Data Structures

FINAL PROJECT REPORT

Genshin Impact

Artifact Rater

Prepared by:

Kokopium

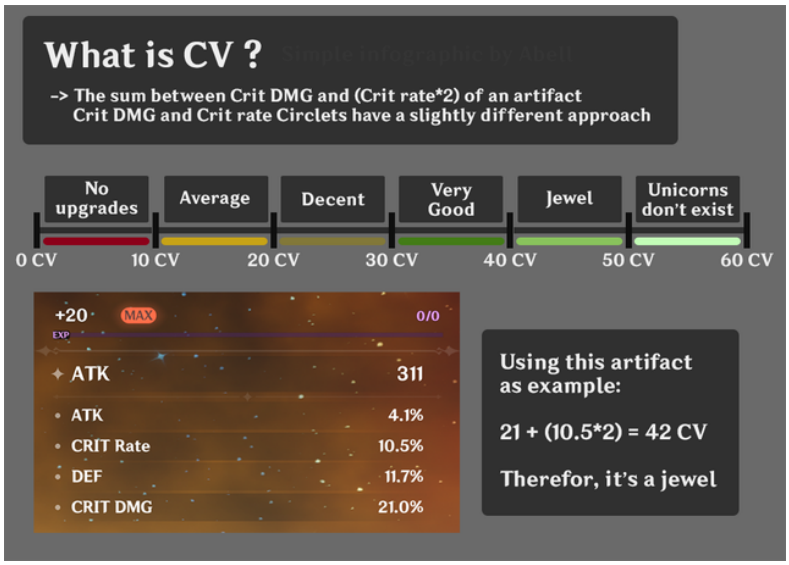
(Tiffany & Nicholas)

Even semester 2022

Computer Science Program

Binus University International

Problem Description:



Genshin Impact is currently one of the most popular games in the world with approximately 9 million users being active daily out of the 60 million registered users. One thing that we as experienced players observed is that many players (despite being new or veteran players) struggle in deciding on what makes an artifact worth keeping. Artifacts essentially act as armor for your characters to equip as it improves the stats of the character that equips it. In other words, it will assist your characters to deal more damage and survive longer. Obtaining artifacts is done through consuming resins (in-game currency) which takes time to replenish.

There are many aspects that can contribute to the value of one artifact, and one of the main factors is the crit value an artifact possesses. The higher the crit value an artifact has, the more valuable it becomes as it is difficult to obtain artifacts with high crit values since it is all based on the game's Random Number Generated system. Lots of players that lack knowledge on managing artifacts would usually level up and keep artifacts that are deemed as "worthless" as the artifacts either possess 0-low crit value. This not only wastes one's time and resources, but it may hinder one's progress since the game gets progressively more difficult when you level up and are required to fight higher-leveled enemies.

Although the solution will not improve the chances of getting a high crit value artifact, hopefully, it will assist the confused player base of the Genshin community in managing the artifacts they have through our Genshin Artifact Rater.

Proposed Alternative Data Structures:

- **Binary Search Tree:**

  The first data structure we thought of would be the BST or the Binary Search Tree. This data structure allows us to have a comparable key to each node, where each key in any node is larger than the keys in all nodes in the left subtree and smaller than the keys in all nodes of the right subtree. The reason why a binary search tree would be great for this program is that binary search trees allow for fast insertion and deletion when balanced. BST is also really efficient, and the code is simple compared to other data structures. This data structure is great for our program as our program involves lots of insertions, deletions, and indexing of each object, however, a BST comes with the drawback of being slower when accessing an element compared to an array.

- **Linked List (doubly):**

  The second data structure we thought would suit to solve the problem is a doubly-linked list. It allows convenient access from a list node to the next node and other nodes ahead. This is through storing two pointers where one is the node following it and the other pointer to the node preceding it. Our artifact rater requires a data structure that would allow the user to store the data of their artifacts and allow them to add in between, before, or after a certain artifact. We also need a data structure that gives them the flexibility to update the artifacts' stats in case they decided to level it up and decides to use our program to re-evaluate the artifacts again. Another important operation is to allow the user to delete or remove an artifact from the list as our main objective is to have the users decide which artifacts to throw away based on our ratings.

Theoretical analysis of Binary Search Tree and how it affects our program.

