CS 201, Spring 2019
Homework Assignment 3

Due: 23:55, May 6, 2019

In this homework, you will implement a library system by using linked lists. The library system stores information about students and books. The students will be able to check out books from the library and will be able to return them to the library. This system **MUST** have three linked lists. For each student, the system stores an id, name, and a list of books that she/he checked out. Each book is represented by its id, title, and year. The system **MUST** use a circular doubly linked list with no dummy head node to store the students, and for each student, a circular singly linked list with no dummy head node to store the books checked out by this student. There **MUST** be an additional circular singly linked list with no dummy head node that stores the books that are not checked out by any student. All lists can be unsorted. You can assume that there is a single copy of each book.

The library system will have the following functionalities; the details of these functionalities are given below:

1. Add a book

2. Delete a book

3. Add a student

4. Delete a student

5. Check out a book by a student

6. Return a book

7. Show the list of all books

8. Show detailed information about a particular book

9. Show detailed information about a particular student

PART A (30 points):

To take the final exam, you MUST submit at least this part and MUST get at least half of its points.

This part includes the following three functionalities:

1. Add a book

2. Delete a book

3. Show the list of all books

When this part is graded, you are expected to display all of the books in the system as being not checked out. Therefore, for this part, you may assume that all of the books in the system are not checked out and there is no student information in the system. Thus, in this part, there **MUST** be one circular singly linked list with no dummy head node that stores the books that are not checked out by any student. This list can be unsorted. You can assume that there is a single copy of each book.

PART B (70 points):

In this part, you will complete the implementation of the entire library system (all of the 9 functionalities listed above).

The details of all of the functionalities are given below:

Add a book: The library system will allow to add a new book to its collection with the following information: book id, book title, and the year it was published. The book ids must be unique positive integers. Hence, the system should check whether or not the specified book id already exists (i.e., whether or not it is the id of another book), and if the book exists, it should not allow the operation and display a warning message.

Delete a book: The library system will allow to delete an existing book specified with its book id. If the book does not exist (i.e., if there is no book with the specified id), the system should display a warning message. Note that it should be possible to delete books which are checked out. If the book to be deleted is already checked out by a student, it will be deleted from the checkout linked list of the student. Otherwise, it will be deleted from the linked list of books that are not checked out by any student.

Add a student: The library system will allow to add a new student with the following information: student id and name. The student ids must be unique positive integers. Therefore, the system should check whether or not the specified student id already exists (i.e., whether or not it is the id of another student), and if the student exists, it should not allow the operation and display a warning message.

Delete a student: The library system will allow to delete an existing student with student id. If the student does not exist (i.e., if there is no student with the specified id), the system should display a warning message. If the student has checked out books, those books must be returned to the system as part of the delete operation for the student.

Checkout a book by a student: The library system will allow to check out a particular book by a particular student. For that, the book id and the student id have to be specified. The system should first check whether or not this book exists; if it does not, it should not allow the operation and display a warning message. The system should also check whether or not this student exists; if he/she does not, it should not allow the operation and display a warning message. If both the book and the student exist and the book is not already checked out, the check out operation must be performed. Note that a book can be checked out by only one student. However, a student can check out multiple books.

Return a book: The library system will allow a student to return a book. For that, the book id has to be specified. If the book does not exist, the system will display a warning message. It will also check whether or not this book is checked out; if not, a warning message will be displayed, otherwise, the operation will be carried out.

Show the list of all books: The library system will allow to display the list of all the books. This list includes, for each book, the book id, book title, and publication year. For each book that is checked out, id and name of the student who checked out the book should also be displayed.

Show detailed information about a particular book: The library system will allow to specify a book id and display detailed information about that particular book. This information includes the book id, book title, publication year and whether or not the book was checked out. If the book is checked out, id and name of the student who checked out the book should be displayed. If the book does not exist (i.e., if there is no book with the specified book id), the system should display a warning message.

Show detailed information about a particular student: The library system will allow to specify a student id and display detailed information about that particular student. This information includes the student id, student name, and the list of books checked out by this student including the book id, book title, and publication year. If the student does not exist (i.e., if there is no student with the specified student id), the system should display a warning message.

