Hong Kong Diploma of Secondary Education

Examination 20XX

Information and Communication Technology

(Coursework)

Option D: Software Development

Title: School library system

Candidate ID No:

Contents

1. Introduction

1.1 Situation

Our school would like to set up an online library system which is to replace the current library system. As a project manager of this new system, I am to provide solutions for the new requirements.

1.2 User Requirements

After collecting user requirements, a feasibility study was conducted to access whether the solution is possible. From the results provided by system analysts, here are the requirement specifications:

R1: The system authenticates librarians and users by user IDs and passwords.

R2: Users and librarians have access to the system through school intranet and school library.

R3: Circulation is same as the old system and should be made more efficient.

R4: Library resources records should be managed efficiently.

R5: Custom Library Settings: The period of resource borrowing, fine for late return resources, number of resources can be borrowed by users each time, number of resources can be reserved by a user each time, maximum number of renewals for each borrowed resource and, number of days before each reserve is cancelled.

R6: Users are allowed to search library resources online by different criteria and reserves can be made accordingly.

R7: Users are allowed to see their currently reserved resources and borrowed resources. Reserves or borrows can be cancelled or renewed accordingly.

R8: Librarians are allowed to manage user records: name, class and class number.

R9: Both users and librarians are allowed to change their passwords.

R10: The system should be fault tolerant.

R11: The system runs on Microsoft Windows.

R12: The system should manage up to 10000 book records

R13: Librarians should be able to view a list of late return resources and lists of resources in other different status.

2. System Design

2.1 Hardware, Software and User Interface

Hardware:

Hardware	Reasons
Personal computers	The system only involves simple text processing, an average personal computer nowadays can handle task easily.
School intranet	The system runs in school intranet, file sharing in the net is important for this system
Barcode reader	A barcode reader can make the circulation runs smoothly by minimizing errors and time of Book ID and Student ID entry.

Software:

- ✧ Custom software developed by school team is chosen because of the following reasons:
- ✧ Special requirements of school library have to be fulfilled
- ✧ Student records must not be leaked
- ✧ Extensive features provided by commercial software are not necessary, developing software by school team can save the cost of buying software
- ✧ Software developed by school team has low hardware requirements
- ✧ Custom features will be added in the future, so the school must own the software in order to modify the program(copyright)

User Interface:

Command Line Interface is used according to the following reasons:

- ✧ Custom software is compiled in pascal, which uses command line interface
- ✧ Operations are in maximum efficiency by using command line interface, which meets the user requirements of efficiency
- ✧ Command input can be made easier and more user-friendly because the software is implemented by the school team, this remedied the drawback of user-unfriendly of command line interface
- ✧ System resources can be saved
- ✧ The software can be run in low resolution display unit
- ✧ Commands to be entered in each stage is shown on screen. Commands and data are validated once they are input to the system. This minimizes the chance of error input and avoids GIGO.

2.2 System Process

There are 2 types of end users in this library system: users who are using the library service and librarians who are responsible to manage the library resources.

For users:

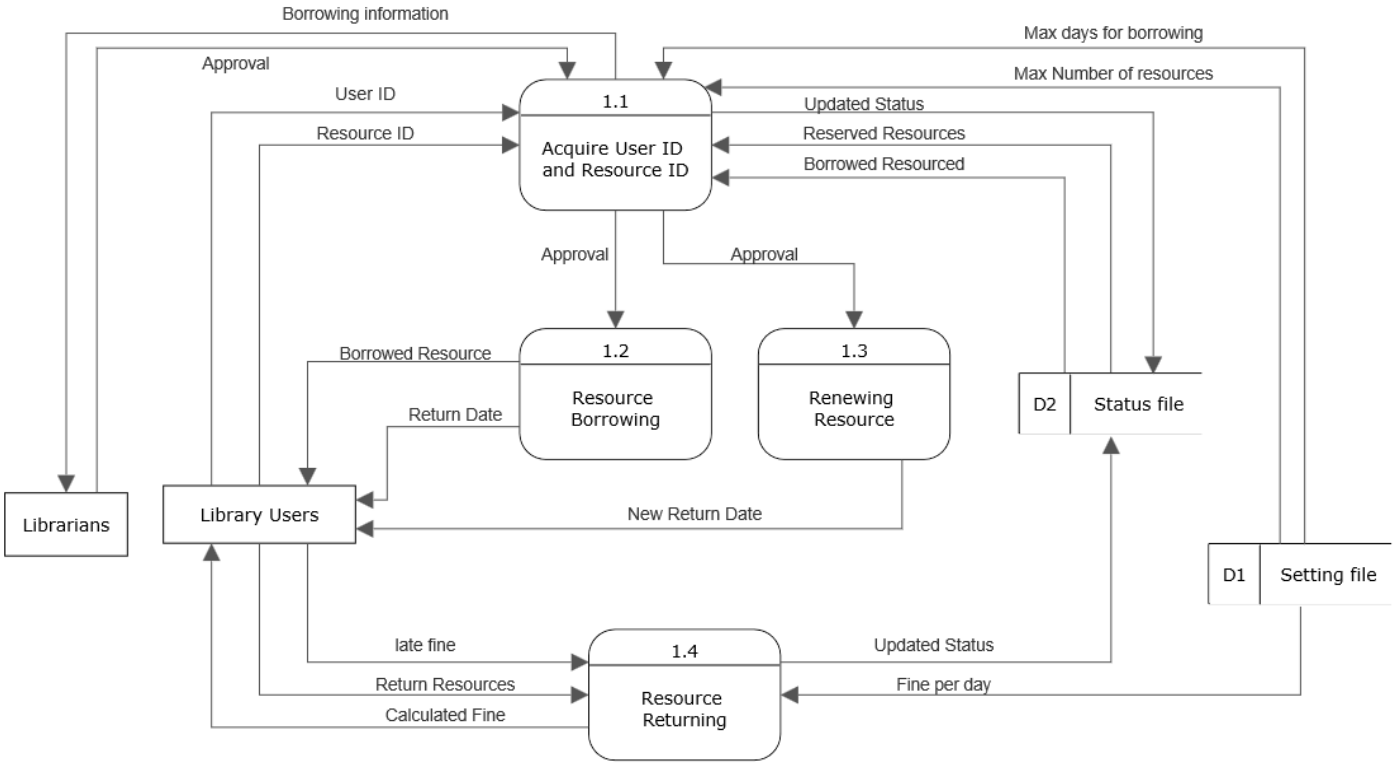
- ✧ They can borrow and return their library resources from the library system by using their user ID and the resource ID at the checkout counter.
- ✧ On the online platform, users can search for a library resource by using different search criteria and reserve the resources that they desire.
- ✧ Also, they can see the library resources they have reserved or borrowed on the platform.
- ✧ Further actions such as cancelling an online reserve or renewal of borrowed resources can be done at the same screen.

For librarians:

- ✧ They are responsible for approving borrowing of library resources at the checkout counter.
- ✧ The system reads a user ID and a resource ID and librarians will decide whether the borrowing, renewal and returning of the resource should continue.
- ✧ After getting the approval form the librarians, the system will update the resource status record accordingly.
- ✧ Resource information such as resource name, resource author and year of publication and user information such as class, class number and password for logging into the online platform can also be updated by librarians.
- ✧ Different library settings can also be made by librarians.

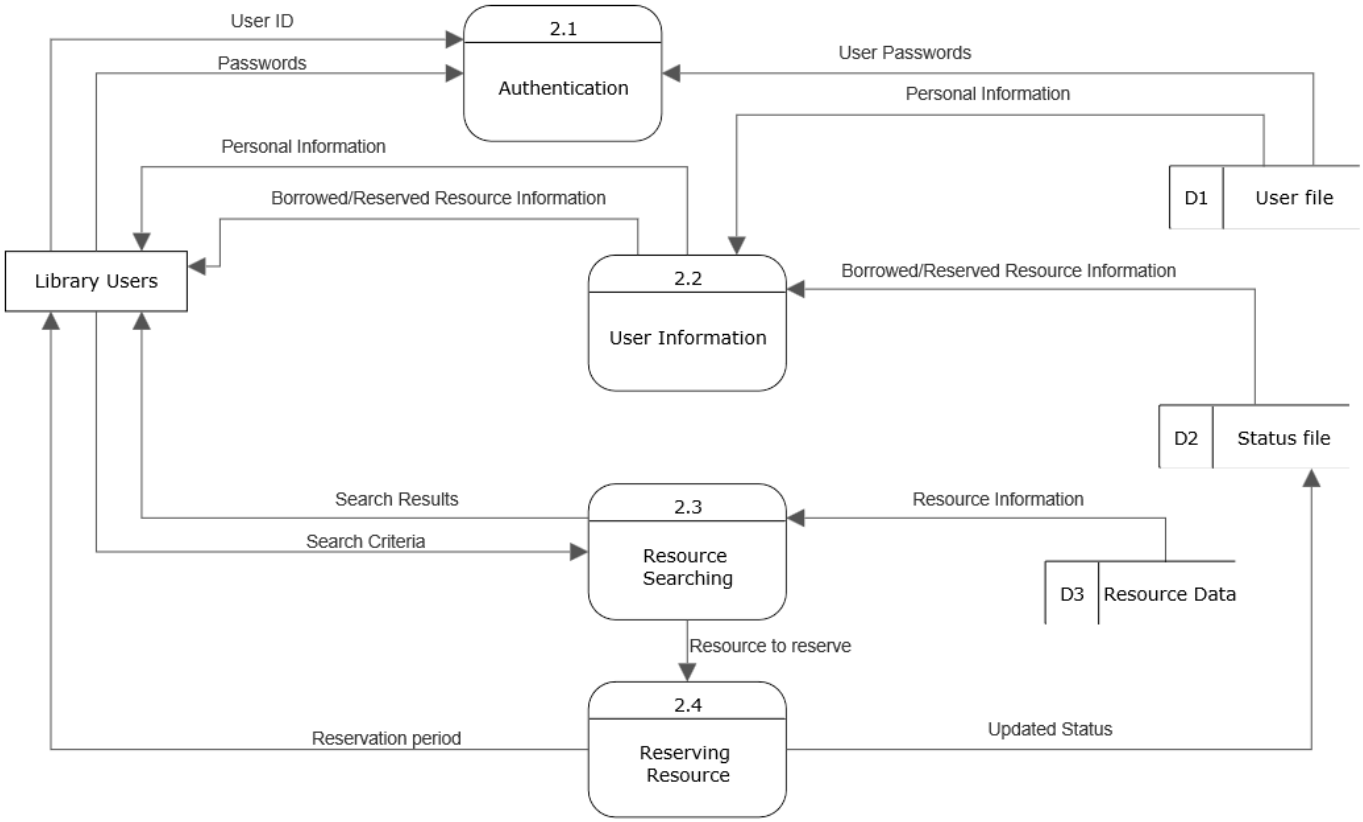
The system can be further divided into 4 major parts: Checkout counter, online platform, Record Management and library settings:

Checkout counter:



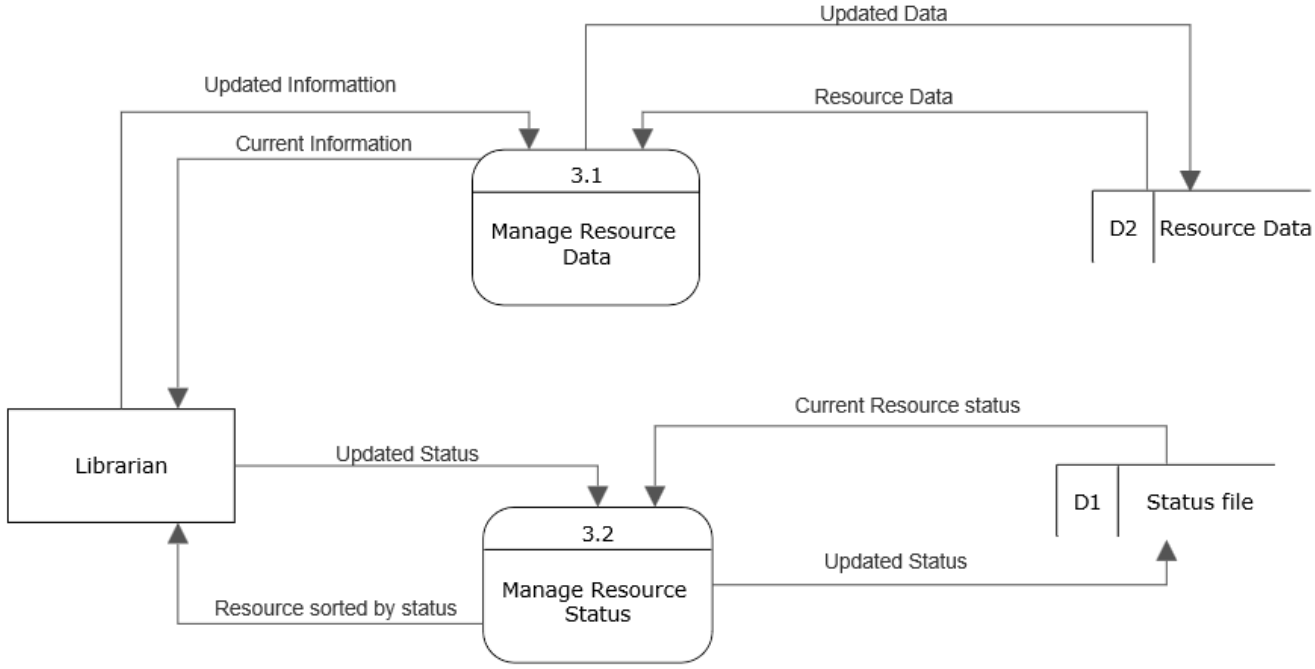
- Circulation of library involves on-shelf resources borrowing, reserved resources borrowing, return borrowed resources and renewal of borrow record.
- It is done by librarians in the library checkout counter.
- Users find their desired resources in the shelves and bring the resources and their ID cards to the checkout counter to perform borrowing, returning and renewal of the borrowing.
- Also, reserved resources can be collected in the checkout counter in case any other users have brought the reserved resources to the checkout counter.

Online Platform:



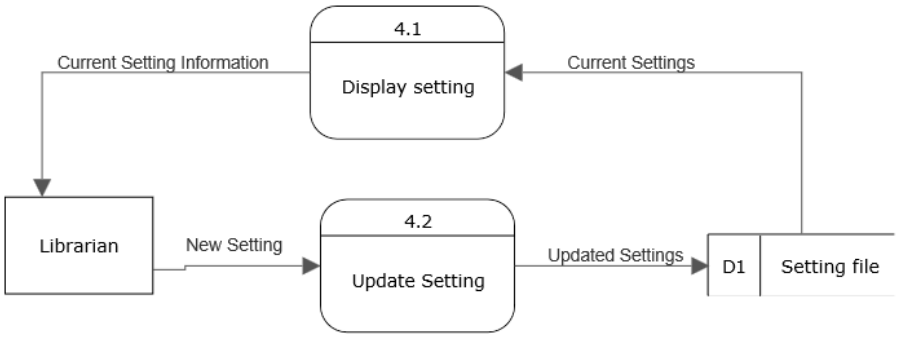
- ✧ Online platform allows users to find resources in the library in the intranet.
- ✧ It can search library resources according to the ISBN, which has to be exact; or search library resource according to different information of the resource: resource type, resource name, owner of the resource and the publisher, which can be a key word.
- ✧ Further actions can be made once the resource can be searched. The resource can be reserved for the users. I
- ✧ If a resource has been reserved, only the user who reserved it can successfully borrow the resource from the checkout counter.
- ✧ The period of reservation is reminded, in which user must go to the school library and take the reserved resource before the reservation is automatically removed from the system.
- ✧ Users can browse the resources they have borrowed or reserved.
- ✧ For borrowed resources, the can renew the borrowing through online platform given they have none of the resources is currently late and the limits of renewal have not been reached.
- ✧ For reserved resources, they can cancel the reservation in the online platform.

Record Management:

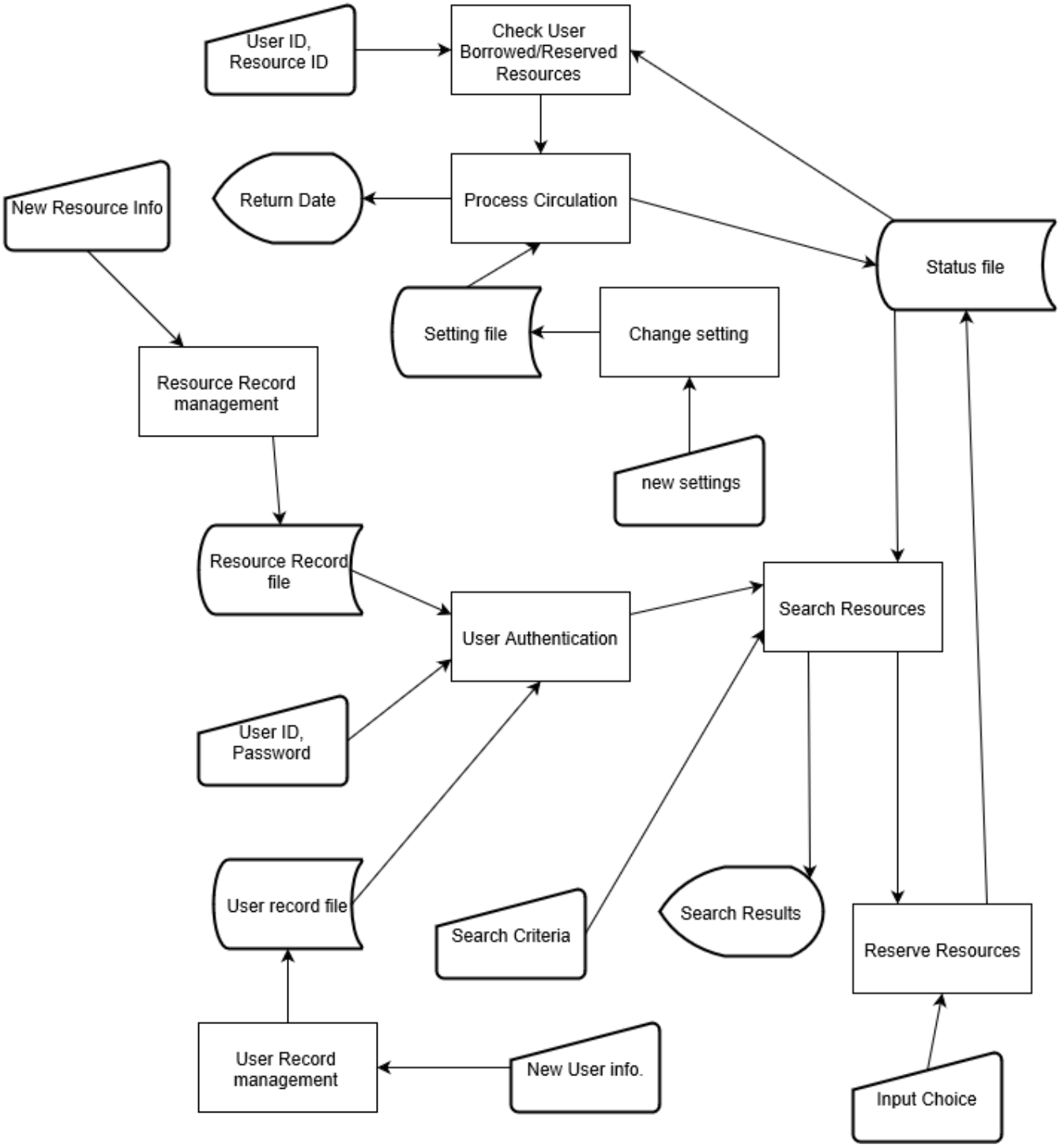


- ✧ Record Management is a process of managing library resource status.
- ✧ It is only accessible by librarians.
- ✧ It allows librarians to modify the status of resources that are not borrowed. This includes updating the status of on-shelf resources, off-shelf resources, and reserved resources.
- ✧ In addition, it allows librarians to modify the records of library resources and users.
- ✧ For library resource, it can change the resource name, resource type, the owner of the resources and publisher.
- ✧ It can generate a list of late returning resources and borrowed resources.

Library settings:



- ✧ A number of library settings can be set by librarians.
- ✧ This includes number of days of the borrowing, length of the reservation period, number of resources can be borrowed a user each time, amount of fine increased each day and number of reserves can be made by a user.



A System flow chart outlines the structure of the library system

2.3 Data management

Data is managed in various ways. Below is a summary table of all the data involved in the system.

User Data:

Data	Description
User ID	A unique code given to every user
User name	Name of the users
User Class	Class of User, e.g. 1A, 2B
Class number	Class number, e.g. 21, 22
Password	Password for Online platform

Resource Data:

Data	Description
Resource ID	A unique code given to every resource currently owned by the library
Resource ISBN	A code which identifies a publication
Resource type	Indicates resource type: Books, Movies, Magazines, Textbooks or others
Resource name	Name/title of the resource
Resource owner	Creator of the resource: author, director, writer etc.
Resource publisher	E.g. Book publisher, movie studio

Resource Status:

Status	Description
On-shelf	Indicates a resource is on the book shelf and is available
Off-shelf	Indicates a resource is off the book shelf and is not available
Reserved	Indicates a resource is only available to the reserved user. It can either be on-shelf or off-shelf while being reserved.
Borrowed	Indicates a resource is not in library
Date	Indicates the deadline of returning the resource, or taking a reserved resource from the library

The data is then stored and accessed in different files. This facilitates all the operations of the library. The files include: Resource status file, a Resource Record file, a User Record file and a library setting file.

Resource Status file:

Data Store	Explanation
Resource ID	Identify individual resource
Resource ISBN	Identify a form of copy of the resource
User ID	Record the user who borrowed/reserved the resource
Date	Record the deadline of returning the resource, or taking a reserved resource from the library
Status	Indicate if the resource is borrowed, on-shelf or off-shelf

Every resources in the library has a record in this file. The Resource ID acts as primary key in this file.

Resource Record file:

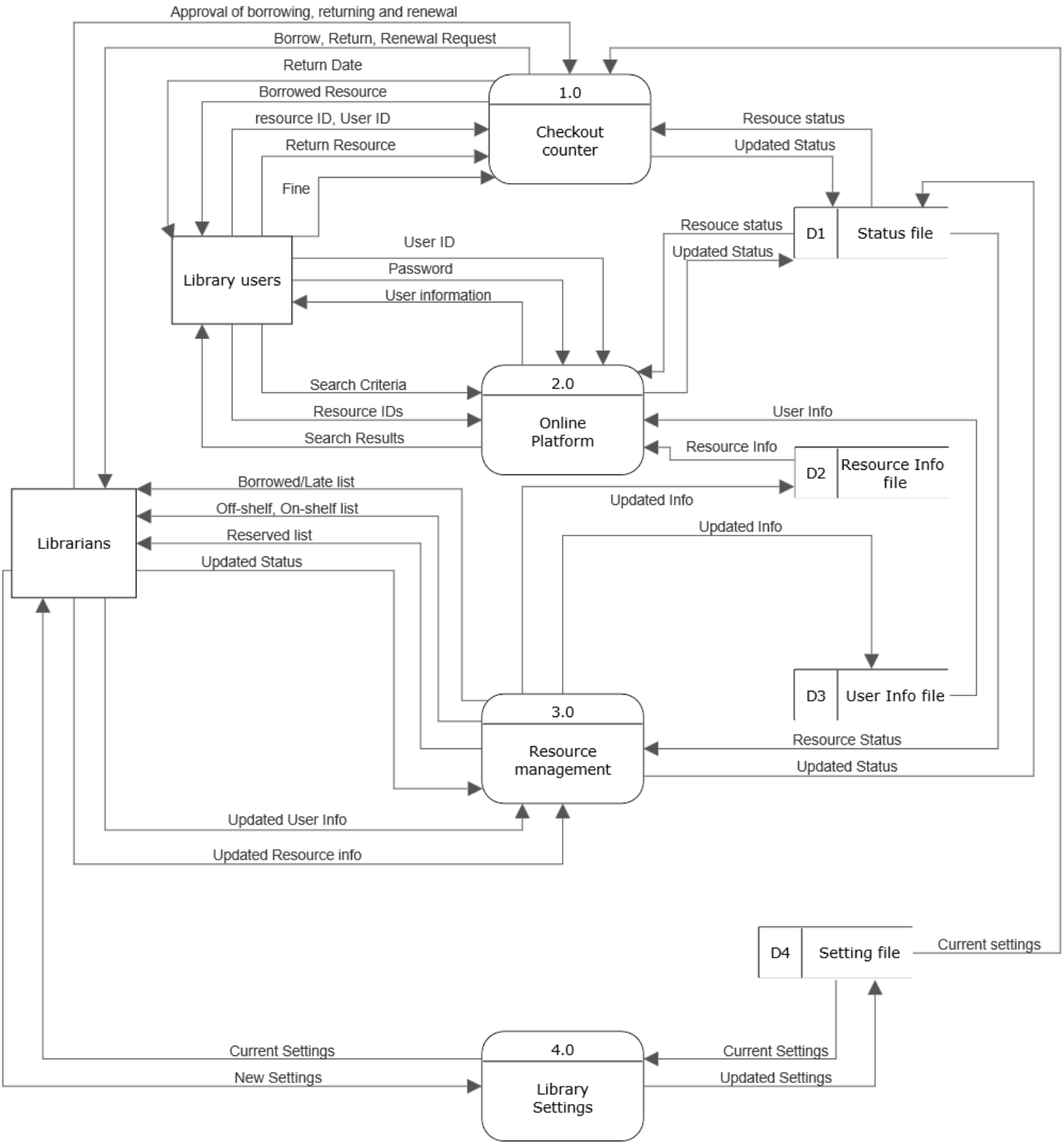
Data Store	Explanation
Resource ISBN	Identify a copy of the resource
Resource type	Stores the type of the resource for searching and displaying purpose
Resource name	Stores the name of the resource for searching and displaying purpose
Resource owner	Stores the owner of the resource for searching and displaying purpose
Resource publisher	Stores the publisher of the resource for searching and displaying purpose

Every different forms of copy has a record in this file. The primary key is Resource ISBN.