Definition:

A binary search tree typically is a special binary tree with left subtrees containing only the keys which are lesser than the key of the node, meanwhile, the right subtree contains only the keys which are greater than the key of the node. The other left and right subtree also need to follow these binary search tree rules. The four basic operations of a BST are Searching, Insertion, Deletion, and Traversals.

Searching in a BST is done by comparing key values and determining whether they are equal to a root key, if so then the search would be successful, if lesser than the root key then search the key in the left subtree and if the key is greater than root key then search in the right subtree until finding the key value which is equal to a root key. If this process fails, the element would be determined not present in the BST. Given that the tree is balanced, it has a worst-case complexity of O(logN)

Insertion in a BST also involves the comparison of the key values. If the key value is lesser than or equal to the root key then go to the left subtree, find an empty space following the search algorithm and insert the data if the key is greater than the root key then go to the right subtree, find an empty space following the search algorithm and insert the data. Has a worst-case complexity of O(N).

Deletion in a BST uses a search algorithm to find the node. Then, find the number of children nodes to be deleted. If the node to be deleted is a leaf node, then delete it. If the node to be deleted has one child node then delete the node and place the child of the node at the position of the deleted node. If the node to be deleted has two children, find the inorder successor or inorder predecessor of the node according to the nearest value to the node that will be deleted. Replace the node with the in-order successor or predecessor. Has a worst-case complexity of O(N).

Traversal in a BST There are 4 ways to traverse a binary search tree.
Level Order Traversal: Traversing each node level by level in order of its appearance.
Pre-order Traversal: Traversing in the order of root and then left subtree and then right subtree.
Inorder Traversal: Traversing in the order of left subtree and then root and then right subtree.
Post Traversal: Traversing the nodes in the order of the left subtree and then the right subtree and then the root.

Application of Binary Search trees: Indexing, Searching, Implementation of other various data structures, implement dictionaries.

Advantages of Binary Search Tree: A binary search tree is fast in insertion and deletions when it is balanced which helps with our program as it utilizes these insertion and deletion operations a lot to add and remove multiple artifacts, and a binary search tree allows for fast and efficient executions.

Disadvantages of Binary Search Tree: The main disadvantage is that we should make sure that the implementation is a balanced tree, otherwise it may cause problems by making non-logarithmic searches and degenerating into a linear search on an array, which ruins the point of having the binary search tree. Another disadvantage that actually hurts our program is the fact that accessing an element in the binary search tree is slightly slower than accessing an element in an array. A BST can also be an imbalance or degenerated which can cause an increase in complexity

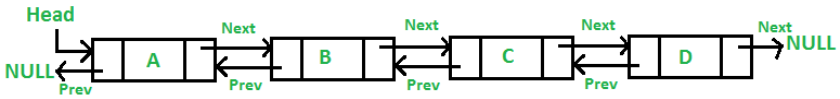
Theoretical analysis of Doubly Linked List and how it affects our program

Definition: A doubly linked list is considered a linear data structure and is a variation of a linked list where it makes navigation possible in both ways (forward and backward) easily. Each node apart from storing its data contains two fields called links where the first link points to the prev node while the second link points to the next node. The first and last node of the list has its prev and next link pointed to null to mark the start and end of a list.

- Link: can store a data called an element
- Prev: Each link contains a link to the previous link called prev
- Next: Each link contains a link to the next link called next
- LinkedList: has the connection link to the first link called first and the last link called last.

Basic Operations:

- **Insertion:** It is used to add new elements to the list which can be added at the beginning of the linked list, end of the linked list, or after a given node. The time complexity for insertion best and worse case for the insertion of an element is O(1).
- **Deletion:** This is used to remove an element from the list where the node to be deleted can be the head node, tail node, or any nodes in between. The time complexity for deletion best and worse case for deletion of an element is O(1).
- **Traversal:** This is used to access each element in the list. It is traversed when the contents of the linked list are printed out. It can traverse forwards and backward, meaning it can be accessed in both directions. The time complexity for traversal however is O(n) as the worst case.



Application of Doubly Linked list:

- Used in navigation systems traversing back and forward is required
- Used by browsers (visit webpages) with the back and forward button
- Applications for undo and redo
- Game deck of cards

Advantages of Doubly Linked List:

The doubly linked list has a bidirectional traversal which is not possible in a singly linked list. It is also easy to do the deletion operation compared to the singly linked list since all it takes is the

pointer to be deleted. It also makes memory reallocation simple and lastly, reversing the linked list is also easy. It is more efficient compared to a singly linked list.

