

DATE:

GRADE:



COMPUTER PROGRAMMING 2 [PROJECT 2]

TECHNICAL DOCUMENTATION

TITLE: Librarian

TEACHER: DANIEL KRASNOKUCKI

STUDENT: TOMASZ WIECZOREK

CONTACT: tomawie972@student.polsl.pl tel. 692 082 158

1. TASK DESCRIPTION

A database for a library. Write a program that supports the work of a library with a few hundred of readers. The program shall hold data for all books currently outside the library: who has them, for how long etc. It shall also find readers who keep their books too long and answer the question who has a specified book.

2. PROJECT ANALYSIS

At first I was thinking about what information I will need to implement a program. I have decided to create three classes:

- Readers, consisting of following variables (all of them are public):

- `int ID;`
- `char first_name[15];`
- `char surname[20];`
- `char street[20];`
- `char house_number[7];`
- `int postcode;`
- `char city[20];`
- `Date birthdate;`
- `int sex; //1 male, 0 female`
- `char ID_number[10];`
- `struct Readers *next;`

- Books, consisting of:

- `int ID;`
- `char title[50];`
- `char author[20];`
- `int year;`
- `char publisher[20];`
- `char genre[10];`

- `int borrowed;`
- `struct Books *next;`
- Borrowings, consisting of:
 - `int ID;`
 - `int reader_ID;`
 - `int book_ID;`
 - `Date from_when;`
 - `Date until_when;`
 - `Date return_date;`
 - `struct Borrowings *next;`
- and auxiliary structure `Date`:
 - `int year;`
 - `int month;`
 - `int day;`

I have chosen following structures to get optimal information about readers, books and borrowings and work on them.

Most of functions in my program works on singly linked lists, that allows to go through all elements of the list.

Librarian stores data in subfolder `data`, at files `readers.txt`, `books.txt` and `borrowings.txt`

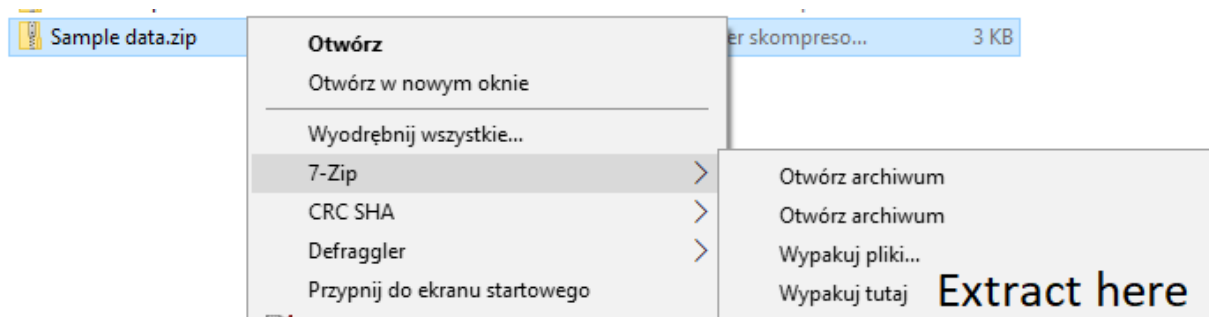
3. EXTERNAL SPECIFICATION

Application *Librarian* allows to run database for library, that contains information about readers, books and borrowings.

First run

If you want at first to learn how Librarian works, you can run it with exemplary data. To run application with test data, you have to extract archive "*Sample data.zip*" (e.g. by 7-zip) that is in the same folder as *Librarian.exe* to the same folder (*Extract here*). It will create folder data, that contains exemplary program data.

NOTE! Application can't be opened during the extraction, because it won't work.



