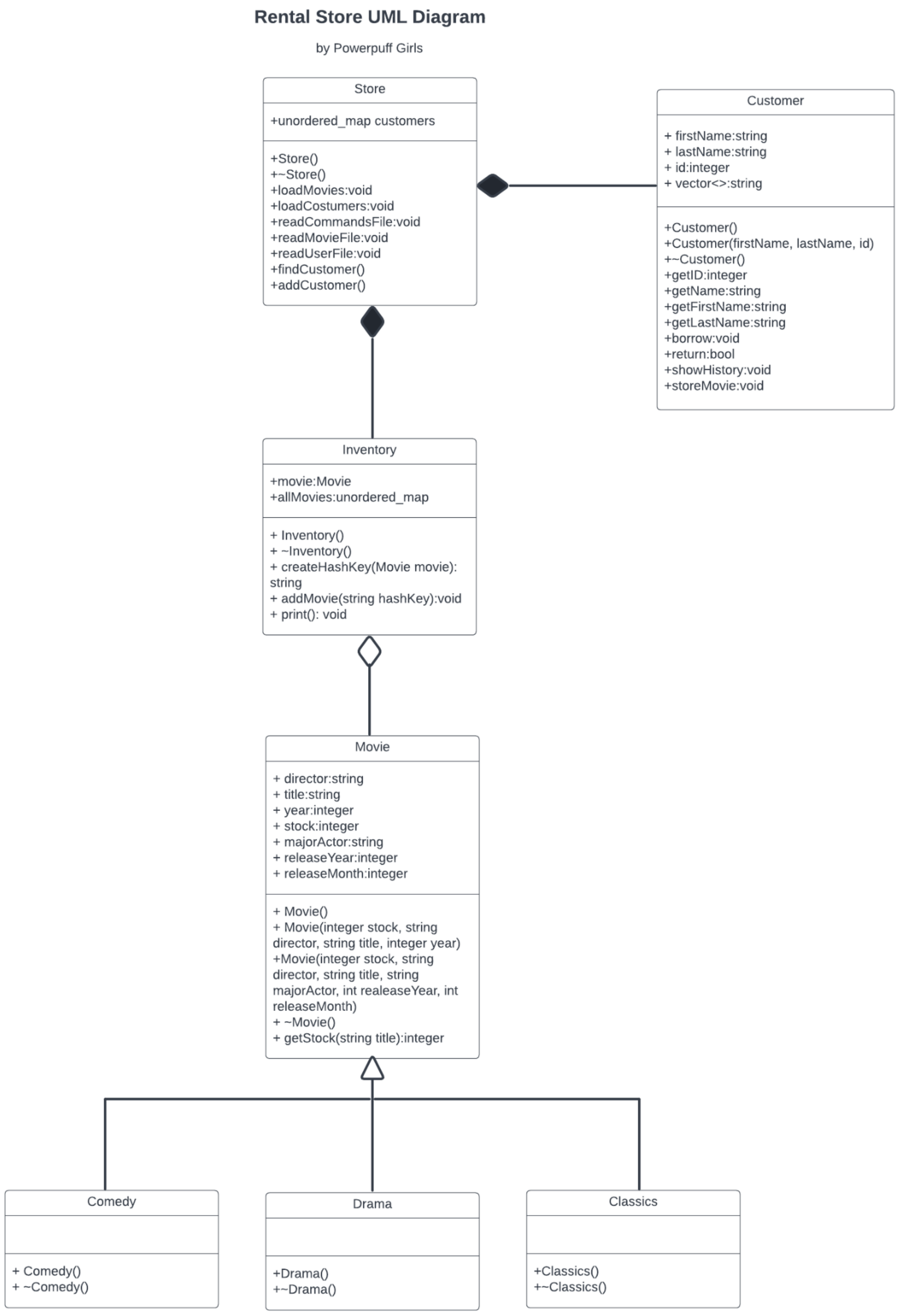
Group Name: Powerpuff Girls
Members: Aislinn O'Connell, Kirti Chandra, and Mahsa Mohajeri
Assignment #4 Proposal

