

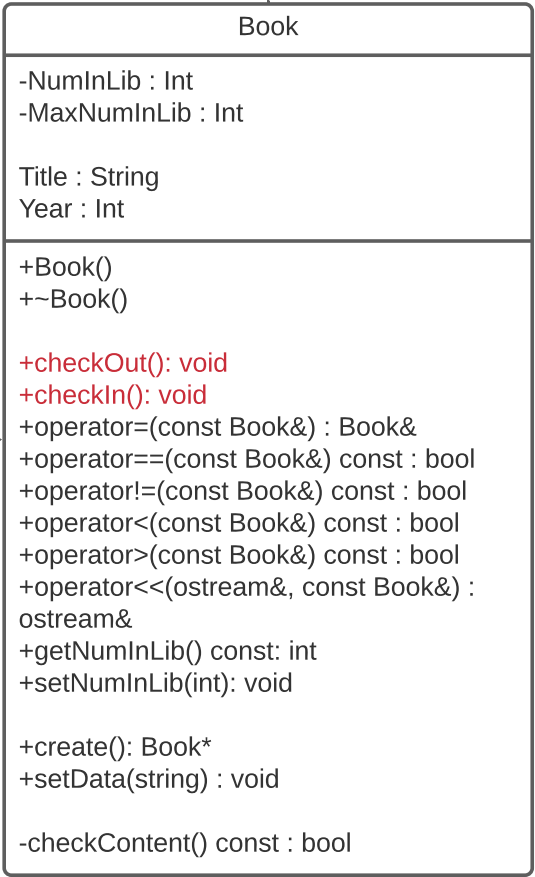
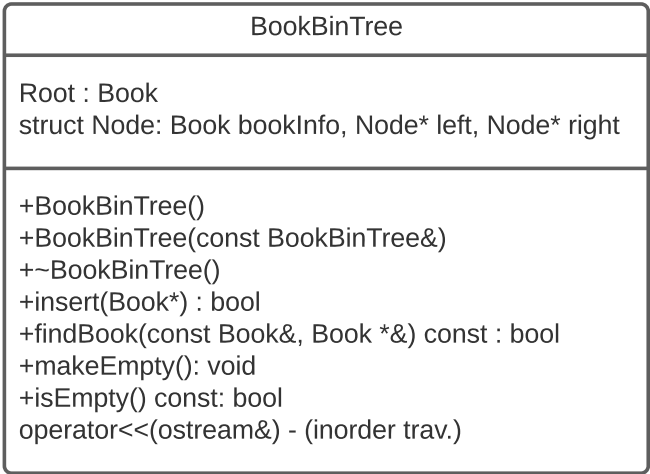
Make BookBinTree an Abstract class and use pointers!  
Create a Book pointer and instantiate it as a new Periodical, Fiction, or Childrens book

- Reuse the tree in 3 ways to hold the three different type of Books

Operators: Compared by date(year, then month) , then by title

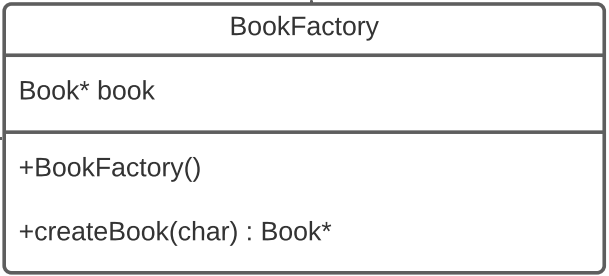
Operators: Compared by author, then by title

Operators: Comaped by title, then by author

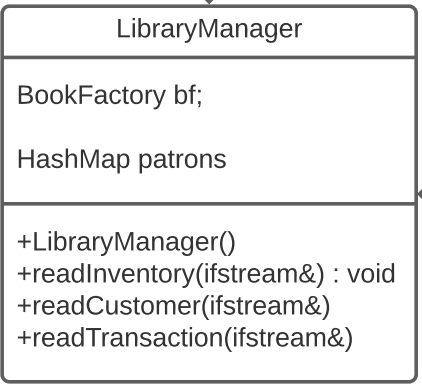


- Create simply creates a new object w/ no data in it
- setData is what populates the object with the correct data

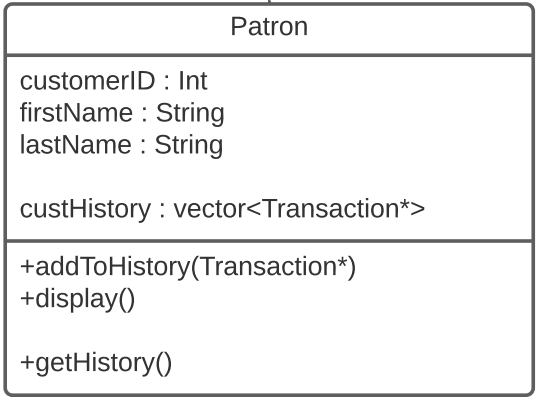
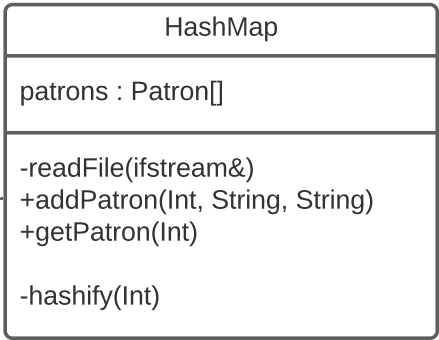
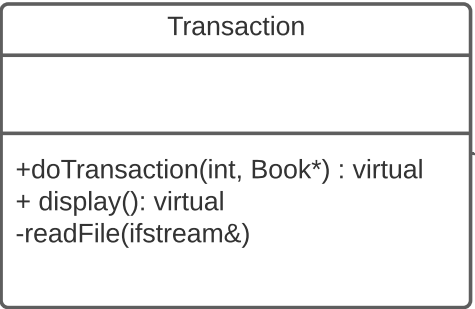
create() returns a new Book object. Example:  
**virtual Stuff\* create() const { return new AObject; }**  
  
checkContent - returns true if content is valid, returns false if not



- createBook in factory calls on create() in Book class and takes in parameter that is the type of book!



Having a Transaction Factory does not comply with Command design pattern



added vector to store transaction history for each customer

HashTable for Customer Transcations  
• column - patron #  
• rows - transaction #

- We are validating if the type of book is valid or not in the LibraryManager class
- All other validation is done in the book class itself

Need some kind of data structure to store the transctations done for each customer  
• idea: we could just use a vector??