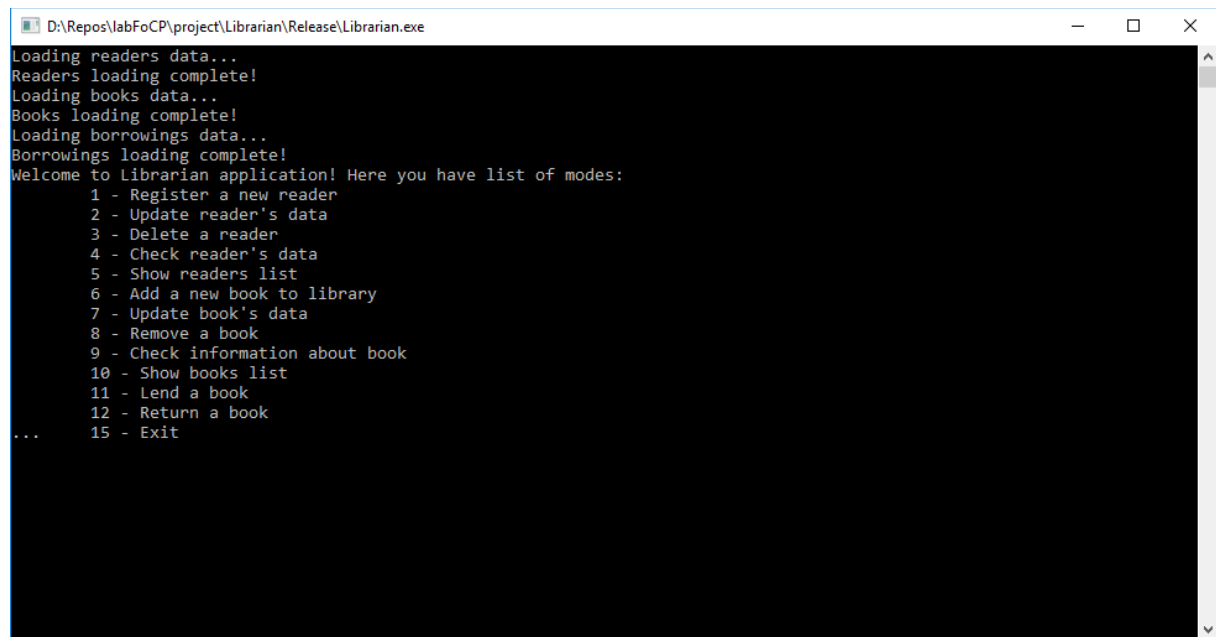
Removing whole data

To remove whole program data (readers, books and borrowings list) just remove folder *data*.

Run an application

To run program you should double-click executable file "*Librarian.exe*". After that, program will load files with data of readers, books and borrowings. If that files don't exist, program will create them.

After loading program will show list of modes you can use:



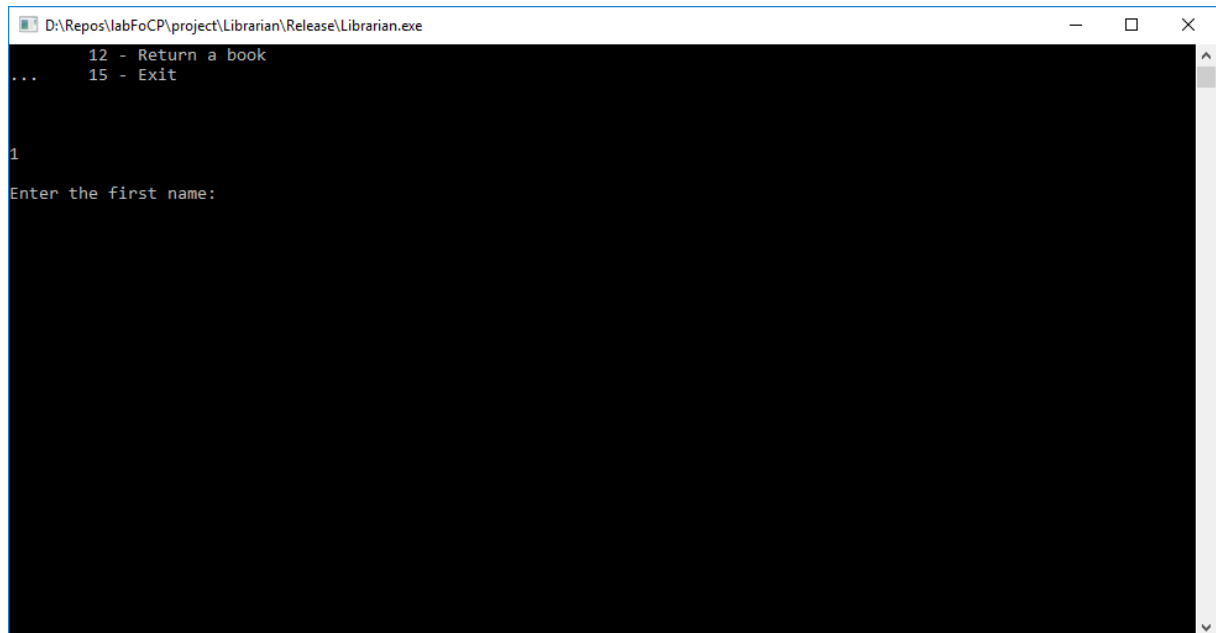
```
D:\Repos\labFoCP\project\Librarian\Release\Librarian.exe
Loading readers data...
Readers loading complete!
Loading books data...
Books loading complete!
Loading borrowings data...
Borrowings loading complete!
Welcome to Librarian application! Here you have list of modes:
  1 - Register a new reader
  2 - Update reader's data
  3 - Delete a reader
  4 - Check reader's data
  5 - Show readers list
  6 - Add a new book to library
  7 - Update book's data
  8 - Remove a book
  9 - Check information about book
 10 - Show books list
 11 - Lend a book
 12 - Return a book
... 15 - Exit
```

