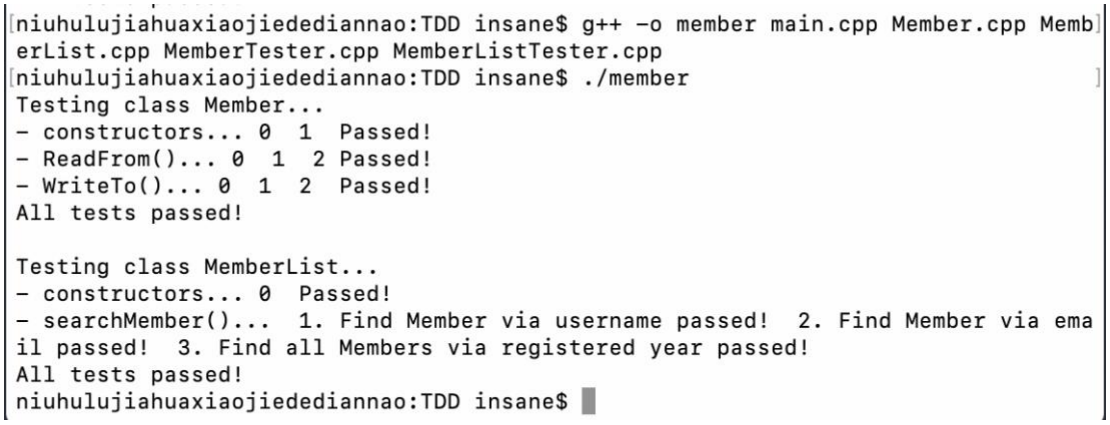
Name: Huayue Hua
USC ID: 9817961224
Email: huayuehu@usc.edu

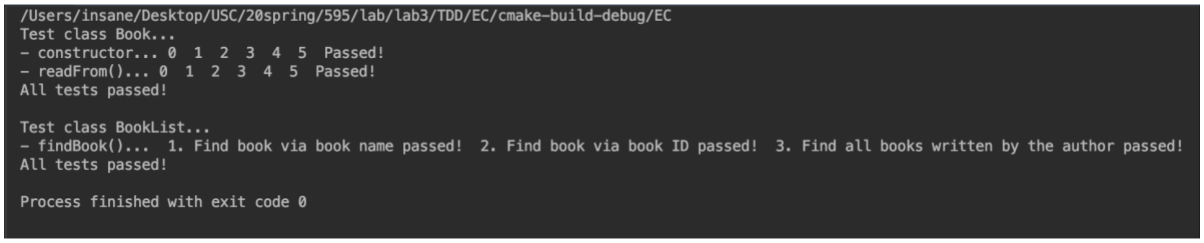
Lab3

1. To run Mandatory part for file “Lab3_p1_9817961224”, the following command should be entered:
   g++ -o lab3 Lab3_p1_9817961224.cpp
   ./lab3 input.txt

2. To run extra credit TDD Member update part, the following should be entered:
   g++ -o member main.cpp MemberTester.cpp Member.cpp MemberList.cpp MemberListTester.cpp
   ./member
   Result:


```
niuhulujiahuaxiaojiedediannao:TDD insane$ g++ -o member main.cpp Member.cpp Memb
erList.cpp MemberTester.cpp MemberListTester.cpp
niuhulujiahuaxiaojiedediannao:TDD insane$ ./member
Testing class Member...
- constructors... 0  1  Passed!
- ReadFrom()... 0  1  2 Passed!
- WriteTo()... 0  1  2  Passed!
All tests passed!

Testing class MemberList...
- constructors... 0  Passed!
- searchMember()...  1. Find Member via username passed!  2. Find Member via ema
il passed!  3. Find all Members via registered year passed!
All tests passed!
niuhulujiahuaxiaojiedediannao:TDD insane$
```

3. To run extra credit TDD Book part, the following should be entered:
   g++ -o main mainBook.cpp BookTester.cpp Book.cpp BookList.cpp BookListTester.cpp
   ./main
   Result:


```
/Users/insane/Desktop/USC/20spring/595/lab/lab3/TDD/EC/cmake-build-debug/EC
Test class Book...
- constructor... 0  1  2  3  4  5  Passed!
- readFrom()... 0  1  2  3  4  5  Passed!
All tests passed!

Test class BookList...
- findBook()...  1. Find book via book name passed!  2. Find book via book ID passed!  3. Find all books written by the author passed!
All tests passed!

Process finished with exit code 0
```

4. To run extra credit Q2 and Q3 for file “Lab3_p2_9817961224”, the following command should be entered:
   g++ -o ec Lab3_p2_9817961224.cpp
   ./ec input.txt

5. Time complexity for extra credit Q2:
   To check independece of current and given NumberSet, it will create a Hash table filling with current NumberSet elements firstly, then search every element from given NumberSet in this Hash table. If one of the elements has appeared in hash table, return false; otherwise, return true after completing all elements search. The time complexity for creating a hash table is O(n), searching an element is O(1). Therefore, the total time complexity is O(n) + n*O(1) = O(n) + O(n) = O(n).