Disadvantages of Doubly Linked List:

It uses more memory in comparison to a singly-linked list or an array implementation. It also doesn't allow direct access since the elements in the memory are stored randomly making the elements have to be accessed sequentially.

Time complexity comparison (Worst Case)

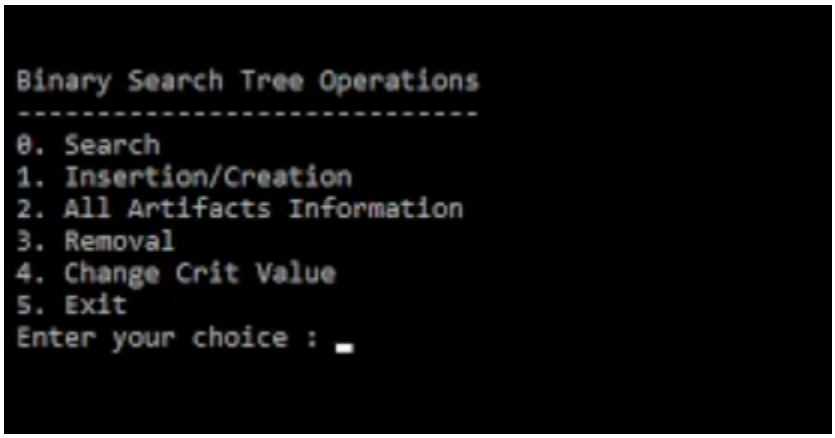
Data Structure	Access	Insertion	Deletion	Search
Binary Search Tree	O(N)	O(N)	O(N)	O(N)
Doubly Linked List	O(N)	O(1)	O(1)	O(N)

Time complexity comparison (Best case)

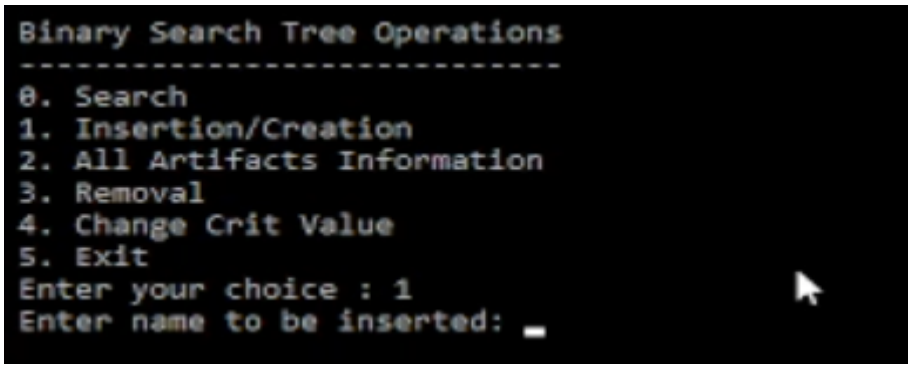
Data Structure	Access	Insertion	Deletion	Search
Binary Search Tree	O(logn)	O(logn)	O(logn)	O(logn)
Doubly Linked List	O(1)	O(1)	O(1)	O(1)

Program manual BST + Results!