Now, you can choose which one you want to use, writing a number of mode (e.g. 1) and pressing *Enter*.

When mode is chosen program can ask you about some information to proceed next steps. When all needed information are collected, program will present results. After that application will bring you back to list of modes.

Registering a new reader

To register new reader write *1* at list of modes and press Enter. *Librarian* will ask you about necessary data to register new reader:



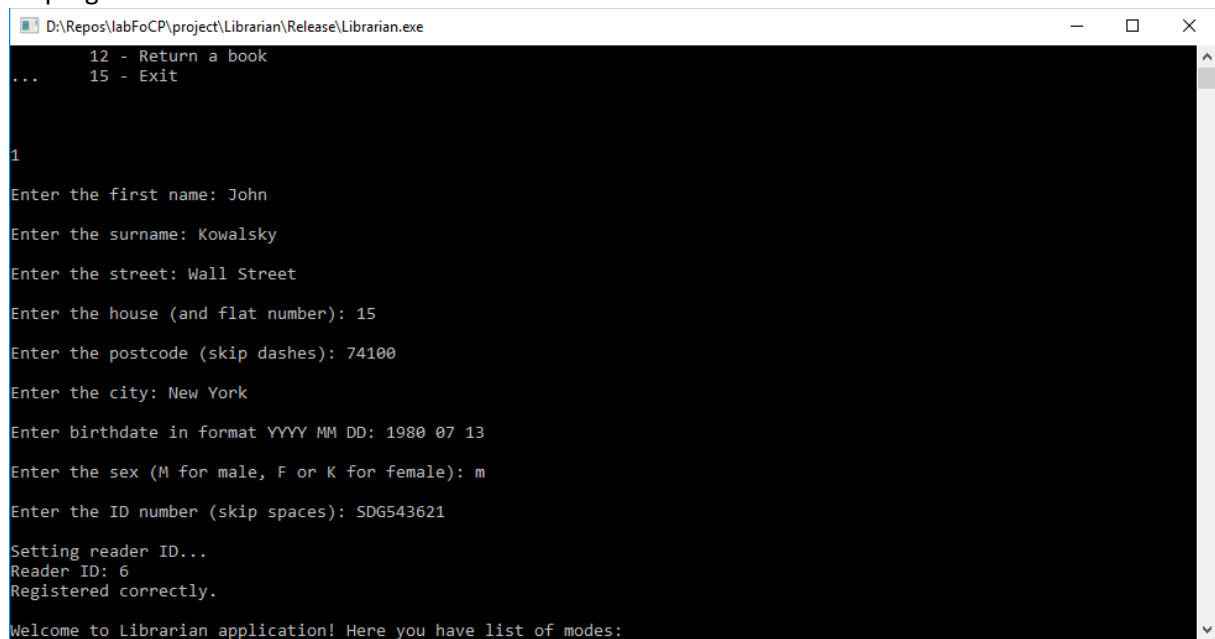
```
D:\Repos\labFoCP\project\Librarian\Release\Librarian.exe
12 - Return a book
... 15 - Exit

1
Enter the first name:
```

You will need to enter first name, surname, street, house number, postcode, city, birthdate, sex and ID number.

