C++ Programming	Student number	21600193
Homework 4	Name	Kim, Hyo Rim

1. Comment

- 1) struct book{}: Make the information of the book to be saved into struct format. 'title', 'p_Year', 'author', 'edition', 'b_Person' are declared as string type. Declare 'I_Day' and 'index' as int. At this time, 'b_Person' and 'I_Day' specify default values. int index represents the number of stored books.
- 2) int main():Declare b_List, the pointer type of struct book. Use the 'load_m' function to store the contents of the file in b_List. Use the while statement to continue the menu until the 'exit' command is called. Each command uses a switch to call the required function. In this case, the command uses the touuper function so that it is not case sensitive.
- 3) void printMenu(): Show the menu.
- 4) string * p_Arg(string line): Divide the input command so that it can be used in the program and save it.
- 5) book *load_m(string FileName): It receives the file name and stores the file information in a struct book. At this point, struct book uses static variables to make the saved book available to other functions as well.
- 6) void save_m(string fileName, book * bookList): Save the struct book as a file name.
- 7) void print_m(book * bookList): Receives a struct book and prints out the Book Catalog and a statement specifying the return time limit for the book.
- 8) book * insert_m(book * bookList, string * arg): The new book information inputted from the user through the command is stored in the existing information and outputted.
- 9) book *lend_m(book * bookList, string * arg): It stores information of a book borrowed by another person.
- 10) book *passDay_m(book * bookList): Subtract 1 from 'I_Day' of the book where 'b_Person' is not "None".
- 11) book * returned_m(book * bookList, string * arg): When the returned book is entered, the 'I_Day' and 'b_Person' of the book are returned to their default values.