User Record file:

Data store	Explanation
User ID	Identify a user for Online platform login
User name	Stores the name of user
User Class	Stores the class of user
Class number	Stores the class number of user
Password	Stores password set by each user

Every users of the library has a record in this file. The primary key is User ID.



This is a level 1 context diagram showing the data flow between each modules.

Notes:

Resource info: Data store in resource record file.

User info: Data store in user record file.

3. System implementation

3.1 Software development

The system software is implemented by using pascal. It consists of a total of four modules which provide the functions specified by system design. The software implementation process is divided into eight phases which were implemented in chronological order as following:

Phase 1: Data Store implementation

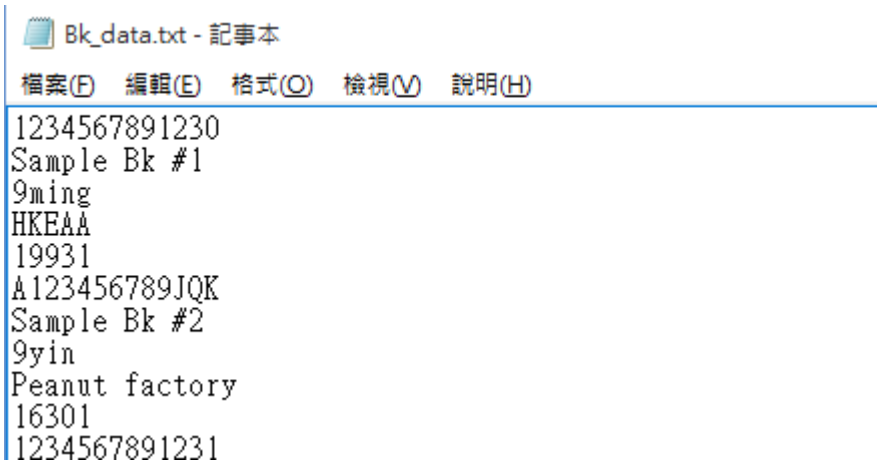
Data Store is the backbone of the library system. It is therefore first implemented in the software development. In this phase, Status file and Record file are implemented.

Step 1: Implementing Data Structure of Resource Information

Data Structure:

Data Store	Variable name	Field length and explanation
Resource ISBN	ISBN	String[13]
Resource type	Res_typ	Char: meaning 1 : Book 2 : Movie 3 : Magazine 4 : Text Book 5 : Others
Resource name	Book_title	String[50]
Resource owner	Book_Author	String[50]
Resource publisher	Book_publisher	String[50]
Publication year	Year_pub	String[4]

To minimize the number of lines in text file, the data is stored as following:



A Resource record occupies 5 lines in a text file

Line1: ISBN
Line2: Book_title
Line3: Book_Author
Line4: Book_publisher
Line5: Year_pub, Res_typ

The Data Store of resource data is then put into record structure. The Record name is called BookDataType, which is declared as follwing:

```
BookDataType = record
                    ISBN: string[13];
                    Book_title, Book_Author, Book_publisher: string[50];
                    year_pub: string[4];
                    Res_typ: char;
               end;
```

Mechanism:

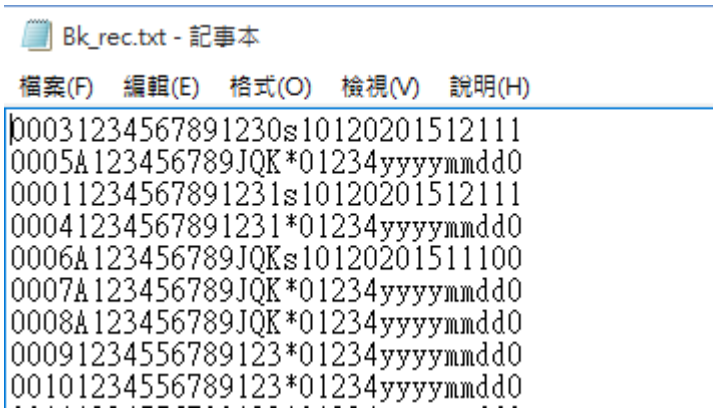
An array of this record structure is used to process all the resource information in the library, namely Resource name, Resource ISBN, Resource owner, Resource Type and Year of publication.

It is a static record file with ISBN as primary key. ISBN is searched in this file and hence, the information of the resources can be fetched accordingly.

Step 2: Implementing the Data Structure of Resource Status

Data Store	Variable name	Field length and explanation
Resource ID	BK_ID	String[4]
Resource ISBN	ISBN	String[13]
User ID	Stud_ID	String[6]
Date	date	String[8]
Status	Status	integer: Meaning -3: Reserved and Off-shelf -2: Reserved and On-shelf -1: Off-shelf/Not available 0 : On-shelf/Available 1 .. 10: Borrowed

To minimize the number of lines in the text file, data is stored as following:



A resource status record occupies one line in the text file.

Line1: BK_ID, ISBN, Stud_ID, Date, Status

The Data Store of resource status is then put into record structure. The Record name is called BookStatType, which is declared as follwing:

```
BookStatType = record
                    Status: integer;
                    Stud_ID: string[6];
                    BK_ID: string[4];
                    ISBN: string[13];
                    date: string[8]
                end;
```

Mechanism:

As the center of the library system, it handles borrowing, reserving and returning of resources. An array of the records is used to store the status of all the resources in the library. It is dynamic and works as the following:

Borrowing:

- ✧ While the status of the book = 0, the Resource can be borrowed.
- ✧ When the resource is first borrowed, the status value increases by 1. The Stud_ID of the record will store the user_ID of the user. The Date of the record will store the return date of the resource.
- ✧ Every time when the borrowing is renewed, the status value increases by 1. The return date is also updated.

Reserving:

- ✧ While the status of the book = 0, the Resource can be reserved.
- ✧ If the resource is reserved via online platform, the status value will become -2. The Stud_ID of the record will store the user_ID of the user. The Date of the record will store the date of cancellation of the reservation.
- ✧ If the resource is going to be borrowed by other user during reservation period, the librarian will stop the borrowing and take away the resource. The status of the resource will become -3.

Returning:

- ✧ When the resource is returned by a user, the status will be set to -1. The Date of the record will be set to yyyymmdd. The Stud_ID of the record will be set to *01234.
- ✧ If the resource is to be borrowed or reserved again, the status will be set to 0.

End of Phase1.

Phase 2: Implementation of data loading and saving procedures

Every time when the program is executed, status and record is loaded to the array of status and records respectively. After each actions involve changes of records and status, the changes are saved. This facilitates the data processing of the library system and hence implemented in phase 2.

Step 1: Implementing the loading procedure

Resource status and record is loaded from text files into status and Information records via this procedure. It is also responsible for checking the current date and count how many resources and titles are currently in the library whenever it is called.

It is implemented as a procedure with no data entity and no interactions with users and librarians. (Procedure Reload;)

Mechanism:

Generally, a while loop is used to load all the required records. Until the loading of records from one file has finished, the loading of another records from another file will start. Loading form different files is done in order as following:

Checking the Current date:

- ✧ A procedure from pascal library called GetDate is used to get integral vaules of year, month and day of the current date.
- ✧ The three integral values are transformed into one 8 digit string value. Yyyy + mm + dd → yyyymmdd.
- ✧ The string value of date is stored in global string variable: ToDae.

Loading the Resource Information:

- ✧ Resource information records are loaded form Bk_data.txt by using 'while not eof'.
- ✧ Plain loading and no algorithm involved.

Accounting

- ✧ When a resource information record is loaded successfully, a global variable storing total number of titles in the library: ISBN_total will increase by 1.
- ✧ After loading is finished, total number of resources equals to ISBN_total.

Loading the Resource Status:

- ✧ Resource status records are loaded from Bk_rec.txt by using 'while not eof'.
- ✧ If a record with resource ID's first digit = '-', the record is ignored and not loaded into the array of resource status record.
- ✧ If a record with status = -2 or -3, and the date of the record is smaller than global variable ToDae, the date of the record is reset to yyyyddmm and the Stud_ID of the record is reset to *01234. In this case, if status = -2, it will be reset to 0, or if status = -3, it will be reset to -1.
- ✧ When a resource status record is loaded successfully, a global variable storing total number of titles in the library: Book_total will increase by 1.
- ✧ After loading is finished, total number of resources equals to Book_total.

Step 2: Implementing the saving procedure

Resource status and record is saved to text files via this procedure in designated formats.

It is also implemented no interactions with users and librarians but it has data entity to indicate which part of the saving procedure is required:

Procedure OverWrite (OverWriteFile: integer);

Mechanism:

Generally, a for-do loop is used to save all the required records. It includes different parts among which only one is used when this procedure is called. A flag: OverWriteFile is used to indicate which part of the procedure will be used. The different parts are listed as following:

Saving the Resource Information:

- ✧ When the flag: OverWriteFile = 1, this part is executed.
- ✧ Resource information records are saved to Bk_data.txt by using for-do loop. The number of saving operations equals to sum of titles in the library.
- ✧ Plain saving and no algorithm involved.

Saving the Resource Status:

- ✧ When the flag: OverWriteFile = 2, this part is executed.
- ✧ Resource status records are saved to Bk_data.txt by using for-do loop. The number of saving operations equals to the number of resources in the library.
- ✧ Plain saving and no algorithm involved.

End of Phase 2

Phase 3: Implementation of Resource data management

After the implementation of the first 2 phases, only syntax errors can be fixed. After this phase is finished, the first unit test can be conducted. Hence, putting it as phase 3 can minimize the amount of errors to be fixed in later phases.

Resource data is updated when there is addition of titles and resources to the current library. Resource data is also updated when there are changes on the resource information and removals of resources from the library.

Therefore, Resource data management is divided into addition of resources and resource titles, changes in resource information and removal of resources.

Step 1: Implementing addition of resources and resource titles procedure

Addition of resources and resource titles requires librarians. Hence a user interface is implemented.

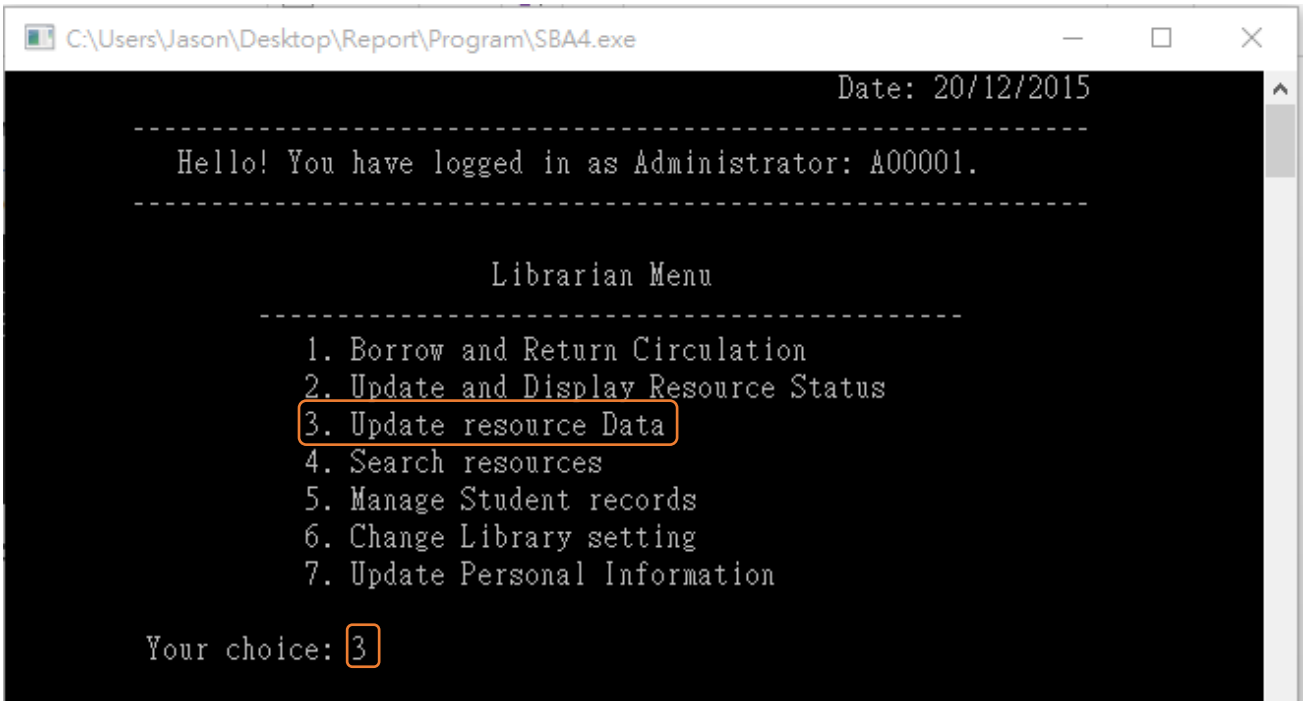
Both procedures implemented in the previous phase will be called.

No data flow from other modules as it is a separate procedure.

Mechanism:

A series of questions are used to obtain all the information required to add a new title to the library. Librarians input all the information of the resources by answering the questions:

At Librarian Menu:



Enter 3 to access this module.

Flow of inputting information of the new title:

```
C:\Users\Jason\Desktop\Report\Program\SBA4.exe
                                                    Date: 20/12/2015
    ------------------------------------------------------------

                        Resource Data Management
            ---------------------------------------------

            ISBN/Product code: 123456789AAAA
```

✧ Resource ISBN of the new title is entered

```
C:\Users\Jason\Desktop\Report\Program\SBA4.exe
                                                    Date: 20/12/2015
    ------------------------------------------------------------

                        Resource Data Management
            ---------------------------------------------

            ISBN/Product code: 123456789AAAA
    ------------------------------------------------------------
    >>Required Resource code/ISBN Not found!
    Add New Resource Record?[Y/N]
```

✧ The system reminds the librarian the ISBN cannot be found in the library and asks if the librarian wants to add a new title.

```
C:\Users\Jason\Desktop\Report\Program\SBA4.exe
                                                    Date: 20/12/2015
    ------------------------------------------------------------

                        Resource Data Management
            ---------------------------------------------

            ISBN/Product code: 123456789AAAA
    ------------------------------------------------------------
    >>Required Resource code/ISBN Not found!
    Add New Resource Record?[Y/N] y
```

✧ If yes, the following information will be entered:

```
Adding New Resource: Resource Type
-------------------------------------------

1. Book
2. Movie
3. Magazine
4. Text Book
5. Others

Your choice: 1
```

✧ Resource Type of the new title is entered.

```
Adding New Book
-------------------------------------------

Resource name/Book title: Sample title #11
Resource owner/Book author: Sample Author
Publisher/Producer name: Sample Publisher
Year of publish: 2015
```

✧ Resource title of the new title is entered.

✧ Resource owner of the new title is entered.

✧ Publisher is entered.

✧ Year of publish is entered.

```
Adding New Book
-------------------------------------------

Resource name/Book title: Sample title #11
Resource owner/Book author: Sample Author
Publisher/Producer name: Sample Publisher
Year of publish: 2015
Confirm The above information?[Y/N] y
>>New Resource Record have been added!
```

✧ Confirmation required.

```
A123456789123
Sample Bk #5
Sample Chan
Sample Publications
19222
123456789AAAA
Sample title #11
Sample Author
Sample publisher
20151
```

- ✧ If yes, a new resource title will be added to the library.

Adding new title to the library:

- ✧ The total number of titles in the library increases by 1. Hence, global variable: ISBN_total := ISBN_total + 1.
- ✧ A new record is added to the existing records.
- ✧ Call Procedure OverWrite(1) to save the new records to the text file.
- ✧ The total number of resources in the library increases by 1. Hence, global variable: Bk_total := Bk_total + 1.
- ✧ An algorithm is then used to find the smallest available Bk_ID in the library.
- ✧ The new resource is assigned with the smallest Bk_ID found.
- ✧ Call Procedure OverWrite(2) to save the new status records to the text file.

Step 2: Implementing Amendments on resource titles

A user Interface is required.

Both procedures implemented in the previous phase will be called.

No data flow from other modules as it is a separate procedure.

Mechanism:

It is also done by a series of questions. Librarians input the amended information by answering the questions on the screen. If the field of information does not need amendment, '-' is entered to ignore the change.

Flow of inputting amended information of the titles:

```
1234556789123
Sample Bk #4
36ming
45ming
13313
A123456789123
Sample Bk #5
Sample Chan
Sample Publications
19222
```

```
            Resource Data Management
-------------------------------------------

ISBN/Product code: A123456789123
```

✧ Resource ISBN of the title is entered.

```
            Update Resource Information
-------------------------------------------
  Resourec type: Movie
  Resource title: Sample Bk #5
  Resource owner/ Book owner: Sample Chan
  Publisher: Sample Publications
  Year of publish: 1922
-------------------------------------------

        1. Edit information
        2. Add/Remove the Resources

    Your choice:
```

✧ The ISBN is found by the system.

✧ Librarian choose to amend the information of the title.

```
          Update Resource Type
-------------------------------------------

    1. Book
    2. Movie
    3. Magazine
    4. Text Book
    5. Others

  Your choice: 1
```

✧ Updated resource type is entered.

```
                Updating Book Information
     ------------------------------------------

 Resource name/Book title: Sample #11
 Resource owner/Book author: Sample Writer
 Publisher/Producer: Test Publisher
 Year of publish: 2016
```

- ✧ Updated resource title is entered.
- ✧ Updated resource owner is entered.
- ✧ Updated Publisher is entered.
- ✧ Updated Year of publish is entered.

```
                Updated Information
    ------------------------------------------

ISBN/Product code: A123456789123
Resource name/Book title: Sample #11
Resource Type: Book
Resource owner/Book author: Sample Writer
Publisher/Producer: Test Publisher
Year of publish: 2016

      Confirm changes? y
```

- ✧ Confirmation required.

```
A123456789123
Sample #11
Sample Writer
Test Publisher
20161
123456789AAAA
Sample title #11
Sample Author
Sample Publisher
20151
```

- ✧ Resource information is updated.

Updating resource information:

- ✧ The existing record is replaced by the amended record.
- ✧ Call Procedure OverWrite(1) to save the new records to the text file.

Step 3: Implementing addition and removal of resources with existing title

A user Interface is required.

Both procedures implemented in the previous phase will be called.

No data flow from other modules as it is a separate procedure.

Mechanism

When doing addition of such titles, number of resources to be added will asked. When doing removal of such titles, a list of the existing resources with the title will be displayed. The resources to be removed will be entered.

Flow of adding resources:

```
            Resource Data Management
---------------------------------------------

ISBN/Product code: 1234567891230
```

- ✧ Resource ISBN of the title is entered.

```
            Update Resource Information
---------------------------------------------
  Resourec type: Book
  Resource title: Sample title #11
  Resource owner/ Book owner: Sample Author
  Publisher: Sample Publisher
  Year of publish: 2015
---------------------------------------------

        1. Edit information
        2. Add/Remove the Resources

   Your choice: 2
```

- ✧ The ISBN is found by the system.

```
          Update Number of Resources
 ---------------------------------------------
    Resourec type: Book
    Resource title: Sample title #11
    Resource owner/ Book owner: Sample Author
    Publisher: Sample Publisher
    Year of publish: 2015
 ---------------------------------------------

        1. Add Resource titles
         2. Delete Resources titles

    Your choice: 1
```

```
          Your choice: 1
 ---------------------------------------------
>>Number of the Resources in library: 1
  Number of Resources to add? 2
```

✧ Librarian enters the number of resources to be added.

```
 ---------------------------------------------
 >>Number of the Resources in library: 1
   Number of Resources to add? 2
 >>Resource(s) added successfully!
```

```
Bk_rec.txt - 記事本
檔案(F)  編輯(E)  格式(O)  檢視(V)  說明(H)

00031234567 89AAAA*01234yyyymmdd-1
0012 1234567891230*01234yyyymmdd-1
0019 1234567891230*01234yyyymmdd-1
```

✧ Resource records will be updated.

Updating resource information:

✧ The total number of resources in the library increases by the value which the librarian has entered. Hence, global variable: Bk_total := Bk_total + n.

✧ An algorithm is then used to find the smallest available Bk_IDs in the library.

✧ The new resources are assigned with the smallest Bk_IDs found.

✧ Call Procedure OverWrite(2) to save the new status records to the text file.

Flow of removing resources:

- ✧ Resource ISBN of the title is entered.
- ✧ The ISBN is found by the system.

```
-------------------------------------------
  Resourec type: Book
  Resource title: Sample title #11
  Resource owner/ Book owner: Sample Author
  Publisher: Sample Publisher
  Year of publish: 2015
-------------------------------------------

      1. Add Resource titles
      2. Delete Resources titles

  Your choice: 2
```

```
                Removing Resources
      -------------------------------------------

Resource ID: 0002  Borrowed by s10120 until 31/12/2015
Resource ID: 0012  Off shelf
Resource ID: 0019  Off shelf

Enter a Resource ID to remove:
```

- ✧ A list of resources with the ISBN will be displayed

```
Resource ID: 0002  Borrowed by s10120 until 31/12/2015
Resource ID: 0012  Off shelf
Resource ID: 0019  Off shelf

Enter a Resource ID to remove: 0019
-------------------------------------------------------
>>Resource removed successfully
```

- ✧ Librarian enters the Resource ID of the resource to be removed

```
Bk_rec.txt - 記事本
檔案(F)  編輯(E)  格式(O)  檢視(V)  說明(H)

0003123456789AAAA*01234yyyymmdd-1
0012123456789123 0*01234yyyymmdd-1
-12-123456789123 0*01234yyyymmdd-1
```

- ✧ Resource records will be updated.

Updating resource information:

- ✧ The total number of resources in the library decreases by 1. Hence, global variable: Bk_total := Bk_total -1.
- ✧ The first character of Bk_ID of the record will be replaced by '-', which denotes the record will be ignored next time the Procedure Reload is called.
- ✧ Call Procedure OverWrite(1) to save the new records to the text file.
- ✧ The Procedure Reload is called and a new list is displayed.

End of Phase 3

Phase 4: Implementation of Resource status management

After the first 3 phases of implementation, the system can now handle the resource records. New titles and resources can be added to or removed from the system. A unit test is conducted after the first three phases and several syntax errors are fixed.

Before resource status can be managed, some resource records must be added to the system. The data management function implemented in the first 3 phases are therefore used to add the required resource records.

Resource status is updated automatically when borrowing, reserving and returning. However, sometimes status has to be updated by librarians manually. This happens when resources are put to the shelves or removed from the shelves. Librarians can also remove a reservation made by users. This also requires manual update of resource status.

Step 1: Implementing a dummy procedure

General flow of changing the status is similar. A dummy procedure is hence implemented. Dialogs and status to be set in the flow can be substituted. The setting of status involves several changes. They are, from On-shelf to Off-shelf, from Off-shelf to On-shelf, from reversed to off-shelf and from reserved to on-shelf.

Procedure OnOfflist(Onf: char);

Mechanism:

A flag (Onf) is first entered to the dummy procedure to indicate which changes will be involved. Then a list of resources grouped by the required status will be displayed. Librarian enters a resource ID displayed on the screen to update the status of the library resources.

Onf value and corresponding substitute actions:

Onf: char value	List displayed	Change of status
'3'	List of on-shelf resources	On-shelf → Off-shelf Status file: 0 → -1
'4'	List of off-shelf resources	Off-shelf → On-shelf Status file: -1 → 0
'5'	List of Reserved resources (On-shelf)	Reserved → On-shelf Status file: -2 → 0
'6'	List of Reserved resources (Off-shelf)	Reserved → Off-shelf Status file: -3 → -1

At Administrator menu:

```
-------------------------------------------------------------
  Hello! You have logged in as Administrator: A00001.
-------------------------------------------------------------

                    Librarian Menu
        ---------------------------------------------
          1. Borrow and Return Circulation
          2. Update and Display Resource Status
          3. Update resource Data
          4. Search resources
          5. Manage Student records
          6. Change Library setting
          7. Update Personal Information

 Your choice: 2
```

Flow of changing status:

```
          Display Resource Status
---------------------------------------------
    1. Display Borrowed resource list
    2. Display Late Return resource list

       Update & Display Resource Status
---------------------------------------------
    3. Manage On shelf Resource
    4. Manage Off shelf Resource
    5. Manage reserved Resource(On Shelf)
    6. Manage reserved Resource(Off Shelf)

Your Choice: 3
```

✧ Librarian chooses to update resources with the required status.

```
            Update & Display Resource Status
      ---------------------------------------------
          Resource ID: 0001 On shelf
          Resource ID: 0004 On shelf
          Resource ID: 0006 On shelf
          Resource ID: 0007 On shelf
          Resource ID: 0008 On shelf
          Resource ID: 0009 On shelf
          Resource ID: 0010 On shelf
          Resource ID: 0011 On shelf

 Resource to be off-shelf:
```

✧ A list of resources with the required status is displayed

```
            Update & Display Resource Status
      ---------------------------------------------
         Resource ID: 0001 On shelf
         Resource ID: 0004 On shelf
         Resource ID: 0006 On shelf
         Resource ID: 0007 On shelf
         Resource ID: 0008 On shelf
         Resource ID: 0009 On shelf
         Resource ID: 0010 On shelf
         Resource ID: 0011 On shelf

Resource to be off-shelf: 0011
```

✧ Librarian chooses to the resources on the list to update their status

```
            Update & Display Resource Status
      ---------------------------------------------
         Resource ID: 0001 On shelf
         Resource ID: 0004 On shelf
         Resource ID: 0006 On shelf
         Resource ID: 0007 On shelf
         Resource ID: 0008 On shelf
         Resource ID: 0009 On shelf
         Resource ID: 0010 On shelf
         Resource ID: 0013 On shelf

Resource to be off-shelf:
```

```
           Display Resource Status
---------------------------------------------
    1. Display Borrowed resource list
    2. Display Late Return resource list

       Update & Display Resource Status
---------------------------------------------
    3. Manage On shelf Resource
    4. Manage Off shelf Resource
    5. Manage reserved Resource(On Shelf)
    6. Manage reserved Resource(Off Shelf)

Your Choice: 4
```

```
            Update & Display Resource Status
      ---------------------------------------------
          Resource ID: 0005 Off shelf
          Resource ID: 0011 Off shelf
          Resource ID: 0003 Off shelf
          Resource ID: 0012 Off shelf

 Resource to be put on shelf:
```

✧ System process the change

```
00091234556789123*01234yyyymmdd0
00101234556789123*01234yyyymmdd0
0011123455678912 3*01234yyyymmdd-1
```

✧ Changes are saved to resource status file

System process in changing the status:

✧ Two char variables: ShoStat meaning ShowedStatus and ReplaStat meaning ReplaceStatus are used to hold the required status and the updated status.

