Systems Analysis and Design

Instructor : Huang, Chuen-Min

Teamwork2 ver.1

Group 6

ID	Name
A10523006	Maggie
A10523049	Peggy
B10423003	Kurumi
B10423029	Bean
B10523020	Kendy
B10523030	Jerry
B10523053	Lynn
M10723001	Joe

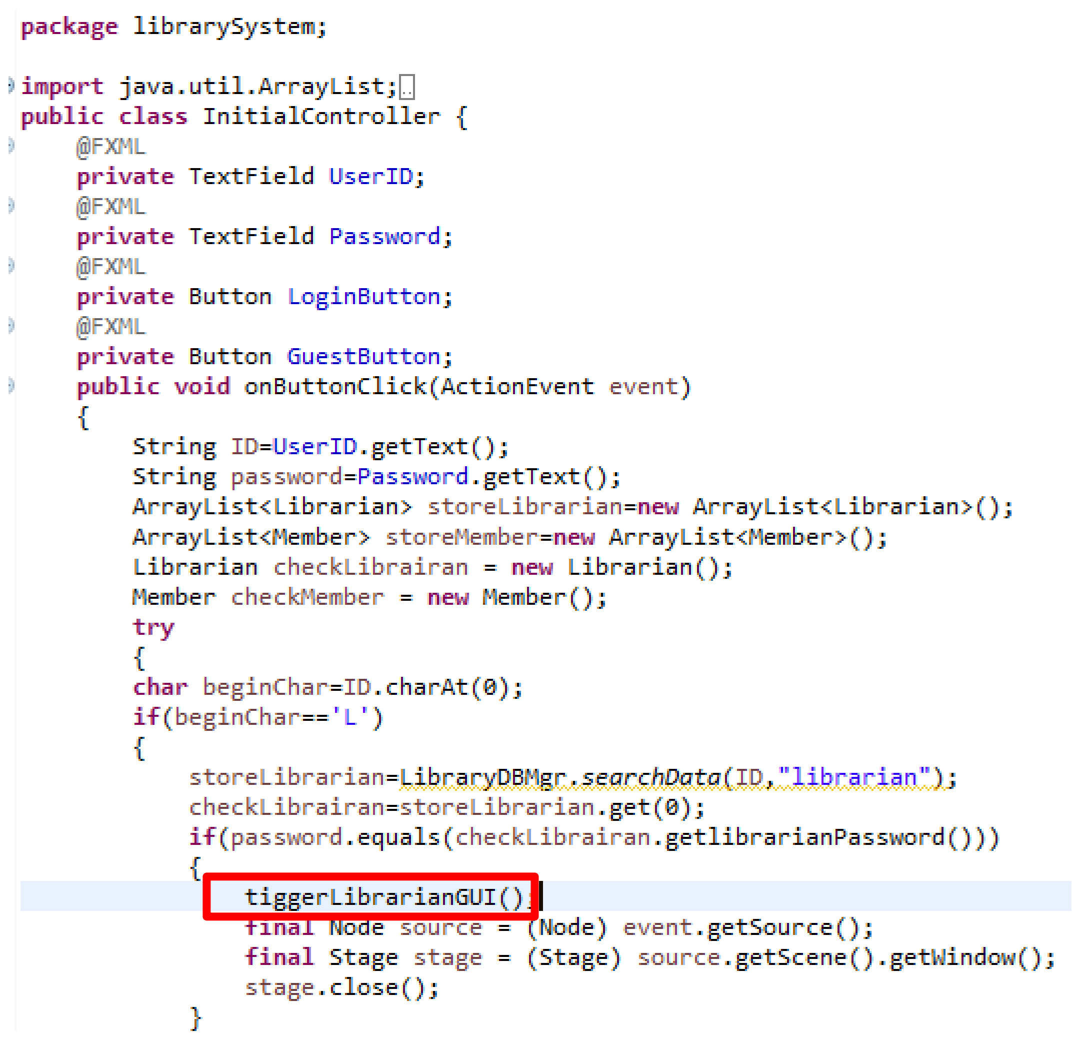
Date 2018/05/29

Content

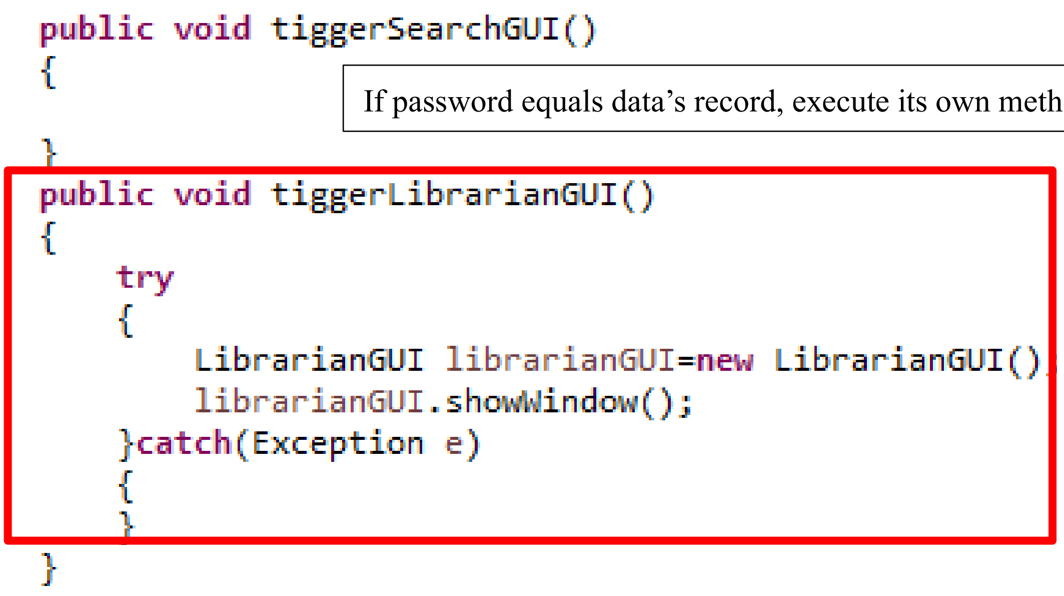
1) Please explain the Law of Demeter (LoD) by using of your project.

1. to itself (O itself)

class InitialController



```
package librarySystem;

import java.util.ArrayList;
public class InitialController {
    @FXML
    private TextField UserID;
    @FXML
    private TextField Password;
    @FXML
    private Button LoginButton;
    @FXML
    private Button GuestButton;
    public void onButtonClick(ActionEvent event)
    {
        String ID=UserID.getText();
        String password=Password.getText();
        ArrayList<Librarian> storeLibrarian=new ArrayList<Librarian>();
        ArrayList<Member> storeMember=new ArrayList<Member>();
        Librarian checkLibrairan = new Librarian();
        Member checkMember = new Member();
        try
        {
        char beginChar=ID.charAt(0);
        if(beginChar=='L')
        {
            storeLibrarian=LibraryDBMgr.searchData(ID,"librarian");
            checkLibrairan=storeLibrarian.get(0);
            if(password.equals(checkLibrairan.getlibrarianPassword()))
            {
                tiggerLibrarianGUI();
                final Node source = (Node) event.getSource();
                final Stage stage = (Stage) source.getScene().getWindow();
                stage.close();
            }
```



