Data Structures
Create-Your-Own Final Project
Book Inventory & Checkout Tracker

Data Structures:

1. Hash Map

   The hash map was implemented to store and sort books by an author's name. The hash map resolves collisions by expanding, rather than using chaining or probing. This means every lookup is O(1) (when searching for an author, not their books) because there is no need to search through a linked list or probe for an entry. It is a mapping of author names to linked lists of books. To be clear, the linked list is NOT for chaining entries in the hash map, it is ONLY for storing the books given the author. The underlying is an array that is dynamically doubled as needed. Author names are hashed according to the sum of the ordinals of the characters in his or her name.

2. Linked List

   As described above, the linked list is exclusively used to store the books added by a given author. Every book node includes only the title of the book and the author's name is stored in the linked list itself (rather than the nodes)

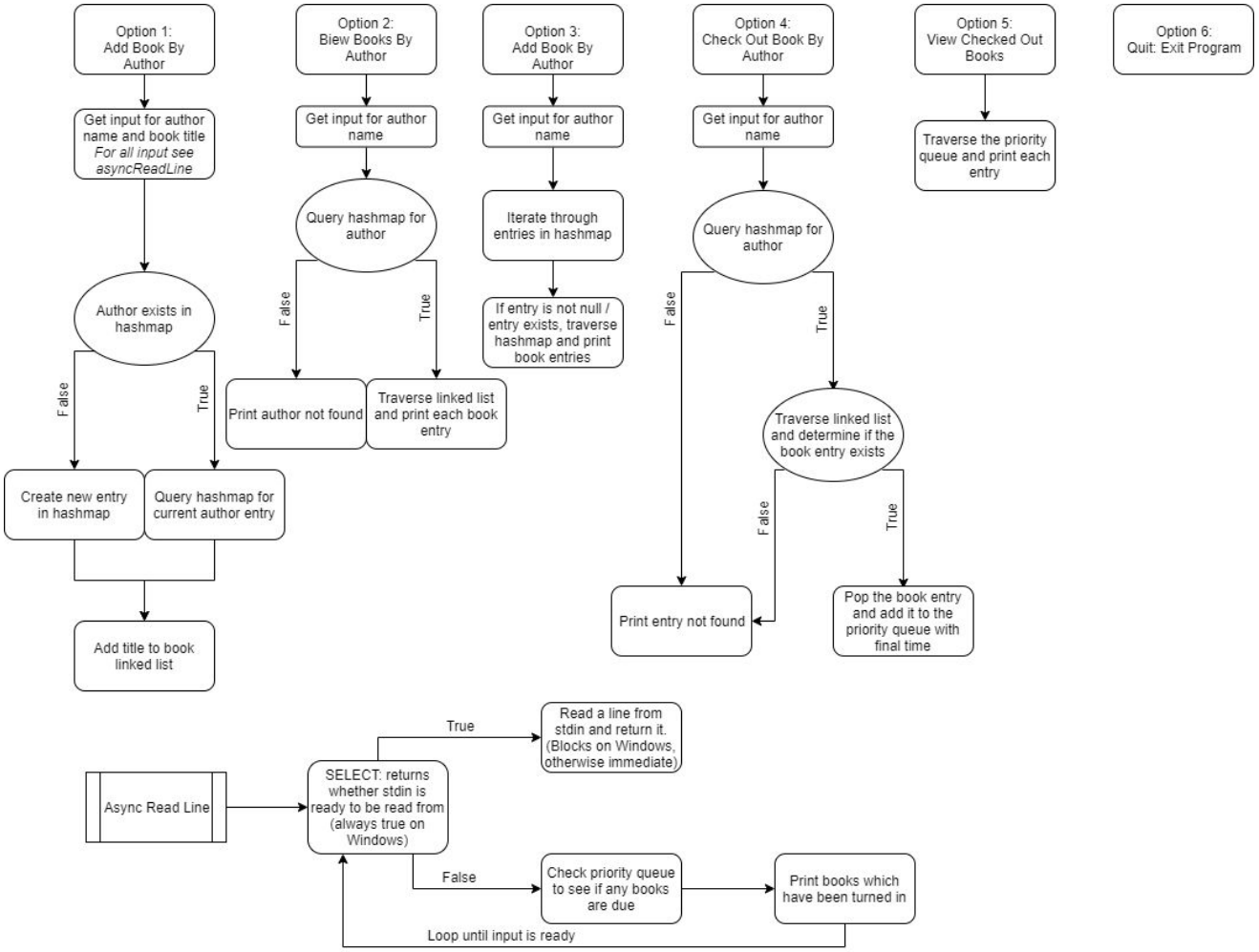
3. Heap

   The heap is used to implement a priority queue. This queue stores information on books which have been checked out. Each checked out book includes the author, the title, the time at which it will be turned in, and the name of the person who has checked it out. The priority queue is used to tell if any books are due, by checking whether the current time and seeing whether it has passed the projected time.

Methodology:

The program runs a CLI with 6 options:

1. Add a book by an author
2. View the available books by an author (not including checked out books)
3. View all available books
4. Check out a book
5. View checked out books
6. Quit



Data:

All data is input from standard in. For sample input please see `sample_input.txt`, simple inputs such as author names, book titles, user names, and generated times for the priority queue.

Results:

The final outputs are most simply seen by following some of the sample inputs. The system effectively manages adding new books and could easily be expanded to include much more in depth information and other dynamic inputs. There are a number of formatting details which are platform dependent. The application is best run on a Unix system. Windows will cause several issues with formatting which are difficult to prevent given the limitations on the way the OS handles IO. The application will work on Windows, but it will cause some unusual outputs.