**CPT-287 Group Project Report**

**By: Sam Atienza, Ryan Schoonover, Seth Wolf, Zach Deall**

**Project 1A: Movie Management System**

**System Design**

This program runs off a text-based menu in the console. Each function of the program is controlled by user input. For example, if the user were to input “D”, the program would display the movies. This system utilizes many data structures in the java language including linked lists, iterators, and deques. In this project, the linked lists data structure was used twice. One linked list was used to hold movies that had been received, but not released, and another linked list was used to hold movies that had been received and released. These linked lists were navigated via the use of the next data structure, iterators. In this project, four iterators were used. The first iterator, it1, was used to help navigate through the linked list responsible for holding the released movies. The second iterator was responsible for helping in the navigation of the linked list holding received movies. The third iterator was created to help navigate release. Finally, the last iterator, it2 was used in the edit class to help navigate through the text file and assist in the retrieval of the certain fields. Along with linked lists and iterators, deques were also used in this project. Many deques were used for various purposes, but most were utilized for the purpose of copying other data structures as our team thought this would be the most efficient way to do so.

**UML Diagram**

Graphical user interface, application

Description automatically generated

**Test Cases #1**

|  |  |  |  |
| --- | --- | --- | --- |
| **Step No.** | **Test Input** | **Expected Output** | **Actual Output** |
| **1** | D | “You have selected to display movies.”  Movies Released:  moviesReleased  Movies Received:  moviesReceived | “You have selected to display movies.”  Movies Released:  moviesReleased  Movies Received:  moviesReceived |
| **2** | C | “You have selected to count movies.”  “Enter given date -> “ | “You have selected to count movies.”  “Enter given date -> “ |
| **3** | 06/20/2019 | “There is one released movie before 06/20/2019” | “There is one released movie before 06/20/2019” |
| **4** | X | “You have selected to terminate the program.” | “You have selected to terminate the program. |

**Test Case #2**

|  |  |  |  |
| --- | --- | --- | --- |
| **Step No.** | **Test Input** | **Expected Output** | **Actual Output** |
| **1** | A | “You have selected to add a movie.” | “You have selected to add a movie.” |
| **2** | The Boy 2 | “Please enter movie’s name.” | “Please enter movie’s name.” |
| **3** | 02/21/2020 | “Please enter movie’s release date.” | “Please enter movie’s release date.” |
| **4** | Horror | “Please enter movie’s description.” | “Please enter movie’s description.” |
| **5** | 02/05/2020 | “Please enter movie’s receive date.” | “Please enter movie’s receive date.” |
| **6** | X | “You have selected to terminate the program.” | “You have selected to terminate the program. |

**Contributions and Future Improvements**

**Sam Atienza –** Theatre class, Text based menu, Project Repository, Project Report

**Ryan Schoonover –** Theatre class, Add class, Movies class

**Seth Wolf –** Theatre class, Add class, Count class, Display class, Movies class

**Zach Deall –** Theatre class, Edit class

While the system was built to be as efficient as possible, room for improvement always exists in systems like these. For starters, to reduce lines of code and redundancies in the program, the case system in Java could have been utilized instead of using multiple if statements. This system could have also been improved by having more than just the main method in the Theatre class. Having a unique class for each function the system is capable of executing only expands on the amount of storage required for this project, thus making it run slower in less powerful machines. Overall, the system was built to be as efficient as possible often utilizing data structures to execute user commands.