Below is the required `public` part of the `LibrarySystem` class that you must write in this assignment. The name of the class <u>must</u> be `LibrarySystem`, and <u>must</u> include these public member functions. We

will use these functions to test your code. The interface for the class must be written in a file called LibrarySystem.h and its implementation must be written in a file called LibrarySystem.cpp. You can define additional public and private member functions and data members in this class. You can also define additional classes in your solution.

```cpp
class LibrarySystem {

public:
    LibrarySystem();
    ~LibrarySystem();

    void addBook( const int bookId, const string name, const int year );
    void deleteBook( const int bookId );

    void addStudent( const int studentId, const string name );
    void deleteStudent( const int studentId );

    void checkoutBook( const int bookId, const int studentId );
    void returnBook( const int bookId );

    void showAllBooks();
    void showBook( const int bookId );
    void showStudent( const int studentId );
};
```

Here is an example test program that uses this class and the corresponding output. We will use a similar program to test your solution so make sure that the name of the class is LibrarySystem, its interface is in the file called LibrarySystem.h, and the required functions are defined as shown above. You can assume that the book ids and student ids are entered as positive integers.

Example test code:

```cpp
#include "LibrarySystem.h"

int main() {

    LibrarySystem LS;

    LS.addBook( 1000, "Machine Learning", 2017 );
    LS.addBook( 1200, "Data Mining", 2015 );
    LS.addBook( 1300, "Problem S. with C++", 2015 );
    LS.addBook( 1400, "C++ How to Program", 2016 );
    LS.addBook( 1200, "Data Mining", 2015 );
    LS.deleteBook( 1300 );
    LS.deleteBook( 2000 );
    LS.addBook( 1500, "Pattern Recognition", 2000 );
    cout << endl;
    LS.addStudent( 21900000, "Furkan Huseyin" );
    LS.addStudent( 21900011, "Aynur Dayanik" );
    LS.addStudent( 21900011, "Cigdem Gunduz Demir" );
    LS.addStudent( 21900020, "Gokhan Akcay" );
    LS.addStudent( 21900001, "Cihan Akcay" );
    LS.addStudent( 21900005, "Gozde Nur Gunesli" );
    LS.deleteStudent( 21900011 );
    LS.deleteStudent( 21900050 );
    cout << endl;
```

3

```
        LS.checkoutBook( 1200, 21900000 );
        LS.checkoutBook( 1400, 21900020 );
        LS.checkoutBook( 2050, 21900011 );
        LS.checkoutBook( 1000, 21900444 );
        LS.checkoutBook( 1500, 21900000 );
        LS.checkoutBook( 1400, 21900001 );
        cout << endl;
        LS.showStudent( 21900000 );
        cout << endl;
        LS.showStudent( 21900005 );
        cout << endl;
        LS.showStudent( 21900011 );
        cout << endl;
        LS.showBook( 1000 );
        LS.showBook( 1200 );
        cout << endl;
        LS.showAllBooks();
        cout << endl;
        LS.returnBook( 1200 );
        LS.returnBook( 1000 );
        cout << endl;
        LS.checkoutBook( 1200, 21900020 );
        LS.checkoutBook( 1000, 21900000 );
        cout << endl;
        LS.showAllBooks();
        cout << endl;
        LS.deleteStudent( 21900020 );
        cout << endl;
        LS.deleteBook( 1000 );
        cout << endl;
        LS.showStudent( 21900000 );
        cout << endl;
        LS.showAllBooks();

        return 0;
}
```