✧ Upon receiving the Onf value, an algorithm is used to substitute appropriate status value to the ShoStat and ReplaStat variable. E.g. When Onf = '3', 0 → ShoStat, -1 → ReplaStat.

✧ A search algorithm is then used to search resource IDs with status = ShoStat in the resource status file and the resource IDs with status matched ShoStat are held in a temporary array.

✧ The array of Resources IDs are displayed as a list of IDs.

✧ Upon receiving a resource ID, the status of the resource will be substituted by ReplaStat. i.e. ReplaStat → Status of resource ID.

Minor feature:

✧ An algorithm is used to display only eight of the list of resource IDs.

```
            Update & Display Resource Status
      ---------------------------------------------
          Resource ID: 0005 Off shelf
          Resource ID: 0001 Off shelf
          Resource ID: 0004 Off shelf
          Resource ID: 0006 Off shelf
          Resource ID: 0007 Off shelf
          Resource ID: 0008 Off shelf
          Resource ID: 0009 Off shelf
          Resource ID: 0010 Off shelf

 Resource to be put on shelf: *.*
```

```
          Display Resource Status
-------------------------------------------
    1. Display Borrowed resource list
    2. Display Late Return resource list

       Update & Display Resource Status
-------------------------------------------
    3. Manage On shelf Resource
    4. Manage Off shelf Resource
    5. Manage reserved Resource(On Shelf)
    6. Manage reserved Resource(Off Shelf)

Your Choice: 3
```

```
            Update & Display Resource Status
      -------------------------------------------
         Resource ID: 0005 On shelf
         Resource ID: 0001 On shelf
         Resource ID: 0004 On shelf
         Resource ID: 0006 On shelf
         Resource ID: 0007 On shelf
         Resource ID: 0008 On shelf
         Resource ID: 0009 On shelf
         Resource ID: 0010 On shelf

 Resource to be off-shelf:
```

✧ Instead of entering the resource IDs to update the status, '*.*' can be entered to update the status of all the displayed Resources.

End of Phase4

Phase 5: Implementation of Circulation software

After 4 phases of implementation, the software for handling status and resource information (Data Management Module) has been completed. The second unit test is conducted before this stage and several bugs are fixed. The system can now handle status and resource information correctly.

In phase 5, either circulation software or online platform of the system can be implemented. However, due to lack of programmers, only one of the above can be implemented in phase 5 and circulation is chosen in this phase.

The circulation software is used in checkout counter to facilitate efficient Circulation of library resources.

Procedure BNRCirc (Borrow and Return Circulation) is a procedure used to handle the all circulation processes. They include:

- ✧ Borrowing of On-shelf library resource,
- ✧ Borrowing of reserved resources,
- ✧ Renewal of borrowing,
- ✧ Returning of borrowed resources and
- ✧ Collection of fine.

Step 1: Implementing Date processing algorithm

In any process related to circulation, date processing is required. Hence, a date processing algorithm is implemented before any processes above.

The date processing is to calculate the date after a specific number of days after the current day. i.e. Today = 2015/11/29 + 7 days = 2015/12/6

Mechanism:

An algorithm is needed to add the number days to the current date. However, Pascal cannot check if a calculated date is correct. E.g. 2015/11/31. Hence, another algorithm is needed to check if the calculated date exist.

In FreePascal, there is a function in the compiler library that can check if the date exist. IsValidDate (yyyy, mm, dd: integer): Boolean. However, in dev-pascal, this function is absent and this function has to be also implemented manually.

Date format: yyyymmdd

- ✧ A string[8] is used to hold a date, in form of yyyymmdd.
- ✧ Comparison between days can be made directly. i.e. date1 > date2
- ✧ Storing date is efficient as only one variable is involved.

IsValidDate Function:

```
Function IsValidDate(yyyy, mm, dd: integer):boolean;
var Monthlist: array[1..12] of integer;
    i : integer;
begin
  For i := 1 to 7 do
    if odd(i) then
      Monthlist[i] := 31
    else
      Monthlist[i] := 30;
  For i := 8 to 12 do
    if not odd(i) then
      Monthlist[i] := 31
    else
      Monthlist[i] := 30;
  Monthlist[2] := 28;
  If yyyy MOD 4 = 0 then
    Monthlist[2] := 29;
  If (yyyy MOD 100 = 0) and not (yyyy MOD 400 = 0) then
    Monthlist[2] := 28;
  IsValidDate := (dd <= Monthlist[mm]) and (mm < 13)
end;
```

- A Monthlist is generated to store the number days in a month.
  E.g. Monthlist[1] = 31, Monthlist[9] = 30.
- Special cases are considered, e.g. leap year once 4 year, once 100 year and Monthlist[2] is changed accordingly.
- If the dd(days) is smaller than the number days in that month(largest possible days), the date is valid. Monthlist[mm] >= dd returns true.

Add Date algorithm: CverDate(<date string>, <number of days forward>);

- The date string: string[8], in form of yyyymmdd, is broken down into integers: yyyy, mm and dd which store the integral value of day, month and year.
- A loop is used to increment the dd value, each time by 1, until the number of increment = <number of days forward>.
- In each loop, IsValidDate function is used to check the yyyy, mm, dd values. This makes sure the addition of date is done correctly
- The yyyy, mm and dd reassembles into 8-digit string, yyyymmdd.

Checking number days between two dates:

- Add Date algorithm: CverDate is used.
- To calculate the days between date1 and date2 (date1 > date2), date2 is added until it equals to date1. i.e. In a repeat loop, CverDate(date2, i), increment i, until date1 = CverDate(date2, i).
- After the looping, i equals to the number of days between two dates.

Step 2: Implementing flow of circulation

Before any of the circulation process can be performed, Resource ID and User ID must be acquired. Also, data validation and verification on the two inputted data must be performed. Hence, a flow of circulation is implemented to maximize the efficiency of circulation.

Mechanism:

As a whole, a repeat loop is used to make different circulation processes continuous. The system will process all the circulation processes of one user first, then proceed to another user. Time sharing is not employed as this will make the circulation process chaotic.

At librarian menu:

```
-------------------------------------------------------------
  Hello! You have logged in as Administrator: A00001.
-------------------------------------------------------------

                    Librarian Menu
        ---------------------------------------------
          1. Borrow and Return Circulation
          2. Update and Display Resource Status
          3. Update resource Data
          4. Search resources
          5. Manage Student records
          6. Change Library setting
          7. Update Personal Information

 Your choice: 1
```

Flow of circulation:

- ✧ Users of the library line up and wait for the checkout counter handling the requests.
- ✧ The user approaches the checkout counter.

```
C:\Users\Jason\Desktop\Report\Program\SBA4.exe                    —
                                                  Date: 20/12/2015
      -------------------------------------------------------------

                        Circulation software
            ---------------------------------------------

              User ID: s10120
```

- ✧ Librarians get his/her User ID which is on the student ID card and it is inputted into the circulation system by using a barcode reader.

```
                                                    Date: 20/12/2015
-------------------------------------------------------------------
                      Borrowed Resources
          ---------------------------------------------
   ID     Return Date      Type          Resource name
 0016      03/01/2016      Book      Sample #11
 0017      31/12/2015      Book      Sample #11
 0002      31/12/2015      Book   Sample title #11

                     Reserved Resources
          ---------------------------------------------
   ID   Cancel Before      Type          Resource name
 0005      26/12/2015      Book    Sample Bk #2
 0006      26/12/2015      Book    Sample Bk #2
 0013      26/12/2015      Book      Sample #11
 0018      26/12/2015      Book      Sample #11

 Enter a Resource ID:
```

- ✧ User ID is validated and a list of the user's reserved resources and a list of the user's borrowed resources are displayed to the librarian.
- ✧ Librarian checks the user's borrowed and reserved resources and decides if the circulation process should continue.

Accounting:

- ✧ When a user ID is verified, the system will count the number of reserved resources, number of borrowed resources and number of late return resources of the user.
- ✧ This process is not shown to the librarian.

System process in circulation software:

- ✧ Upon getting the User ID from barcode reader, the User ID is searched from the user record file.
- ✧ If the user ID is found, the user ID is then searched in Resource status file. If records with the field Stud_ID = User ID, the resource IDs of the records are put into a temporary array.
- ✧ Each time when Stud_ID = User ID, the system checks the resource status of the records. An algorithm is used to count the number of borrowed resources of the user and store the result in NumBorr, the number of reserved resources of the user and store the result in NumBooked and the number of late return resources and store the result in NumLate.
- ✧ The system then display the resource IDs in the temporary array grouped by status on the screen. The two groups are: User's borrowed resources and user's reserved resources.

Step 3: Implementing borrowing processes of library resource

Borrowing processes include borrowing of on-shelf library resource, borrowing of reserved resources and renewal of borrowings. These three circulation processes are grouped under borrowing processes as they are similar.

The borrowing processes follow the input of user ID to the system. The resource IDs of the resources to be borrowed are entered to system.

Flow of borrowing on-shelf library resources:

```
                                                Date: 20/12/2015
-------------------------------------------------------------
                    Borrowed Resources
        ---------------------------------------------
   ID    Return Date     Type         Resource name
 0016     03/01/2016     Book     Sample #11
 0017     31/12/2015     Book     Sample #11
 0002     31/12/2015     Book  Sample title #11

                   Reserved Resources
        ---------------------------------------------
   ID  Cancel Before     Type         Resource name
 0005     26/12/2015     Book   Sample Bk #2
 0006     26/12/2015     Book   Sample Bk #2
 0013     26/12/2015     Book     Sample #11
 0018     26/12/2015     Book     Sample #11

 Enter a Resource ID: 0009
```

- ✧ The resource IDs of the resources are entered one by one to the system.
- ✧ The system checks the number of late resources and the number of borrowed resources of the user.(An Asterisk at the Return Date)
- ✧ If the number of late resources = 0 and number of borrowed resources does not exceed the limitation of the library, the borrowing continues.

```
                 On-shelf Resource Borrowing
        ---------------------------------------------

          Book ISBN/Product code: 1234556789123
          Book ID: 0009

-------------------------------------------------------------
Borrow the above Resource?[Y/N]
```

- ✧ Librarian confirms the borrowing of the resources.
- ✧ Perform system process of borrowing described in phase 1.

Flow of borrowing reserved library resources:

```
                                                    Date: 20/12/2015
 ------------------------------------------------------------
                      Borrowed Resources
          ---------------------------------------------
    ID    Return Date     Type         Resource name
  0016     03/01/2016     Book     Sample #11
  0017     31/12/2015     Book     Sample #11
  0002     31/12/2015     Book  Sample title #11

                      Reserved Resources
          ---------------------------------------------
    ID  Cancel Before     Type         Resource name
  0005     26/12/2015     Book   Sample Bk #2
  0006     26/12/2015     Book   Sample Bk #2
  0013     26/12/2015     Book     Sample #11
  0018     26/12/2015     Book     Sample #11

  Enter a Resource ID: 0005
```

```
                                                    Date: 20/12/2015
------------------------------------------------------------

                  Reserved Resource Borrowing
         ---------------------------------------------

           Book ISBN/Product code: A123456789JQK
           Book ID: 0005

------------------------------------------------------------
Take Reserved Resource?[Y/N]
```

- ✧ The resource IDs of the reserved resources are entered one by one to the system.
- ✧ If reserved resource is off-shelf, the librarian fetches the reserved resources from the repository.

```
                User Information
   ---------------------------------------------
              User ID: s10169
              Class: 6G
              Class Number: 34
              Name: Yu Tai Man

>>Press Enter to proceed circulation.
```

```
                                                    Date: 20/12/2015
-------------------------------------------------------------------

                          Circulation Process
-------------------------------------------------------------------

 Enter a Resource ID: 0005
>>Reserved Resource! Please Recover the Resource immediately!

 Enter a Resource ID:
```

- ✧ The system checks if the resource is reserved by the user. i.e. Stud_ID field in Resource status resocrd = user ID entered.
- ✧ The system checks the number of late resources and the number of borrowed resources of the user.
- ✧ If the number of late resources = 0 and number of borrowed resources does not exceed the limitation of the library, the borrowing continues.
- ✧ Perform system process of borrowing described in phase 1.

Flow of renewing the borrowing:

```
                                                    Date: 20/12/2015
-------------------------------------------------------------------
                        Borrowed Resources
          ---------------------------------------------
   ID     Return Date      Type          Resource name
 0016      03/01/2016      Book      Sample #11
 0017      31/12/2015      Book      Sample #11
 0002      31/12/2015      Book   Sample title #11

                       Reserved Resources
          ---------------------------------------------
   ID   Cancel Before      Type          Resource name
 0005      26/12/2015      Book    Sample Bk #2
 0006      26/12/2015      Book    Sample Bk #2
 0013      26/12/2015      Book      Sample #11
 0018      26/12/2015      Book      Sample #11

 Enter a Resource ID: 0016
```

- ✧ The resource IDs of the borrowed resources are entered one by one to the system.
- ✧ The system checks the number of late resources and the number of renewals that have been made.

```
                                                    Date: 20/12/2015
-------------------------------------------------------------------

                        Returning & Renewing
            ---------------------------------------------

              Book ISBN/Product code: A123456789123
              Book ID: 0016

            ---------------------------------------------

                  1. Renew the borrowing.
                  2. Return the resource.

            Your choice: 1
-------------------------------------------------------------------
>>The borrowing has been borrowed 4 times
Renew the borrowing?[Y/N] y
```

✧ If the number of late resources = 0 and number of renewals does not exceed the limitation of the library, the borrowing continues.

✧ Perform system process of borrowing described in phase 1.

Step 4: Implementing returning process of library resource

Returning process involves checking of return date and, if the resource is returned late, fine is also calculated.

The returning process is also followed by the input of user ID to the system.

Flow of returning process:

```
                                                    Date: 20/12/2015
-------------------------------------------------------------------
                        Borrowed Resources
            ---------------------------------------------
   ID     Return Date      Type          Resource name
 0016      03/01/2016      Book      Sample #11
 0017      31/12/2015      Book      Sample #11
 0002      31/12/2015      Book   Sample title #11

                        Reserved Resources
            ---------------------------------------------
   ID   Cancel Before      Type          Resource name
 0005      26/12/2015      Book    Sample Bk #2
 0006      26/12/2015      Book    Sample Bk #2
 0013      26/12/2015      Book      Sample #11
 0018      26/12/2015      Book      Sample #11

 Enter a Resource ID: 0017
```

```
                    Returning & Renewing
        ---------------------------------------------

          Book ISBN/Product code: A123456789123
          Book ID: 0017

        ---------------------------------------------

              1. Renew the borrowing.
              2. Return the resource.

        Your choice: 2
 ------------------------------------------------------------
 Return Borrowed Resource?[Y/N] y
```

✧ The resource IDs of the borrowed resources are entered one by one to the system.

```
                                          Date: 20/12/2015
-------------------------------------------------------------
                    Borrowed Resources
        ---------------------------------------------
  ID    Return Date     Type        Resource name
 0016     03/01/2016    Book     Sample #11
 0002    31/11/2015*    Book  Sample title #11

                   Reserved Resources
        ---------------------------------------------
  ID  Cancel Before     Type        Resource name
 0005     26/12/2015    Book   Sample Bk #2
 0006     26/12/2015    Book   Sample Bk #2
 0013     26/12/2015    Book     Sample #11
 0018     26/12/2015    Book     Sample #11

 Enter a Resource ID: 0002
```

✧ An algorithm (implemented in Step 1) is used to check if the resource is late return and return the number of days the resource is late → i.

```
                    Returning & Renewing
          ------------------------------------------

            Book ISBN/Product code: 1234567891230
            Book ID: 0002

          ------------------------------------------

                2. Return the resource.

 ----------------------------------------------------------
 >>Late Return Resource!
 >>Late Penalty: 10.00
 Return Borrowed Resource?[Y/N]
```

- ✧ If the resource is late return, the amount of fine is also calculated. i.e. i*penalty increased per day.
- ✧ After collecting the fine, the return process is finished.

End of Phase 5

Phase 6: implementation of Online platform

After Phase 5, the circulation software is completed. The third unit test is conducted before the start of phase 6 and errors are fixed. The software can now process circulation of library resources correctly.

In phase 6, online platform of the library is implemented. Some algorithms implemented in phase 5 are reused.

Online platform allows user to look for their desired resources in the school intranet and reserve the resources. It also allows user to check their reserved resources and borrowed resources. Cancellation of reserve and renewal can also be made by users online.

Users have to log in to the online platform by using their User ID and passwords before using the online platform.

Step 1: implementing flow of searching and reserving

The purpose of searching is to search for resources that match the descriptions by users. The descriptions can be keywords in resource name, resource author, resource publisher and year of publish.

Another search method is by resource ISBN, the ISBN search cannot be done by keywords but has to be exact.

Resources titles with information that match all of the above fields are displayed as search results. Users enter and resource IDs of the resource they want to reserve.

Mechanism:

Similar to amending resource information, user are asked Searching of resources in the library is done by user answering a set of questions related to the resources they want to enquire. The answers are search keywords in different search fields. The search field can also be ignored by entering a '-'.

In User Menu:

```
                                        Date: 20/12/2015
-------------------------------------------------------------
  Hello! You have logged in as Library User: s10120.
-------------------------------------------------------------

                        User Menu
        ---------------------------------------------
           1. Online Resource Enquiry and Borrowing
           2. Access Your library Resource
           3. View personal information

 Your choice: 1
```

Flow of searching by descriptions:

```
                Search Library resource
-------------------------------------------

      1. Search by Resource code/ISBN
      2. Directed Search

   Your choice: 2
```

✧ User chooses the search method: search by descriptions.

```
          Search Resource: Resource Type
-------------------------------------------

      1. Book
      2. Movie
      3. Magazine
      4. Text Book
      5. Others

   Your choice: -
```

✧ User enters the resource type. ('-' denotes ignoring the field)

```
                  Searching Resource
      -------------------------------------------

  Resource name: Sam
  Resource Owner/Book Author: ple
  Resource Publisher: Sample
  Year of publish: -
```

✧ User enters the keywords in resource title

✧ User enters the keywords in resource owner/author

✧ User enters the keywords in Resource Publisher

✧ User enters the year publish.

✧ A search algorithm processes the search keywords.

```
                Search Results
 ---------------------------------------------

        ISBN                Resource Name
1. 1234567891230   Sample title #11
2. 123456789AAAA   Sample title #11

    Resource to enquire:
```

✧ A list of resource titles are displayed as search results

Flow of searching by Resource ISBN:

```
            Search Library resource
---------------------------------------------

      1. Search by Resource code/ISBN
      2. Directed Search

   Your choice: 1
```

✧ User chooses the search method: search by ISBN.

```
          Search By Resource code/ISBN
 ---------------------------------------------

 Enter A Resource code/ISBN: A123456789123
```

✧ User enters the exact ISBN code.

```
                Search Results
 ---------------------------------------------

        ISBN                Resource Name
1. A123456789123   Sample #11

    Resource to enquire:
```

✧ The only resource title with the ISBN code is displayed as search result.

Flow of reserving the resources:

```
                Search Results
 ---------------------------------------------

        ISBN                Resource Name
1. 1234567891230   Sample title #11
2. 123456789AAAA   Sample title #11

    Resource to enquire:
```

✧ Users choose the resource title in the result page.

```
             Resource Information
 ---------------------------------------------

   Resourec type: Book
   Resource title: Sample title #11
   Resource owner/ Book owner: Sample Author
   Publisher: Sample Publisher
   Year of publish: 2015


>>Press Enter to continue
```

```
                Searching Resource
     ---------------------------------------------

 Resource name: Sam
 Resource Owner/Book Author: ple
 Resource Publisher: Sample
 Year of publish: -
```

✧ All of the information related to the chosen resource title is displayed.

✧ User checks the resource information and proceed to reserving page.

```
                Resource reserving
      ---------------------------------------------

Resource ID: 0003  On shelf

         Resource ID:
```

✧ A list of resources with the same title are displayed.

```
                                                    Date: 20/12/2015
-------------------------------------------------------------------

                        Resource reserving
            ------------------------------------------

   Resource ID: 0003  On shelf

              Resource ID: 0003
```

```
-------------------------------------------------------------------
Confirm Reservation?[Y,N] y
>>Resource has been reserved for you!
>>Please take your borrowed resource within 6 day(s)
```

- ✧ User checks the status of the resources and enter the resource IDs to reserve the library resources.
- ✧ System described in Phase 1 is performed to update the record.

Step 2: Implementing the search algorithm of resource searching

Processing the description search, an algorithm is needed to perform the search with keywords in different fields.

Matching Resource information:

Search keywords:

Resource type	Resource title	Author	Publisher	Year of Publish
'-': ignored	Title A	B	Sher A	1999

Given: 4 resource titles

Resource type	Resource title	Author	Publisher	Year of Publish
Book	Book Title a	or BAuth	sher aPubli	1999
Movie	Title B	Author A	Publisher B	1999
Movie	B title	Author A	sher a SHER A	2000
Magazine	Title A	Baby Bit Bb	Publisher B	2000

Search Results: Title A.

- ✧ The highlighted part is the part that matched the search keywords.
- ✧ The search is not case-sensitive.
- ✧ The resources that have highlighted parts in all of their fields are displayed as search results. i.e. Title A.

Algorithm of searching:

- ✧ A For-do loop is used to check all the resource titles in the library.
  For i := 1 to ISBN_total do
- ✧ An array of Boolean is used to determine if the resources in library match the search keywords in each field.
- ✧ Resource Information matching of each search field is performed stepwise:

Initialize:

i	1	2	3	4	6	7	8	9	10
Searchlist[i]	true	true	true	true	true	true	true	true	true

Match resource type:

i	1	2	3*	4	6*	7	8*	9*	10
Searchlist[i]	true	true	false	true	false	true	false	false	true

Match resource name:

i	1	2*	3	4	6	7	8	9	10
Searchlist[i]	true	false	false	true	false	true	false	false	true

Match resource author/owner:

i	1*	2	3	4	6	7	8	9	10
Searchlist[i]	false	false	false	true	false	true	false	false	true

Match resource publisher (ignored):

i	1	2	3	4	6	7	8	9	10
Searchlist[i]	false	false	false	true	false	true	false	false	true

Match resource year of publish:

i	1	2	3	4	6	7*	8	9	10
Searchlist[i]	false	false	false	true	false	false	false	false	true

- ✧ Result: The 4th resource information record and the 10th resource information record.
- ✧ And gate is used in each step, if none of the keyword is matched in one field, the record is eliminated. i.e. Searchlist[i] := <Field is Matched> And Searchlist[i].
- ✧ Another array: Arranged_list is used to store the successful search and display to the user's screen.
- ✧ Resource IDs are also searched in status file to display available resource of the title. i.e. Resource IDs with information record 4 and 10 are searched.

Step 3: Implementing Reserved and Borrowed Resource information

Users are allowed to check their borrowed and reserved resources at this page, follow up actions: renewal and cancelling the reserve can also be performed.

A list of borrowed resources and a list of reserved resources are displayed on the same page. Resource IDs are entered to update to perform follow up actions.

Flow of displaying and updating reserved and borrowed Resources:

- ✧ The User ID of the user is searched in Resource Status file. i.e. Borrowed resources and reserved resources, Stud_ID field is searched.
- ✧ A list of user's borrowed resources and reserved resources can be found.
- ✧ User enters the resource IDs to perform follow up actions.
- ✧ Enter IDs of reserved resource → cancel reservation.
- ✧ Enter IDs of borrowed resource → renewal.
- ✧ System Process of borrowing and reserving is performed. (Described in Phase 1)

End of Phase 6

Phase 7: Implementation of User Authentication

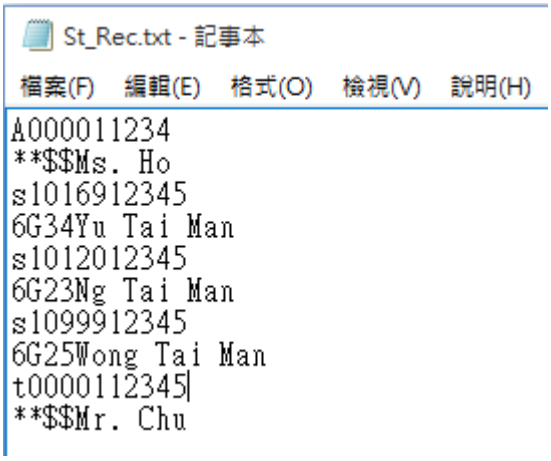
With the previous 6 phases of the implementations, the important features of the library system: Online platform and Circulation software are finished. After the forth unit test, these features can work correctly.

There are two types of end users: Users of the library service and librarians. An authentication process is required to distinguish the two types of end users. Apart from that, users of the library services have their accounts to log into the Online Platform, hence, Authentication process is also required for login actions.

