ENGR 102 Final Project Proposal
Team 19-502

Archer, Joshua, Nick, Khoi, & Bryce

March 28, 2022

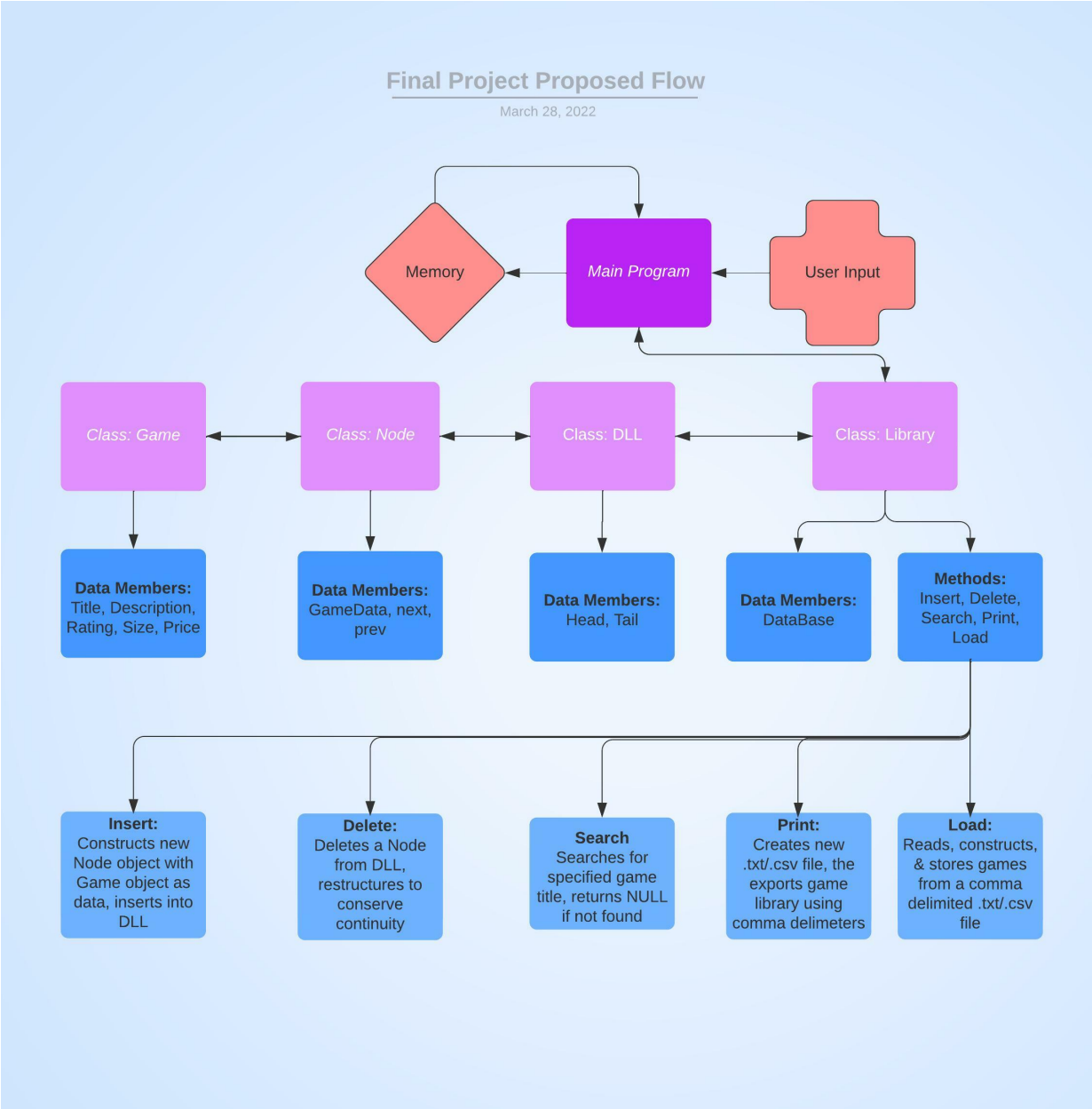
Game Library

For our final project, our team has decided to choose a game library. The game library would emulate what would typically be found on consoles within the stores. We got the idea when we saw the "inventory tracker" suggestion on Project_Ideas.pdf. Library type databases are an excellent way to exercise our knowledge of python to make a database which, to the user, seems very simplistic and easy to use, while taking advantage of python's most powerful tools in tandem with Data Structures & Algorithms to ensure each operation is as fast as it can be. We are all passionate about video games and have spent our lives looking at and using these game libraries, so to recreate one would allow us to better understand the mechanisms by which these libraries are built, the challenges that come with making a speed optimized game library, as well as how developers go about organizing a program of this size, making a simplistic looking, though secretly complex system for the users to find and learn about their favorite games.

The foundation of the library will be based off of either doubly linked lists within a vector of size 26 (for alphabetical sorting), or will be sorted alphabetically within some sort of binary search tree, depending on which makes more sense following more research into the topic. Currently, though, we believe doubly linked lists within a vector would most likely be the best option specifically for insertion or deletion, as it runs in constant time rather than O(log(n)) (n is the number of game objects stored in the library) like a binary search tree, though we are still doing research given searching/accessing specific elements for the vector/DLL pair is O(n) rather than the BSTs O(log(n)).

The main program will consist of 3 classes: Games, DLL (Doubly Linked List) OR BST (Binary search tree), and Library. The games class will represent each of the game entries, which will have data members such as title, description, rating, price, download size, etc. The DLL (or BST) class will be used as a custom data structure to speed up our program as it is faster in the worst-case scenario than a normal array. The library class is the highest abstract class within the program; it will host a vector of DLLs, while each DLL holds n class Game objects. Functions will include insert, delete, search, print, & load. Print will create a .txt/.csv document with all or specified parts of the game library, while load will take in a .txt/.csv of listed games, automatically sort each entry in lexicographical order. When "quit" is entered into the program, it will create a "memory" file, where each of the games added prior to the program shutting down will be saved in a .txt/.csv file to be loaded into the program when it launches next. Further features will be added, though we are deciding as we do much more research. Even if no other features are added, we are happy with the learning opportunity this challenge provides and are excited to begin.

Flowchart on next page



The graphic above is the prototype flowchart of the program's hierarchy