2. Code

```
#include <iostream>
#include <string>
#include<iomanip>
#include <fstream>
using namespace std;
//DONE: ONE - make
struct book{
        string title;
        string p_Year;
        string author;
        string edition;
        string b_Person = "None";
        int I_Day = 0;
        int index;
};
void printMenu();
string * p_Arg(string line);
book *load_m(string FileName);
void save_m(string fileName, book * bookList);
void print_m(book * bookList);
book * insert_m(book * bookList, string * arg);
book *lend_m(book * bookList, string * arg);
book *passDay_m(book * bookList);
book * returned_m(book * bookList, string * arg);
int getExit();
int main(){
```

```
int setExit = 1;
book * b_List;
string openFile = "input.txt";
b_List = load_m(openFile);
while(setExit){
        string argLine, * arg;
        printMenu();
        getline(cin, argLine);
        arg = p_Arg(argLine);
        char n1 = arg[0][0];
        char n2 = arg[0][1];
        switch(toupper(n1)){
        case 'l':
                                b_List = insert_m(b_List, arg);
                                break;
        case 'L':
                                b_List = lend_m(b_List, arg);
                                break;
        case 'S':
                                save_m(arg[1], b_List);
                                break;
        case 'R':
                                b_List = returned_m(b_List, arg);
```

```
break;
             case 'P':
                           switch (toupper(n2)) {
                                  case 'A':
                                                b_List = passDay_m(b_List);
                                                break;
                                  case 'R':
                                                print_m(b_List);
                                                break;
                           }
                           break;
             case 'E':
                                  setExit = getExit();
                                  break;
             }
      }
      return 0;
void printMenu(){
      cout << " 1. INSERT BookTitle; Author; PubYear; Edition" << endl;</pre>
      cout << " 2. LEND BookTitle; Person Borrowing; How many days" << endl;
      cout << " 3. SAVE new_filename.txt" << endl;</pre>
      cout << " 4. RETURNED BookTitle" << endl;</pre>
      cout << " 5. PASSDAY" << endl;</pre>
      cout << " 6. PRINT" << endl;</pre>
      cout << " 7. EXIT" << endl << endl;</pre>
      cout << " ========= " << end|;
      cout << " >>";
```

```
string * p_Arg(string line){
        int f_{col}, i = 0;
        static string arg[6];
        while(1){
                if (i == 0) {
                        f_col = line.find(" ");
                        arg[i] = line.substr(0, f_col);
                        line.erase(0, f_col + 1);
                        j++;
                }
                else {
                        if (line.length() > 0) {
                                f_col = line.find(";");
                                 if (f_{col} == -1) {
                                         arg[i] = line;
                                         break;
                                 }
                                arg[i] = line.substr(0, f_col);
                                line.erase(0, f_{col} + 2);
                                 j++;
                        }
                        else
                                break;
                }
        }
        return arg;
```

```
//DONE: THREE - function Load
book *load_m(string FileName){
        //open file and store list
        static book bookList[50];
        int i = 0;
        string line;
        ifstream inData;
        inData.open(FileName.c_str());
        if(!inData){
                cout << FileName << " does not exist." << endl;</pre>
        }
        else{
                while(!inData.eof()){
                        getline(inData, line);
                         int f_{col}, j = 0;
                        string arr[6];
                        while(1){
                                 if (line.length() > 0) {
```

```
f_col = line.find(";");
                                 if (f_{col} == -1) {
                                         arr[j] = line;
                                         break;
                                 }
                                 arr[j] = line.substr(0, f_col);
                                line.erase(0, f_col+2);
                                 j++;
                }
                else
                        break;
                }
                bookList[i].title = arr[0];
                bookList[i].p_Year = arr[1];
                bookList[i].author = arr[2];
                bookList[i].edition = arr[3];
                bookList[i].b_Person = arr[4];
                bookList[i].I_Day = stoi(arr[5]);
                j++;
        }
        j---;
        bookList[0].index = i;
        print_m(bookList);
}
inData.close();
return bookList;
```

```
//DONE: THREE - function Save
void save_m(string fileName, book * bookList){
              // use ofstream and save text file
              ofstream outData;
              int i = 0;
              outData.open(fileName.c_str());
              while (i < bookList[0].index - 1) {</pre>
                             outData << bookList[i].title << "; " << bookList[i].p_Year << "; " <<
bookList[i].author << "; "<< bookList[i].edition << "; "<< bookList[i].b_Person << "; "<< bookList[i].b_Person << "; "<< bookList[i].b_Person << "; "<< bookList[i].edition << "; "< bookList[i].edition </ >
bookList[i].I_Day << endl;</pre>
                             j++;
              }
              outData << bookList[i].title << "; "<< bookList[i].p_Year << "; "<< bookList[i].author
<< "; "<< bookList[i].edition << "; " << bookList[i].b_Person << "; "<< bookList[i].l_Day;</pre>
              outData.close();
}
//DONE: THREE - function Print
void print_m(book * bookList){
              string pass[50];
              string today[50];
               int pass_j = 0, today_j = 0;
```

```
<< end1
       cout
                                                                            Book
                                                                                 Catalog
                              ======" << endl;
       cout << setw(8) << "Title" << setw(24) << "Author" << setw(25) << "Publised Year" <<
setw(10) << "Edition" << setw(10) << "Borrower" << setw(16) << "Days Borrowed" << endl;</pre>
       for(int i = 0; i < bookList[0].index; i++){
               if(bookList[i].b_Person != "None"){
                       if(bookList[i].I_Day == 0){
                              today[today_j] = bookList[i].title;
                              today[today_j+1] = bookList[i].b_Person;
                              today_j += 2;
                       }
                      else if(bookList[i].l_Day <0){</pre>
                              pass[pass_j] = bookList[i].title;
                              pass[pass_j+1] = bookList[i].b_Person;
                              pass_j += 2;
                       }
               }
               bookList[i].author.length()) << bookList[i].author << setw(22 - bookList[i].author.length()) <<
bookList[i].p_Year << setw(22 - bookList[i].p_Year.length()) << bookList[i].edition << setw(9</pre>
- bookList[i].edition.length() + bookList[i].b_Person.length()) << bookList[i].b_Person <</pre>
setw(12 - bookList[i].b_Person.length()) << bookList[i].l_Day << endl;</pre>
       }
       cout
                 <<
                                                                                       End
                                 ======" << endl;
       for(int i = 0; i < pass_j; i += 2)
               cout << pass[i]<< " SHOULD HAVE BEEN RETURNED ALREADY by " << pass[i+1]<< endl;
       for (int i = 0; i < today_j; i += 2)
               cout << today[i]<< " should be returned today by " << today[i+1]<< endl;</pre>
```

```
cout << endl;</pre>
}
//DONE: THREE - function Insert
book * insert_m(book * bookList, string * arg){
        bookList[bookList[0].index].title = arg[1];
        bookList[bookList[0].index].author = arg[2];
        bookList[bookList[0].index].p_Year = arg[3];
        bookList[bookList[0].index].edition = arg[4];
        bookList[0].index++;
        cout << end| << " | << "Inserted | << arg[1] << " successfully!" << end| << end|;
        print_m(bookList);
        return bookList;
//DONE: THREE - function Lend
book *lend_m(book * bookList, string * arg){
        int j = 0;
        for(int i = 0; i < bookList[0].index; <math>i++){
                if(bookList[i].title == arg[1])
                        j++;
        }
        if(j == 0){
```

```
cout << "NO SUCH BOOK!" << endl;</pre>
                return bookList;
        }
        cout << endl <<"
                                                        ====== Books Cuurrently Lent
        cout << setw(8) << "Title" << setw(24) << "Author" << setw(25) << "Publised Year" <<
setw(10) << "Edition" << setw(10) << "Borrower" << setw(16) << "Days Borrowed" << endl;</pre>
        for(int i = 0; i < bookList[0].index; <math>i++){
                if(bookList[i].title == arg[1]){
                        bookList[i].b_Person = arg[2];
                        bookList[i].I_Day = stoi(arg[3]);
                }
                if(bookList[i].b_Person != "None")
                             << "
                        cout
                                                   << bookList[i].title << setw(23)</pre>
bookList[i].title.length() + bookList[i].author.length()) << bookList[i].author << setw(22 -
bookList[i].author.length()) << bookList[i].p_Year << setw(22 - bookList[i].p_Year.length()) <<
bookList[i].edition << setw(9 - bookList[i].edition.length() + bookList[i].b_Person.length())</pre>
<< bookList[i].b_Person << setw(12 - bookList[i].b_Person.length()) << bookList[i].l_Day <</pre>
end1;
        }
                cout
                                                                                           End
                                      ======" << end| << end|;
        return bookList;
//DONE: THREE - function Passday
```

```
book *passDay_m(book * bookList){
       cout << endl <<"
                                            ====== Books Cuurrently Lent
                              =====" <<endl;
       cout << setw(8) << "Title" << setw(24) << "Author" << setw(25) << "Publised Year" <<
setw(10) << "Edition" << setw(10) << "Borrower" << setw(16) << "Days Borrowed" << endl;</pre>
       string pass[50];
       string today[50];
       int pass_j = 0, today_j = 0;
       for(int i = 0; i < bookList[0].index; <math>i++){
                if(bookList[i].b_Person != "None"){
                       bookList[i].I_Day--;
                       if(bookList[i].I_Day == 0){
                               today[today_j] = bookList[i].title;
                               today[today_j+1] = bookList[i].b_Person;
                               today_j += 2;
                       }
                       else if(bookList[i].I_Day <0){</pre>
                               pass[pass_j] = bookList[i].title;
                               pass[pass_j+1] = bookList[i].b_Person;
                               pass_j += 2;
                       }
                              " << bookList[i].title << setw(23)</pre>
                       cout
bookList[i].title.length() + bookList[i].author.length()) << bookList[i].author << setw(22 -
bookList[i].author.length()) << bookList[i].p_Year << setw(22 - bookList[i].p_Year.length()) <<
bookList[i].edition << setw(9 - bookList[i].edition.length() + bookList[i].b_Person.length())</pre>
<< bookList[i].b_Person << setw(12 - bookList[i].b_Person.length()) << bookList[i].l_Day <</pre>
endl;
```