It is important to text postcode only as numbers – application doesn't allow to using e.g. dashes. Birthdate has to be set in the form of YYYY MM DD, where YYYY is year, MM is month, DD is day of birth, where all of them are numbers (e.g. 1980 07 13)

When all necessary data are collected, *Librarian* will inform you what ID new reader got. After that program will back to the list of modes.



```
D:\Repos\labFoCP\project\Librarian\Release\Librarian.exe

12 - Return a book
... 15 - Exit

1

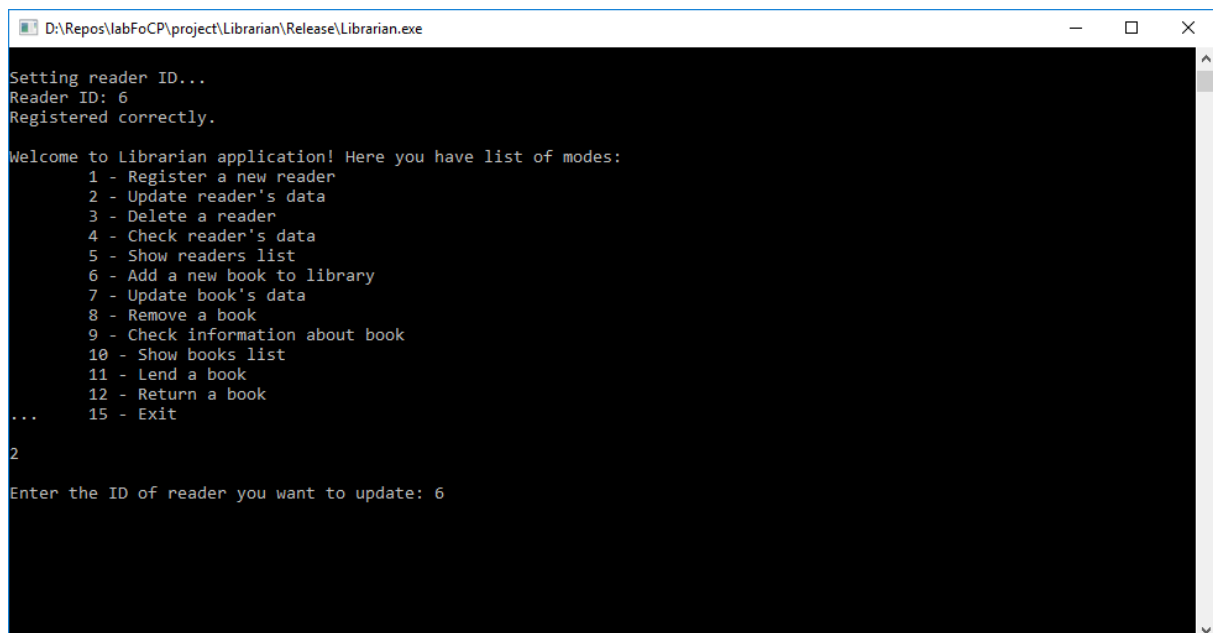
Enter the first name: John
Enter the surname: Kowalsky
Enter the street: Wall Street
Enter the house (and flat number): 15
Enter the postcode (skip dashes): 74100
Enter the city: New York
Enter birthdate in format YYYY MM DD: 1980 07 13
Enter the sex (M for male, F or K for female): m
Enter the ID number (skip spaces): SDG543621

Setting reader ID...
Reader ID: 6
Registered correctly.

Welcome to Librarian application! Here you have list of modes:
```

Updating reader's data

When you need to update some of reader's data, write 2 at modes list and press Enter. Librarian will ask you about his ID:



```
D:\Repos\labFoCP\project\Librarian\Release\Librarian.exe

Setting reader ID...
Reader ID: 6
Registered correctly.

Welcome to Librarian application! Here you have list of modes:
1 - Register a new reader
2 - Update reader's data
3 - Delete a reader
4 - Check reader's data
5 - Show readers list
6 - Add a new book to library
7 - Update book's data
8 - Remove a book
9 - Check information about book
10 - Show books list
11 - Lend a book
12 - Return a book
... 15 - Exit

2

Enter the ID of reader you want to update: 6
```

Text it and press Enter. When program get information, it will print actual data from database:

```
D:\Repos\labFoCP\project\Librarian\Release\Librarian.exe
Welcome to Librarian application! Here you have list of modes:
 1 - Register a new reader
 2 - Update reader's data
 3 - Delete a reader
 4 - Check reader's data
 5 - Show readers list
 6 - Add a new book to library
 7 - Update book's data
 8 - Remove a book
 9 - Check information about book
10 - Show books list
11 - Lend a book
12 - Return a book
... 15 - Exit

2

Enter the ID of reader you want to update: 6
Actual data of reader with ID: 6
First name: John
Last name: Kowalsky
Street: Wall Street
House number: 15
Postcode: 74100
City: New York
Birthdate: 1980 7 13
ID number: SDG543621

Confirm that you want to update that reader (Y/N): Y
```

Now you should confirm if you want to update that reader by pressing Y (for “yes”) or N (for “no”) and confirm by Enter.

