

**Software Design 4.2**

**By**

**Gourav Ganguly**

**(A00268742)**

**Mr. Declan Byrne, Lecturer**

**Bachelor of Engineering (Hons.)**

**in SOFTWARE ENGINEERING**

**ATHLONE INSTITUTE OF TECHNOLOGY**

**Overview**

Nowadays people use different online services/subscriptions like Netflix, Amazon Prime, etc. And every user subscribes to their own favorite services. So, there is a need for managing the subscriptions for which I have created Subscription Management System through which we can perform CRUD operations for the User and his/her subscriptions. It also has an Admin panel from where we can see the details of each Users and only Admin has the access of deleting the user.

A user has two main details:

1. UserName (Key)
2. Subscriptions (Value)

For storing the above details, I have used TreeMap and for storing unique subscriptions a LinkedHashSet object is passed as a value in the TreeMap.



Some features:

* CRUD on User and subscriptions
* Data stored is Serialized (TreeMap already implements Serializable)
* Data displayed in Tables
* Interactive user interface
* Separate Customer and Admin user interfaces
* Unique users and subscriptions (No duplicate subscription)

Collection framework classes used to develop the system are:

1. TreeMap
2. LinkedHashSet

**Key Classes and Code Snippets**

**Project Structure:**

A screenshot of a cell phone

Description automatically generated

**Key classes:**

* **Home.java**: Contains the GUI and buttons actions
* **TreeMapStorage.java**: Contains Data structures for storing data, methods for CRUD, methods for dataModel for tables and FileInput and FileOutput for Serializing.



Fig. TreeMapStorage Class snippet



Fig. SearchData class which returns model for GUI table



Fig. Putting Data in File using Serializable



Fig. Adding UserName and Subscription method



Fig. Other methods in TreeMapStorage class

**Collection Framework classes used and their performances**

1. **TreeMap:** The TreeMap in Java is used to implement Map interface and NavigableMap along with the Abstract Class. The map is sorted according to the natural ordering of its keys, or by a Comparator provided at map creation time, depending on which constructor is used. This proves to be an efficient way of sorting and storing the key-value pairs.

**Performance:** TreeMap offers **O(log N)** lookup and insertion. Keys are ordered, so if you need to iterate through the keys in sorted order, you can. This means that keys must implement the Comparable interface. TreeMap is implemented by a Red-Black Tree.

1. **LinkedHashSet:** A LinkedHashSet is an ordered version of HashSet that maintains a doubly-linked List across all elements. When the iteration order is needed to be maintained this class is used. When iterating through a HashSet the order is unpredictable, while a LinkedHashSet lets us iterate through the elements in the order in which they were inserted. When cycling through LinkedHashSet using an iterator, the elements will be returned in the order in which they were inserted.

**Performance:** LinkedHashSet gives insertion, removing and retrieving operations performance in order **O(1).**

**Comparison between HashSet and LinkedHashSet:**

Total time to insert 1000 elements in HashSet in sec : 3752794  
Total time to insert 1000 elements in LinkedHashSet in sec : 1293794

**Screenshots**

A screenshot of a cell phone

Description automatically generated

Fig. Home Screen

A screenshot of a cell phone

Description automatically generated

Fig. Add Customer and subscription

A screenshot of a cell phone

Description automatically generated

Fig. Edit UserName or delete subscription

A screenshot of a social media post

Description automatically generated

Fig. Admin Panel

**Conclusion and problems encountered**

While doing this project I encountered various problems such as:

* Structuring and formatting data in TreeMap
* Deciding the correct data structure for my application
* Making data model from TreeMap

Through this project I understood how data structures is important in solving various real-life problems an improve it’s efficiency and robustness. By using the collection framework we can easily implement these complex data structures which helps us save time and memory.