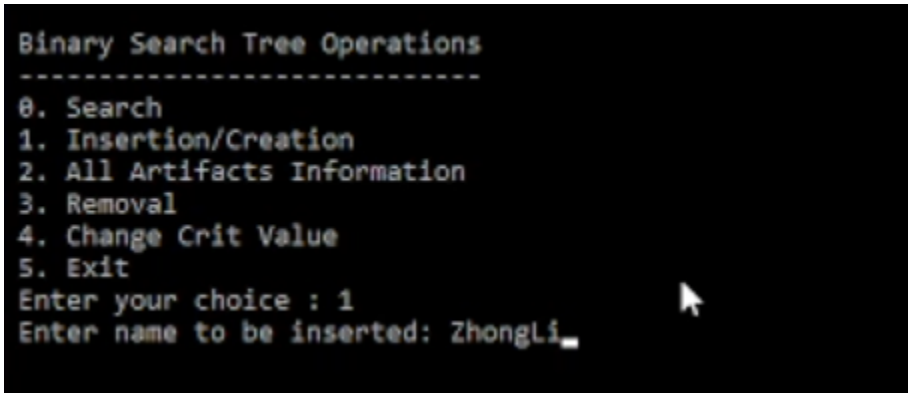
Main menu



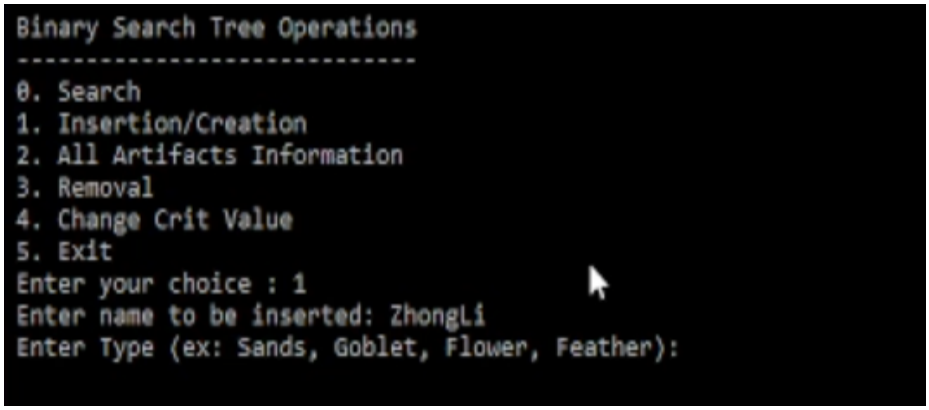
Step 1: Choice 1 will then prompt you with multiple information insertions starting with the name of the artifact.



Enter a name for the artifact

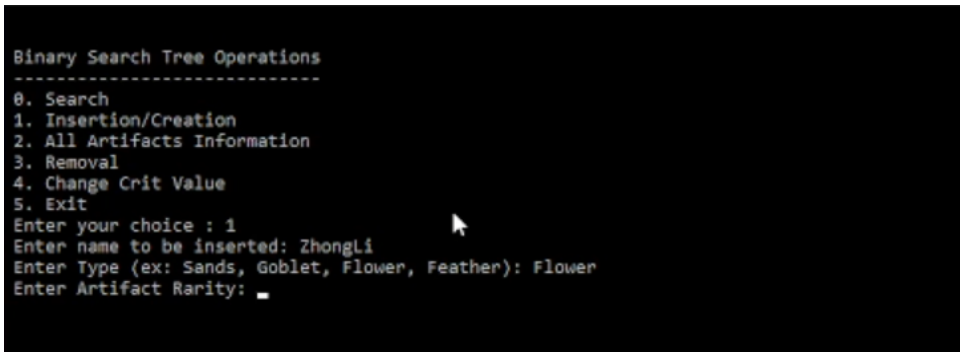


After enter the artifact time, you will have to enter an artifact type



Enter the artifact type

After entering the artifact you will have to enter the artifact rarity



Enter Artifact rarity

```
Binary Search Tree Operations
-----------------------------
0. Search
1. Insertion/Creation
2. All Artifacts Information
3. Removal
4. Change Crit Value
5. Exit
Enter your choice : 1
Enter name to be inserted: ZhongLi
Enter Type (ex: Sands, Goblet, Flower, Feather): Flower
Enter Artifact Rarity: 5
```

After entering the artifact rarity, you have to enter the artifact level.

```
Binary Search Tree Operations
-----------------------------
0. Search
1. Insertion/Creation
2. All Artifacts Information
3. Removal
4. Change Crit Value
5. Exit
Enter your choice : 1
Enter name to be inserted: ZhongLi
Enter Type (ex: Sands, Goblet, Flower, Feather): Flower
Enter Artifact Rarity: 5
Enter Artifact Level:
```

Enter the artifact level.

```
Binary Search Tree Operations
-----------------------------
0. Search
1. Insertion/Creation
2. All Artifacts Information
3. Removal
4. Change Crit Value
5. Exit
Enter your choice : 1
Enter name to be inserted: ZhongLi
Enter Type (ex: Sands, Goblet, Flower, Feather): Flower
Enter Artifact Rarity: 5
Enter Artifact Level: 20
```

After entering the artifact level you have to enter the crit rate now in integers and doubles.

```
Binary Search Tree Operations
-----------------------------
0. Search
1. Insertion/Creation
2. All Artifacts Information
3. Removal
4. Change Crit Value
5. Exit
Enter your choice : 1
Enter name to be inserted: ZhongLi
Enter Type (ex: Sands, Goblet, Flower, Feather): Flower
Enter Artifact Rarity: 5
Enter Artifact Level: 20
Enter crit rate (integer):
```