If you chosen N, program will go back to modes list.

If you chosen Y, program will ask you about new reader’s data. You need to enter them like registering data:

```
D:\Repos\labFoCP\project\Librarian\Release\Librarian.exe
Enter the ID of reader you want to update: 6
Actual data of reader with ID: 6
First name: John
Last name: Kowalsky
Street: Wall Street
House number: 15
Postcode: 74100
City: New York
Birthdate: 1980 7 13
ID number: SDG543621

Confirm that you want to update that reader (Y/N): Y
Set new data:

Enter the first name: John
Enter the surname: Kowalsky
Enter the street: Wall Street
Enter the house (and flat number): 108
Enter the postcode (skip dashes): 10001
Enter the city: New York
Enter birthdate in format YYYY MM DD: 1980 7 13
Enter the sex (M for male, F or K for female): M
```

When all necessary data are entered, program will confirm edition by message:

Reader updated.

Delete a reader

If you need to remove reader, *Librarian* allows to do it.

IMPORTANT! If reader has a borrowed book, it is impossible to remove him!

Please, choose mode 3. *Librarian* will ask you about his reader's ID. Enter it and confirm by enter. Program will show you his data and ask to confirm deletion:

```
11 - Lend a book
12 - Return a book
... 15 - Exit

3
Enter the ID of reader you want to remove: 8
First name:    Marian
Last name:     Pazdzioch
Street:        Cwiartki
House number:  3/3
Postcode:      50001
City:          Wroclaw
Confirm that you want to delete that reader (Y/N):
```

To confirm press *Y* and Enter, program will remove reader. To cancel press *N* and Enter.

Checking reader's data and borrowings



If you want to check reader's data and his borrowings, choose mode 4. *Librarian* will ask you about reader's ID:

```
4
Enter the ID of reader you want to check:
```

Write his ID and press Enter. Application will show personal data and list of all his borrowings:

```
Enter the ID of reader you want to check: 1
First name:    Marian
Last name:     Starosolski
Street:        Dziwna
House number:  15
Postcode:      95216
City:          Dziwnowo
Birthdate:     2015 1 1

Borrowing ID   Book ID       Borrowing date    Until when       Return date
1              1             2017 1 27        2017 1 28        2017 1 31!
```

Red color of text and exclamation mark at the date of return means that book hasn't been returned on time.  Values at  date of return or means, that book hasn't been returned yet.

Showing readers list

If you want to see readers list, you shall choose mode no. 5. It will print list of readers as list in the following form:

- ID
- first name
- surname

```
1
Marian
Starosolski
2
Marianna
Starosolska
3
Genowefa
Mruczek
```

Adding a book

If you bought new books for the library, you probably want to add them to database. To add a new book to *Librarian's* database, you have to choose mode no. 6.

This mode work analogously to mode 1 (registering a reader). Program will ask you about book's data – title, author, year of publishing, publisher and genre. When data are collected, application will set book's ID.

```
Enter a title: Syzyfowe prace
Enter an author: Stefan Zeromski
Enter the year of publishing: 1975
Enter a publisher: BLS
Enter the genre: novel
Book ID: 4
Book added correctly.
```

Update book's data

To update book's data, you should to choose mode number 7. Program will ask you about book's ID, and then will show you actual data:

```
7
Enter the ID of book you want to update: 3
Actual data of book with ID: 3
Title:      Igrzyska Smierci. Kosoglos
Author:     Suzanne Collins
Year:       2014
Publisher:  Media Rodzina
Genre:      sci-fi
```

Then *Librarian* will ask you to confirm that you want update that book:

```
Confirm that you want to update that book (Y/N): _
```

Press Y and Enter to confirm, N and Enter to deny. When confirmed, program will ask about updated book's data. When all data are collected, you will get information that book has been updated.

```
Book updated.
```

Removing a book

If for some reasons you have to remove book from library (e.g. it is destroyed), you can remove it from library.