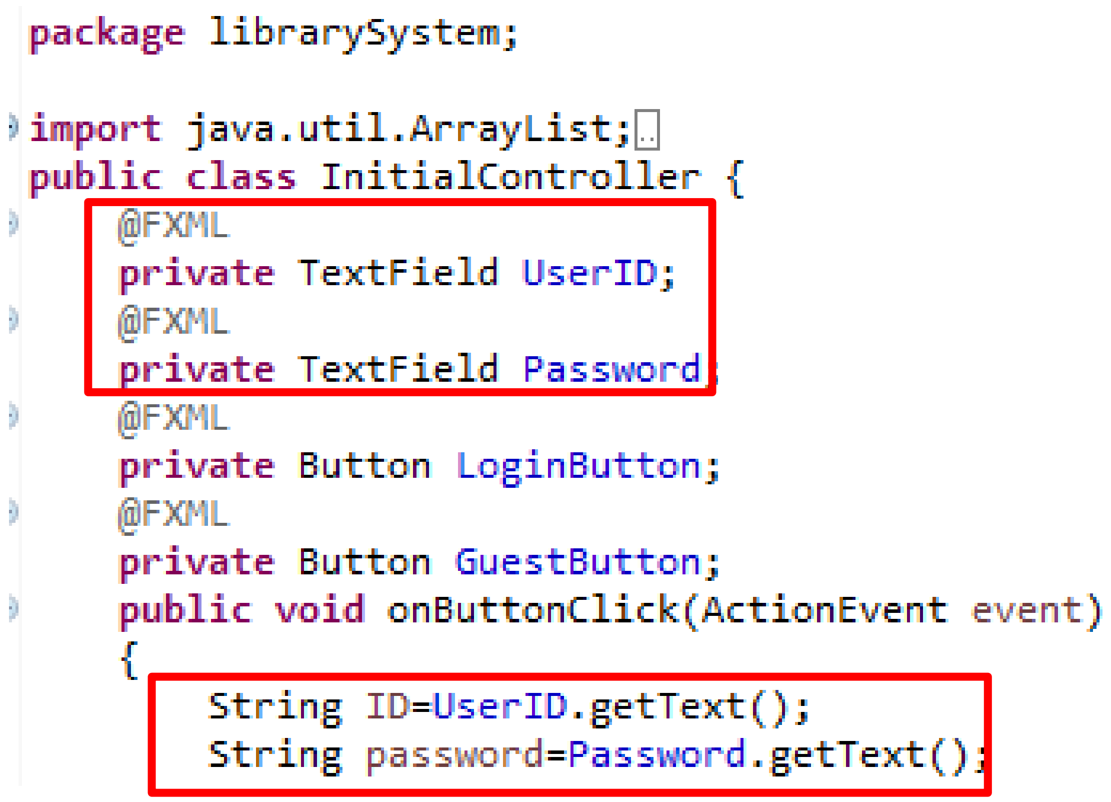
```
public void tiggerSearchGUI()
{

}
public void tiggerLibrarianGUI()
{
    try
    {
        LibrarianGUI librarianGUI=new LibrarianGUI();
        librarianGUI.showWindow();
    }catch(Exception e)
    {
    }
}
```

If password equals data's record, execute its own method.

2. to objects contained in attributes of itself or a superclass (Any objects created/instantiated within M)

Class InitialController create object UserID & Password itself, then it can use getText() in its method.

class InitialController



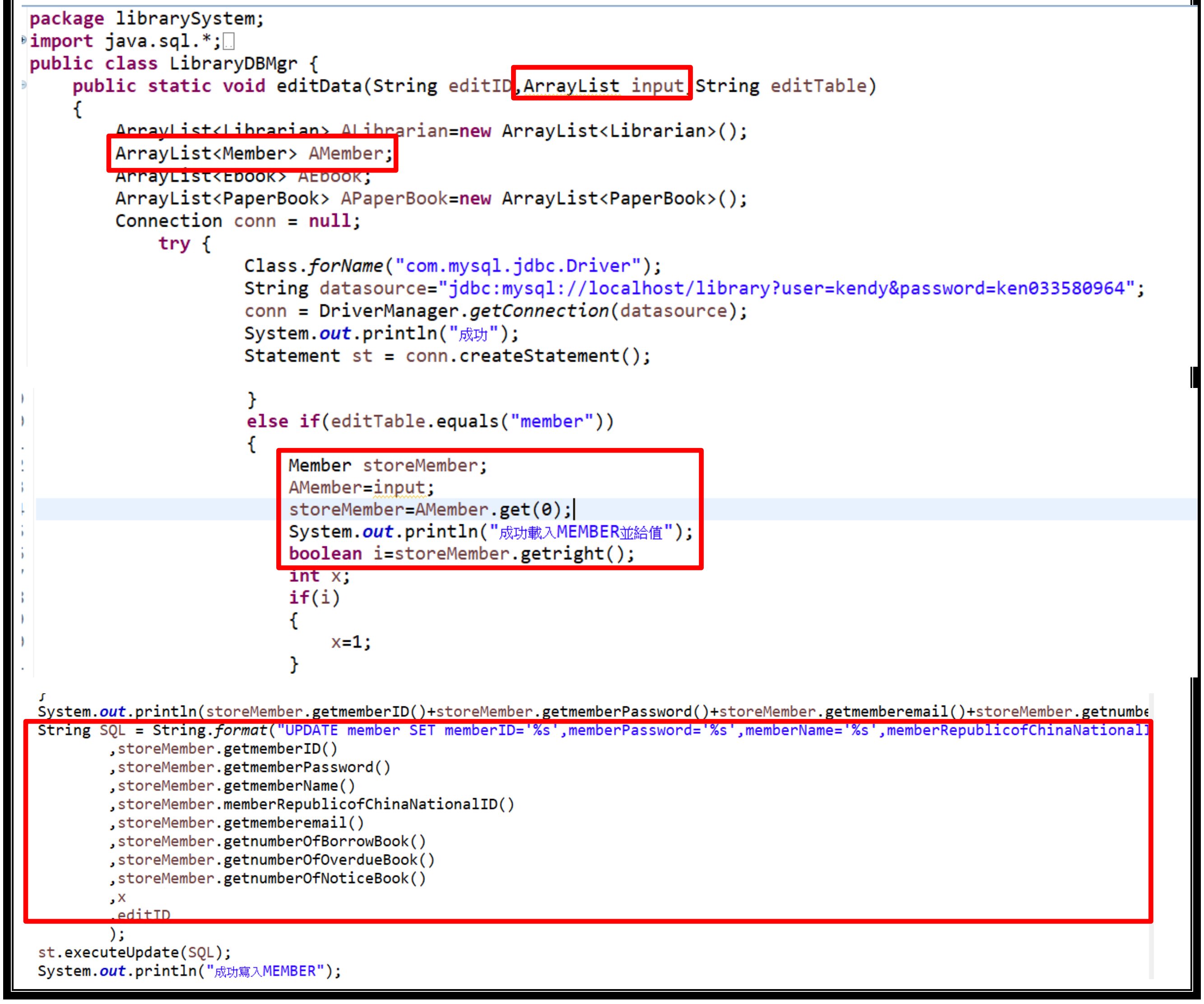
```
package librarySystem;

import java.util.ArrayList;
public class InitialController {
    @FXML
    private TextField UserID;
    @FXML
    private TextField Password;
    @FXML
    private Button LoginButton;
    @FXML
    private Button GuestButton;
    public void onButtonClick(ActionEvent event)
    {
        String ID=UserID.getText();
        String password=Password.getText();
```

3. to an object that is passed as a parameter to the method (M's parameters)

Method editdata() gets an ArrayList that named input. Then method editdata() changes the ArrayList's name as AMember.

Method editdata() has an object Member, its name is storeMember. Then ArrayList<Member> AMember equals to input (let input's type to Member) and get first AMember's Array to storeMember.