Enter A crit rate in either integer or double.

```
Binary Search Tree Operations
-----------------------------
0. Search
1. Insertion/Creation
2. All Artifacts Information
3. Removal
4. Change Crit Value
5. Exit
Enter your choice : 1
Enter name to be inserted: ZhongLi
Enter Type (ex: Sands, Goblet, Flower, Feather): Flower
Enter Artifact Rarity: 5
Enter Artifact Level: 20
Enter crit rate (integer): 3.5
```

After the crit rate you have to enter the crit damage value of the artifact substats

```
Binary Search Tree Operations
-----------------------------
0. Search
1. Insertion/Creation
2. All Artifacts Information
3. Removal
4. Change Crit Value
5. Exit
Enter your choice : 1
Enter name to be inserted: ZhongLi
Enter Type (ex: Sands, Goblet, Flower, Feather): Flower
Enter Artifact Rarity: 5
Enter Artifact Level: 20
Enter crit rate (integer): 3.5
Enter crit damage (integer):
```

Enter the crit damage

```
Binary Search Tree Operations
-----------------------------
0. Search
1. Insertion/Creation
2. All Artifacts Information
3. Removal
4. Change Crit Value
5. Exit
Enter your choice : 1
Enter name to be inserted: ZhongLi
Enter Type (ex: Sands, Goblet, Flower, Feather): Flower
Enter Artifact Rarity: 5
Enter Artifact Level: 20
Enter crit rate (integer): 3.5
Enter crit damage (integer): 35.5
```

Back to the main menu

```
Binary Search Tree Operations
-----------------------------
0. Search
1. Insertion/Creation
2. All Artifacts Information
3. Removal
4. Change Crit Value
5. Exit
Enter your choice :
```

Step 2: After choosing option 2 the program will show the artifact you have stored in the binary search tree.

```
Binary Search Tree Operations
-----------------------------
0. Search
1. Insertion/Creation
2. All Artifacts Information
3. Removal
4. Change Crit Value
5. Exit
Enter your choice : 2

                                    Artifact  Information
-------------------------------------------------------------------------------------------------
LEVEL         RARITY          NAME            CRITVALUE               TYPE            RATING
20            5               ZhongLi         42                      Flower          Jewel
```

Here is what happens when you create a second artifact

```
Binary Search Tree Operations
-----------------------------
0. Search
1. Insertion/Creation
2. All Artifacts Information
3. Removal
4. Change Crit Value
5. Exit
Enter your choice : 1
Enter name to be inserted: HuTao
Enter Type (ex: Sands, Goblet, Flower, Feather): Feather
Enter Artifact Rarity: 5
Enter Artifact Level: 20
Enter crit rate (integer): 3.5
Enter crit damage (integer): 7.8
```

Printing it out looks like this:

```
Binary Search Tree Operations
----------------------------
0. Search
1. Insertion/Creation
2. All Artifacts Information
3. Removal
4. Change Crit Value
5. Exit
Enter your choice : 2

                                    Artifact  Information
-------------------------------------------------------------------------------------------------
 LEVEL          RARITY          NAME            CRITVALUE               TYPE            RATING
 20             5               HuTao           14                      Feather         Average
 20             5               ZhongLi         42                      Flower          Jewel
```

Step 3: **Option zero lets you search based on the name of the Artifact**

```
 Binary Search Tree Operations
 -----------------------------
 0. Search
 1. Insertion/Creation
 2. All Artifacts Information
 3. Removal
 4. Change Crit Value
 5. Exit
 Enter your choice : 0
 Enter the name of the Artifact to search for:
ZhongLi
The Crit value for ZhongLi is 42
```

Step 4: Deleting an element using the artifact name

```
 Binary Search Tree Operations
 -----------------------------
 0. Search
 1. Insertion/Creation
 2. All Artifacts Information
 3. Removal
 4. Change Crit Value
 5. Exit
 Enter your choice : 3
 Enter data to be deleted : HuTao
```

Resulting print out

```
Binary Search Tree Operations
----------------------------
0. Search
1. Insertion/Creation
2. All Artifacts Information
3. Removal
4. Change Crit Value
5. Exit
Enter your choice : 2

                                    Artifact  Information
-------------------------------------------------------------------------------------------------
 LEVEL          RARITY          NAME            CRITVALUE               TYPE            RATING
 20             5               ZhongLi         42                      Flower          Jewel
```