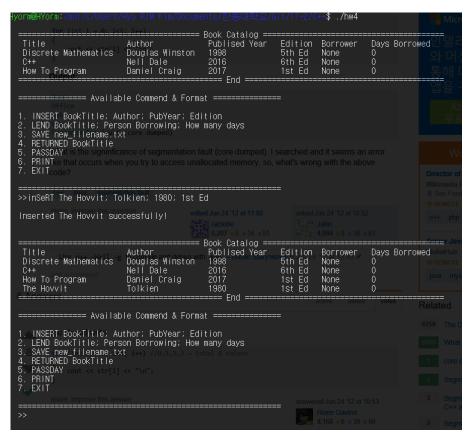
```
}
       }
       cout
                <<
                                                                                    End
                                ======= << endl;
       for(int i = 0; i < pass_j; i += 2)
              cout << pass[i]<< " SHOULD HAVE BEEN RETURNED ALREADY by " << pass[i+1]<< endl;</pre>
       for(int i = 0; i < today_j; i += 2)
              cout << today[i]<< " should be returned today by " << today[i+1]<< endl;</pre>
       cout << endl;</pre>
       return bookList;
}
//DONE: THREE - function Returned
book * returned_m(book * bookList, string * arg){
       int j = 0;
       for(int i = 0; i < bookList[0].index; <math>i++){
              if(bookList[i].title == arg[1]){
                      if(bookList[i].b_Person == "None"){
                             cout << end!<< "ATTENTION No one borrowed that book!!" << endl;</pre>
                             return bookList;
                      }
                      else{
                      j++;
                      Retured =====
                      cout \ll setw(8) \ll "Title" \ll setw(24) \ll "Author" \ll setw(25) \ll
```

```
"Publised Year" << setw(10) << "Edition" << setw(10) << "Borrower" << setw(16) << "Days Borrowed"
<< endl;
                       bookList[i].b_Person = "None";
                       bookList[i].I_Day = 0;
                        cout << "
                                                   << bookList[i].title << setw(23)</pre>
bookList[i].title.length() + bookList[i].author.length()) << bookList[i].author << setw(22 -
bookList[i].author.length()) << bookList[i].p_Year << setw(22 - bookList[i].p_Year.length()) <<
bookList[i].edition << setw(9 - bookList[i].edition.length() + bookList[i].b_Person.length())</pre>
<< bookList[i].b_Person << setw(12 - bookList[i].b_Person.length()) << bookList[i].l_Day <</pre>
end1;
                               << "
                                                                                           End
                        cout
                                    =======" << end| << end|;
                        }
                }
        }
        if(j == 0)
                cout << end!<< "ATTENTION No Such Book!"<< endl;</pre>
        return bookList;
}
//DONE: THREE - fundtion Exit
int getExit(){
        return 0;
```

