	The Xianzhou Loufu
	11. March7th	4 Star	Preservation	Ice	Astral Express
	12. Misha	4 Star	Destruction	Ice	Penacony
	13. Pela	4 Star	Nihility	Ice	Belobog
	14. Ruanmei	5 Star	Harmony	Ice	Herta Space Station
	15. Yanqing	5 Star	The Hunt	Ice	The Xianzhou Loufu
	16. Aventurine	5 Star	Preservation	Imaginary	IPC
	17. Dr.ratio	5 Star	The Hunt	Imaginary	Intelligentsia Guild
	18. Imbibitorlunae	5 Star	Destruction	Imaginary	Astral Express
	19. Luocha	5 Star	Abundance	Imaginary	The Xianzhou Loufu
	20. Welt	5 Star	Nihility	Imaginary	Astral Express
	21. Yukong	4 Star	Harmony	Imaginary	The Xianzhou Loufu
	22. Acheron	5 Star	Nihility	Lightning	Self Annihilator
	23. Arlan	4 Star	Destruction	Lightning	Herta Space Station
	24. Bailu	5 Star	Abundance	Lightning	The Xianzhou Loufu
	25. Jingyuan	5 Star	Erudition	Lightning	The Xianzhou Loufu
	26. Kafka	5 Star	Nihility	Lightning	Stellaron hunter
	27. Serval	4 Star	Erudition	Lightning	Belobog
	28. Tingyun	4 Star	Harmony	Lightning	The Xianzhou Loufu
					

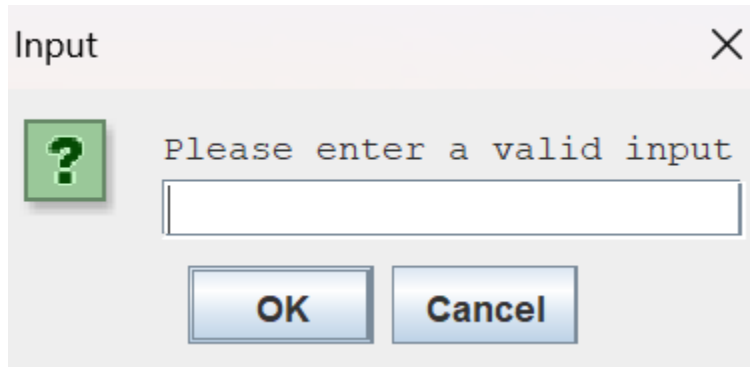
Message						×
	29. Argenti	5 Star	Erudition	Physical	Knights of Beauty	
	30. Clara	5 Star	Destruction	Physical	Belobog	
	31. Hanya	4 Star	Harmony	Physical	Xianzhou Loufu	
	32. Luka	4 Star	Nihility	Physical	Belobog	
	33. Natasha	4 Star	Abundance	Physical	Belobog	
	34. Robin	5 Star	Harmony	Physical	Penacony	
	35. Sushang	4 Star	The Hunt	Physical	The Xianzhou Loufu	
	36. Fuxuan	5 Star	Preservation	Quantum	The Xianzhou Loufu	
	37. Lynx	4 Star	Abundance	Quantum	Belobog	
	38. Qingque	4 Star	Erudition	Quantum	The Xianzhou Loufu	
	39. Seele	5 Star	The Hunt	Quantum	Belobog	
	40. Silverwolf	5 Star	Nihility	Quantum	Stellaron Hunter	
	41. Sparkle	5 Star	Harmony	Quantum	Masked Fools	
	42. Xueyi	4 Star	Destruction	Quantum	The Xianzhou Loufu	
	43. Blackswan	5 Star	Nihility	Wind	Garden of Recollection	
	44. Blade	5 Star	Destruction	Wind	Stellaron Hunter	
	45. Bronya	5 Star	Harmony	Wind	Belobog	
	46. Danheng	4 Star	The Hunt	Wind	Astral Express	
	47. Huohuo	5 Star	Abundance	Wind	The Xianzhou Loufu	
	48. Sampo	4 Star	Nihility	Wind	Belobog	
						

Explanation: As shown above, the characters are sorted by their element alphabetically.

Action Q:



Explanation: The last letter in the list of actions is the quit option, this leads to the stoppage of the program, after showing this message,

Invalid Inputs:

For error handling, everytime the user enters an input that is not one of the valid inputs, the system will prompt the user to enter a valid input until the condition is met.

2. Git Website

Github link: <https://github.com/gamakagami/DSA-Char-DB>

3. Presentation files

Powerpoint link: https://www.canva.com/design/DAGHC9JZWaY/P_amHeRb-x8__dbU8w9Eaw/edit

Spreadsheet link:

docs.google.com/spreadsheets/d/1vm9aSuVw4x2HO_wc9nJgHMEpBVNsiFo151cCCITafFM/edit