Step 5: Option 4 lets you change the crit value for a stats upgrade.

```
Binary Search Tree Operations
-----------------------------
0. Search
1. Insertion/Creation
2. All Artifacts Information
3. Removal
4. Change Crit Value
5. Exit
Enter your choice : 4
Enter the name of the Artifact of which the crit value you wish to change:
ZhongLi

Enter the new Crit Value:
50

Crit value changed successfully.
```

This will show after printing it out

```
Binary Search Tree Operations
-----------------------------
0. Search
1. Insertion/Creation
2. All Artifacts Information
3. Removal
4. Change Crit Value
5. Exit
Enter your choice : 2

                                    Artifact  Information
-------------------------------------------------------------------------------------------------
 LEVEL          RARITY          NAME            CRITVALUE               TYPE            RATING
 20             5               ZhongLi         50                      Flower          Jewel
```

Step 6: **Type 5 to exit.**

```
Binary Search Tree Operations
-----------------------------
0. Search
1. Insertion/Creation
2. All Artifacts Information
3. Removal
4. Change Crit Value
5. Exit
Enter your choice : 5
```

Program manual Doubly Linked List + Results!

```
Welcome to Kokopium's Genshin Impact Artifact Rater!

Which operation would you like to perform? Enter 1-9. Enter 0 to exit.
1. Append Artifact
2. Prepend Artifact
3. Insert Artifact After
4. Delete Artifact
5. Search for Artifact
6. Update Artifact
7. Print All Artifacts
8. Rate Artifact
9. Clear Screen
```

Step 1: Run the program and view the options. You may choose to append or prepend an artifact.

Other options include: inserting an artifact after another existing artifact, deleting an artifact by name, searching for an artifact by name, updating an existing artifact's crit rate, crit damage, and level by inputting the artifact's name. You may also choose to print all the artifacts stored in the list or have our program rate the artifact for you!

Step 2: Enter 1 to add an artifact to the list. Proceed to enter all required data such as name, type, crit rate, crit damage, level, and rarity of the artifact.

Result: The artifact has been successfully added to the list.

```
This is the APPEND Artifact Operation
Enter a new artifact name:
Emblem
what is the artifact type? (ex: sands, hat, goblet, flower, feather)
sands
Enter a new artifact's crit rate:
2.8
Enter a new artifact's crit damage:
7.8
Enter a new artifact's level:
20
Enter a new artifact's rarity:
5
Artifact Appended as Head Artifact
```

Step 3:

Enter 2 to prepend/add an artifact at the start of the list. Proceed to enter all required data.

```
PREPEND Artifact Operation
Enter the data of the Artifact that needs to be Prepended
Enter a new artifact name:
Shimenawa
what is the artifact type? (ex: sands, hat, goblet, flower, feather)
goblet
Enter the new artifact's crit rate:
5.8
Enter the new artifact's crit damage:
27.3
Enter the new artifact's level:
20
Enter the new artifact's rarity:
5
Artifact Prepended
```

Step 4: Enter 3 to insert an artifact after another artifact (insertion in between). Proceed to enter all required data.

```
INSERT Artifact AFTER Operation
Enter the name of an existing Artifact which you want to Insert this New Artifact after:
Shimenawa
Enter the name of the new artifact you want to insert:
Maidens
what is the artifact type? (ex: sands, hat, goblet, flower, feather)
flower
Enter the new artifact's crit rate:
15.5
Enter the new artifact's crit damage:
12.4
Enter the new artifact's level:
20
Enter the new artifact's rarity:
5
Artifact Inserted in Between
```

Result (If the artifact does not exist):

```
3
INSERT Artifact AFTER Operation
Enter the name of an existing Artifact which you want to Insert this New Artifact after:
Pale
Enter the name of the new artifact you want to insert:
Flame
what is the artifact type? (ex: sands, hat, goblet, flower, feather)
hat
Enter the new artifact's crit rate:
14.2
Enter the new artifact's crit damage:
7.9
Enter the new artifact's level:
20
Enter the new artifact's rarity:
5
No Artifact exists with the artifact name Pale
```

Step 5: Enter 7 to print the artifact list.