IMPORTANT! You can't remove borrowed book!

To remove book please choose mode 8. *Librarian* will ask you about book's ID:

```
Enter the ID of book you want to remove: 3
Title:      Igrzyska Smierci. Kosoglos
Author:     Suzanne Collins
Year:       2014
Publisher:   Media Rodzina
Genre:      sci-fi
Confirm that you want to delete that book (Y/N):
```

You ought to confirm if it is the book you want to delete.

Checking information about book and its borrowings

If you want to check information about book, choose mode no. 9. *Librarian* will ask you about ID of book:

```
9
Enter the ID of book you want to check:
```

Then, program will show information about book – title, author, year of publishing, publisher, genre, its state and list of its borrowings:

```
Title:      Igrzyska Smierci. Kosoglos
Author:     Suzanne Collins
Year:       2014
Publisher:   Media Rodzina
Genre:      sci-fi
Borrowed:   YES, by reader with ID: 5

Borrowing ID  Reader ID  Borrowing date  Until when  Date of return
2             5             2017 1 31      2017 3 2    0 0 0
```

Values 0 0 0 at date of return means that book hasn't been returned yet. Exclamation mark at date of return means that time to return of book has been exceeded.

Showing books list

To show books list you shall choose mode 10. It will print a list of books in form:

- ID
- title
- author
- year of publishing
- publisher
- genre
- state if borrowed and by who

```
1
Alicja w Krainie Czarow
Lewis Carroll
1865
Tenniel
absurd
Not borrowed
2
FniN TMK
Rafal Kosik
2005
Powergraph
sci-fi
Not borrowed
3
Igrzyska Smierci. Kosoglos
Suzanne Collins
2014
Media Rodzina
sci-fi
Borrowed by reader 5
4
Syzyfowe prace
Stefan Zeromski
1975
BLS
novel
Not borrowed
```

Lending/Borrowing a book

If you want to lend/borrow a book, you must choose mode number *11*. Program will ask you about readers ID:

```
10 - Show books list
11 - Lend a book
12 - Return a book
... 15 - Exit

11

Enter reader's ID: _
```

When reader's ID is set, program will ask about book ID. When given, *Librarian* will propose deadline of return:

```
Enter reader's ID: 7

Enter book's ID: 4
Return date proposal: Thu Mar  2 22:11:01 2017
Do you agree? (Y/N):
```

Default, it is in 30 days from day of borrowing. To confirm date, press *Y* and Enter. If you would like to change returns date, press *N* and Enter. Program will ask you about deadline in format YYYY MM DD:

```
Do you agree? (Y/N): Y
Write return date in format YYYY MM DD: _
```

```
Do you agree? (Y/N): Y
Borrowing ID: 3
Borrowed correctly.
```

When correct date is given or chosen suggested by program, it will return message:

Returning book to a library

If reader wants to return book to a library, you should choose mode no. *12*. When chosen, it will ask you to reader ID and book ID. After that it will check if there exist borrowing that responds to given data and that hasn't been returned. If found, program will reply with borrowing date and question to confirm return.

```
12
Enter reader's ID: 5
Enter book's ID: 3
Borrowed on 2017 1 31
Confirm return (Y/N):
```

If confirmed – returns date of borrowing is set to day of return.

Closing application

If you want to close application, it is highly recommended to close it using mode 15 – it is the safest way to close file without losing data.

Possible errors

- **There is no reader with that ID! / There is no book with that ID!**

```
There is no book with that ID!  
Enter the ID of reader you want to remove: 50  
There is no reader with that ID!
```

In database there is no reader/book with such ID. Check if you entered properly (e.g. by mode 5 – showing readers list)

- **Book is already borrowed! / Book is already lent.**

```
Book is already borrowed! Deletion is not possible!  
Book's return is necessary to remove it.  
Book is already lent, it's not possible to lend lent book!
```

Book is already borrowed and it is impossible to be deleted/borrowed until returned. You can check when book should be at the latest available my mode 9 – checking book's data.

- **Reader hasn't returned all books!**

```
Confirm that you want to delete that reader (Y/N): Y  
Reader hasn't returned all books! Not possible to remove him.
```

Reader hasn't returned at least one book to library. Program doesn't allow to remove reader if he hasn't returned all books. You can check what books he has, using mode 4 – checking reader's data and his borrowings

- **Wrong answer!**

```
Confirm that you want to update that book (Y/N): milk  
Wrong answer!
```

Check if you entered answer for question in proper way.