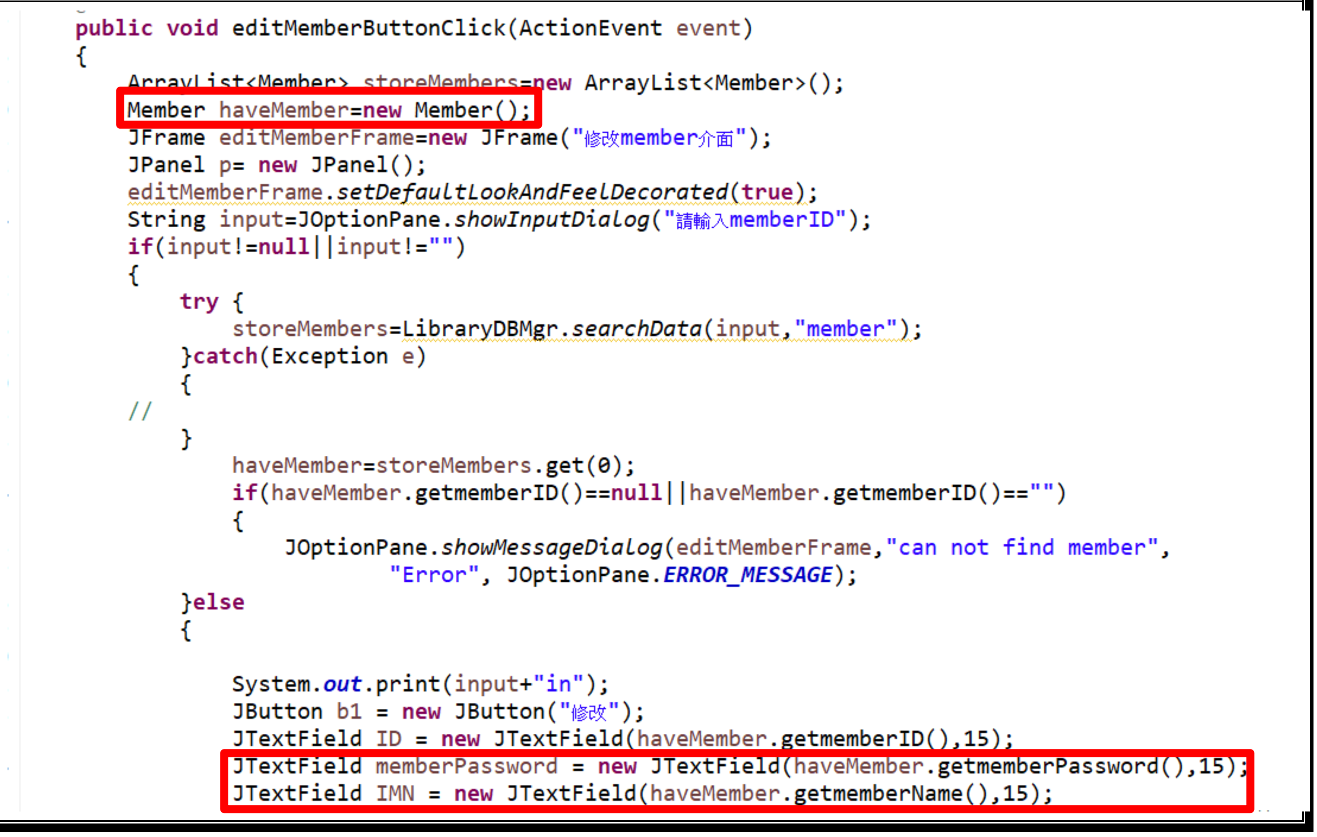
class LibrarianDBMgr



```
package librarySystem;
import java.sql.*;
public class LibraryDBMgr {
    public static void editData(String editID,ArrayList input,String editTable)
    {
        ArrayList<Librarian> ALibrarian=new ArrayList<Librarian>();
        ArrayList<Member> AMember;
        ArrayList<Ebook> AEbook;
        ArrayList<PaperBook> APaperBook=new ArrayList<PaperBook>();
        Connection conn = null;
            try {
                    Class.forName("com.mysql.jdbc.Driver");
                    String datasource="jdbc:mysql://localhost/library?user=kendy&password=ken033580964";
                    conn = DriverManager.getConnection(datasource);
                    System.out.println("成功");
                    Statement st = conn.createStatement();

                    }
                    else if(editTable.equals("member"))
                    {
                        Member storeMember;
                        AMember=input;
                        storeMember=AMember.get(0);
                        System.out.println("成功載入MEMBER並給值");
                        boolean i=storeMember.getright();
                        int x;
                        if(i)
                        {
                            x=1;
                        }

System.out.println(storeMember.getmemberID()+storeMember.getmemberPassword()+storeMember.getmemberemail()+storeMember.getnumbe
String SQL = String.format("UPDATE member SET memberID='%s',memberPassword='%s',memberName='%s',memberRepublicofChinaNationalI
        ,storeMember.getmemberID()
        ,storeMember.getmemberPassword()
        ,storeMember.getmemberName()
        ,storeMember.memberRepublicofChinaNationalID()
        ,storeMember.getmemberemail()
        ,storeMember.getnumberOfBorrowBook()
        ,storeMember.getnumberOfOverdueBook()
        ,storeMember.getnumberOfNoticeBook()
        ,x
        ,editID
        );
st.executeUpdate(SQL);
System.out.println("成功寫入MEMBER");
```

4. to an object that is created by the method (O’s direct component objects)

When addMemberButtonClick method execute, it will create object and use its method.

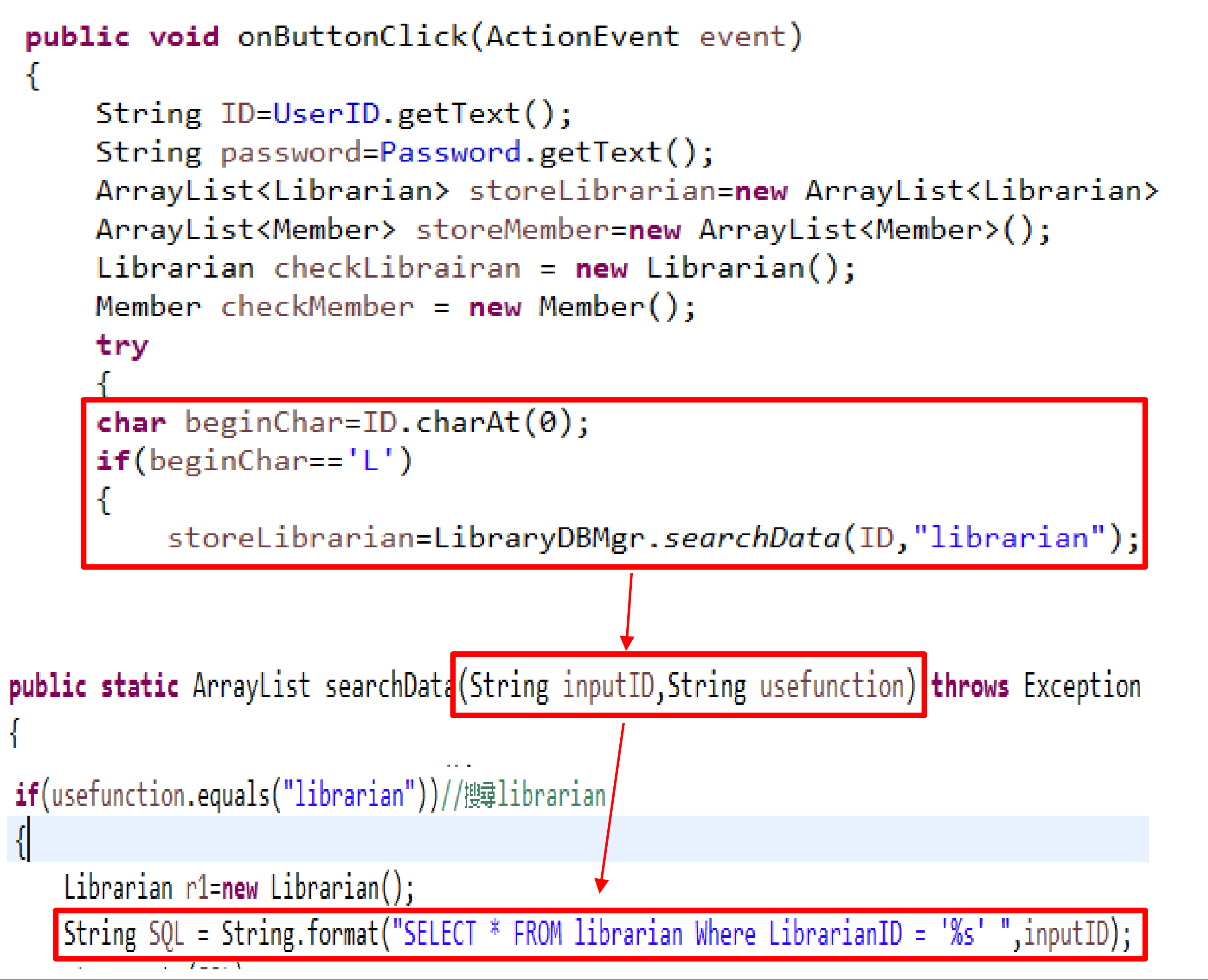
class LibrarianController


```java
public void editMemberButtonClick(ActionEvent event)
{
    ArrayList<Member> storeMembers=new ArrayList<Member>();
    Member haveMember=new Member();
    JFrame editMemberFrame=new JFrame("修改member介面");
    JPanel p= new JPanel();
    editMemberFrame.setDefaultLookAndFeelDecorated(true);
    String input=JOptionPane.showInputDialog("請輸入memberID");
    if(input!=null||input!="")
    {
        try {
            storeMembers=LibraryDBMgr.searchData(input,"member");
        }catch(Exception e)
        {
    //
        }
            haveMember=storeMembers.get(0);
            if(haveMember.getmemberID()==null||haveMember.getmemberID()=="")
            {
                JOptionPane.showMessageDialog(editMemberFrame,"can not find member",
                        "Error", JOptionPane.ERROR_MESSAGE);
        }else
        {

            System.out.print(input+"in");
            JButton b1 = new JButton("修改");
            JTextField ID = new JTextField(haveMember.getmemberID(),15);
            JTextField memberPassword = new JTextField(haveMember.getmemberPassword(),15);
            JTextField IMN = new JTextField(haveMember.getmemberName(),15);
```

2) There are six (or seven) types of interaction coupling, each falling on different parts of a good-to-bad continuum. Choose three pieces of your project to describe what types of the coupling they belong to.