Result:

Can be seen that all the artifacts added previously are inside the list and everything is printed out properly. The list proves that step 1, 2, and 3 was a success. In step 1, “Emblem” is added to the list however it is now at the bottom since in step 2 we prepended “Shimenawa” which is why it is at the top. In step 3, we decided to insert “Maidens” after “Shimenawa” which is why it is in the middle as shown in the picture.

```
7

YOUR ARTIFACTS LIST :
<><><><><><><><><><><><><><><><><><><>
Artifact name: Shimenawa
Type: goblet
Crit value: 38.9
Level: 20
Rarity: 5
<><><><><><><><><><><><><><><><><><><>
Artifact name: Maidens
Type: flower
Crit value: 43.4
Level: 20
Rarity: 5
<><><><><><><><><><><><><><><><><><><>
Artifact name: Emblem
Type: sands
Crit value: 13.4
Level: 20
Rarity: 5
<><><><><><><><><><><><><><><><><><><>
```

Step 6: Enter 5 to search for an artifact by its name. Proceed to enter the artifact name you are looking for.

```
5
SEARCH Artifact By artifact name Operation
Enter the artifact's name that you want to search for:
Shimenawa

The Artifact Shimenawa is found in the List!

<><><><><><><><><><><><><><><><><><><><>
Artifact name: Shimenawa
Type: goblet
Crit value: 38.9
Level: 20
Rarity: 5
<><><><><><><><><><><><><><><><><><><><>
```

Result (if found): The artifact "Shimenawa" is being searched here and can be seen that it has been found successfully since it's already inside the list.

Result (If not found): The artifact "Crimson" is being searched here and can be seen that if it is not in the list, the program will give the results below.

```
5
SEARCH Artifact By artifact name Operation
Enter the artifact's name that you want to search for:
Crimson
No Artifact exists with the artifact name: Crimson
```

Step 7: Enter 6 to update the artifact's crit rate, crit damage, and level. Proceed to enter the artifact's name that you would like to update.

```
Update Artifact By artifact name Operation
Enter the artifact's name that you want to update the stats for:
Emblem
Enter the updated crit rate:
8.2
Enter the updated crit damage:
7.8
Enter the updated level:
20
The Artifact's crit value and level has been Updated Successfully
```

```
YOUR ARTIFACTS LIST :
<><><><><><><><><><><><><><><><><><><><>
Artifact name: Shimenawa
Type: goblet
Crit value: 38.9
Level: 20
Rarity: 5
<><><><><><><><><><><><><><><><><><><><>
Artifact name: Maidens
Type: flower
Crit value: 43.4
Level: 20
Rarity: 5
<><><><><><><><><><><><><><><><><><><><>
Artifact name: Emblem
Type: sands
Crit value: 24.2
Level: 20
Rarity: 5
<><><><><><><><><><><><><><><><><><><><>
```

Result (if successfully updated):

We can enter 6 to print out the list again and check if the artifact got updated. As can be seen, the crit value for artifact "Emblem" has changed from 13.4 to 24.2.

Result (if the artifact to be updated is not found in the list):

```
6
Update Artifact By artifact name Operation
Enter the artifact's name that you want to update the stats for:
Baka
Enter the updated crit rate:
5.1
Enter the updated crit damage:
29.3
Enter the updated level:
20
Cannot find the artifact with the name : Baka
```

The artifact "Baka" cannot be updated because it is not in the list in the first place.

Step 8: Enter 4 to do the deletion. I decided to delete the first artifact in the list which happens to be Shimenawa.

Result (if artifact name exists):

```
DELETE Artifact By artifact name Operation
Enter the name of the Artifact to be deleted:
Shimenawa
Artifact Shimenawa has been successfully deleted.
```

```
YOUR ARTIFACTS LIST :
<><><><><><><><><><><><><><><><><><><>
Artifact name: Maidens
Type: flower
Crit value: 43.4
Level: 20
Rarity: 5
<><><><><><><><><><><><><><><><><><><>
Artifact name: Emblem
Type: sands
Crit value: 24.2
Level: 20
Rarity: 5
<><><><><><><><><><><><><><><><><><><>
```

As you can see, the artifact "Shimenawa" has been removed from the list and we can print the list to see that it's not there anymore.

Result (if the artifact name to be deleted doesn't exist):

```
4
DELETE Artifact By artifact name Operation
Enter the name of the Artifact to be deleted:
uwu
No Artifact exists with artifact name: uwu
```

Step 9: Enter 8 to rate an artifact that is already in the list.