- **Can't be borrowed to the day in past!**

```
Can't be borrowed to the day in past! Try again.
```

You have given day of return as a day in past. You need to give day in future.

- **Out of memory!**

Program couldn't allocate memory to data (during loading from data files) or to new data – try to free Random Access Memory (RAM), disabling another programs.

4. INTERNAL SPECIFICATION

Structures:

```
• struct Date {           //structure contains date's information

    int year;              //variable to store year

    int month;             //variable to store month

    int day;               //variable to store day

};
```

Classes:

```
• class Readers

{
public:

    int ID;
    string first_name;
    string surname;
    string street;
    string house_number;
    int postcode;
    string city;
    Date birthdate;
    int sex; //1 male, 0 female
    string ID_number;
    Readers *next; //pointer to the next reader on the list.

    Readers(); //default constructor - it's empty
    ~Readers(); //default destructor - it's empty

    int reg(); //method to register new reader
```

```

    int set_ID(); //method that sets new ID
    Readers get_info(); //method that sets readers data given by user

    int update(); //updating reader's data
    int check(); //checking reader's data
    int show_list(); //shows readers list

    int del(); //removing readers

};
• class Books //structure of books data
{
public:
    int ID;
    string title;
    string author;
    int year;
    string publisher;
    string genre;
    int borrowed;
    Books *next;

    Books();
    ~Books();

    Books get_info(); //gets book's data from user
    int set_ID(); //sets ID
    int add(); //adds book

    int update(); //updates book
    int check(); //checks book
    int show_list(); //shows list

    int del(); //removes book

};

```

- `class Borrowings //structure that contains information about borrowings`

```

{
public:
    int ID;
    int reader_ID;
    int book_ID;
    Date from_when;
    Date until_when;
    Date return_date;
    Borrowings *next;

    int lend_a_book(); //to lend a book
    int check_if_after_return_date(Borrowings *borrowing); //checks if reader
is late
    int check_if_returned_after_date(); //checks if reader was late
    int return_book(); //returns book to library

    Borrowings();
    ~Borrowings();
};

```

Global pointers:

- `extern Readers *readers_head, *p_readers, *last_reader; //reader_head - first reader on the list`

```

//p_reader - pointer to one of the reader
//last_reader - pointer to last reader on the list

```
- `extern Books *books_head, *p_books, *last_book;`

```

//Books *books_head, *p_books, *last_book;
//pointer to first element of books list, pointer that goes through the list, pointer to the last book on the list
extern Borrowings *borrowings_head, *p_borrowings, *last_borrowing;
//pointer to first element of borrowings list, pointer that goes through whole list, last borrowing on the list

```

Functions/Methods

Most used methods were written above in classes section. In general int functions/methods return 0, when function was is true(e.g. done correct or query is true), and 1, when an error occurred. An exception is function `int check_if_after_return_date(Borrowings *borrowing)` that returns 0 or 1 as true, depending on situation

Functions to work on files:

- `int loadReaders();` //function that loads readers from file
- `int saveReaders();` //function that saves readers to file
- `int loadBooks();` //function that loads books from file
- `int saveBooks();` //function that saves books to file
- `int loadBorrowings();` //function that loads borrowings from file
- `int saveBorrowings();` //function that saves borrowings to file

Other functions:

- `void cui();` //shows interface at the start of programm (choosing program mode
- `void free_memory();` //function used to free memory if there are out of memory errors and when program is closing
- `int if_int();` //checks if given data (e.g birthdate) is intiger.
- `Date today();` //function that returns todays date structure
- `Date return_date_proposal();` //function that returns proposal date of book's return deadline (30 days from today) as structure

5. SOURCE CODE

Source code is attached to project in archive "Source code".

6. TESTING

Program was tested in any mode that is available in application. Some of them is visible in *User's manual (3. External specification)* However, for some data it couldn't work because of undetected errors.

Exemplary data files, that were created using *Librarian*, are added to report in archive "*Sample data.zip*", which is included to archive "*Application.zip*".

7. CONCLUSIONS

However my project is ready for now, I think that it could be more optimized. There are many possibilities to improve program, such as more functions, like searching reader by personal data, books by title or author.