Overview:

Our approach to creating a functioning rental store program is to follow an object-oriented approach that will utilize inheritances and classes. We decided that we will create a Store Class, which is essentially the main interface for the program. The Store Class will pass in the Customer Class and the Inventory Class, in order to be able to track the customers in the Store Class and add books to the Inventory Class. The Customer Class will store the data for a singular customer and will pass in no other classes. The Inventory Class will have access to the Movie Class in order to be able to store Movie objects. The Classics Class, Comedy Class, and Drama Class, will all inherit the Movie Class. Both the Comedy and Drama Classes share the same data fields, but the Classics Movie needs different data fields so there will be other classes added to the Classics Movie on top of the functionalities provided by the Movie Class. Our main will contain a created Store Class and then call upon the function to read in the customer and movie text files, then the commands text file.

Class Diagram:



Class Descriptions:

- Store: The interface where the data files are read in and assessed to call upon the needed object or command.
    - Store(): Default constructor.
    - ~Store(): Destructor
    - loadMovies(): Prints out the movies in the inventory.
    - loadCustomers(): Prints out the customers in the system.
    - readCustomerFile(): Reads in the data file and calls upon or creates the corresponding object.
    - readMovieFile(): Reads in the data file for the movies.
    - readCommandsFile(): Reads in the data file and creates the corresponding commands from the inherited objects.
    - unordered_map customers: Holds all the customers for the movie rental store.
    - findCustomer(): Returns the address to the specific customer in the hashmap.
    - addCustomer(): Creates a new Customer object and adds it to the hashmap.
- Customer: Stores the information for a single user of the rental store.
    - Customer(): Default constructor.
    - Customer(firstName, lastName, id): Creates a constructor with information to identify the customer.
    - ~Customer(): Destructor.
    - string firstName: Stores the user's first name.
    - string lastName: Stores the user's last name.
    - int id: Stores the id number of the user.
    - vector<string> history: Tracks which movies that the user has checked out and returned.
    - getID(): Returns the id number of the user.
    - getName(): Returns the user first and last name.
    - getFirstName(): Returns the first name of the user.
    - getLastName(): Returns the last name of the user.
    - borrow(): Adds movie back into total inventory.
    - return(): Removes movie from total inventory.

    - showHistory(): Prints out the total rental history of the user.
    - storeMovie(): Adds movie rental or return into the history vector.
- Inventory: Holds the details of what movies there are, their type, director, stock, year released, major actor
    - Inventory(): Default constructor.
    - ~Inventory(): Destructor.
    - unordered_map allMovies: A hashmap to store all the movies.
    - createHashKey(string director, string title, string year): Creates a hash key given the director, title, and year of a movie and returns the hash key in the form of a string.
    - addMovie(string hashKey): Adds a movie to the hashmap given the hash key associated with the movie data.
    - print(): Prints out the total inventory.
- Movie: Creates a default movie object (with no type)
    - Movie(): Default constructor.
    - Movie(director, title, year): Creates a movie object from the given details of a movie.
    - ~Movie(): Destructor.
    - string director: Stores the director of a movie.
    - string title: Stores the title of a movie.
    - integer year: Stores the year of release of a movie.
    - integer stock: stores the count of a particular movie.
    - getStock(string title): returns the stock value as an integer.
- Classics: Stores the information of a single classics movie
    - Classics(): Default constructor.
    - ~Classics(): Destructor
    - string majorActor: Stores the major actor of a movie
    - integer releaseYear: Stores the year of release of a movie
    - integer releaseMonth: Stores the month of release of a movie
- Comedy: Stores the information of a single comedy movie
    - Comedy(): Default constructor.

    - ~Comedy: Destructor.
- Drama: Stores the information of a single drama movie
    - Drama(): Default Constructor.
    - ~Drama(): Destructor.