1. Control Coupling

If someone wants to login this system, we need to check user's inputted ID. If the user uses librarian's ID, then method onButtonClick() will send two variables to method searchData(). One is user's ID, the other is what we need to find from database.

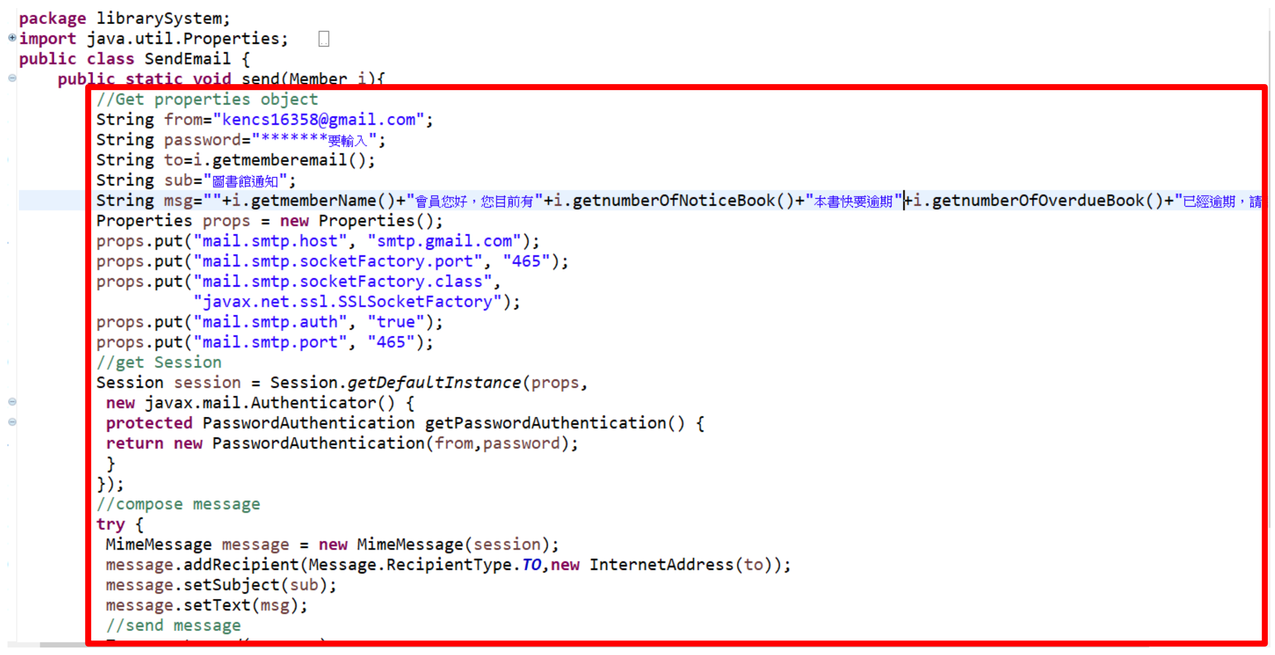
class InitialController


```
public void onButtonClick(ActionEvent event)
{
    String ID=UserID.getText();
    String password=Password.getText();
    ArrayList<Librarian> storeLibrarian=new ArrayList<Librarian>
    ArrayList<Member> storeMember=new ArrayList<Member>();
    Librarian checkLibrairan = new Librarian();
    Member checkMember = new Member();
    try
    {
    char beginChar=ID.charAt(0);
    if(beginChar=='L')
    {
        storeLibrarian=LibraryDBMgr.searchData(ID,"librarian");


public static ArrayList searchData(String inputID,String usefunction) throws Exception
{
 if(usefunction.equals("librarian"))//搜尋librarian
 {
     Librarian r1=new Librarian();
     String SQL = String.format("SELECT * FROM librarian Where LibrarianID = '%s' ",inputID);
```

2. Interaction, Data Coupling

This is a class for sending an E-mail to member. All the parameter is decided by basic variable. And in this class, it doesn't need to call other class. It just handles send E-mail by its own.

class SendEmail

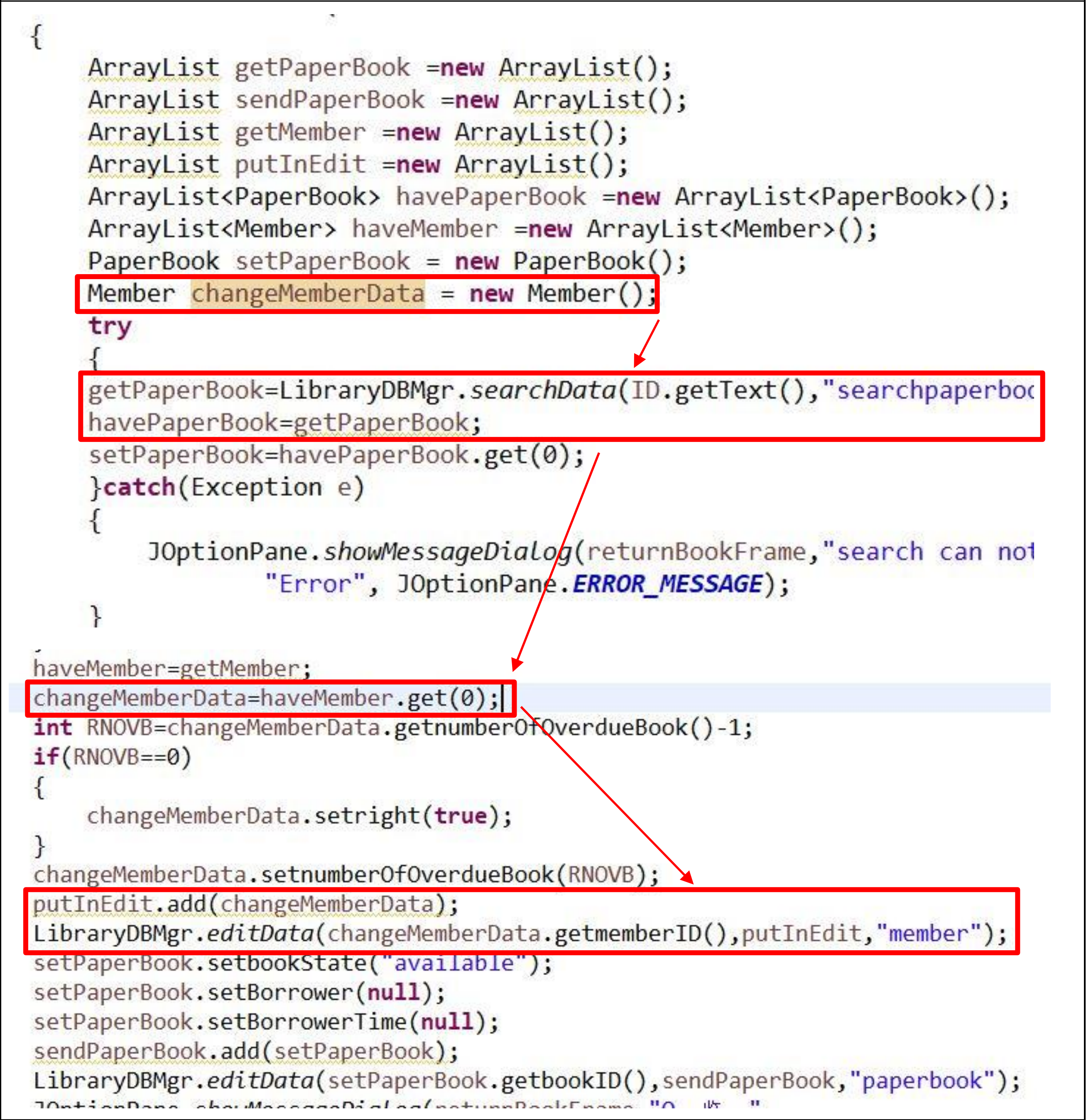


```java
package librarySystem;
import java.util.Properties;
public class SendEmail {
    public static void send(Member i){
        //Get properties object
        String from="kencs16358@gmail.com";
        String password="*******要輸入";
        String to=i.getmemberemail();
        String sub="圖書館通知";
        String msg=""+i.getmemberName()+"會員您好，您目前有"+i.getnumberOfNoticeBook()+"本書快要逾期"+i.getnumberOfOverdueBook()+"已經逾期，請
        Properties props = new Properties();
        props.put("mail.smtp.host", "smtp.gmail.com");
        props.put("mail.smtp.socketFactory.port", "465");
        props.put("mail.smtp.socketFactory.class",
                  "javax.net.ssl.SSLSocketFactory");
        props.put("mail.smtp.auth", "true");
        props.put("mail.smtp.port", "465");
        //get Session
        Session session = Session.getDefaultInstance(props,
         new javax.mail.Authenticator() {
         protected PasswordAuthentication getPasswordAuthentication() {
         return new PasswordAuthentication(from,password);
         }
        });
        //compose message
        try {
         MimeMessage message = new MimeMessage(session);
         message.addRecipient(Message.RecipientType.TO,new InternetAddress(to));
         message.setSubject(sub);
         message.setText(msg);
         //send message
```