Step 1: Implementing User Record Data Store

Data Store	Variable name	Field length and explanation
User ID	Stud_ID	String[6]
Class and Class Number	ClassNum	String[4]
Password	date	String[20]

User record file: Stud_Rec.txt:



A User record occupies 2 lines:

Line 1: Stud_ID, password

Line 2: Class ands Number, User Name

This minimizes the number of lines in the text file.

User types: Student Users, Teacher Users and Librarians

- For Student user, first digit of user ID is denoted by ‘s’. e.g. s10010
- Student records use the all four record fields:
- For teacher users, the first digit of the User ID is denoted by ‘t’. e.g. t10010
- Class and Class number field is not used. **$$ is stored.
- For Administrator, first digit of user ID is denoted by ‘A’. e.g. A00001.
- Class and Class number field is also not used. **$$ is stored.

The Data Store of user record is then put into record structure. The Record name is called StudRecType, which is declared as follwing:

```
StudRecType    = record
                    Stud_ID: string[6];
                    ClasNum: string[4];
                    name: string[50];
                    pw :string[20];
                end;
```

Mechanism:

A static user record file is implemented. User records are stored in this data store. The primary key in this Data Store is Stud_ID. Each user record consists of Class, Class Number of the user, name of the user and the password to log into the system.

User IDs are searched in this this file. Authentication process fetches user IDs and password to verify the user’s identity.

Step 2: implementing User record management process.

The process is similar to Resource information management. Two main management procedures are addition of User Records and amendments of user records.

Mechanism:

A series of questions are used to obtain all the information required to add a new user record or amendments of the user information. Librarians input all the information of the users by answering the questions.

Flow of management processes:

- Flow of adding user records and amending user information are identical to adding resource information and amending the resource.

Step 3: Implementing user menus: Librarians and Library Users

Librarians and Library users require different services from the library system. Hence, two user interfaces are needed to distinguish their needs.

Librarians have access to all of the functionalities of the library software, except they cannot reserve or borrow library resources.

Functionalities required by librarians:

- ✧ Circulation software
- ✧ Manage library resource status
- ✧ Manage library resource information
- ✧ Online platform(Only Searching function)
- ✧ Manage User Records
- ✧ Manage Library settings
- ✧ Change Account Password

Functionalities required by users (both teacher and student):

- ✧ Online Platform
- ✧ Change Account Password

Mechanism:

Two User Menus are implemented. The Menus list all the functionalities that can be accessed by the account and users enter the choice to use these functionalities. By User Authentication, the two types of end users: library users and librarians, are directed to two different user menus, hence, they can have access to different functionalities of the library software.

Step 4: Implementing Authentication process:

End of Phase 7

Phase 8: Implementation of Library Setting

The library sets limitations on users, they include:

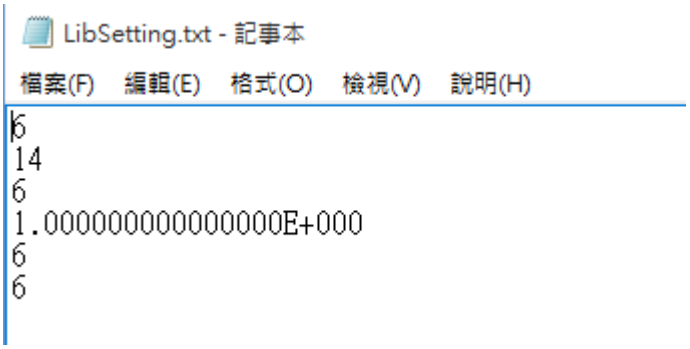
1. number of days of borrowing last,
2. number of days of reservation lasts,
3. number of resources that can be borrowed by a user.
4. number of resource that can be reserved by a user,
5. Amount of fine increases per day per late resources
6. number of continuous renewals can be made

In previous phases, these limitations are used. E.g. Circulation requires 1, 3, 5, 6. Reservation and renewal on Online Platform requires 1, 2, 4, 6. However, in previous phases, these limitations are constant and cannot be changed. In the last phase, a process is made to change these limitation variables.

Step 1: implementing Library Setting Data Store

Data Store	Variable name	Field length and explanation
1	Borrow_days	Integer
2	Request_day	Integer
3	Max_book	Integer
4	Max_request	Integer
5	Penal_per_day	real
6	Max_conti	Integer

A text file is used to store these settings: LibSetting.txt



The setting file consists of 6 lines.

Line 1: Max_book

Line 2: borrow_days

Line 3: request_day

Line 4: penal_per_day

Line 5: max_conti

Line 6: max_request

Mechanism:

Another data loading process is implemented in Loading and saving procedure. (Implemented in phase 2). The settings of library is loaded to the software as global variables when the library software is run. Other processes can use these variables when processing specific tasks.

Step 2: implementing change of library settings

Librarians can change the setting of the library. The change of settings only affects the library processes made after the change. i.e. Return dates of borrowed resources are not changed even if there are changes in the borrow_days variable.

For reference purpose, the current library settings are also displayed when librarians are making the changes.

Mechanism:

A series of questions are used to obtain all the information required to change the library settings. Librarians input the new settings by answering the questions.

At Librarian Menu:

```
-------------------------------------------------------------
  Hello! You have logged in as Administrator: A00001.
-------------------------------------------------------------

                    Librarian Menu
        ---------------------------------------------
          1. Borrow and Return Circulation
          2. Update and Display Resource Status
          3. Update resource Data
          4. Search resources
          5. Manage Student records
          6. Change Library setting
          7. Update Personal Information

 Your choice: 6
```

Flow of changing of library settings:

```
                                                    Date: 20/12/2015
 ------------------------------------------------------------

                    Current Library Settings
          ---------------------------------------------
Maximum number of Resources can be borrowed by a user: 8
Number of Days each borrow lasts: 14
Number of Days before the online reserve is cancelled: 6
Amount of Fine increased each day: $0.50
Maximum number of renewal: 5
Maximum number of online reserves made by each student: 6

 Change Settings?[Y/N]
```

✧ System displays the current settings of the library.

```
                                                    Date: 20/12/2015
 ------------------------------------------------------------

                    Current Library Settings
          ---------------------------------------------
Maximum number of Resources can be borrowed by a user: 8
Number of Days each borrow lasts: 14
Number of Days before the online reserve is cancelled: 6
Amount of Fine increased each day: $0.50
Maximum number of renewal: 5
Maximum number of online reserves made by each student: 6


                    New Library Settings
          ---------------------------------------------
Maximum number of Resources can be borrowed by a user: 8
Number of Days each borrow lasts: -
Number of Days before the online reserve is cancelled: -
Amount of Fine increased each day(in $): 0.5
Maximum number of renewal: 5
Maximum number of online reserves made by each student: 4
Confirm changes?[Y/N]
```

✧ Librarian enters the new setting of the library one by one.

✧ Librarian enters '-' to ignore a change

✧ Librarian confirms the change

```
                Current Library Settings
        ----------------------------------------
Maximum number of Resources can be borrowed by a user: 8
Number of Days each borrow lasts: 14
Number of Days before the online reserve is cancelled: 6
Amount of Fine increased each day: $0.50
Maximum number of renewal: 5
Maximum number of online reserves made by each student: 4

Change Settings?[Y/N]
```

- System saves the change to the setting file. (This Process is identical to saving process in implemented in phase 2)

End of Phase 8

3.2 System conversion

After the 8 phases of implementation, system test is conducted. Several bugs are spotted and they are fixed after the test. At this stage, the library system is ready to be in use. User acceptance test is also conducted and the new system is approved to take over the old system.

However, resource information and user information is yet to be inputted into the new system. Therefore, system conversion is conducted before the new library system can be in full use.

Old system:

- ✧ Library resource information is put in a written record book.
- ✧ Library status record is put in a record spreadsheet.
- ✧ No Online platform and reservation is not allowed.

Conversion strategy: Direct Cutover conversion

- ✧ The is limitation on the resource set by the school, parallel conversion is not feasible
- ✧ School library is the only organization would use the software, piloted conversion is also not feasible as there is no pilot organization.
- ✧ The conversion would be done in long holiday, there is plenty of time to do the conversion process. Hence, phased conversion is not required.
- ✧ The Online platform is a new feature to the library system, it can be added to the library system at any time. Hence, conversion only involves library resource data handling and circulation. Direct cutover does not cause large workload.

Conversion method:

- ✧ The old resource information records are written on a record book. Despite being inefficient, resource information has to be inputted one by one to the new system. The new Resource information management software is used to add new records.
- ✧ Once all the resource data information has been added to the library, the system has also created one resource status record for each resource title record added to the library. For resource titles having two or more resources in the library, the number of resources of these titles have to be changed.
- ✧ The status of the borrowed resources (borrowed during holiday) will have to be changed manually.
- ✧ Barcode labels are stuck to every resources which are currently in the library. Borrowed resources will have barcode labels stuck after they are returned.

Conversion Process:

- ✧ Data entry clerks are hired to enter the resource information.
- ✧ Librarians then verify the inputted information and correct the number of resources per titles.
- ✧ Librarians check the borrowed resources and update the status of these resources.
- ✧ With the correct information records and status records, barcode labels are printed with the Resource IDs assigned to each resource.

Training of librarians and promotion of new online platform:

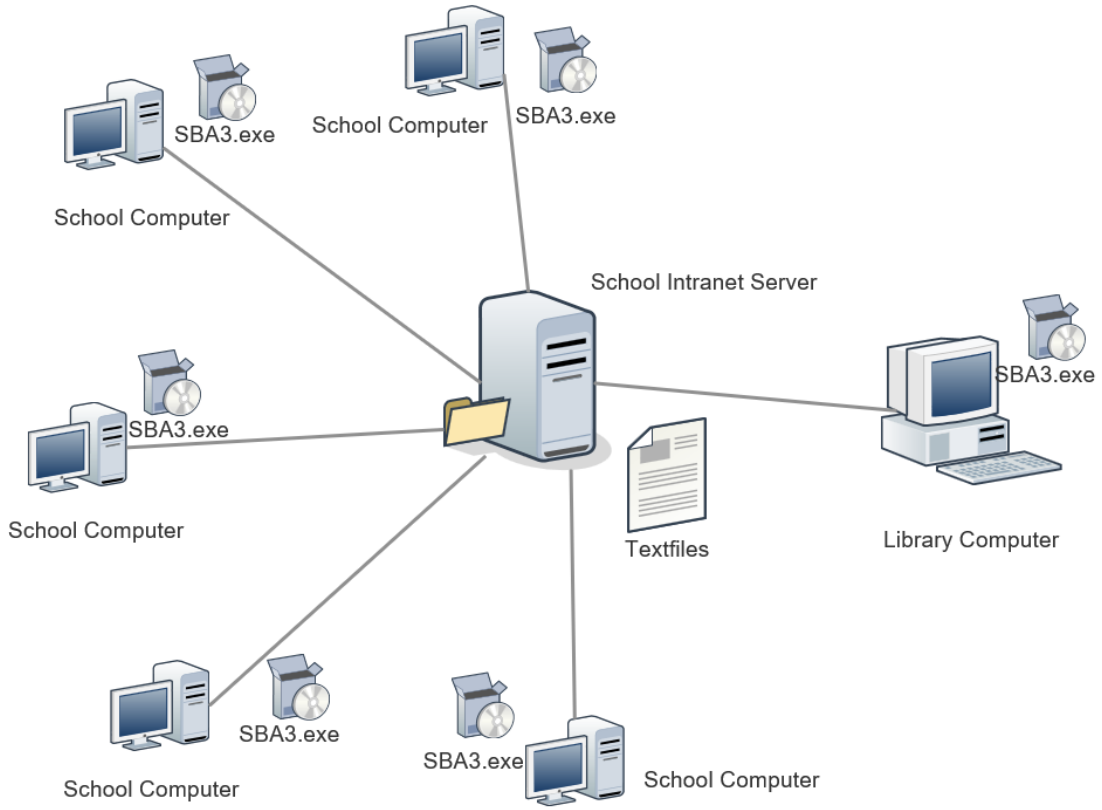
- ✧ Librarians are trained to use the library system software to process circulation.
- ✧ User menus are written for librarians to use the new library system software.
- ✧ Before the Online Platform is launched, announcement will be made and a short user guide is distributed to students and teachers.

3.3 System Installation

The library system is now ready to process circulation and online enquiries. The last stage in system implementation process is to install the new software to the computers in the school.

To use the library system software, a user or librarian have to execute the execution file (SBA3.exe) with the 4 text files under the same directory. But before implementing the system installation, some modifications are made on the software. The directory of the 4 files are set under a hidden directory in the school public drive.

Structure of School library system:



- ✧ The Library system software is installed to every school computer connected to the school intranet server and Library computer.
- ✧ The 4 Record files: Bk_data.txt, Bk_rec.txt, LibSetting.txt and Stud_rec.txt are put under the same directory in the school intranet server.
- ✧ The directory storing the 4 record files is hidden and only accessible by the library system software. (SBA3.exe)
- ✧ By using a user account, it is only possible to reserve and enquire library resources.
- ✧ By using an administrator account, resource information status and resource information can be changed.

4. Testing and evaluation

Unit testing is done during the implementation of the library system program. Starting from phase 3, after the implementation of each phase, a unit test is conducted to ensure no logical and syntax error appears in that phase of implementation.

After the library system program was implemented, system test should be conducted to check if the library system can handle the circulation and online reservation service.

Lastly, User Acceptance test should be conducted. The purpose is to find out if the new library system meets the user requirements.

After these tests have been passed, the new library system was approved to take over the old system and system conversion begins.

4.1 Testing Plans for Unit Tests

The unit test is conducted by the programmer, the purpose is to ensure no errors in individual module.

Syntax errors are spotted by the compiler and are corrected immediately.

Logical errors are detected by executing the module and inputting different test cases. After being spotted, programmer will find a solution to the problem.

Test Plan 1: Resource Management module

Functions	The tests determine
Resource Information Management	✧ Distinguish existent resource ✧ Correct addition of resource title ✧ Correct edition of resource information ✧ Correct edition of number of resource title
Resource Status Management	✧ Correct display of current status ✧ Correct changing of status ✧ Correct switching of pages ✧ Correct respond to commands i.e. '*.*'

Resource Information Management:

Testing correct distinguish existent resource:

At this page, the followings are tested.

Input	Nature	Expected output	Test Output	Follow-up actions
Existing ISBN	Normal input	Proceed to change resource information page	As expected	nil
Non-existing ISBN	Normal input	Proceed to add resource record page	As expected	Nil
ISBN with length not equal to 13 i.e. length > 13 or length < 13	Invalid input	Remind about the invalid input, require input again	When length < 13, the output is correct, but when the length > 13, the check is passed	Set length of the input string to 14 instead of 13

Correct addition of resource title:

```
                Adding New Movie
    ------------------------------------------

Resource name/Book title: Testing title
Resource owner/Book author: Testing Author
Publisher/Producer name: test
Year of publish: 1234
Confirm The above information?[Y/N] y
```

After confirming the addition, the following is tested

Input	Nature	Expected output	Test Output	Follow-up actions
Follow the flow of adding new title and confirm the addition	Normal Input	The new record is added	As expected	nil

Correct change on number of resources:

For adding number of resources:

```
                1. Add Resource titles
                2. Delete Resources titles

           Your choice: 1
-------------------------------------------
>>Number of the Resources in library: 2
  Number of Resources to add?
```

At this screen, the followings are entered:

Input	Nature	Expected output	Test Output	Follow-up actions
Positive Integer: 1 - 100	Normal input	Notify the addition is done successfully. The number of records are added	As expected	nil
Positive Integer: 101 - 9999 Note that: Max record = 9999	Extreme cases	System do not allow the addition Librarian asked to input again	As expected	nil
Non-integer: ‘a’, ‘aa’, ‘hello’. Negative integer, Zero	Invalid input	System do not allow the addition Librarian asked to input again	As expected	nil

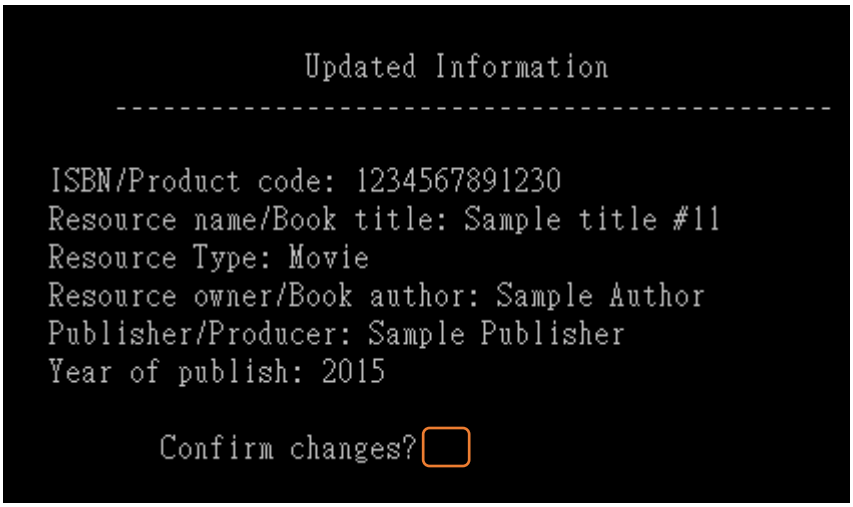
For deleting resources:

```
                Removing Resources
      -------------------------------------------

Resource ID: 0002  On shelf
Resource ID: 0012  On shelf

Enter a Resource ID to remove:
```

At this screen, the followings are entered:

Input	Nature	Expected output	Test Output	Follow-up actions
The resource ID shown on the screen	Normal input	The resource ID is removed from the screen, library record file is updated. For borrowed resources, the Resource ID is not removed.	As expected	nil
Resource ID not on the screen	Invalid input	The system requires the user to enter the resource ID again	As expected	nil

Correct edition of resource information:



After confirming the change, the following is tested

Input	Nature	Expected output	Test Output	Follow-up actions
Follow the flow of changing information and confirm the addition	Normal Input	The new record is added	As expected	nil

Resource Status Management:

Correct display of current status:

```
          Display Resource Status
-------------------------------------------
    1. Display Borrowed resource list
    2. Display Late Return resource list

       Update & Display Resource Status
-------------------------------------------
    3. Manage On shelf Resource
    4. Manage Off shelf Resource
    5. Manage reserved Resource(On Shelf)
    6. Manage reserved Resource(Off Shelf)

Your Choice:
```

Input	Nature	Expected output	Test Output	Follow-up actions
Numbers on the screen: 1 - 6	Normal Input	A list of resources sorted by status	As expected	nil
Number not on the screen: -1, -2, 1000	Invalid Input	Display Invalid Input	As expected	nil

Correct changing of status

```
            Update & Display Resource Status
      -------------------------------------------
          Resource ID: 0001 On shelf
          Resource ID: 0004 On shelf
          Resource ID: 0006 On shelf
          Resource ID: 0007 On shelf
          Resource ID: 0008 On shelf
          Resource ID: 0009 On shelf
          Resource ID: 0010 On shelf
          Resource ID: 0011 On shelf

 Resource to be off-shelf:
```

Input	Nature	Expected output	Test Output	Follow-up actions
ID on the screen	Normal Input	ID removed from the screen, textfile updated	As expected	nil
ID not on the screen: 0012, 0003	Invalid Input	Another input dialog	As expected	nil

Correct switching of pages:

```
            Update & Display Resource Status
      ----------------------------------------
          Resource ID: 0013 On shelf
          Resource ID: 0014 On shelf
          Resource ID: 0015 On shelf
          Resource ID: 0017 On shelf
          Resource ID: 0018 On shelf
          Resource ID: 0020 On shelf
          Resource ID: 0002 On shelf
          Resource ID: 0012 On shelf

 Resource to be off-shelf:
```

Input	Nature	Expected output	Test Output	Follow-up actions
Hit Enter without inputting anything	Normal Input	Switched to next page	As Expected	nil

Correct Use of commands:

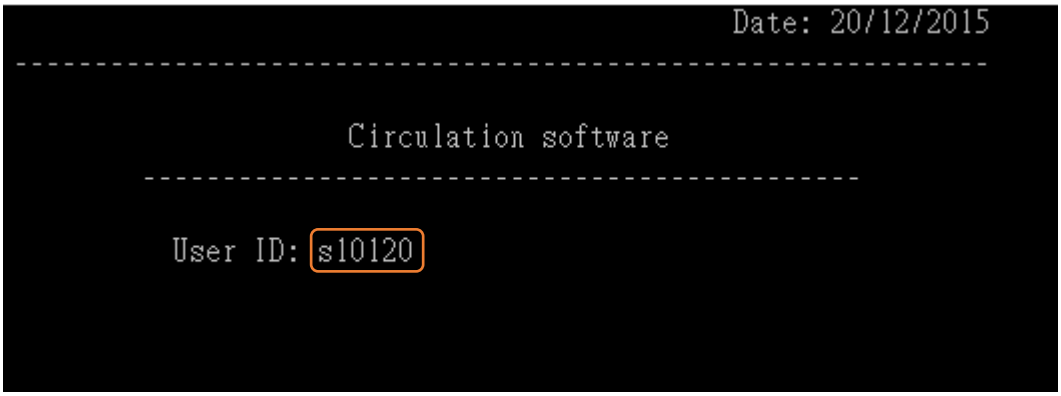
```
                                          Date: 20/12/2015
-------------------------------------------------------------

            Update & Display Resource Status
      ----------------------------------------
          Resource ID: 0013 On shelf
          Resource ID: 0014 On shelf
          Resource ID: 0015 On shelf
          Resource ID: 0017 On shelf
          Resource ID: 0018 On shelf
          Resource ID: 0020 On shelf
          Resource ID: 0002 On shelf
          Resource ID: 0012 On shelf

  Resource to be off-shelf: *.*
```

Input	Nature	Expected output	Test Output	Follow-up actions
'*.*'	Normal Input	'*.*' will update the status of IDs on the screen	As Expected	nil

Test Plan 2: Circulation module

Functions	The tests determine
Input of User ID and Resource ID	✧ Correct display of user's borrowed resources and reserved resources ✧ Correct identification of circulation flow
Circulation Process: Borrowing, returning and renewing	✧ Correct Circulation

Correct display of user's borrowed resources:



At this screen the followings are entered

Input	Nature	Expected output	Test Output	Follow-up actions
Correct student ID	Normal input	A list of borrowed and reserved resources of the user	As expected	nil
Incorrect Student ID: Admin, k10120	Invalid Input	Warning message and another input dialog	As expected	nil
Librarian ID: A00001	Extreme case	Warning message and another input dialog	No warning message	Warning message is added.

Correct identification of circulation flow:

```
                                                Date: 20/12/2015
-------------------------------------------------------------
                    Borrowed Resources
        ---------------------------------------------
  ID    Return Date     Type         Resource name
0016     03/01/2016     Book     Sample #11

                   Reserved Resources
        ---------------------------------------------
  ID  Cancel Before     Type         Resource name
0005     26/12/2015     Book   Sample Bk #2
0003     26/12/2015     Book  Sample title #11

Enter a Resource ID:
```

At this page the followings are entered:

Input	Nature	Expected output	Test Output	Follow-up actions
Borrowed resource ID: 0016	Normal Input	Direct to return and renew page	As expected	nil
Reserved resource ID: 0005	Normal Input	Direct to Reserved borrowing page	As expected	nil
On-shelf resource IDs	Normal Input	Direct to On-shelf borrowing page	As expected	nil
Off-shelf resource IDs	Boundary Input	Warning message and another input dialog	The screen displays nothing	Warning message and new input dialog added
Incorrect ID: F123, K006	Invalid input	Warning message And another Input dialog	As expected	nil

Correct Circulation: Borrowing

```
                                                    Date: 20/12/2015
 ------------------------------------------------------------

                  On-shelf Resource Borrowing
         ---------------------------------------------

           Book ISBN/Product code: A123456789123
           Book ID: 0018

 ------------------------------------------------------------
 Borrow the above Resource?[Y/N]
```

At this page

Input	Nature	Expected output	Test Output	Follow-up actions
‘Y’, ‘N’ are entered	Normal Input	Show borrowing succeed. Record file updated	As expected	Nil
1233, 21, gg are entered	Invalid Input	Warning message Another Input dialog	As expected	Nil
‘n’, ‘y’ are entered	Boundary case	Show borrowing succeed. Record file updated	System treated as error input	‘n’, ‘y’ are added to the correct set. IN[‘n’, ‘y’, ‘Y’, ‘N’, ‘-’]

Correct Circulation: Returning

```
------------------------------------------------------------

                  Returning & Renewing
        ---------------------------------------------

          Book ISBN/Product code: A123456789123
          Book ID: 0016

        ---------------------------------------------

              2. Return the resource.

------------------------------------------------------------
Return Borrowed Resource?[Y/N]
```

At this page, the following are entered.