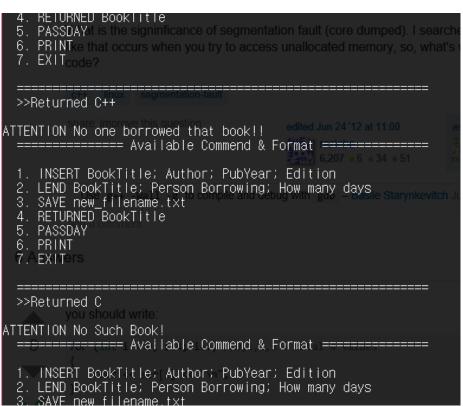
3. Result

1) input.txt

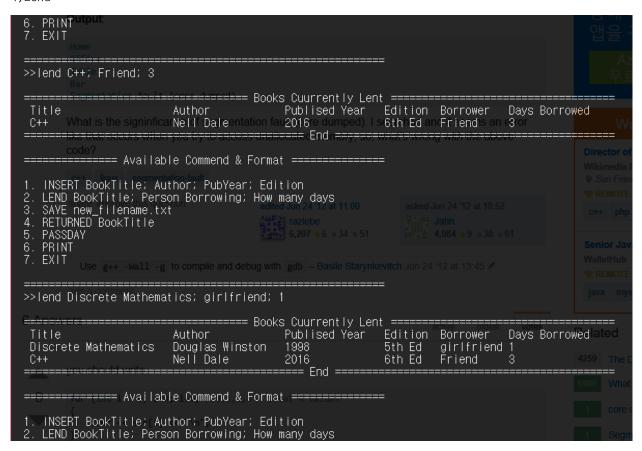
2)Start & Insert



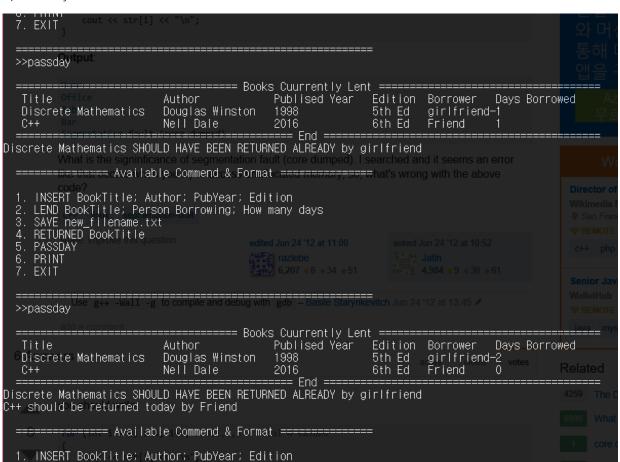
3)Returned



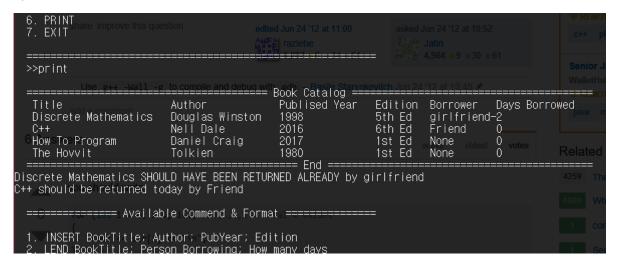
4)Lend



5) Passday



6)Print



7)Returned correctly



8) Save