3. Stamp Coupling

System uses class LibraryDBMgr to change paperbook's state. First of all, system creates member object changeMemberData. System will find a book it wants to change from database, and puts this data in variable havaPaperBook. Then use LibraryDBMgr's method editData() to change book's state. The member object changeMemberData will be its parameter.

class LibrarianController



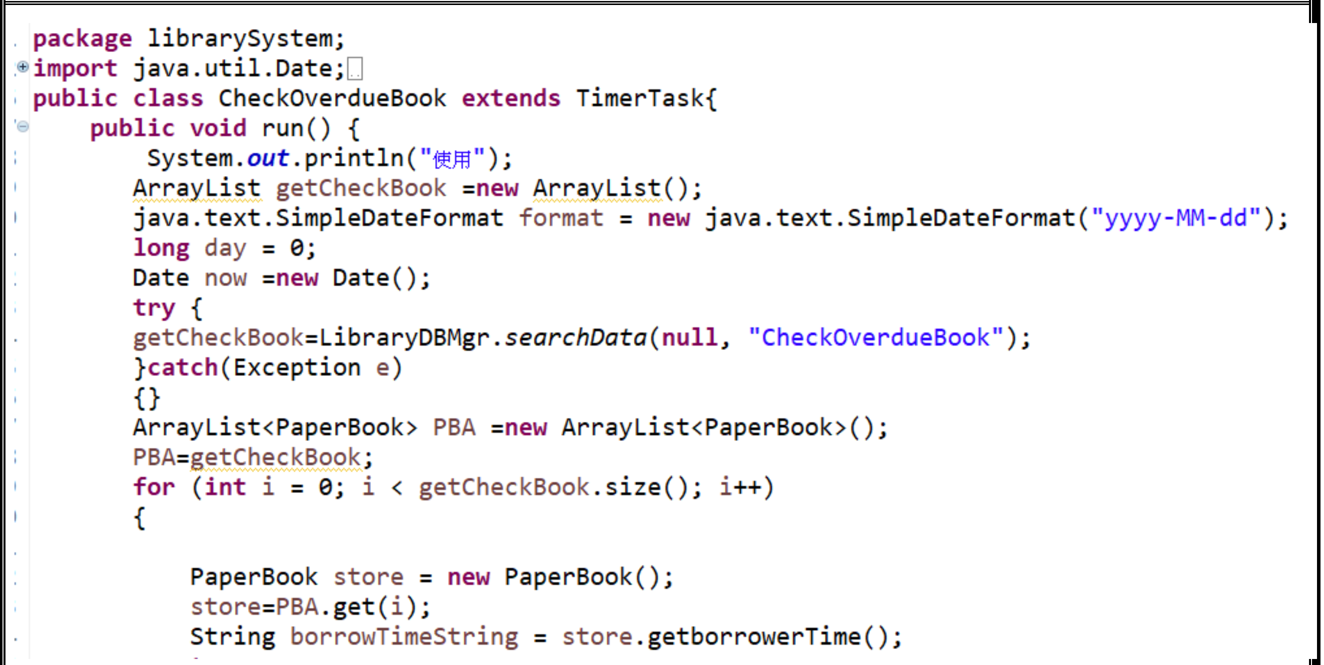
```
{
    ArrayList getPaperBook =new ArrayList();
    ArrayList sendPaperBook =new ArrayList();
    ArrayList getMember =new ArrayList();
    ArrayList putInEdit =new ArrayList();
    ArrayList<PaperBook> havePaperBook =new ArrayList<PaperBook>();
    ArrayList<Member> haveMember =new ArrayList<Member>();
    PaperBook setPaperBook = new PaperBook();
    Member changeMemberData = new Member();
    try
    {
    getPaperBook=LibraryDBMgr.searchData(ID.getText(),"searchpaperboc
    havePaperBook=getPaperBook;
    setPaperBook=havePaperBook.get(0);
    }catch(Exception e)
    {
        JOptionPane.showMessageDialog(returnBookFrame,"search can not
                "Error", JOptionPane.ERROR_MESSAGE);
    }

haveMember=getMember;
changeMemberData=haveMember.get(0);
int RNOVB=changeMemberData.getnumberOfOverdueBook()-1;
if(RNOVB==0)
{
    changeMemberData.setright(true);
}
changeMemberData.setnumberOfOverdueBook(RNOVB);
putInEdit.add(changeMemberData);
LibraryDBMgr.editData(changeMemberData.getmemberID(),putInEdit,"member");
setPaperBook.setbookState("available");
setPaperBook.setBorrower(null);
setPaperBook.setBorrowerTime(null);
sendPaperBook.add(setPaperBook);
LibraryDBMgr.editData(setPaperBook.getbookID(),sendPaperBook,"paperbook");
```

3) There are seven types of method cohesion, choose three pieces of your project to describe what types of the cohesion they belong to.