Input	Nature	Expected output	Test Output	Follow-up actions
'Y', 'N' are entered	Normal Input	Show returning succeed. Record file updated	As expected	Nil
1233, 21, gg are entered	Invalid Input	Warning message Another Input dialog	As expected	Nil
'n', 'y' are entered	Boundary case	Show returning succeed. Record file updated	As expected	Nil

Correct Circulation: Reserved Resource borrowing

```
                                                    Date: 20/12/2015
-------------------------------------------------------------------

                    Reserved Resource Borrowing
            ---------------------------------------------

              Book ISBN/Product code: A123456789JQK
              Book ID: 0005

-------------------------------------------------------------------
Take Reserved Resource?[Y/N]
```

At this page, the followings are entered

Input	Nature	Expected output	Test Output	Follow-up actions
'Y', 'N' are entered	Normal Input	Show resource taken Record file updated	As expected	Nil
1233, 21, gg are entered	Invalid Input	Warning message Another Input dialog	As expected	Nil
'n', 'y' are entered	Boundary case	Show resource taken Record file updated	As expected	Nil

Correct Circulation: Reserved Resource borrowing

```
                                                    Date: 20/12/2015
-------------------------------------------------------------------

                      Returning & Renewing
          ------------------------------------------------

            Book ISBN/Product code: 1234567891230
            Book ID: 0012

          ------------------------------------------------

                1. Renew the borrowing.
                2. Return the resource.

          Your choice: 1
-------------------------------------------------------------------
>>The borrowing has been borrowed 1 times
Renew the borrowing?[Y/N]
```

At this page, the following are entered:

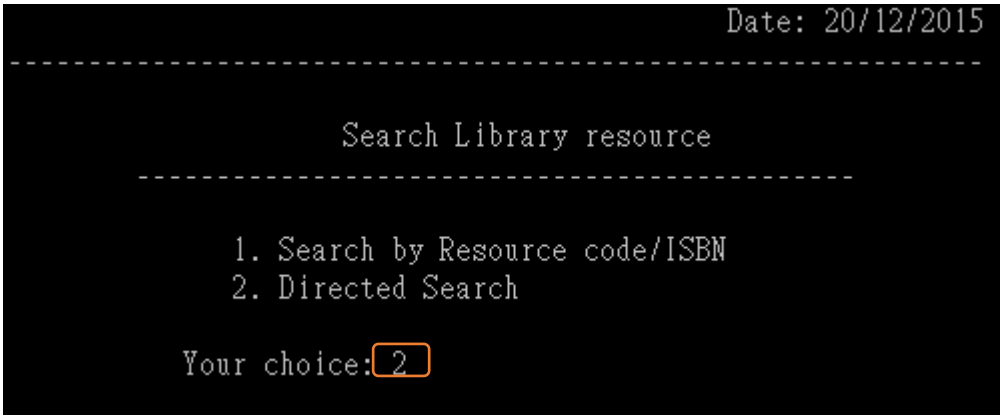
Input	Nature	Expected output	Test Output	Follow-up actions
'Y', 'N' are entered	Normal Input	Show number of times of renewal Record file updated	As expected	Nil
1233, 21, gg are entered	Invalid Input	Warning message Another Input dialog	As expected	Nil
'n', 'y' are entered	Boundary case	Show number of times of renewal Record file updated	As expected	Nil

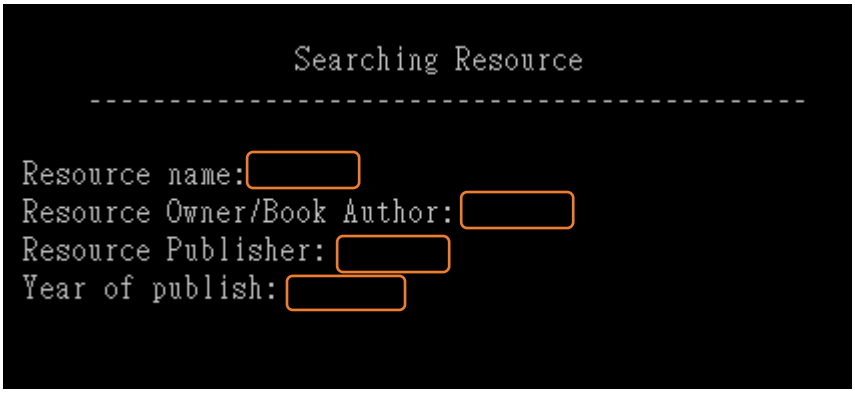
Test Plan 2: Online Platform module

Functions	The tests determine
Searching of resources	✧ Correct searching which matches all the key words in different field ✧ Correct omitting the unsearched field
Resource Reserving	✧ Correct showing the available resource in the library ✧ Correct reserving of resources
Showing Reserved and borrowed Library Resource	✧ Correct showing of the resources ✧ Correct follow-up actions by users: cancel reserving, continue borrowing

Searching of resources:

Correct searching:





At the above page, the following are entered:

Input	Nature	Expected output	Test Output	Follow-up actions
Resource Type: '-' Resource Name: Sam Author: ple Publisher: '-' Year: 2015	Normal Input	Resource IDs matching all the search fields	As expected	Nil
Resource Type: Resource Name: Author: Publisher: Year:	Boundary case	Display all records	No records can be displayed	Add empty input as field to ignore.
Resource Type: 3 Resource Name: 123 Author: Samp1e Publisher: 6G club Year: ABCD	Invalid Input	Warning message and abort the searching	Searching continues and no record is displayed	Add warning message. Change algorithm to abort this type of searching
Resource Type: '-' Resource Name: '-' Author: '-' Publisher: '-' Year: '-'	Normal Input	Display all records	As expected	Nil

Resource Reserving:

Correct showing the available resource in the library:

```
                                                    Date: 20/12/2015
-------------------------------------------------------------------

                        Search Results
          ---------------------------------------------

                ISBN                  Resource Name
        1. 1234567891230   Sample title #11
        2. A123456789JQK   Sample Bk #2
        3. 1234567891231   Sample Bk #3
        4. 1234556789123   Sample Bk #4
        5. A123456789123   Sample #11
        6. 123456789AAAA   Sample title #11

            Resource to enquire:
```

At this page, the followings are entered:

Input	Nature	Expected output	Test Output	Follow-up actions
The choices on the screen. i.e. 1 - 6	Normal Input	Direct to reserving page. Showing available resources	As expected	nil
The choices not on the screen: 7, -1, 9	Invalid input	Warning message, Another input dialog	As expected	nil

Correct reserving of resources:

```
                                                    Date: 20/12/2015
-------------------------------------------------------------------

                        Resource reserving
          ---------------------------------------------

    Resource ID: 0005  Reserved for s10120 until 26/12/2015
    Resource ID: 0006  On shelf
    Resource ID: 0007  On shelf
    Resource ID: 0008  On shelf

            Resource ID:
```

At the above page, the following are entered:

Input	Nature	Expected output	Test Output	Follow-up actions
Resource IDs on the screen	Normal Input	For on-shelf, successful reserve For others, warning message and another input box	As expected	nil
Resource ids not on the screen	Invalid input	warning message and another input box	As expected	nil

Showing Reserved and borrowed Library Resource:

Correct showing of the resources:

```
                                        Date: 20/12/2015
-------------------------------------------------------------
  Hello! You have logged in as Library User: s10120.
-------------------------------------------------------------

                        User Menu
        ------------------------------------------
          1. Online Resource Enquiry and Borrowing
          2. Access Your library Resource
          3. View personal information

 Your choice:
```

Input	Nature	Expected output	Test Output	Follow-up actions
Input '2' to access the resources	Normal Input	Showing the reserved and borrowed resources	As expected	nil

Correct showing of the resources:

```
                                        Date: 20/12/2015
-------------------------------------------------------------

                    Borrowed Resources
        ------------------------------------------
   ID    Return Date     Type         Resource name
 0016     03/01/2016     Book     Sample #11
 0012     03/01/2016     Book  Sample title #11

                    Reserved Resources
        ------------------------------------------
   ID  Cancel Before     Type         Resource name
 0005     26/12/2015     Book   Sample Bk #2
 0003     26/12/2015     Book  Sample title #11


 Enter Resource ID From Above:
```

Input	Nature	Expected output	Test Output	Follow-up actions
Enter the resource IDs on the screen	Normal Input	For Reserved resources, the reserve will be cancelled. For Borrowed resource, the return date will be updated	As expected	Nil
Enter resource IDs not on the screen	Boundary case	Warning message, another input dialog appears	As expected	nil

Other Modules implemented in this system has been tested individually. But due to the relative insignificance, they are omitted in this unit test section.

End of unit test section.

4.2 Possible System tests and user acceptance tests

Although it is not possible to conduct these tests in this project, some these tests should be conducted to ensure user requirements have been fulfilled.

Possible System tests	Foreseeable outcome and solutions
Volume test: 10 computers simultaneously open the program and access to the same file: BK_rec.txt.	**Outcome**: The textfile will contain incorrect status as the many updating done in a short time. **Solutions**: Divide the single status file into 10 status files, each hold small sections of resource records. This will avoid heavy traffic to one textfile.
Storage test: 9999 resource status records are stored in the resource data file: Bk_rec.txt	**Outcome**: A txt file has size limit of 1 GB. Also, array of records also has size limit. For 9999 resource records, some records cannot be loaded to the system at the start of the program. **Solutions**: Nil. It is not solvable if the platform of implementation is pascal, which is not designed for large amount of data storing. To truly overcome the problem, DBMS should be employed.
Performance time test: Large amount of library resource status records are stored. e.g. >5000 records	**Outcome:** The system will suffer from heavy delay if there is too many records in the record files. This is because all records are loaded to the software and the loading load will increase dramatically. **Solutions:** Implement an algorithm to selectively load the require record. This should improve the loading time.

Possible User Acceptance tests	Foreseeable outcome and solutions
Librarians and student users are invited to use the software and provide feedbacks during the holiday.	**Outcome:** The user menu may not be user-friendly. There may not be enough instructions for user to understand how to use the system. **Solution:** Add instructions for every functions to instruct library users. Write a short user menu for librarians to manage library resources.

4.3 Self-Evaluation

The project development is finally finished. The system can be put to use at any time, and as the project manager, I will evaluate the system according to different criteria.

Criteria	Evaluations	Improvements
User-friendliness of the library system software	**For users**, the instructions on the system is clear and obvious. There is also a hint page for user at the login page, which further enhances the user-friendliness. **For librarians**, there is few instructions. Librarians have to read the user menu in order to use some commands.	Too much instructions will decrease the readability. The way to add instructions can be randomly display 1 - 2 instructions at each screen, same as showing hints at the login page.
Effectiveness of data storing and data processing	The **data handling** is effective. By using array of records, it has similar effect as using tables in DBMS. Records can be located by using a key field. Yet, the records are not scalable in user's aspect. Fields cannot be added or deleted at user's level. The **data storing** is not as effective as DBMS.	The fields in array of records can be ordinal values. e.g. Instead of using Book_name and Author as fields, A, B, C, D should be used as fields. The actual field name can use a parallel array to store. e.g. Field_name: array['A'..'Z'] of string.
Efficiency of data storing and data processing	The processing data is high. The program will preload all the records into the array of records. The **processing** of array of records will cause no delay. Yet, since the loading process will preload all records. The loading time will be long. The **saving process** will also consume a lot of time.	The records can be sorted according to some fields. This way, the system need not to preload all the records and can locate the record by algorithm. Searching can also be done more quickly.
Convenience of using the system	For **librarians**, the system has some commands to process large volume of resource data. It is convenient to manage the resource status and information. For **library users**, the system can reserve resources easily. However, a book can only hold 1 reservation. It is inconvenient for user to reserve for popular resources.	Data structure can be altered. Instead of keeping the reserve field at status records, the reserve can be kept in resource data records. Multiple slots of reservations can be provided for each title.

5. Discussions and Conclusions

5.1 Comments on the software development process

The software development is divided into 8 phases, in which sub-programs are implemented in order. The sub-programs are tested immediately and assembled into the main program after passing the tests.

The advantage of this is that the testing in later phases will be easier as bugs are not accumulated. In addition, the assembling of different modules is done at the end of the implementation. The effort of assembling different sub-programs into a large one is reduced.

However, it does require careful planning. Some core features have to be implemented. For instance, the data processing of the library system, which is a core feature, is first implemented. The builds up the framework of the system first and the feasibility of the solution is also assessed at the beginning. If other minor features are implemented at the beginning, the system may not be a feasible solution as core feature may be difficult to implement.

The implementation process involves me only in this project. Yet, in real situation, coordination between different departments and different programming teams is important. Thus, programming codes should be understood by programmers easily. But in this project, many codes are written due to instinct. Some codes cannot be comprehended even by me.

5.2 Improvements

Improvements can be made on the validation process. There are although warning messages to tell the users about the error, they are not clear because they do not tell the user how to correct the errors.

There are also rooms of improvements for the user interface. Though Pascal does not have GUI, it is still possible to make the user menu more attractive by using textcolor function and appropriate design of the interface. Keypress functions can also be utilized. Instead of inputting a dash to return to the previous page, escape can be pressed. This improvements can be made on the user interface

However, there are also some loopholes that cannot be fixed after the system design stage. For instance, the inefficient data storage and handling of the library system software and the use of ordinal fields cannot be fixed unless a new development cycle is started.

5.3 Future developments

This project follows a pure waterfall model for development. The implementation process follows the system design completely without change. For instance, the idea of sing ordinal fields in array came during the implementation process. Nevertheless, I gave up this idea even it would be better.

In the next development cycle, a sashimi model can be employed. This would be especially useful to discover faults in system design stage. A plan of changing design should also be developed before the start of the whole project. This would avoid the chaos caused due to a sudden change in system design.

Future development can also use 4GL programming languages instead of 3GL such as pascal. There are DBMS software which can effectively manage the library data. This can save the programming effort of implementing a data store.

5.4 Reflections

The software development project is by far the biggest task I have accomplished. It lasts almost a year. Fortunately, I can finish this project bit by bit throughout the year. If the development project is cramped in a small period of time, it might not be completed. This project again reminds me about the importance of time management.

The projects also teach me to be a consistent learner. After finishing this project, I feel proud of myself. I think my software is very good. However, when I see the library system used by the Hong Kong public library and the actual library system used by my school, my library system is but old and simple. In this information age, technologies advance in unbelievable speed. We have to be humble and catch up to the technology outside and thus, be better equipped.

The last thing I have learnt is the importance of copyrights. I used to think software do not deserve such a high price. Yet, in this project, I can feel the effort spent in programming. Even a system as small as mine would require a year to develop, let alone software that can be in practical use? It is very important to respect copyrights.

6. Reference and Acknowledgement

From websites:

1. http://www.freepascal.org/docs-html/rtl/crt/clreol.html
2. http://www.freepascal.org/docs-html/rtl/dos/getdate.html
3. http://www.freepascal.org/docs-html/rtl/sysutils/uppercase.html

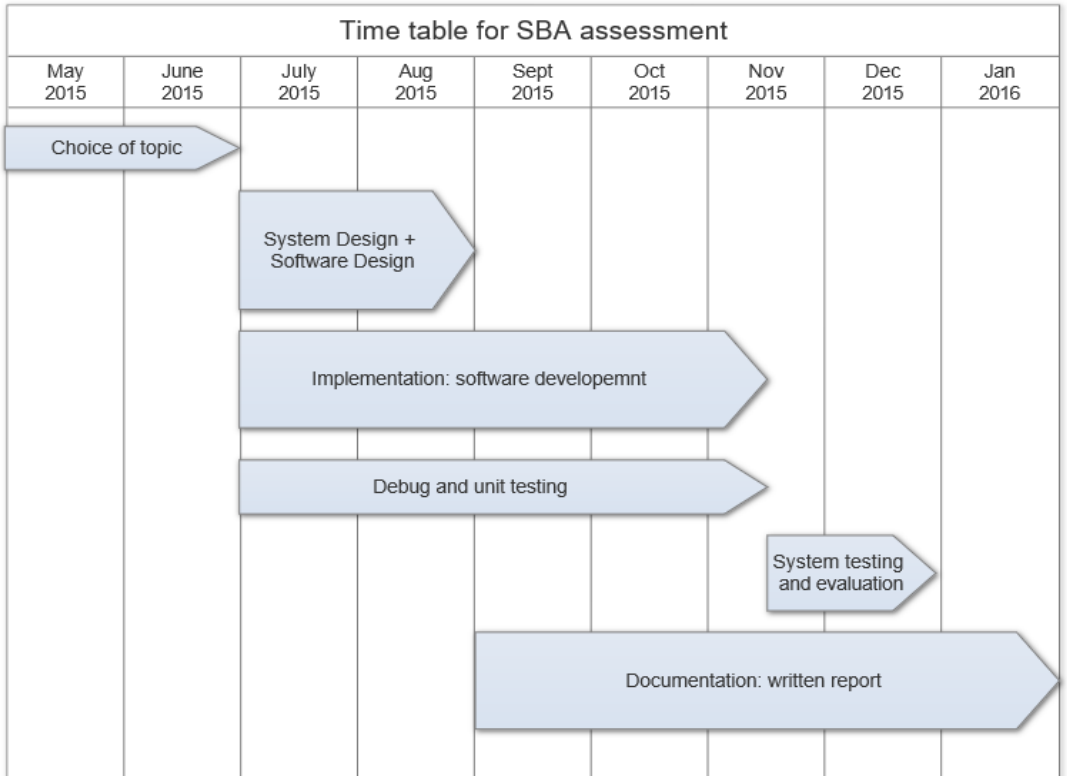
From books:

1. NSS ICT Elective D1 Software Development
2. NSS ICT Elective D2 Software Development

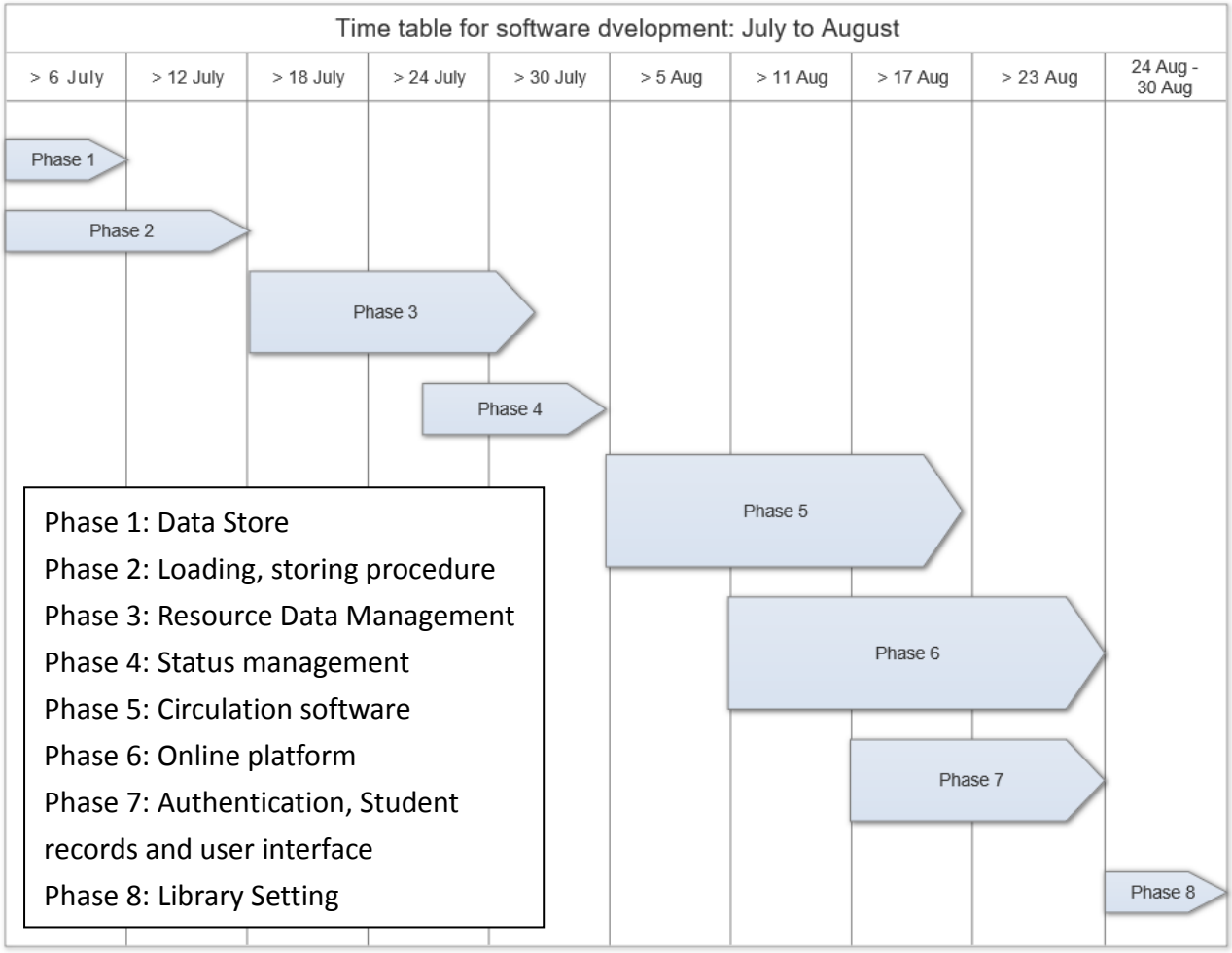
Acknowledgement:

1. ICT teacher of my school
2. Last year's ICT students
3. My classmates helping the debug

Appendix 1: Working Schedule



- In May, a set of topics were given and some uninterested topics were eliminated.
- In June, the feasibility of working on a particular topic is determined and a choice was made.
- In July and August, System design, implementation and debugging of the library software were conducted alternately.
- Before September, the prototype of the software was finished. It included all the designed functions. At the start of September, since all the major designs of the software were finished, the writing of report could start.
- In September and October, it is the fine-tuning of the software. User interface was enhanced, password hiding and on-screen tip displaying functions were added.
- From mid-November to the end of December, the system is tested by classmates and evaluated by myself.
- In January, the written report and programs were finalized and handed in to the supervising teacher.



- In each phase, the software function going to be implemented part was first designed. They were then implemented. A unit test on the implemented modules is conducted immediately after finishing the module implementation.
- Phase 1 and Phase 2 are both about text files processing, the design of implementation of both phases conducted at the same time.
- Phase 4 started after implementing phase 3.
- Phase 6 started designed after designing in phase 5.
- Phase 7 started designed after designing in phase 6.