Output of the example test code:

```
Book 1000 has been added
Book 1200 has been added
Book 1300 has been added
Book 1400 has been added
Book 1200 already exists
Book 1300 has not been checked out
Book 1300 has been deleted
Book 2000 does not exist
Book 1500 has been added

Student 21900000 has been added
Student 21900011 has been added
Student 21900011 already exists
Student 21900020 has been added
Student 21900001 has been added
Student 21900005 has been added
```

```
Student 21900011 has been deleted
Student 21900050 does not exist

Book 1200 has been checked out by student 21900000
Book 1400 has been checked out by student 21900020
Book 2050 does not exist for checkout
Student 21900444 does not exist for checkout
Book 1500 has been checked out by student 21900000
Book 1400 has been already checked out by another student

Student id: 21900000 student name: Furkan Huseyin
Student 21900000 has checked out the following books:
Book id    Book name              Year
1200       Data Mining            2015
1500       Pattern Recognition    2000

Student id: 21900005 student name: Gozde Nur Gunesli
Student 21900005 has no books

Student 21900011 does not exist

1000       Machine Learning       2017      Not checked out
1200       Data Mining            2015      Checked out by student 21900000

Book id    Book name              Year      Status
1000       Machine Learning       2017      Not checked out
1200       Data Mining            2015      Checked out by student 21900000
1400       C++ How to Program     2016      Checked out by student 21900020
1500       Pattern Recognition    2000      Checked out by student 21900000

Book 1200 has been returned
Book 1000 has not been checked out

Book 1200 has been checked out by student 21900020
Book 1000 has been checked out by student 21900000

Book id    Book name              Year      Status
1000       Machine Learning       2017      Checked out by student 21900000
1200       Data Mining            2015      Checked out by student 21900020
1400       C++ How to Program     2016      Checked out by student 21900020
1500       Pattern Recognition    2000      Checked out by student 21900000

Book 1400 has been returned
Book 1200 has been returned
Student 21900020 has been deleted

Book 1000 has been returned
Book 1000 has been deleted

Student id: 21900000 student name: Furkan Huseyin
Student 21900000 has checked out the following books:
Book id    Book name              Year
1500       Pattern Recognition    2000

Book id    Book name              Year      Status
1200       Data Mining            2015      Not checked out
```

```
1400      C++ How to Program    2016     Not checked out
1500      Pattern Recognition   2000     Checked out by student 21900000
```

Notes:

1. This assignment is due by 23:55 on Monday, May 6, 2019. You should upload your homework to the upload link on Moodle before the deadline. This upload link will be available between April 19 and May 9. No hardcopy submission is needed. The standard rules about late homework submissions apply. Please see the course syllabus for further discussion of the late homework policy as well as academic integrity.

2. You ARE NOT ALLOWED to modify the given parts of the `LibrarySystem` class. However, if necessary, you may define additional data members and member functions. Moreover, you ARE NOT ALLOWED to use any global variables.

3. For this assignment, you must use your own implementation of linked lists. In other words, you cannot use any existing linked list code from other sources such as the `list` class in the C++ standard template library (STL). However, you can adapt the linked list codes in the Carrano book. You will get no points if you do not use linked lists as indicated.

4. You will get no points if you make array-based implementations (fixed-sized arrays, dynamically allocated arrays, data structures such as vector from the standard library.

5. Your code must not have any memory leaks. You will lose points if you have memory leaks in your program even though the outputs of the operations are correct.

6. In this assignment, you must have separate interface and implementation files (i.e., separate `.h` and `.cpp` files) for your class. We will test your implementation by writing our own driver `.cpp` file which will include your header file. For this reason, your class' name MUST BE "`LibrarySystem`" and your files' name MUST BE "`LibrarySystem.h`" and "`LibrarySystem.cpp`". You should upload these two files (and any additional files if you wrote additional classes in your solution) as a single archive file (zip or rar). The submissions that do not obey these rules will not be graded. You must also write your own driver file, which must be different from the one that we gave above, to test each of your functions. You MUST submit this test code as well in the archive file.

7. You are free to write your programs in any environment (you may use either Linux or Windows). On the other hand, we will test your programs on "`dijkstra.ug.bcc.bilkent.edu.tr`" and we will expect your programs to compile and run on the "`dijkstra`" machine. If we could not get your program properly work on the "`dijkstra`" machine, you would lose a considerable amount of points. Therefore, we recommend you to make sure that your program compiles and properly works on "`dijkstra.ug.bcc.bilkent.edu.tr`" before submitting your assignment.

8. This homework will be graded by your TA Gozde Nur Gunesli (nur.gunesli at bilkent edu tr). Thus, you may ask your homework related questions directly to her.