1. Function Cohesion

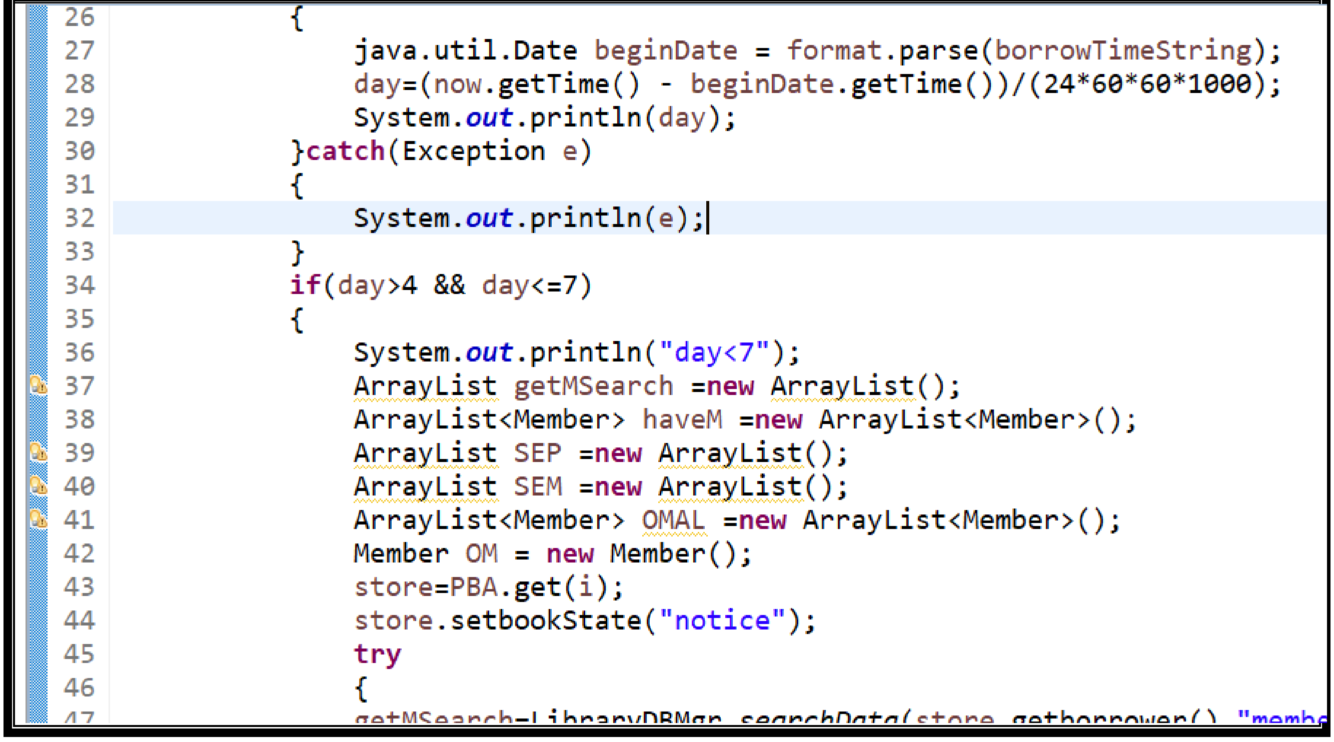
This method's only function is to check book borrow situation.

class CheckOverdueBook - 1



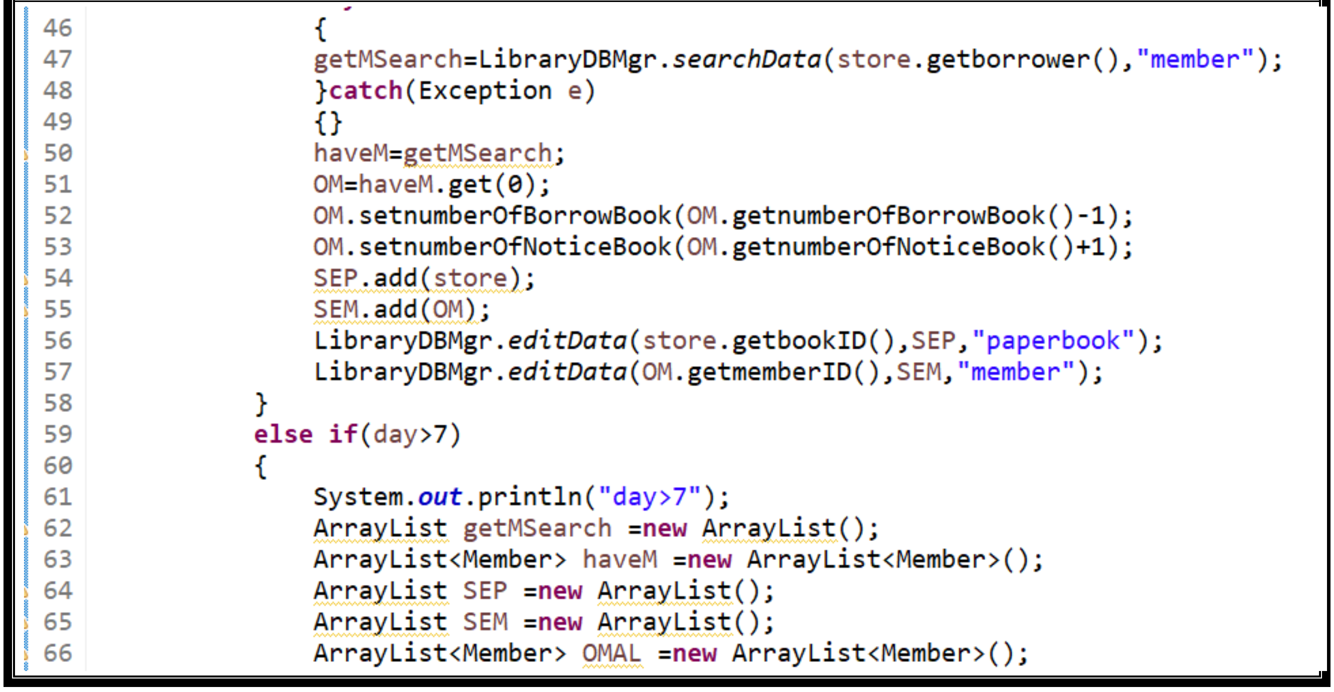
```
package librarySystem;
import java.util.Date;
public class CheckOverdueBook extends TimerTask{
    public void run() {
        System.out.println("使用");
       ArrayList getCheckBook =new ArrayList();
       java.text.SimpleDateFormat format = new java.text.SimpleDateFormat("yyyy-MM-dd");
       long day = 0;
       Date now =new Date();
       try {
       getCheckBook=LibraryDBMgr.searchData(null, "CheckOverdueBook");
       }catch(Exception e)
       {}
       ArrayList<PaperBook> PBA =new ArrayList<PaperBook>();
       PBA=getCheckBook;
       for (int i = 0; i < getCheckBook.size(); i++)
       {

           PaperBook store = new PaperBook();
           store=PBA.get(i);
           String borrowTimeString = store.getborrowerTime();
```

class CheckOverdueBook - 2



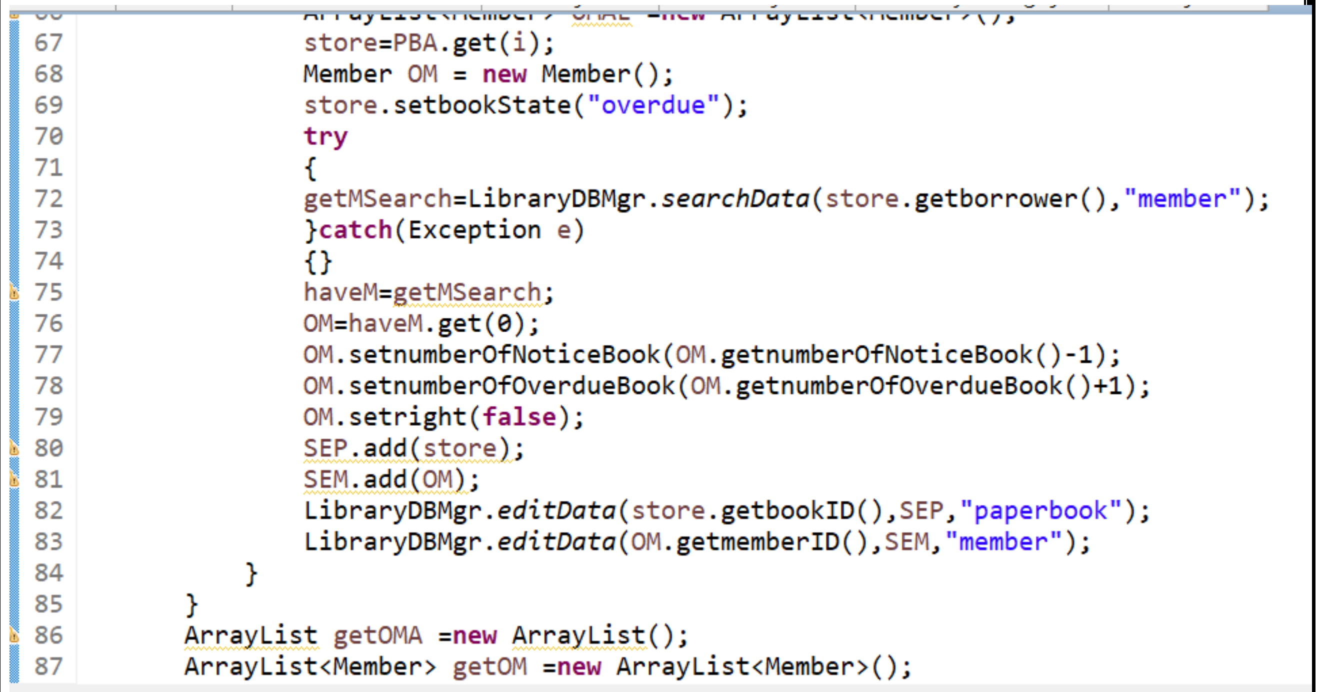
```
            {
                java.util.Date beginDate = format.parse(borrowTimeString);
                day=(now.getTime() - beginDate.getTime())/(24*60*60*1000);
                System.out.println(day);
            }catch(Exception e)
            {
                System.out.println(e);
            }
            if(day>4 && day<=7)
            {
                System.out.println("day<7");
                ArrayList getMSearch =new ArrayList();
                ArrayList<Member> haveM =new ArrayList<Member>();
                ArrayList SEP =new ArrayList();
                ArrayList SEM =new ArrayList();
                ArrayList<Member> OMAL =new ArrayList<Member>();
                Member OM = new Member();
                store=PBA.get(i);
                store.setbookState("notice");
                try
                {
                getMSearch=LibraryDBMgr.searchData(store.getborrower(), "membe
```