Appendix 2: Program codes

```
Program Library_system;
Uses Crt, DOS, SysUtils ;
type
   BookDataType = record
                        ISBN: string[13];
                        Book_title, Book_Author, Book_publisher: string[50];
                        year_pub: string[4];
                        Res_typ: char;
                     end;
   BookStatType = record
                        Status: integer;
                        Stud_ID: string[6];
                        BK_ID: string[4];
                        ISBN: string[13];
                        date: string[8]
                     end;
   StudRecType    = record
                        Stud_ID: string[6];
                        ClasNum: string[4];
                        name: string[50];
                        pw :string[20];
                     end;

var BookDataArr: array[1..10000] of BookDataType;
      BookRecArr: array[1..10000] of BookStatType;
      BookIDStatArr: array[1..9999] of boolean;
      StudRecArr:array[1..2000] of StudRecType;
      ResTyplist: array['1'..'5']of string[13];
      BkStatlist: array[-10..10] of string[9];
      Borrlist: array[1..200] of integer;        {temp list}
      Requetlist: array[1..200] of integer;    {temp list}
      Bk_total, ISBN_total, St_total, ni, nj : integer;
      ToDae : string[8];    {Today's date in yyyymmdd}
      max_book, borrow_days, request_day, max_conti, max_request, RowPos: integer; {library setting}
      penal_per_day: real;
```

```
procedure CverDate(var InDateStr:string[8]; FonDay: integer);
var dd, mm, yyyy, reva, i :integer;
      temp: string[4];

Function IsValidDate(yyyy, mm, dd: integer):boolean;
var Monthlist: array[1..12] of integer;
      i : integer;
begin
   For i := 1 to 7 do
      if odd(i) then
         Monthlist[i] := 31
      else
         Monthlist[i] := 30;
   For i := 8 to 12 do
      if not odd(i) then
         Monthlist[i] := 31
      else
         Monthlist[i] := 30;
   Monthlist[2] := 28;
   If yyyy MOD 4 = 0 then
      Monthlist[2] := 29;
   If (yyyy MOD 100 = 0) and not (yyyy MOD 400 = 0) then
      Monthlist[2] := 28;
   IsValidDate := (dd <= Monthlist[mm]) and (mm < 13)
end;

begin
   val(copy(InDateStr, 1, 4), yyyy, reva);
   val(copy(InDateStr, 5, 2), mm, reva);
   val(copy(InDateStr, 7, 2), dd, reva);
   For i := 1 to FonDay do
   begin
      If IsValidDate(yyyy, mm, dd + 1) then
         dd := dd + 1
      else
      begin
         dd := 1;
```

```
        if IsValidDate(yyyy, mm + 1, dd) then
            mm := mm + 1
        else
        begin
            mm := 1;
            yyyy := yyyy + 1
        end
    end
  end;
  str(yyyy, InDateStr);
  str(mm, temp);
  If mm < 10 then
    temp := '0' + temp;
  InDateStr := InDateStr + temp;
  str(dd, temp);
  If dd < 10 then
    temp := '0' + temp;
  InDateStr := InDateStr + temp;
end;


Procedure Reload;
var i, temp, reva : integer;
    BookData, BookRec, StudRec, LibraSet: text;
    tempBKID, yc, mc, dc: string[8];
    d, m, y, reav: word;
begin
  GetDate(y, m, d, reav);
  str(y, yc);
  str(m, mc);
  str(d, dc);
  If m < 10 then
    mc := '0' + mc;
  If d < 10 then
    dc := '0' + dc;
  ToDae := yc + mc + dc;
  {-------------------get date part ends-------------------}
  Assign(BookData, 'Bk_data.txt');
```

```
Assign(BookRec, 'Bk_Rec.txt');
Assign(StudRec, 'St_Rec.txt');
Assign(LibraSet, 'LibSetting.txt');
Reset(BookData);
For i := 1 to 10000 do
   BookIDStatArr[i] := FALSE;
i := 0;
While not eof(BookData) do
begin
   i := i + 1;
   with BookDataArr[i] do
   begin
      readln(BookData, ISBN);
      readln(BookData, Book_title);
      readln(BookData, Book_Author);
      readln(BookData, Book_publisher);
      readln(BookData, year_pub, Res_typ)
   end
end;
Close(BookData);
ISBN_total := i;
i := 1;
Reset(BookRec);
While not eof(BookRec) do
begin
   with BookRecArr[i] do
      readln(BookRec, BK_ID, ISBN, Stud_ID, date, Status);
   If BookRecArr[i].BK_ID[1] <> '-' then
   begin
      tempBKID := copy(BookRecArr[i].BK_ID, 2, 8);
      val(tempBKID, temp, reva);
      BookIDStatArr[temp] := TRUE;
      If BookRecArr[i].Status = -2    then
         If BookRecArr[i].Date < ToDae then
         begin
            BookRecArr[i].Stud_ID := '*01234';
            BookRecArr[i].Status := 0;
            BookRecArr[i].Date := 'yyyymmdd';
```

```
            end;
        If BookRecArr[i].Status = -3 then
            If BookRecArr[i].Date < ToDae then
            begin
                BookRecArr[i].Stud_ID := '*01234';
                BookRecArr[i].Status := -1;
                BookRecArr[i].Date := 'yyyymmdd';
            end;
        i := i + 1
    end;
  end;
  Bk_total := i - 1;
  Close(BookRec);
  i := 1;
  Reset(StudRec);
  While not eof(StudRec) do
  begin
    with StudRecArr[i] do
    begin
      readln(StudRec, Stud_ID, pw);
      readln(StudRec, ClasNum, name)
    end;
    i := i + 1
  end;
  St_total := i - 1;
  Close(StudRec);
  Reset(LibraSet);
  readln(LibraSet, max_book);
  readln(LibraSet, borrow_days);
  readln(LibraSet, request_day);
  readln(LibraSet, penal_per_day);
  readln(LibraSet, max_conti);
  readln(LibraSet, max_request);
  Close(LibraSet)
end;



Procedure OverWrite(OverWriteFile:integer);
```

```
var i: integer;
      BookData, BookRec, StudRec, LibraSet: text;
begin
   Assign(BookData, 'Bk_data.txt');
   Assign(BookRec, 'Bk_Rec.txt');
   Assign(StudRec, 'St_Rec.txt');
   Assign(LibraSet, 'LibSetting.txt');
   i := 0;
   If OverWriteFile = 1 then
   begin
      Rewrite(BookData);
      While i < ISBN_total do
      begin
         i := i + 1;
         with BookDataArr[i] do
         begin
            writeln(BookData, ISBN);
            writeln(BookData, Book_title);
            writeln(BookData, Book_Author);
            writeln(BookData, Book_publisher);
            writeln(BookData, year_pub, Res_typ)
         end
      end;
      close(BookData);
   end;
   if OverWriteFile = 2 then
   begin
      Rewrite(BookRec);
      While i < Bk_total do
      begin
         i := i + 1;
         with BookRecArr[i] do
            writeln(BookRec, BK_ID, ISBN, Stud_ID, date, Status)
      end;
      close(BookRec)
   end;
   If OverWriteFile = 3 then
   begin
```

```
      Rewrite(StudRec);
      While i < St_total do
      begin
         i := i + 1;
         with StudRecArr[i] do
         begin
            writeln(StudRec, Stud_ID, pw);
            writeln(StudRec, ClasNum, name)
         end;
      end;
      close(StudRec)
   end;
   If OverWriteFile = 4 then
   begin
      Rewrite(LibraSet);
      writeln(LibraSet, max_book);
      writeln(LibraSet, borrow_days);
      writeln(LibraSet, request_day);
      writeln(LibraSet, penal_per_day);
      writeln(LibraSet, max_conti);
      writeln(LibraSet, max_request);
      close(LibraSet)
   end
end;


Function SearchISBN(target:string[13];SearchFile, Start_pos: integer):integer;    {Bk_total, ISBN_total} {BookDataArr,
BookRecArr}
var found: boolean;
      i : integer;
begin
   found := FALSE;
   i := Start_pos;
   If Searchfile = 1 then
      While (not found) and (i <= ISBN_total)    do
      begin
         if BookDataArr[i].ISBN = target then
         begin
```

```
            found := true;
            SearchISBN := i
        end;
        i := i + 1
    end
  else
    While (not found) and (i <= Bk_total)    do
    begin
        if BookRecArr[i].ISBN = target then
        begin
            found := true;
            SearchISBN := i
        end;
        i := i + 1
    end;
  if not found then
    SearchISBN := 0
end;


Function SearchBookID(target:string[4];SearchFile, Start_pos: integer):integer;
var found: boolean;
    i : integer;
begin
  found := FALSE;
  i := Start_pos;
  If Searchfile = 1 then
  begin
    While (not found) and (i <= Bk_total)    do
    begin
        if BookRecArr[i].Bk_ID = target then
        begin
            found := true;
            SearchBookID := i
        end;
        i := i + 1
    end
  end;
```

```
   if not found then
      SearchBookID := 0
end;



Function SearchStudID(target:string[6];SearchFile, Start_pos: integer):integer;    {Bk_total, ISBN_total} {BookDataArr,
BookRecArr}
var found: boolean;
      i : integer;
begin
   found := FALSE;
   i := Start_pos;
   If Searchfile = 1 then
   begin
      While (not found) and (i <= Bk_total)    do
      begin
         if BookRecArr[i].Stud_id = target then
         begin
            found := true;
            SearchStudID := i
         end;
         i := i + 1
      end
   end
   else
      While (not found) and (i <= St_total)    do
      begin
         if StudRecArr[i].Stud_id = target then
         begin
            found := true;
            SearchStudID := i
         end;
         i := i + 1
      end;
   if not found then
      SearchStudID := 0
end;
```

```
Procedure Display_bk_info(Bk_index: integer);
begin
   with BookDataArr[Bk_index] do
   begin
      writeln('                              Resourec type: ', ResTyplist[res_typ]);
      writeln('                              Resource title: ', Book_title);
      writeln('                              Resource owner/ Book owner: ', Book_author);
      writeln('                              Publisher: ', Book_publisher);
      writeln('                              Year of publish: ', year_pub);
   end
end;



Procedure UpStud_Record(moe, St_index: integer);        {core}
var Stud_id:string[6];
      choice: char;
      ClasNum: string[4];
      name: string[50];
      temp1, temp2 :string[2];

Procedure alt_Name_ClassNum;
begin
   writeln(' ':53,'Date: ', copy(ToDae, 7, 2), '/', copy(ToDae, 5, 2), '/', copy(ToDae, 1, 4));
   writeln('              ------------------------------------------------------------');
   writeln;
   writeln('                                          Update User Information');
   writeln('                              ---------------------------------------------');
   writeln;
   write('                     New Student Name: ');
   readln(name);
   if name = '-' then
      name := StudRecArr[St_index].name;
   write('                     New Class: ');
   readln(temp1);
   if temp1 = '-' then
      temp1 := copy(StudRecArr[St_index].ClasNum, 1, 2);
   write('                     New Class Number: ');
```

```
readln(temp2);
if temp2 = '-' then
   temp2 := copy(StudRecArr[St_index].ClasNum, 3, 2);
ClasNum := temp1 + temp2;
ClrScr;
writeln(' ':53,'Date: ', copy(ToDae, 7, 2), '/', copy(ToDae, 5, 2), '/', copy(ToDae, 1, 4));
writeln('                ------------------------------------------------------------');
writeln;
writeln('                                                Updated Information');
writeln('                              ---------------------------------------------');
writeln;
writeln('                                    Name: ', name);
writeln('                                    Class: ', temp1);
writeln('                                    Class Number: ', temp2);
writeln;
repeat
   write('                        Confirm changes?[Y/N] ');
   RowPos := WhereX;
   readln(choice);
   If not (choice IN['y', 'Y', 'n', 'N', '-']) then
   begin
      GotoXY(RowPos + 5, WhereY - 1);
      writeln('            >>Invalid Choice')
   end
until choice IN['y', 'Y', 'n', 'N', '-'];
If choice IN['y', 'Y'] then
begin
   StudRecArr[St_index].name := name;
   StudRecArr[St_index].ClasNum := ClasNum;
   OverWrite(3);
   write('                        >>Successfully changed!');
   readln
end;
If choice IN['n', 'N', '-'] then
begin
   Reload;
   write('                        >>Previous Record has been restored!');
   readln
```

```
  end
end;

Procedure admin_alt;
begin
  repeat
    Stud_id := '';
    choice := 'n';
    writeln(' ':53,'Date: ', copy(ToDae, 7, 2), '/', copy(ToDae, 5, 2), '/', copy(ToDae, 1, 4));
    writeln('                ------------------------------------------------------------');
    writeln;
    writeln('                                                User Record Management');
    writeln('                              ---------------------------------------------');
    writeln;
    repeat
      write('                              Student ID: ');
      RowPos := WhereX;
      readln(Stud_id);
      RowPos := length(Stud_id) + RowPos;
      If not (((length(Stud_id) = 6) AND (Stud_id[1] IN['s', 't'])) OR (Stud_id = '-')) then
      begin
        GotoXY(RowPos, WhereY - 1);
        writeln('          >>Invalid ID')
      end
    until ((length(Stud_id) = 6) AND (Stud_id[1] IN['s', 't'])) OR (Stud_id = '-');
    If Stud_id = '-' then
       choice := '-';
    St_index := SearchStudID(Stud_id, 2, 1);
    If not (Stud_id[1] IN['s', 't']) then
      St_index := -1;
    If St_index > 0 then
    begin
      ClrScr;
      writeln(' ':53,'Date: ', copy(ToDae, 7, 2), '/', copy(ToDae, 5, 2), '/', copy(ToDae, 1, 4));
      writeln('                ------------------------------------------------------------');
      writeln;
      writeln('                                                User Record Management');
      writeln('                              ---------------------------------------------');
```

```
writeln;
  with StudRecArr[St_index] do
  begin
    writeln('                    Student Name: ', name);
    If Stud_ID[1] = 's' then
    begin
      writeln('                    Class: ', copy(ClasNum, 1, 2));
      writeln('                    Class Number: ', copy(ClasNum, 3, 2))
    end
  end;
  writeln;
  repeat
    write('                    Change Student Information?[Y/N] ');
    readln(Choice);
  until choice IN['n', 'N', 'Y', 'y', '-'];
  ClrScr;
  If choice IN['Y', 'y'] then
    repeat
      alt_Name_ClassNum;
      ClrScr
    until choice IN['n', 'N', 'Y', 'y', '-']
end;
If (St_index = 0) AND (choice <> '-')    then
begin
  repeat
    writeln('                    >>Required student id not found!');
    write('                    >>Add New Student Record? [Y/N] ');
    readln(choice);
  until choice IN['n', 'N', 'Y', 'y', '-'];
  ClrScr;
  If choice IN['y', 'Y'] then
  begin
    repeat
      writeln(' ':53,'Date: ', copy(ToDae, 7, 2), '/', copy(ToDae, 5, 2), '/', copy(ToDae, 1, 4));
      writeln('          ------------------------------------------------------------');
      writeln;
      writeln('                                    Adding New Student ID: ', Stud_id);
      writeln('                    ---------------------------------------------');
```

```
            writeln;
            write('                    Student Name: ');
            readln(name);
            write('                    Class: ');
            readln(temp1);
            write('                    Class Number: ');
            readln(temp2);
            ClasNum := temp1 + temp2;
            write('                    >>Confirm Student Information?[Y/N] ');
            readln(choice);
            If choice IN ['n', 'N'] then
              ClrScr;
          until choice IN['Y', 'y', '-'];
          if choice IN['Y', 'y'] then
          begin
            St_total := St_total + 1;
            StudRecArr[St_total].name := name;
            StudRecArr[St_total].ClasNum := ClasNum;
            StudRecArr[St_total].pw := '1234';
            StudRecArr[St_total].stud_id := Stud_id;
            OverWrite(3);
            write('                    >>New Student Record have been added!');
          end;
          if choice IN['n', 'N', '-'] then
            write('                    >>No Student Record have been added');
          readln;
          ClrScr
        end
      end
  until (choice = '-') AND (Stud_id = '-')
end;

Procedure Pw_int(var pw:string[20]);
var key: char;
begin
  GotoXY(WhereX - length(pw), WhereY);
  ClrEol;
  pw := '';
```

```
  repeat
    key := readkey;
    If (key = #45) AND (pw = '') then
    begin
      pw := '-';
      write('-')
    end;
    If (((key >#64) AND (key < #91)) OR ((key >#96) and (key < #123)) OR ((key >#47) and (key < #58))) AND
(length(pw) < 20) AND (pw <> '-') then
    begin
      pw := pw + key;
      write('*')
    end;
    If (key = #8) and (length(pw) > 0) then
    begin
      pw := copy(pw, 1, length(pw) - 1);
      GotoXY(WhereX - 1, WhereY);
      ClrEol;
    end
  until key = #13;
end;

Procedure Pw_alt;
var Old_pw, New_pw, temp_pw: string[20];
begin
  Old_pw := '';
  New_pw := '';
  temp_pw := '';
  writeln(' ':53,'Date: ', copy(ToDae, 7, 2), '/', copy(ToDae, 5, 2), '/', copy(ToDae, 1, 4));
  writeln('              ------------------------------------------------------------');
  writeln;
  writeln('                                                Changing Password');
  writeln('                              ---------------------------------------------');
  writeln;
  write('                          Old Password: ');
  RowPos := WhereX;
  While (Old_pw <> StudRecArr[St_index].pw) AND (Old_pw <> '-') do
  begin
```

```
    Pw_int(Old_pw);
    If (Old_pw <> StudRecArr[St_index].pw) AND (Old_pw <> '-') then
    begin
        write('            >>Password Not Match!');
        readln;
        GotoXY(RowPos + length(Old_pw), WhereY - 1)
    end
end;
writeln;
If Old_pw <> '-' then
begin
    write('                        New Password: ');
    While (length(temp_pw) <= 3) AND (temp_pw <> '-') do
    begin
        Pw_int(temp_pw);
        If (length(temp_pw) <= 3) AND (temp_pw <> '-') then
        begin
            write('            >>Password too short!');
            readln;
            GotoXY(RowPos + length(temp_pw), WhereY - 1)
        end
    end;
    If temp_pw <> '-' then
    begin
        writeln;
        write('                        Confirm New Password: ');
        RowPos := WhereX;
        While (temp_pw <> New_pw) AND (New_pw <> '-') do
        begin
            Pw_int(New_pw);
            If (temp_pw <> New_pw) AND (New_pw <> '-') then
            begin
                write('            >>Password Not Match!');
                readln;
                GotoXY(RowPos + length(New_pw), WhereY - 1)
            end
        end
    end
```

```
  end;
  If length(New_pw) > 3    then
  begin
    StudRecArr[St_index].pw := New_pw;
    Overwrite(3);
    writeln;
    write('                    >>Successfully Changed Password!');
    readln
  end;
end;

Procedure stud_alt;
begin
  repeat
    writeln(' ':53,'Date: ', copy(ToDae, 7, 2), '/', copy(ToDae, 5, 2), '/', copy(ToDae, 1, 4));
    writeln('                ------------------------------------------------------------');
    writeln;
    writeln('                                                Your Information');
    writeln('                              ---------------------------------------------');
    writeln;
    with StudRecArr[St_index] do
    begin
      writeln('                                                Student Name: ', name);
      If Stud_ID[1] = 's' then
      begin
        writeln('                                                Class: ', copy(StudRecArr[St_index].ClasNum, 1, 2));
        writeln('                                                Class Number: ', copy(StudRecArr[St_index].ClasNum, 3, 2))
      end;
    end;
    writeln;
    repeat
      write('                              Change Login Password?[Y/N] ');
      RowPos := WhereX;
      Readln(choice);
      If not (choice IN['Y', 'y', '-', 'n', 'N']) then
      begin
        GotoXY(RowPos + 5, WhereY - 1);
        writeln('            >>Invalid Choice');
```

```
        end
      until Choice in['y', 'Y', '-', 'n', 'N'];
      ClrScr;
      If choice IN ['y', 'Y'] then
        Pw_alt;
      CLrScr
    until Choice = '-'
end;

begin
    Reload;
    Case moe of
      1: admin_alt;
      2: stud_alt;
      3: Pw_alt
    end
end;



Procedure DisplayBkStat(ISBN:string[13]);
var pst: integer;
begin
    pst := 0;
    repeat
      pst := SearchISBN(ISBN, 2, pst + 1);
      If pst > 0 then
      begin
        write('                    Resource ID: ', BookRecArr[pst].Bk_ID,'    ', BkStatlist[BookRecArr[pst].Status]);
        If BookRecArr[pst].Status IN[1..10] then
          write(' by ', BookRecArr[pst].Stud_id, ' until');
        If (BookRecArr[pst].Status = -2) OR (BookRecArr[pst].Status = -3) then
          write(' for ', BookRecArr[pst].Stud_id, ' until');
        If BookRecArr[pst].Date <> 'yyyymmdd' then
          write(copy(BookRecArr[pst].date, 7, 2):3, '/', copy(BookRecArr[pst].date, 5,
2),'/',copy(BookRecArr[pst].date, 1, 4));
        If (BookRecArr[pst].Date < ToDae) and (BookRecArr[pst].Date <> 'yyyymmdd') then
          write('*');
        writeln
```

```
        end
    until pst <= 0;
end;



Procedure UpBook_Record;
var Bk_index : integer;
        ISBN: string[13];
        choice, Res_typ: char;
        Book_title, Book_Author, Book_publisher: string[50];
        year_pub, ass_id: string[4];

Procedure alter_bk_data;          {confirm}
begin
    writeln(' ':53,'Date: ', copy(ToDae, 7, 2), '/', copy(ToDae, 5, 2), '/', copy(ToDae, 1, 4));
    writeln('                ------------------------------------------------------------');
    writeln;
    writeln('                                                Update Resource Type');
    writeln('                              ---------------------------------------------');
    writeln;
    Res_typ := '1';
    While Res_typ <= '5' do
    begin
        writeln('                                        ',Res_typ, '. ', ResTyplist[Res_typ]);
        Res_typ := succ(Res_typ)
    end;
    writeln;
    Repeat
        write('                                 Your choice: ');
        RowPos := WhereX;
        readln(Res_typ);
        If not (Res_typ IN['1', '2', '3', '4', '5', '-']) then
        begin
            GotoXY(RowPos + 5, WhereY - 1);
            writeln('            >>Invalid Choice');
        end
    until Res_typ IN['1', '2', '3', '4', '5', '-'];
    if Res_typ = '-' then
```

```
    Res_typ := BookDataArr[Bk_index].Res_typ;
  Clrscr;
  writeln(' ':53,'Date: ', copy(ToDae, 7, 2), '/', copy(ToDae, 5, 2), '/', copy(ToDae, 1, 4));
  writeln('                ------------------------------------------------------------');
  writeln;
  writeln('                                          Updating ', ResTyplist[Res_typ],' Information');
  writeln('                              ---------------------------------------------');
  writeln;
  write('                      Resource name/Book title: ');
  readln(Book_title);
  if Book_title = '-' then
    Book_title := BookDataArr[Bk_index].Book_title;
  write('                      Resource owner/Book author: ');
  readln(Book_author);
  if Book_author = '-' then
    Book_author := BookDataArr[Bk_index].Book_author;
  write('                      Publisher/Producer: ');
  readln(Book_publisher);
  if Book_publisher = '-' then
    Book_publisher := BookDataArr[Bk_index].Book_publisher;
  write('                      Year of publish: ');
  readln(year_pub);
  if year_pub = '-' then
    year_pub := BookDataArr[Bk_index].year_pub;
  ClrScr;
  writeln(' ':53,'Date: ', copy(ToDae, 7, 2), '/', copy(ToDae, 5, 2), '/', copy(ToDae, 1, 4));
  writeln('                ------------------------------------------------------------');
  writeln;
  writeln('                                            Updated Information');
  writeln('                              ---------------------------------------------');
  writeln;
  writeln('                        ISBN/Product code: ', ISBN);
  writeln('                        Resource name/Book title: ', Book_title);
  writeln('                        Resource Type: ', ResTyplist[Res_typ]);
  writeln('                        Resource owner/Book author: ', Book_author);
  writeln('                        Publisher/Producer: ', Book_publisher);
  writeln('                        Year of publish: ', year_pub);
  writeln;
```

```
   write('                                Confirm changes? ');
   readln(choice);
   If choice IN['y', 'Y'] then
   begin
      BookDataArr[Bk_index].Book_title := Book_title;
      BookDataArr[Bk_index].Book_author := Book_author;
      BookDataArr[Bk_index].Book_publisher := Book_publisher;
      BookDataArr[Bk_index].year_pub := year_pub;
      BookDataArr[Bk_index].Res_typ := Res_typ;
      write('                        >>Records have been updated!');
      readln
   end;
   If choice IN['n', 'N', '-'] then
   begin
      write('                        >>Previous Records have been restored!');
      readln
   end;
   OverWrite(1)
end;

Procedure genNewBkID(var New_id :string[4]);
var j: integer;
begin
   j := 1;
   While BookIDStatArr[j] do
      j := j + 1;
   BookIDStatArr[j] := TRUE;
   str(j, New_id);
   While length(New_id) < 4 do
      New_id := '0' + New_id
end;

Procedure alter_bk_num;      {pst: position of a bk_id record} {Un_StcList}
var j, pst, reva: integer;
begin
   j := 0;
   writeln;
   writeln('                                        1. Add Resource titles');
```

```
writeln'                                    2. Delete Resources titles');
writeln;
repeat
   write('                                Your choice: ');
   RowPos := WhereX;
   readln(choice);
   If not (choice IN['1', '2', '-']) then
   begin
      GotoXY(RowPos + 5, WhereY - 1);
      writeln('            >>Invalid Choice')
   end
until choice IN['1', '2', '-'];
pst := 0;
repeat
   pst := SearchISBN(ISBN, 2, pst + 1);
   j := j + 1
until pst = 0;
j := j - 1;
If choice = '1' then
begin
   writeln('                ------------------------------------------------------------');
   writeln('                >>Number of the Resources in library: ',j);
   repeat
      write('                    Number of Resources to add? ');
      RowPos :=    WhereX;
      readln(ass_id);                                    {use new var}
      RowPos := RowPos + length(ass_id);
      val(ass_id, j, reva);
      If not ((j >= 0) AND (reva = 0)) OR (ass_id = '-') then
      begin
         GotoXY(RowPos, WhereY - 1);
         writeln('            >>Invalid Number')
      end
   until ((j >= 0) AND (reva = 0)) OR (ass_id = '-') ;
   ass_id := '';
   For pst := 1 to j do
   begin
      Bk_total := Bk_total + 1;
```

```
        genNewBkID(ass_id);
        BookRecArr[Bk_total].ISBN := ISBN;
        BookRecArr[Bk_total].Bk_ID := ass_id;
        BookRecArr[Bk_total].Stud_ID := '*01234';
        BookRecArr[Bk_total].date := 'yyyymmdd';
        BookRecArr[Bk_total].Status := -1;
    end;
    OverWrite(2);
    If j > 0 then
    begin
        write('                >>Resource(s) added successfully!');
        readln
    end
end;
ClrScr;
If choice = '2' then
begin
    While (j > 0) and (ass_id <> '-') do
    begin
        pst := 0;
        ass_id := '-';
        ClrScr;
        Reload;
        writeln(' ':53,'Date: ', copy(ToDae, 7, 2), '/', copy(ToDae, 5, 2), '/', copy(ToDae, 1, 4));
        writeln('                -------------------------------------------------------------');
        writeln;
        writeln('                                                Removing Resources');
        writeln('                                -----------------------------------------------');
        writeln;
        DisplayBkStat(ISBN);
        writeln;
        write('                    Enter a Resource ID to remove: ' );
        readln(ass_id);
        If ass_id <> '-' then
            pst := SearchBookID(ass_id, 1, 1);
        If (pst > 0) then
        begin
            writeln('                -------------------------------------------------------------');
```

```
            If NOT (BookRecArr[pst].Status IN[1..10]) then
            begin