Result: The artifact rated in this example is Maidens which is rated as Jewel according to its crit value which happens to be 43.4.

```
8
Rate Artifact By artifact name Operation
Enter the artifact's name that you want us to rate:
Maidens
Our rating:
Jewel
```

(Optional step): press 7 anytime to clear the screen if it's too much.

```
What operation do you want to perform? Select Option number. Enter 0 to exit.
1. append Artifact
2. prepend Artifact
3. insert Artifact After
4. delete Artifact By Name
5. update Artifact By Name
6. print Artifact List
7. Clear Screen
```

Step 10: Enter 0 to exit the program!

```
What operation do you want to perform? Select Option number. Enter 0 to exit.
1. append Artifact
2. prepend Artifact
3. insert Artifact After
4. delete Artifact By Name
5. update Artifact By Name
6. print Artifact List
7. Clear Screen
0
PS C:\Users\tiffa\Documents\BINUS\CS\Data Structures\Assignments>
```

Runtime

BST Operation runtimes

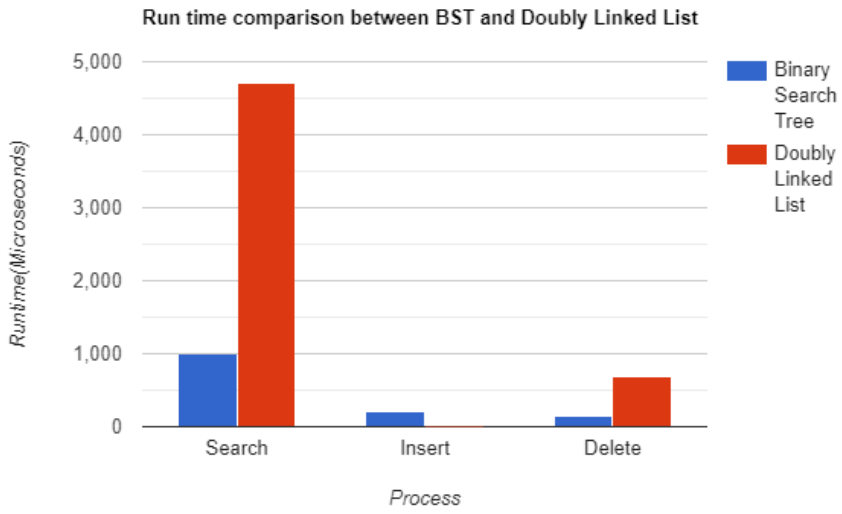
Data size: 100

Operations	Time in Microsec
Search	100, 2000,1000
Print	62000, 43000, 32000,32000
Removal	100,300
Insert	100,200

Doubly Linked List run times

Data size: 100

Operations	Time in Microseconds
Search	4408, 4202, 5606
Insert (append)	4, 4, 3
Delete (middle)	790, 752, 539



A binary search tree is more efficient than the doubly linked list when it comes to searching for an element. Insertion however is faster in a Doubly Linked List.

Teamwork & Contribution:

Nicholas:

- Created the BST program
- Theoretical analysis of Binary Search Tree and how it affects our program
- Recorded Video + editor
- Demo Video voiceover
- Documentation on BST Manual
- Proposed Alternative Data Structure (Binary search Tree) Documentation

Tiffany

- Created the Doubly Linked List program
- Theoretical analysis of Doubly Linked List and how it affects our program
- Documentation on Doubly Linked List Manual
- Demo video voiceover
- Problem Description Documentation
- Proposed Alternative Data Structure (Doubly Linked List) Documentation

File Links:

GitHub: https://github.com/Pandalmation/DS-Final-Project

Video Demo: https://youtu.be/zjQqwf4NGEc

Resources:

https://www.geeksforgeeks.org/applications-advantages-and-disadvantages-of-binary-search-tree/

https://www.geeksforgeeks.org/advantages-disadvantages-and-uses-of-doubly-linked-list/

https://www.daniweb.com/programming/software-development/threads/327525/binary-search-tree-file-reading

https://www.geeksforgeeks.org/complexity-different-operations-binary-tree-binary-search-tree-avl-tree/#:~:text=In%20general%2C%20time%20complexity%20is%20O(h).,complexity%20is%20O(h).

https://www.studytonight.com/data-structures/doubly-linked-list

https://www.geeksforgeeks.org/time-complexities-of-different-data-structures/