class CheckOverdueBook - 3



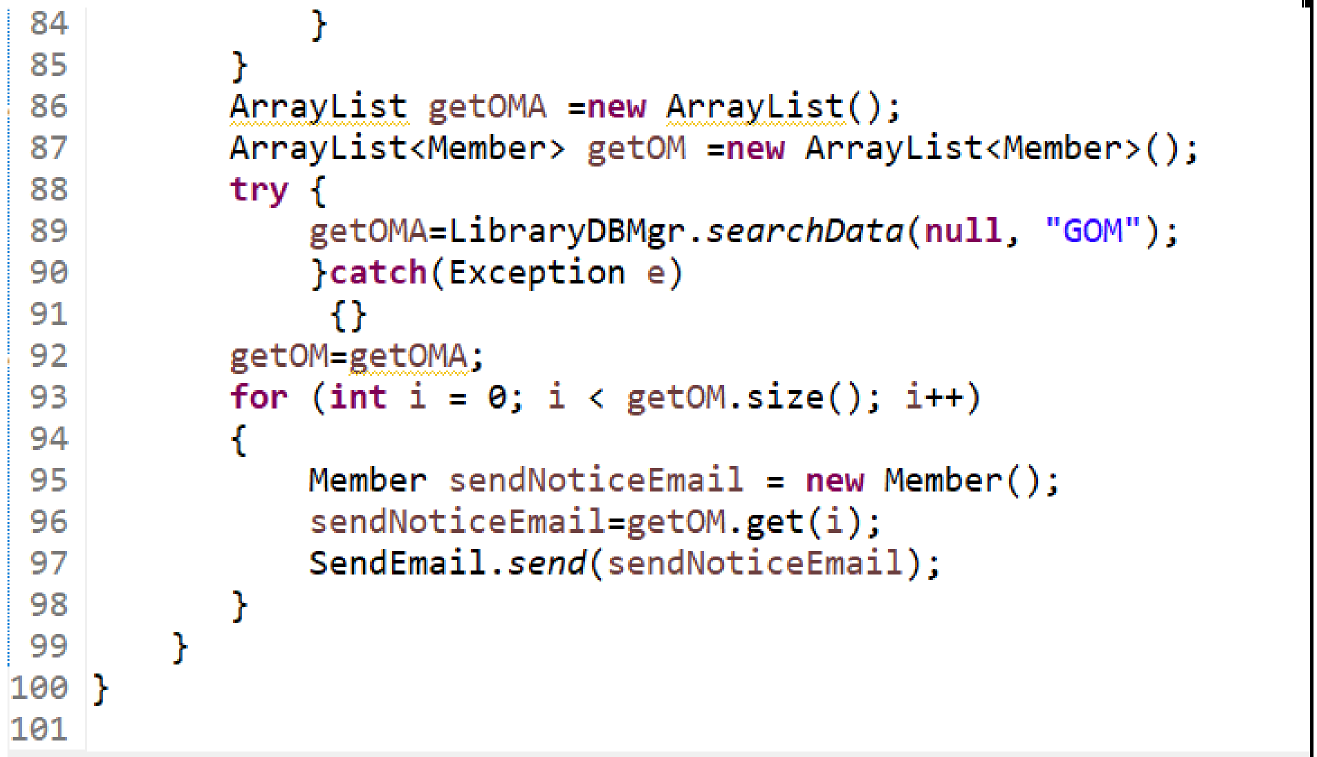
```java
            {
            getMSearch=LibraryDBMgr.searchData(store.getborrower(),"member");
            }catch(Exception e)
            {}
            haveM=getMSearch;
            OM=haveM.get(0);
            OM.setnumberOfBorrowBook(OM.getnumberOfBorrowBook()-1);
            OM.setnumberOfNoticeBook(OM.getnumberOfNoticeBook()+1);
            SEP.add(store);
            SEM.add(OM);
            LibraryDBMgr.editData(store.getbookID(),SEP,"paperbook");
            LibraryDBMgr.editData(OM.getmemberID(),SEM,"member");
        }
        else if(day>7)
        {
            System.out.println("day>7");
            ArrayList getMSearch =new ArrayList();
            ArrayList<Member> haveM =new ArrayList<Member>();
            ArrayList SEP =new ArrayList();
            ArrayList SEM =new ArrayList();
            ArrayList<Member> OMAL =new ArrayList<Member>();
```

class CheckOverdueBook - 4



```java
            store=PBA.get(i);
            Member OM = new Member();
            store.setbookState("overdue");
            try
            {
            getMSearch=LibraryDBMgr.searchData(store.getborrower(),"member");
            }catch(Exception e)
            {}
            haveM=getMSearch;
            OM=haveM.get(0);
            OM.setnumberOfNoticeBook(OM.getnumberOfNoticeBook()-1);
            OM.setnumberOfOverdueBook(OM.getnumberOfOverdueBook()+1);
            OM.setright(false);
            SEP.add(store);
            SEM.add(OM);
            LibraryDBMgr.editData(store.getbookID(),SEP,"paperbook");
            LibraryDBMgr.editData(OM.getmemberID(),SEM,"member");
        }
    }
    ArrayList getOMA =new ArrayList();
    ArrayList<Member> getOM =new ArrayList<Member>();
```

class CheckOverdueBook - 5

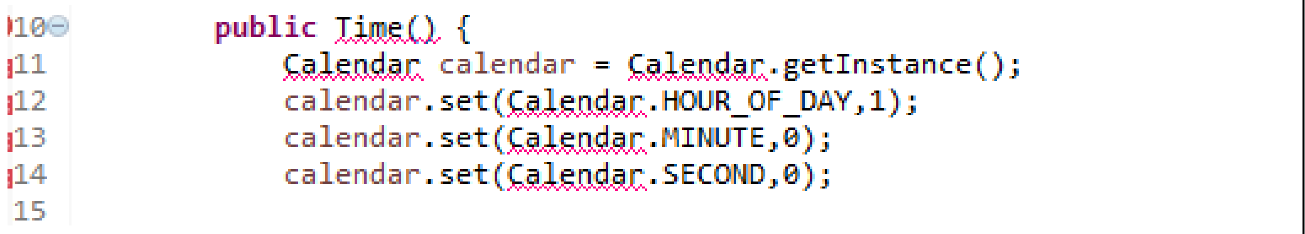


```
            }
        }
        ArrayList getOMA =new ArrayList();
        ArrayList<Member> getOM =new ArrayList<Member>();
        try {
            getOMA=LibraryDBMgr.searchData(null, "GOM");
            }catch(Exception e)
             {}
        getOM=getOMA;
        for (int i = 0; i < getOM.size(); i++)
        {
            Member sendNoticeEmail = new Member();
            sendNoticeEmail=getOM.get(i);
            SendEmail.send(sendNoticeEmail);
        }
    }
}
```

2. Temporal Cohesion

Member functions are called at the same time.