               BookRecArr[pst].Bk_ID := '-12-';
               OverWrite(2);
               write('                >>Resource removed successfully!');
               j := j - 1
            end;
            If (BookRecArr[pst].Status IN[1..10]) then
               write('                >>Resourcoe cannot be removed!');
            readln
         end;
      end
   end
end;

begin
   repeat
      Reload;
      choice := 'n';
      writeln(' ':53,'Date: ', copy(ToDae, 7, 2), '/', copy(ToDae, 5, 2), '/', copy(ToDae, 1, 4));
      writeln('                ------------------------------------------------------------');
      writeln;
      writeln('                                                Resource Data Management');
      writeln('                                ---------------------------------------------');
      writeln;
      Repeat
         ISBN := '';
         write('                                ISBN/Product code: ');
         RowPos := WhereX;
         readln(ISBN);
         RowPos := length(ISBN) + RowPos;
         If not ((length(ISBN) = 13) OR (ISBN = '-')) then
         begin
            GotoXY(RowPos, WhereY - 1);
            writeln('            >>Invalid ISBN')
         end
      until (length(ISBN) = 13) OR (ISBN = '-');          {Check(ISBN) if in practical use, Check(ISBN: sting[13]):Boolean;
validates ISBN code}
```

```
If ISBN = '-' then
   choice := '-';
Bk_index := SearchISBN(ISBN, 1, 1);
If Bk_index > 0 then
begin
   Repeat
      ClrScr;
      writeln(' ':53,'Date: ', copy(ToDae, 7, 2), '/', copy(ToDae, 5, 2), '/', copy(ToDae, 1, 4));
      writeln('              ------------------------------------------------------------');
      writeln;
      writeln('                                          Update Resource Information');
      writeln('                          ---------------------------------------------');
      Display_bk_info(Bk_index);
      writeln('                          ---------------------------------------------');
      writeln;
      writeln('                                     1. Edit information');
      writeln('                                     2. Add/Remove the Resources');
      writeln;
      write('                              Your choice: ');
      readln(Choice);
      RowPos := WhereX;
      ClrScr;
      if Choice = '1' then
      begin
         repeat
            Display_bk_info(Bk_index);
            ClrScr;
            alter_bk_data;
            If not (choice IN['Y', 'y', '-', 'n', 'N']) then
            begin
               GotoXY(RowPos + 5, WhereY - 1);
               writeln('           >>Invalid Choice')
            end;
            ClrScr
         until choice IN['Y', 'y', '-', 'n', 'N'];
         choice := 'd'
      end;
      if Choice = '2' then
```

```
        begin
          repeat
            writeln(' ':53,'Date: ', copy(ToDae, 7, 2), '/', copy(ToDae, 5, 2), '/', copy(ToDae, 1, 4));
            writeln('                ------------------------------------------------------------');
            writeln;
            writeln('                                          Update Number of Resources');
            writeln('                              ---------------------------------------------');
            Display_bk_info(Bk_index);
            writeln('                              ---------------------------------------------');
            alter_bk_num;
            ClrScr
          until choice = '-';
          choice := 'd'
        end
      until choice = '-'
    end;
    If (Bk_index = 0) AND (choice <> '-') then
    begin
      repeat
        writeln('                ------------------------------------------------------------');
        writeln('                >>Required Resource code/ISBN Not found!');
        write('                Add New Resource Record?[Y/N] ');
        RowPos := WhereX;
        readln(choice);
        If not (choice IN['Y', 'y', 'N', 'n', '-']) then
        begin
          GotoXY(RowPos + 5, WhereY - 1);
          writeln('            >>Invalid Choice');
        end;
      until choice IN['Y', 'y', 'N', 'n', '-'];
      ClrScr;
      If choice IN    ['N', 'n', '-'] then
        Bk_index := -1;
      If choice IN['Y', 'y'] then
      begin
        repeat
          writeln(' ':53,'Date: ', copy(ToDae, 7, 2), '/', copy(ToDae, 5, 2), '/', copy(ToDae, 1, 4));
          writeln('                ------------------------------------------------------------');
```

```
writeln;
writeln('                                Adding New Resource: Resource Type');
writeln('                          ---------------------------------------------');
writeln;
Res_typ := '1';
While Res_typ <= '5' do
begin
   writeln('                                ',Res_typ, '. ', ResTyplist[Res_typ]);
   Res_typ := succ(Res_typ)
end;
writeln;
Repeat
   write('                              Your choice: ');
   RowPos := WhereX;
   readln(Res_typ);
   If not (Res_typ IN['1', '2', '3', '4', '5', '-']) then
   begin
      GotoXY(RowPos + 5, WhereY - 1);
      writeln('            >>Invalid Choice');
   end
until Res_typ IN['1', '2', '3', '4', '5', '-'];
ClrScr;
writeln(' ':53,'Date: ', copy(ToDae, 7, 2), '/', copy(ToDae, 5, 2), '/', copy(ToDae, 1, 4));
writeln('                ------------------------------------------------------------');
writeln;
writeln('                                          Adding New ', ResTyplist[Res_typ]);
writeln('                          ---------------------------------------------');
writeln;
write('                     Resource name/Book title: ');
readln(Book_title);
write('                     Resource owner/Book author: ');
readln(Book_author);
write('                     Publisher/Producer name: ');
readln(Book_publisher);
write('                     Year of publish: ');
readln(year_pub);
write('                     Confirm The above information?[Y/N] ');
readln(choice);
```

```
            If choice IN['n', 'N'] then
                ClrScr;
            If choice = '-' then
                Bk_index := -1;
        until choice IN['Y', 'y', '-'];
        if choice IN['Y', 'y'] then
        begin
            ISBN_total := ISBN_total + 1;
            BookDataArr[ISBN_total].ISBN := ISBN;
            BookDataArr[ISBN_total].Book_title := Book_title;
            BookDataArr[ISBN_total].Book_author := Book_author;
            BookDataArr[ISBN_total].res_typ := res_typ;
            BookDataArr[ISBN_total].Book_publisher := Book_publisher;
            BookDataArr[ISBN_total].year_pub := year_pub;
            Bk_total := Bk_total + 1;
            ass_id := '';
            genNewBkID(ass_id);
            BookRecArr[Bk_total].ISBN := ISBN;
            BookRecArr[Bk_total].Bk_ID := ass_id;
            BookRecArr[Bk_total].Stud_ID := '*01234';
            BookRecArr[Bk_total].date := 'yyyymmdd';
            BookRecArr[Bk_total].Status := -1;
            OverWrite(1);
            OverWrite(2);
            write('              >>New Resource Record have been added!');
        end;
        if choice IN['n', 'N', '-'] then
            write('              >>No Resource Record have been added');
        readln;
        ClrScr
      end
    end
  until (choice = '-') AND (ISBN = '-');
end;



Procedure Stud_Acc(Stud_ID:string[6]; var NumBorr, NumBooked, NumLate: integer);
var pst: integer;
```

```
begin
   NumBorr := 0;
   NumBooked := 0;
   NumLate:= 0;
   pst := 0;
   repeat
      pst := SearchStudID(Stud_ID, 1, pst + 1);
      If pst > 0 then
      begin
         If BookRecArr[pst].Status in[1..10] then
         begin
            NumBorr := NumBorr + 1;
            Borrlist[NumBorr] :=    pst
         end;
         If (BookRecArr[pst].Status < -1) and (BookRecArr[pst].Status > -4) then
         begin
            NumBooked := NumBooked + 1;
            Requetlist[NumBooked] :=    pst
         end
      end;
   until pst <= 0;
   For pst := 1 to NumBorr do
      If BookRecArr[Borrlist[pst]].date < ToDae then
         NumLate := NumLate + 1;
end;



Procedure Stud_info(Stud_ID:string[6]);
var pst, i, NumBorr, NumBooked, NumLate: integer;
      DateStr: string[11];
begin
   Stud_Acc(Stud_ID, NumBorr, NumBooked, NumLate);
   If NumBorr > 0 then
   begin
      writeln('                                        Borrowed Resources');
      writeln('                          ---------------------------------------------');
      writeln('                    ID       Return Date         Type                Resource name');
   end;
```

```
   For i := 1 to NumBorr do
   begin
      pst := SearchISBN(BookRecArr[Borrlist[i]].ISBN, 1, 1);
      DateStr := copy(BookRecArr[Borrlist[i]].date, 7, 2) + '/' + copy(BookRecArr[Borrlist[i]].date, 5, 2) + '/' +
copy(BookRecArr[Borrlist[i]].date, 1, 4);
      If BookRecArr[Borrlist[i]].date < ToDae then
         DateStr := DateStr + '*';
      write(BookRecArr[Borrlist[i]].Bk_ID:13 , DateStr:15, ResTyplist[BookDataArr[pst].Res_typ]:9, '    ',
BookDataArr[pst].Book_title:13);
      writeln
   end;
   If NumBooked > 0 then
   begin
      If NumBorr > 0 then
         writeln;
      writeln('                                        Reserved Resources');
      writeln('                          ---------------------------------------------');
      writeln('                    ID    Cancel Before         Type                Resource name');
   end;
   For i := 1 to NumBooked do
   begin
      pst := SearchISBN(BookRecArr[Requetlist[i]].ISBN, 1, 1);
      DateStr := copy(BookRecArr[Requetlist[i]].date, 7, 2) + '/' + copy(BookRecArr[Requetlist[i]].date, 5, 2) + '/' +
copy(BookRecArr[Requetlist[i]].date, 1, 4);
      write(BookRecArr[Requetlist[i]].Bk_ID:13 , DateStr:15, ResTyplist[BookDataArr[pst].Res_typ]:9, '    ',
BookDataArr[pst].Book_title:13);
      writeln
   end;
   writeln
end;



Procedure UpBookStat;
var i, j, k :integer;
      choice: char;
      Bk_IDStr: string[4];
      Displaylist: array[1..8] of integer;
```

```
Procedure Borrowlist(Onf: char);
begin
   i := 1;
   j := 0;
   writeln(' ':53,'Date: ', copy(ToDae, 7, 2), '/', copy(ToDae, 5, 2), '/', copy(ToDae, 1, 4));
   writeln('                ------------------------------------------------------------');
   writeln;
   writeln('                                                Display Resource Status');
   writeln('                                ---------------------------------------------');
   While i <= Bk_total do
   begin
      choice := 'y';
      if (BookRecArr[i].Status in[1..10]) and (Onf = '1') then
      begin
         write('                    Resource ID: ', BookRecArr[i].Bk_ID,' ', BkStatlist[BookRecArr[i].Status],' by ',
BookRecArr[i].Stud_id, ' until ');
         write(copy(BookRecArr[i].date, 7, 2):3, '/', copy(BookRecArr[i].date, 5, 2),'/',copy(BookRecArr[i].date, 1, 4));
         If (BookRecArr[i].Date < ToDae) then
            write('*');
         writeln;
         j := j + 1
      end;
      if (BookRecArr[i].Status in[1..10]) and (Onf = '2') and (BookRecArr[i].Date < ToDae) then
      begin
         write('                    Resource ID: ', BookRecArr[i].Bk_ID,' ', BkStatlist[BookRecArr[i].Status],' by ',
BookRecArr[i].Stud_id, ' until ');
         write(copy(BookRecArr[i].date, 7, 2):3, '/', copy(BookRecArr[i].date, 5, 2),'/',copy(BookRecArr[i].date, 1, 4));
         writeln;
         j := j + 1
      end;
      if (j = 8) OR (i = Bk_total) then
      begin
         write('                    Continue Displaying Records?[Y,N] ');
         readln(choice);
         j := 0;
         If choice IN ['-', 'N', 'n'] then
         begin
            i := Bk_total;
```

```
            ClrScr
        end
        else
        begin
            if i = Bk_total then
                i := 0;
            ClrScr;
            writeln(' ':53,'Date: ', copy(ToDae, 7, 2), '/', copy(ToDae, 5, 2), '/', copy(ToDae, 1, 4));
            writeln('                ------------------------------------------------------------');
            writeln;
            writeln('                                                Display Resource Status');
            writeln('                                ---------------------------------------------');
        end;
    end;
    i := i + 1
  end
end;

Procedure OnOfflist(Onf: char);
var speech: string[50];
    ShoStat, ReplaStat, total_dis: integer;
begin
  if Onf = '4' then
  begin
    speech := '                    Resource to be put on shelf: ';
    ShoStat := -1;
    ReplaStat := 0
  end;
  if Onf = '3' then
  begin
    speech := '                    Resource to be off-shelf: ';
    ShoStat := 0;
    ReplaStat := -1
  end;
  If Onf = '5' then
  begin
    speech := '                    Cancel Online Reserve(On Shelf Books): ';
    ShoStat := -2;
```

```
    ReplaStat := 0
  end;
  If Onf = '6' then
  begin
    speech := '                    Cancel Online Reserve(Off Shelf Books): ';
    ShoStat := -3;
    ReplaStat := -1;
  end;
  Bk_IDStr := '';
  choice := 'n';
  i := 1;
  j := 0;
  total_dis := 0;
  writeln(' ':53,'Date: ', copy(ToDae, 7, 2), '/', copy(ToDae, 5, 2), '/', copy(ToDae, 1, 4));
  writeln('              ------------------------------------------------------------');
  writeln;
  writeln('                                        Update & Display Resource Status');
  writeln('                              ---------------------------------------------');
  While i <= Bk_total do
  begin
    if BookRecArr[i].Status = ShoStat then
    begin
      write('                                    Resource ID: ', BookRecArr[i].Bk_ID,' ', BkStatlist[BookRecArr[i].Status]) ;
      j := j + 1;
      Displaylist[j] := i;
      total_dis := total_dis + 1;
      if (BookRecArr[i].Status = -2) OR (BookRecArr[i].Status = -3) then
      begin
        write(' for ', BookRecArr[i].Stud_id, ' until');
        write(' ',copy(BookRecArr[i].date, 7, 2):4, '/', copy(BookRecArr[i].date, 5, 2),'/',copy(BookRecArr[i].date, 1,
4))
      end;
      writeln
    end;
    if ((j = 8) OR (i = Bk_total)) AND (j > 0) then
    begin
      repeat
        writeln;
```

```
    write(speech);
    readln(Bk_IDStr);
    k := 1;
    While (k <= j) and (BookRecArr[Displaylist[k]].Bk_ID <> Bk_IDStr) do
        k := k + 1;
until (k <= j) OR (Bk_IDStr = '*.*') OR (Bk_IDStr = '')OR (Bk_IDStr = '-');
If Bk_IDStr = '*.*' then
begin
    k := 1;
    While k <= j do
    begin
        BookRecArr[Displaylist[k]].Status := ReplaStat;
        BookRecArr[Displaylist[k]].Stud_ID := '*01234';
        BookRecArr[Displaylist[k]].Date := 'yyyymmdd';
        k := k + 1
    end;
    Overwrite(2);
    write('                    >>Success!');
    readln
end;
If k <= j then
begin
    BookRecArr[Displaylist[k]].Status := ReplaStat;
    BookRecArr[Displaylist[k]].Stud_ID := '*01234';
    BookRecArr[Displaylist[k]].Date := 'yyyymmdd';
    Overwrite(2);
    write('                    >>Success!');
    readln
end;
j := 0;
if Bk_IDStr = '' then
    if i = Bk_total then
        i := 0;
If Bk_IDStr = '-' then
        i :=    Bk_total;
If (Bk_IDStr <>    '-') AND (Bk_IDStr <> '') then
    i := Displaylist[1] - 1;
ClrScr;
```

```
        writeln(' ':53,'Date: ', copy(ToDae, 7, 2), '/', copy(ToDae, 5, 2), '/', copy(ToDae, 1, 4));
        writeln('              ------------------------------------------------------------');
        writeln;
        writeln('                                    Update & Display Resource Status');
        writeln('                          ---------------------------------------------');
      end;
      i := i + 1
    end;
    If total_dis = 0 then
      begin
        If (Onf = '4') then
          write('                  >>No Off shelf Resource in the library!');
        If (Onf = '3') then
          write('                  >>No On shelf Resource in the library!');
        If (Onf = '5') then
          write('                  >>No On Shelf Reserved Book in the library!');
        If (Onf = '6') then
          write('                  >>No Off Shelf Reserved Book in the library!');
        readln
      end;
end;

begin
  repeat
    writeln(' ':53,'Date: ', copy(ToDae, 7, 2), '/', copy(ToDae, 5, 2), '/', copy(ToDae, 1, 4));
    writeln('              ------------------------------------------------------------');
    writeln;
    writeln('                                        Display Resource Status');
    writeln('                          ---------------------------------------------');
    writeln('                                1. Display Borrowed resource list');
    writeln('                                2. Display Late Return resource list');
    writeln;
    writeln('                                    Update & Display Resource Status');
    writeln('                          ---------------------------------------------');
    writeln('                                3. Manage On shelf Resource');
    writeln('                                4. Manage Off shelf Resource');
    writeln('                                5. Manage reserved Resource(On Shelf)');
    writeln('                                6. Manage reserved Resource(Off Shelf)');
```

```
      writeln;
      write('                          Your Choice: ');
      RowPos := WhereX;
      readln(choice);
      If not (choice IN['1', '2', '-', '3', '4', '5', '6']) then
      begin
         GotoXY(RowPos + 5, WhereY - 1);
         write('           >>Invalid Choice');
         readln
      end;
      If choice IN['1', '2'] then
      begin
         ClrScr;
         Borrowlist(choice);
         choice := 't';
      end;
      If choice IN['3', '4', '5', '6'] then
      begin
         ClrScr;
         OnOfflist(choice);
      end;
      ClrScr
   until choice = '-'
end;


procedure BNRCirc;          {book_status: 0:on shelf     1:booked       2:borrowed       3:off-shelf }     {ToDae}
var ID_index, pst, i, NumBorr, NumBooked, NumLate: integer;
      temp: string[6];
      Choice: char;
      tempDate: string[8];

procedure ReturnNContine;
begin
   TempDate := BookRecArr[pst].date;
   While    (ToDae > TempDate) do
   begin
      i := i + 1;
```

```
    CverDate(TempDate, 1)
end;
repeat
    writeln(' ':53,'Date: ', copy(ToDae, 7, 2), '/', copy(ToDae, 5, 2), '/', copy(ToDae, 1, 4));
    writeln('              ------------------------------------------------------------');
    writeln;
    writeln('                                                Returning & Renewing');
    writeln('                          ---------------------------------------------');
    writeln;
    writeln('                              Book ISBN/Product code: ',BookRecArr[pst].ISBN);
    writeln('                              Book ID: ',BookRecArr[pst].BK_ID);
    writeln;
    writeln('                          ---------------------------------------------');
    writeln;
    If (BookRecArr[pst].Status < max_conti) and (NumBorr < max_book) and (NumLate = 0) then
    begin
        writeln('                                  1. Renew the borrowing.');
        writeln('                                  2. Return the resource.');
        writeln;
        repeat
            write('                          Your choice: ');
            RowPos := WhereX;
            readln(choice);
            If not (choice IN['1', '2', '-']) then
            begin
                GotoXY(RowPos + 5, WhereY - 1);
                writeln('            >>Invalid Choice');
            end
        until choice in['1', '2', '-']
    end
    else
    begin
        choice := '2';
        writeln('                                  2. Return the resource.');
        writeln
    end;
    writeln('              ------------------------------------------------------------');
    if i > 0 then
```

```
begin
   writeln('              >>Late Return Resource!');
   writeln('              >>Late Penalty: ', i*penal_per_day: 0: 2)
end;
If choice = '2' then
begin
   repeat
      write('              Return Borrowed Resource?[Y/N] ');
      RowPos := WhereX;
      readln(choice);
      If not (choice IN['n', 'N', 'Y', 'y', '-']) then
      begin
         GotoXY(RowPos + 5, WhereY - 1);
         writeln('              >>Invalid Choice');
      end
   until choice IN['n', 'N', 'Y', 'y', '-'];
   if choice IN['Y', 'y'] then
   begin
      BookRecArr[pst].Stud_ID    := '*01234';
      BookRecArr[pst].Status := -1;
      BookRecArr[pst].date := 'yyyymmdd';
      Overwrite(2);
      write('              >>Book Returned!');
      if i > 0 then
         writeln('              >>Please Collect the Fine!')
   end
   else
      write('              >>Resource is not returned!')
end;
If choice = '1' then
begin
   writeln('              >>The borrowing has been borrowed ', BookRecArr[pst].Status, ' times');
   repeat
      write('              Renew the borrowing?[Y/N] ');
      RowPos := WhereX;
      readln(choice);
      If not (choice IN['n', 'N', 'Y', 'y', '-']) then
      begin
```

```
            GotoXY(RowPos + 5, WhereY - 1);
            writeln('              >>Invalid Choice');
          end
        until choice IN['n', 'N', 'Y', 'y', '-'];
        if choice IN['Y', 'y'] then
        begin
          BookRecArr[pst].Date := ToDae;
          CverDate(BookRecArr[pst].Date, borrow_days);
          BookRecArr[pst].Status := BookRecArr[pst].Status + 1;
          BookRecArr[pst].Stud_id := StudRecArr[ID_index].Stud_ID;
          Overwrite(2);
          write('              >>Renewed Successfully!')
        end
        else
          write('              >>Resource Borrowing is cancelled')
      end;
      If choice IN['n', 'N'] then
      begin
        readln;
        ClrScr
      end
    until choice IN['Y', 'y', '-']
end;

Procedure ResvBkBorr;
begin
  writeln(' ':53,'Date: ', copy(ToDae, 7, 2), '/', copy(ToDae, 5, 2), '/', copy(ToDae, 1, 4));
  writeln('              ------------------------------------------------------------');
  writeln;
  writeln('                                        Reserved Resource Borrowing');
  writeln('                            ---------------------------------------------');
  writeln;
  writeln('                              Book ISBN/Product code: ',BookRecArr[pst].ISBN);
  writeln('                              Book ID: ',BookRecArr[pst].BK_ID);
  writeln;
  writeln('              ------------------------------------------------------------');
  repeat
    write('              Take Reserved Resource?[Y/N] ');
```

```
      RowPos := WhereX;
      readln(choice);
      If not (choice IN['n', 'N', 'Y', 'y', '-']) then
      begin
         GotoXY(RowPos + 5, WhereY - 1);
         writeln('            >>Invalid Choice');
      end
   until choice IN['n', 'N', 'Y', 'y', '-'];
   if choice IN['Y', 'y'] then
   begin
      BookRecArr[pst].Date := ToDae;
      CverDate(BookRecArr[pst].Date, borrow_days);
      BookRecArr[pst].Status := 1;
      Overwrite(2);
      write('                >>Reserved Resource borrowed!')
   end
   else
      write('                >>Resource is not borrowed!')
end;

Procedure BorrRes;
begin
   writeln(' ':53,'Date: ', copy(ToDae, 7, 2), '/', copy(ToDae, 5, 2), '/', copy(ToDae, 1, 4));
   writeln('                ------------------------------------------------------------');
   writeln;
   writeln('                                                On-shelf Resource Borrowing');
   writeln('                                  ---------------------------------------------');
   writeln;
   writeln('                                  Book ISBN/Product code: ',BookRecArr[pst].ISBN);
   writeln('                                  Book ID: ',BookRecArr[pst].BK_ID);
   writeln;
   writeln('                ------------------------------------------------------------');
   repeat
      write('                Borrow the above Resource?[Y/N] ');
      RowPos := WhereX;
      readln(choice);
      If not (choice IN['n', 'N', 'Y', 'y', '-']) then
      begin
```

```
        GotoXY(RowPos + 5, WhereY - 1);
        writeln('            >>Invalid Choice');
      end
    until choice IN['n', 'N', 'Y', 'y', '-'];
    if choice IN['Y', 'y'] then
    begin
      BookRecArr[pst].Date := ToDae;
      CverDate(BookRecArr[pst].Date, borrow_days);
      BookRecArr[pst].Status := 1;
      BookRecArr[pst].Stud_id := StudRecArr[ID_index].Stud_ID;
      Overwrite(2);
      write('            >>Borrow Successfully!')
    end
    else
      write('            >>Resource is not borrowed!')
end;

begin
    repeat
      choice := 'd';
      Reload;
      writeln(' ':53,'Date: ', copy(ToDae, 7, 2), '/', copy(ToDae, 5, 2), '/', copy(ToDae, 1, 4));
      writeln('                ------------------------------------------------------------');
      writeln;
      writeln('                                                    Circulation software');
      writeln('                                ---------------------------------------------');
      writeln;
      repeat
        write('                                User ID: ');
        RowPos := WhereX;
        readln(temp);
        ID_index := SearchStudID(temp, 2, 1);
        If ID_index = 0 then
        begin
          GotoXY(RowPos + 5, WhereY - 1);
          writeln('            >>No Such User!');
        end
      until ((ID_index > 0) OR (temp = '-')) AND (temp[1] <> 'A');
```

```
ClrScr;
if temp = '-' then
   choice := '-';
If (temp <> '-') AND (temp[1] <> 'A') then
begin
   writeln(' ':53,'Date: ', copy(ToDae, 7, 2), '/', copy(ToDae, 5, 2), '/', copy(ToDae, 1, 4));
   writeln('              ------------------------------------------------------------');
   writeln;
   writeln('                                                User Information');
   writeln('                              ---------------------------------------------');
   writeln('                                              User ID: ',StudRecArr[ID_index].Stud_ID);
   If StudRecArr[ID_index].Stud_ID[1] = 's' then
   begin
      writeln('                                              Class: ', copy(StudRecArr[ID_index].ClasNum, 1, 2));
      writeln('                                              Class Number: ', copy(StudRecArr[ID_index].ClasNum, 3, 2))
   end;
   writeln('                                              Name: ',StudRecArr[ID_index].Name);
   writeln;
   write('                        >>Press Enter to proceed circulation.');
   readln;
   ClrScr
end;
While choice <> '-' do
begin
   writeln(' ':53,'Date: ', copy(ToDae, 7, 2), '/', copy(ToDae, 5, 2), '/', copy(ToDae, 1, 4));
   writeln('              ------------------------------------------------------------');
   Stud_info(StudRecArr[ID_index].Stud_ID);
   Stud_Acc(StudRecArr[ID_index].Stud_ID, NumBorr, NumBooked, NumLate);
   If (NumBorr = 0) and (NumBooked = 0) then
   begin
      writeln('                                                Circulation Process');
      writeln('              ------------------------------------------------------------');
      writeln
   end;
   repeat
      write('                Enter a Resource ID: ');
      RowPos := WhereX;
      readln(temp);
```

```
            pst := 0;
            i := 0;
            pst := SearchBookID(temp, 1, pst + 1);
            RowPos := length(temp) + RowPos;
            If (pst = 0) and (temp <> '-')then
            begin
               GotoXY(RowPos, WhereY - 1);
               writeln('           >>No Such Resource ID!');
            end;
            If pst > 0 then
            begin
               If (BookRecArr[pst].Status = -1) OR ((BookRecArr[pst].Status = -3) AND (StudRecArr[ID_index].Stud_ID <>
BookRecArr[pst].Stud_ID)) then
               begin
                  writeln('               >>Off-shelf Resource! Please Recover the Resource immediately!');
                  writeln
               end;
               If (BookRecArr[pst].Status = -2) AND (StudRecArr[ID_index].Stud_ID <> BookRecArr[pst].Stud_ID) then
               begin
                  writeln('               >>Reserved Resource! Please Recover the Resource immediately!');
                  writeln;
                  BookRecArr[pst].Status := -3;
                  Overwrite(2)
               end
            end
         until ((pst > 0) and (BookRecArr[pst].Status > -1)) OR (temp = '-') OR (StudRecArr[ID_index].Stud_ID =
BookRecArr[pst].Stud_ID);
         If temp = '-' then
         begin
            choice := '-';
            temp := 't';
            ClrScr
         end;
         If pst > 0 then
         begin
            i := 0;
            If choice <> '-' then
            begin
```

```
                If (NumBorr >= max_book) then
                begin
                   write('              >>Maximum number of borrowed Resource reached!');
                   readln
                end;
                If (NumLate > 0) then
                begin
                   writeln('              >>Have ',NumLate, ' late return resource(s)!');
                   write('              >>Please return all of them before any more resources can be borrowed!');
                   readln
                end
             end;
             ClrScr;
             If (StudRecArr[ID_index].Stud_ID = BookRecArr[pst].Stud_ID) and (BookRecArr[pst].Status in[1..10]) then
                ReturnNContine;
             If (StudRecArr[ID_index].Stud_ID = BookRecArr[pst].Stud_ID) and ((BookRecArr[pst].Status = -2) OR
(BookRecArr[pst].Status = -3))and (NumBorr < max_book) and (NumLate = 0) then        {Reserved borrow}
                ResvBkBorr;
             If (NumBorr < max_book) and (NumLate = 0) and (BookRecArr[pst].Status = 0) then
{borrow}
                BorrRes;
             If (choice <> '-') and (NumBorr < max_book) and (NumLate = 0) then
                readln;
             ClrScr;
             choice := 't'
          end
       end
   until (choice = '-') AND (temp = '-');
end;



Procedure OptLibSet;
var temp: string[4];
       reva: integer;
       tempreal: real;
       Choice: Char;
begin
   writeln(' ':53,'Date: ', copy(ToDae, 7, 2), '/', copy(ToDae, 5, 2), '/', copy(ToDae, 1, 4));
```

```
writeln('            ------------------------------------------------------------');
writeln;
writeln('                                        Current Library Settings');
writeln('                          ---------------------------------------------');
writeln('            Maximum number of Resources can be borrowed by a user: ', max_book);
writeln('            Number of Days each borrow lasts: ', borrow_days);
writeln('            Number of Days before the online reserve is cancelled: ', request_day);
writeln('            Amount of Fine increased each day: $',penal_per_day:0:2);
writeln('            Maximum number of renewal: ', max_conti);
writeln('            Maximum number of online reserves made by each student: ', max_request);
writeln;
Repeat
   write('            Change Settings?[Y/N] ');
   RowPos := WhereX;
   Readln(Choice);
   If not (Choice IN['y', 'Y', 'n', 'N', '-']) then
   begin
      GotoXY(RowPos + 5, WhereY - 1);
      writeln('          >>Invalid Choice');
   end
until Choice IN['y', 'Y', 'n', 'N', '-'];
If Choice In['y', 'Y'] then
begin
   GotoXY(WhereX, WhereY - 1);
   ClrEol;
   writeln;
   writeln('                                        New Library Settings');
   writeln('                          ---------------------------------------------');
   write('            Maximum number of Resources can be borrowed by a user: ');
   readln(temp);
   val(temp, tempreal, reva);
   If (tempreal >= 0) and (reva = 0) then
      max_book := round(tempreal);
   write('            Number of Days each borrow lasts: ');
   readln(temp);
   val(temp, tempreal, reva);
   If (tempreal >= 0) and (reva = 0)then
   borrow_days := round(tempreal);
```

```
      write('              Number of Days before the online reserve is cancelled: ');
      readln(temp);
      val(temp, tempreal, reva);
      If (tempreal >= 0) and (reva = 0) then
         request_day := round(tempreal);
      write('              Amount of Fine increased each day(in $): ');
      readln(temp);
      val(temp, tempreal, reva);
      If (tempreal >= 0) and (reva = 0) then
         penal_per_day := tempreal;
      write('              Maximum number of renewal: ');
      readln(temp);
      val(temp, tempreal, reva);
      If (tempreal >= 0) and (reva = 0) then
         max_conti := round(tempreal);
      write('              Maximum number of online reserves made by each student: ');
      readln(temp);
      val(temp, tempreal, reva);
      If (tempreal >= 0) and (reva = 0)then
         max_request := round(tempreal);
      write('              Confirm changes?[Y/N] ');
      readln(temp);
      If (temp = 'y') OR (temp = 'Y') then
      begin
         Overwrite(4);
         write('              >>Library Setting Updated!')
      end
      else
      begin
         Reload;
         write('              >>Previous Settings have been Restored!')
      end;
      readln
   end
end;



Procedure IntaFace(moe, ID_index: integer);
```

```
var NumLate, NumBorr, NumBooked: integer;
      ISBN: string[13];
      choice, Res_typ: char;
      Book_title, Book_Author, Book_publisher: string[50];
      year_pub, BK_ID: string[4];
      Arranged_list: array[1..10000] of integer;

Procedure DirSerch(var NumRes:integer);
var SearchField: array[1..5] of boolean;
      Searchlsit: array[1..10000] of boolean;
      tempBool: boolean;
begin
   For ni := 1 to 5 do
      SearchField[ni] := TRUE;
   For ni := 1 to ISBN_total do
      Searchlsit[ni] := TRUE;
   writeln(' ':53,'Date: ', copy(ToDae, 7, 2), '/', copy(ToDae, 5, 2), '/', copy(ToDae, 1, 4));
   writeln('                ------------------------------------------------------------');
   writeln;
   writeln('                                        Search Resource: Resource Type');
   writeln('                              ---------------------------------------------');
   writeln;
   Res_typ := '1';
   While Res_typ <= '5' do
   begin
      writeln('                                        ', Res_typ, '. ', ResTyplist[Res_typ]);
      Res_typ := succ(Res_typ)
   end;
   writeln;
   Repeat
      write('                                 Your choice: ');
      RowPos := WhereX;
      readln(Res_typ);
      If not (Res_typ IN['1', '2', '3', '4', '5', '-']) then
      begin
         GotoXY(RowPos + 5, WhereY - 1);
         writeln('            >>Invalid Choice');
      end
```

```
until Res_typ IN['1', '2', '3', '4', '5', '-'];
Clrscr;
writeln(' ':53,'Date: ', copy(ToDae, 7, 2), '/', copy(ToDae, 5, 2), '/', copy(ToDae, 1, 4));
writeln('                ------------------------------------------------------------');
writeln;
write('                                                Searching ');
If Res_typ = '-' then
   writeln('Resource')
else
   writeln(ResTyplist[Res_typ]);
writeln('                                 ---------------------------------------------');
writeln;
If Res_typ = '-' then
   SearchField[1] := FALSE;
write('                      Resource name: ');
readln(Book_title);
If Book_title = '-' then
   SearchField[2] := FALSE;
write('                      Resource Owner/Book Author: ');
readln(Book_Author);
If Book_Author = '-' then
   SearchField[3] := FALSE;
write('                      Resource Publisher: ');
readln(Book_publisher);
If Book_publisher = '-' then
   SearchField[4] := FALSE;
write('                      Year of publish: ');
readln(year_pub);
If year_pub = '-' then
   SearchField[5] := FALSE;
ClrScr;
If SearchField[1] then
begin
   nj := 1;
   While nj <= ISBN_total do
   begin
      Searchlsit[nj] := Searchlsit[nj] AND (BookDataArr[nj].Res_typ = Res_typ);
      nj := nj + 1
```

```
      end
   end;
   If SearchField[2] then
   begin
      nj := 1;
      While nj <= ISBN_total do
      begin
         tempBool := False;
         If SearchIsit[nj] then
            For ni := 1 to length(BookDataArr[nj].Book_title) - length(Book_title) + 1 do
               tempBool := tempBool OR (Uppercase(copy(BookDataArr[nj].Book_title, ni, length(Book_title))) =
Uppercase(Book_title));
         SearchIsit[nj] := SearchIsit[nj] AND tempBool;
         nj := nj + 1
      end
   end;
   If SearchField[3] then
   begin
      nj := 1;
      While nj <= ISBN_total do
      begin
         tempBool := False;
         If SearchIsit[nj] then
            For ni := 1 to length(BookDataArr[nj].Book_Author) - length(Book_Author) + 1 do
               tempBool := tempBool OR (UpperCase(copy(BookDataArr[nj].Book_Author, ni, length(Book_Author))) =
Uppercase(Book_Author));
         SearchIsit[nj] := SearchIsit[nj] AND tempBool;
         nj := nj + 1
      end
   end;
   If SearchField[4] then
   begin
      nj := 1;
      While nj <= ISBN_total do
      begin
          tempBool := False;
         If SearchIsit[nj] then
            For ni := 1 to length(BookDataArr[nj].Book_publisher) - length(Book_publisher) + 1 do
```

```
                tempBool := tempBool OR (UpperCase(copy(BookDataArr[nj].Book_publisher, ni,
length(Book_publisher))) = UpperCase(Book_publisher));
          Searchlsit[nj] := Searchlsit[nj] AND tempBool;
          nj := nj + 1
       end
    end;
    If SearchField[5] then
    begin
       nj := 1;
       While nj <= ISBN_total do
       begin
            tempBool := False;
          If Searchlsit[nj] then
             For ni := 1 to length(BookDataArr[nj].year_pub) - length(year_pub) + 1 do
                tempBool := tempBool OR (UpperCase(copy(BookDataArr[nj].year_pub, ni, length(year_pub))) =
UpperCase(year_pub));
          Searchlsit[nj] := Searchlsit[nj] AND tempBool;
          nj := nj + 1
       end
    end;
    ni := 0;
    For nj := 1 to ISBN_total do
       if Searchlsit[nj] then
       begin
          ni := ni + 1;
          Arranged_list[ni] := nj
       end;
    NumRes := ni
end;

Procedure SerNEnquir;
var i, j, numcho, pst, reva: integer;
       InStr: string[4];
begin
    repeat
       j := 1;
       i := 0;
       writeln(' ':53,'Date: ', copy(ToDae, 7, 2), '/', copy(ToDae, 5, 2), '/', copy(ToDae, 1, 4));
```

```
writeln('              ------------------------------------------------------------');
writeln;
writeln('                                          Search Library resource');
writeln('                          ---------------------------------------------');
writeln;
writeln('                                  1. Search by Resource code/ISBN');
writeln('                                  2. Directed Search');
writeln;
repeat
   write('                              Your choice: ');
   RowPos := WhereX;
   readln(choice);
      If not (choice IN['1', '2', '-']) then
      begin
         GotoXY(RowPos + 5, WhereY - 1);
         writeln('           >>Invalid Choice');
      end
until choice IN['1', '2', '-'];
ClrScr;
If choice = '1' then
begin
   repeat
      j := 1;
      i := 0;
      writeln(' ':53,'Date: ', copy(ToDae, 7, 2), '/', copy(ToDae, 5, 2), '/', copy(ToDae, 1, 4));
      writeln('              ------------------------------------------------------------');
      writeln;
      writeln('                                          Search By Resource code/ISBN');
      writeln('                             ---------------------------------------------');
      writeln;
      write('                             Enter A Resource code/ISBN: ');
      readln(ISBN);
      While j <= Bk_total do
      begin
         If BookDataArr[j].ISBN = ISBN then
         begin
            i := i + 1;
            Arranged_list[i] := j
```

```
            end;
            j := j + 1
        end;
        ClrScr
    until (length(ISBN) = 13) OR (ISBN = '-')
end;
If choice = '2' then
begin
    DirSerch(i);
end;
If (i = 0) and (choice <> '-') and (ISBN <> '-') then
begin
    writeln(' ':53,'Date: ', copy(ToDae, 7, 2), '/', copy(ToDae, 5, 2), '/', copy(ToDae, 1, 4));
    writeln('                ------------------------------------------------------------');
    writeln;
    writeln('                                                Search Library resource');
    writeln('                                ---------------------------------------------');
    write('                    >>Resource cannot be found!');
    readln
end;
CLrScr;
If i > 0 then
begin
    repeat
        writeln(' ':53,'Date: ', copy(ToDae, 7, 2), '/', copy(ToDae, 5, 2), '/', copy(ToDae, 1, 4));
        writeln('                ------------------------------------------------------------');
        writeln;
        writeln('                                                Search Results');
        writeln('                                ---------------------------------------------');
        writeln;
        writeln('                                        ISBN                        Resource Name');
        For j := 1 to i do
            writeln('                                ',j,'. ', BookDataArr[Arranged_list[j]].ISBN, '      ',
BookDataArr[Arranged_list[j]].Book_title);
        writeln;
        repeat
            numcho := -1;
            write('                                Resource to enquire: ');
```

```
    RowPos := WhereX;
    readln(InStr);
    RowPos := RowPos + length(InStr);
    val(InStr, numcho, reva);
    If not (((reva = 0) and (numcho > 0) and (numcho <= i)) OR (InStr = '-')) then
    begin
        GotoXY(RowPos, WhereY - 1);
        writeln('            >>Invalid Choice');
    end
until ((reva = 0) and (numcho > 0) and (numcho <= i)) OR (InStr = '-');
ClrScr;
If (numcho > 0) and (numcho <= i) then
begin
    writeln(' ':53,'Date: ', copy(ToDae, 7, 2), '/', copy(ToDae, 5, 2), '/', copy(ToDae, 1, 4));
    writeln('                ------------------------------------------------------------');
    writeln;
    writeln('                                                Resource Information');
    writeln('                              ---------------------------------------------');
    writeln;
    Display_bk_info(Arranged_list[numcho]);
    writeln;
    writeln;
    write('                              >>Press Enter to continue');
    readln;
    ClrScr;
    repeat
        writeln(' ':53,'Date: ', copy(ToDae, 7, 2), '/', copy(ToDae, 5, 2), '/', copy(ToDae, 1, 4));
        writeln('                ------------------------------------------------------------');
        writeln;
        writeln('                                                Resource reserving');
        writeln('                              ---------------------------------------------');
        writeln;
        DisplayBkStat(BookDataArr[Arranged_list[numcho]].ISBN);
        pst := - 1;
        writeln;
        write('                              Resource ID: ');
        readln(BK_ID);
        pst := SearchBookID(BK_ID, 1, 1);
```

```
If (pst = 0) and (BK_ID <> '-') and (StudRecArr[ID_index].Stud_ID[1] <> 'A') then
begin
   write('              >>No Such ID for this resource');
   readln;
   ClrScr
end;
If (BookRecArr[pst].Status <> 0) and (StudRecArr[ID_index].Stud_ID[1] <> 'A') and (BK_ID <> '-') then
begin
   write('              >>This resource is not available for online reserve service');
   readln;
   ClrScr
end;
If (StudRecArr[ID_index].Stud_ID[1] = 'A') and (BK_ID <> '-') then
begin
   write('              >>Switch to user Account for borrowing actions');
   readln;
   ClrScr
end;
If (NumBooked >= max_request) and (BK_ID <> '-') and (StudRecArr[ID_index].Stud_ID[1] <> 'A')then
begin
   write('              >>Maximum number of resource(s) have been reached: ', max_request);
   readln;
   ClrScr
end;
If (NumLate > 0) and (BK_ID <> '-')and (StudRecArr[ID_index].Stud_ID[1] <> 'A') then
begin
   write('              >>Late Return Resource(s) have been found: ',NumLate);
   readln;
   ClrScr
end;
If (pst > 0) and (BookRecArr[pst].Status = 0) and (StudRecArr[ID_index].Stud_ID[1] <> 'A') and (BK_ID
<> '-') and (NumLate = 0) and (NumBooked < max_request) then
begin
   repeat
      writeln('              ------------------------------------------------------------');
      write('              Confirm Reservation?[Y,N] ');
      RowPos := WhereX;
      readln(choice);
```

```
                    If not (choice IN['n', 'N', 'Y', 'y', '-']) then
                    begin
                        GotoXY(RowPos + 5, WhereY - 1);
                        writeln('            >>Invalid Choice');
                    end
                until choice IN['n', 'N', 'Y', 'y', '-']
            end;
            If (choice IN['Y', 'y']) and (NumLate = 0) and (NumBooked < max_request) and (BK_ID <> '-') then
            begin
                BookRecArr[pst].Date := ToDae;
                CverDate(BookRecArr[pst].Date, request_day);
                BookRecArr[pst].Status := -2;
                BookRecArr[pst].Stud_id := StudRecArr[ID_index].Stud_ID;
                NumBooked := NumBooked + 1;
                Overwrite(2);
                writeln('                >>Resource has been reserved for you!');
                write('                >>Please take your borrowed resource within ',request_day,' day(s)');
                readln;
                ClrScr
            end
          until ((pst > 0) and (BookRecArr[pst].Status = 0) and (StudRecArr[ID_index].Stud_ID[1] <> 'A') and (choice
IN['Y', 'y'])) OR (BK_ID = '-')
        end;
        ClrScr;
      until InStr = '-'
    end
  until choice = '-';
  choice := 't';
end;

Procedure InfoNConti;
begin
  Reload;
  repeat
    writeln(' ':53,'Date: ', copy(ToDae, 7, 2), '/', copy(ToDae, 5, 2), '/', copy(ToDae, 1, 4));
    writeln('                ------------------------------------------------------------');
    writeln;
    Stud_info(StudRecArr[ID_index].Stud_ID);
```

```
writeln;
write('                Enter Resource ID From Above: ');
RowPos := WhereX;
readln(BK_ID);
ni := 1;
While (ni <= Bk_total) and (BookRecArr[Borrlist[ni]].Bk_ID <> BK_ID) do
   ni := ni + 1;
If (ni > Bk_total) and (BK_ID <> '-') then
begin
   GotoXY(RowPos + length(BK_ID), WhereY - 1);
   write('          >>Invalid Resource ID');
   readln
end;
If ni <= Bk_total then
begin
   If (NumLate = 0) and (BookRecArr[Borrlist[ni]].Status < max_conti) then
   begin
      writeln('              ------------------------------------------------------------');
      write('              Coninue Borrowing Resource: ');
      write(BookRecArr[Borrlist[ni]].Bk_ID, '?[Y/N] ');
      readln(choice);
      If choice In['Y', 'y'] then
      begin
         BookRecArr[Borrlist[ni]].Date := ToDae;
         CverDate(BookRecArr[Borrlist[ni]].Date, borrow_days);
         BookRecArr[Borrlist[ni]].Status := BookRecArr[Borrlist[ni]].Status + 1;
         BookRecArr[Borrlist[ni]].Stud_id := StudRecArr[ID_index].Stud_ID;
         Overwrite(2);
         write('              >>Continue Borrowing succeeded!');
         readln
      end
   end;
   If NumLate > 0 then
   begin
      writeln('              >>Have Late Return Resource(s)!');
      write('              >>No borrowing action is allowed!');
      readln
   end;
```

```
        If BookRecArr[Borrlist[ni]].Status >= max_conti then
        begin
          write('              >>Maximum continue borrowing reached: ',max_conti);
          readln
        end
      end;
      ni := 1;
      While (ni <= Bk_total) and (BookRecArr[Requetlist[ni]].Bk_ID <> BK_ID) do
        ni := ni + 1;
      If ni <= Bk_total then
      begin
        write('              Cancel Reserving the Resource: ');
        write(BookRecArr[Requetlist[ni]].BK_ID, '?[Y/N] ');
        readln(choice);
        If choice In['Y', 'y'] then
        begin
          BookRecArr[Requetlist[ni]].Stud_ID := '*01234';
          If BookRecArr[Requetlist[ni]].Status = -2 then
            BookRecArr[Requetlist[ni]].Status := 0;
          If BookRecArr[Requetlist[ni]].Status = -3 then
            BookRecArr[Requetlist[ni]].Status := -1;
          BookRecArr[Requetlist[ni]].Date := 'yyyymmdd';
          Overwrite(2);
          write('              >>Cancelling Reserve succeeded!');
          readln
        end;
      end;
      ClrScr
    until BK_ID = '-';
    choice := 't'
end;

Procedure StudInt;
begin
    repeat
      writeln(' ':53,'Date: ', copy(ToDae, 7, 2), '/', copy(ToDae, 5, 2), '/', copy(ToDae, 1, 4));
      writeln('              ------------------------------------------------------------');
      writeln('                    Hello! You have logged in as Library User: ',StudRecArr[ID_index].Stud_ID,'.');
```

```
      writeln('              ------------------------------------------------------------');
      writeln;
      writeln('                                                    User Menu');
      writeln('                              ---------------------------------------------');
      writeln('                                    1. Online Resource Enquiry and Borrowing');
      writeln('                                    2. Access Your library Resource');
      writeln('                                    3. View personal information');
      writeln;
      repeat
         write('                 Your choice: ');
         RowPos := WhereX;
         readln(choice);
         If not (choice IN['1'..'3']) then
         begin
            GotoXY(RowPos + 5, WhereY - 1);
            writeln('           >>Invalid Input!');
         end
      until choice IN['1', '2', '3', '-'];
      ClrScr;
      Case choice of
         '1': SerNEnquir;
         '2': InfoNConti;
         '3': UpStud_Record(2, ID_index)
      end;
      Clrscr
   until choice = '-';
   choice := 't'
end;

Procedure AdminInt;
begin
   repeat
      writeln(' ':53,'Date: ', copy(ToDae, 7, 2), '/', copy(ToDae, 5, 2), '/', copy(ToDae, 1, 4));
      writeln('              ------------------------------------------------------------');
      writeln('                    Hello! You have logged in as Administrator: ',StudRecArr[ID_index].Stud_ID,'.');
      writeln('              ------------------------------------------------------------');
      writeln;
      writeln('                                                    Librarian Menu');
```

```
      writeln'                          ----------------------------------------------');
      writeln('                                1. Borrow and Return Circulation');
      writeln('                                2. Update and Display Resource Status');
      writeln('                                3. Update resource Data');
      writeln('                                4. Search resources');
      writeln('                                5. Manage Student records');
      writeln('                                6. Change Library setting');
      writeln('                                7. Update Personal Information');
      writeln;
      repeat
         write('                Your choice: ');
         Rowpos := WhereX;
         readln(choice);
         If not (choice IN['1'..'7']) then
         begin
            GotoXY(RowPos + 5, WhereY - 1);
            writeln('           >>Invalid Input!');
         end
      until (choice IN['1'..'7']) OR (choice = '-');
      ClrScr;
      Case choice of
         '1': BNRCirc;
         '2': UpBookStat;
         '3': UpBook_Record;
         '4': SerNEnquir;
         '5': UpStud_Record(1, 0);
         '6': OptLibSet;
         '7': UpStud_Record(3, ID_index)
      end;
      ClrScr;
   until choice = '-';
end;

begin
   Stud_Acc(StudRecArr[ID_index].Stud_ID, NumBorr, NumBooked, NumLate);
   If moe = 1 then
      StudInt
   else
```

```
        AdminInt
end;



Procedure WriteTips;
var TipsFile: text;
        Tips: string[55];
        drawlist: array[1..70] of integer;
        draw, upper :integer;
begin
    Assign(TipsFile, 'Tips.txt');
    Reset(TipsFile);
    ni := 0;
    nj := 0;
    upper := 0;
    While not eof(TipsFile) do
    begin
        readln(TipsFile);
        upper := upper + 1;
        drawlist[upper] := upper
    end;
    Randomize;
    For ni := 1 to 3 do
    begin
        draw := random(upper) + 1;
        upper := upper - 1;
        Reset(TipsFile);
        For nj := 1 to drawlist[draw] - 1 do
            readln(TipsFile);
        Readln(TipsFile, Tips);
        writeln('                          >>', Tips);
        For nj := draw to upper do
            drawlist[nj] := drawlist[nj + 1];
    end;
    Close(TipsFile);
end;
```

```
Procedure login;
var pw: string[20];
        key: char;
        stud_id: string[6];
        st_index: integer;
begin
    Reload;
    repeat
        pw := '';
        repeat
            writeln(' ':53,'Date: ', copy(ToDae, 7, 2), '/', copy(ToDae, 5, 2), '/', copy(ToDae, 1, 4));
            writeln('                ------------------------------------------------------------');
            writeln;
            writeln('                            School Library System & Online Library Platform');
            writeln('                              ---------------------------------------------');
            writeln;
            writeln('                            Tips:');
            WriteTips;
            writeln;
            writeln('                              ---------------------------------------------');
            writeln;
            write('                                User ID: ');
            RowPos := WhereX;
            readln(stud_id);
            RowPos := RowPos + length(stud_id);
            st_index := SearchStudID(stud_id, 2, 1);
            If st_index = 0 then
            begin
                GotoXY(RowPos, WhereY - 1);
                write('                  >>Invalid User ID!');
                readln;
                ClrScr
            end
            else
            begin
                writeln;
            end
        until st_index > 0;
```

```
      write('                                Password: ');
      RowPos := WhereX;
      repeat
         key := readkey;
         If (((key >#64) and (key < #91)) OR ((key >#96) and (key < #123)) OR ((key >#47) and (key < #58))) AND
(length(pw) < 20) then
         begin
            pw := pw + key;
            write('*')
         end;
         If (key = #8) and (length(pw) > 0) then
         begin
            pw := copy(pw, 1, length(pw) - 1);
            GotoXY(WhereX - 1, WhereY);
            ClrEol;
         end
      until key = #13;
      If StudRecArr[st_index].pw <> pw then
      begin
         RowPos := RowPos + length(pw);
         GotoXY(RowPos, WhereY);
         write('                  >>Invalid Password!');
         readln
      end;
        ClrScr
   until StudRecArr[st_index].pw = pw;
   If StudRecArr[st_index].Stud_ID[1] = 'A' then
      IntaFace(2, st_index)
   else
      IntaFace(1, st_index)
end;



begin
   ResTyplist['1'] := 'Book';
   ResTyplist['2'] := 'Movie';
   ResTyplist['3'] := 'Magazine';
   ResTyplist['4'] := 'Text Book';
```

```
    ResTyplist['5'] := 'Others';
    BkStatlist[0] := 'On shelf';
    For ni := 1 to 10 do
        BkStatlist[ni] := 'Borrowed';
    BkStatlist[-1] := 'Off shelf';
    BkStatlist[-2] := 'Reserved';    {On_shelf}
    BkStatlist[-3] := 'Reserved';    {Off_shelf}
    repeat
        Reload;
        login
    until 0 > 1;
    ClrScr
end.
```