class Time

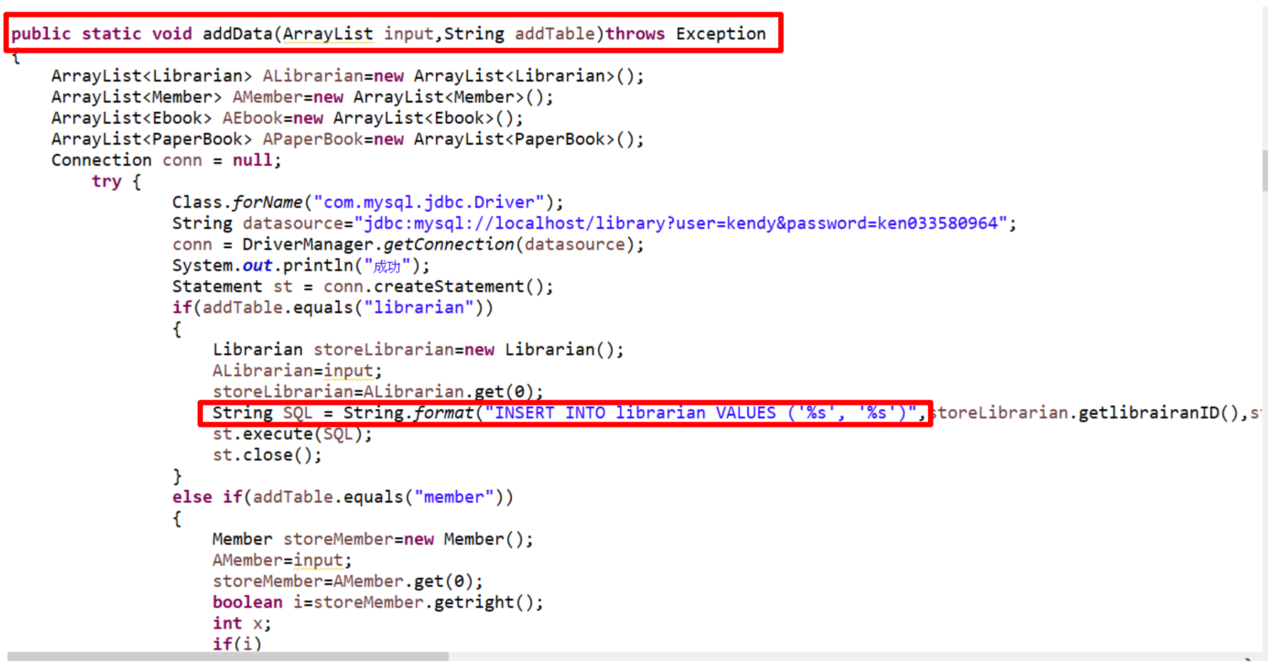


```
    public Time() {
        Calendar calendar = Calendar.getInstance();
        calendar.set(Calendar.HOUR_OF_DAY,1);
        calendar.set(Calendar.MINUTE,0);
        calendar.set(Calendar.SECOND,0);
```

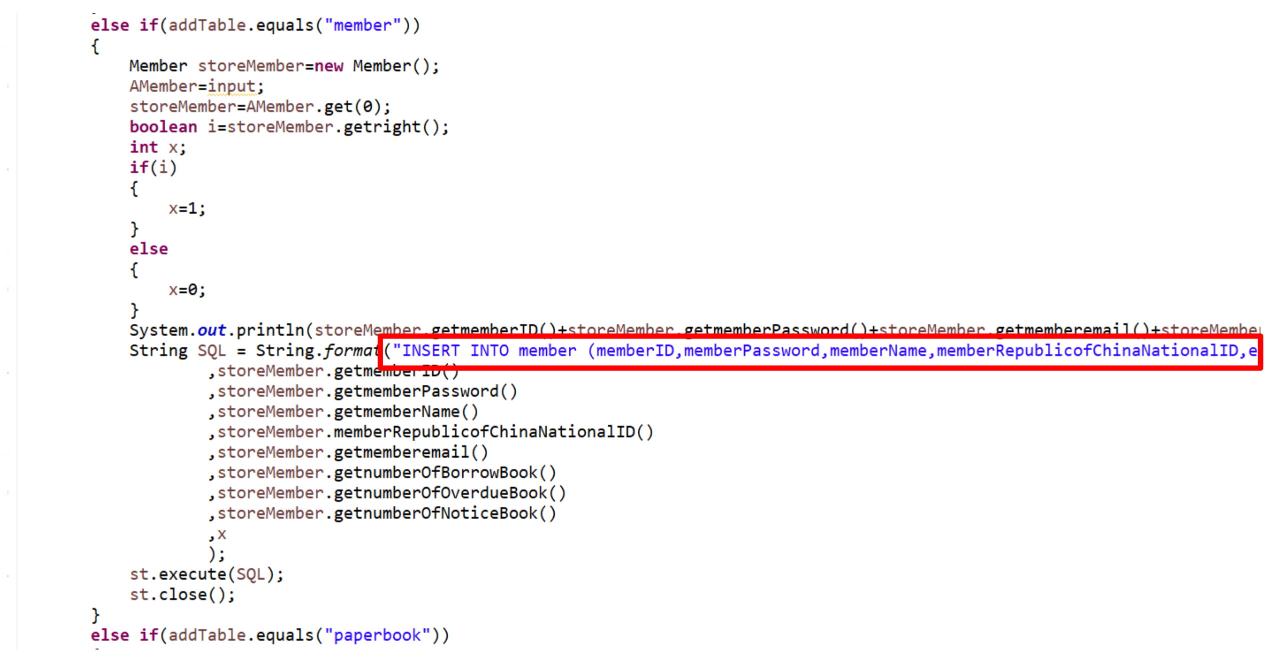
3. Logical Chhesion

Method addDate() can add four data there are member, librarian, paperbook, ebook.

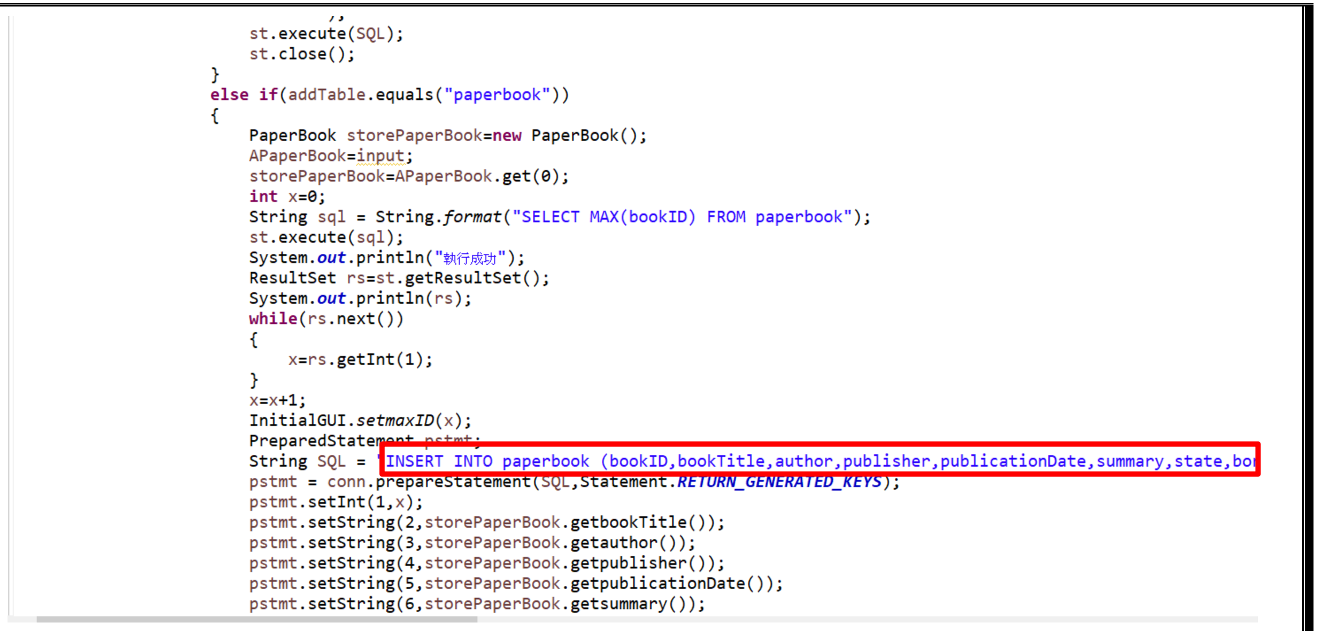
class LibraryDBMgr



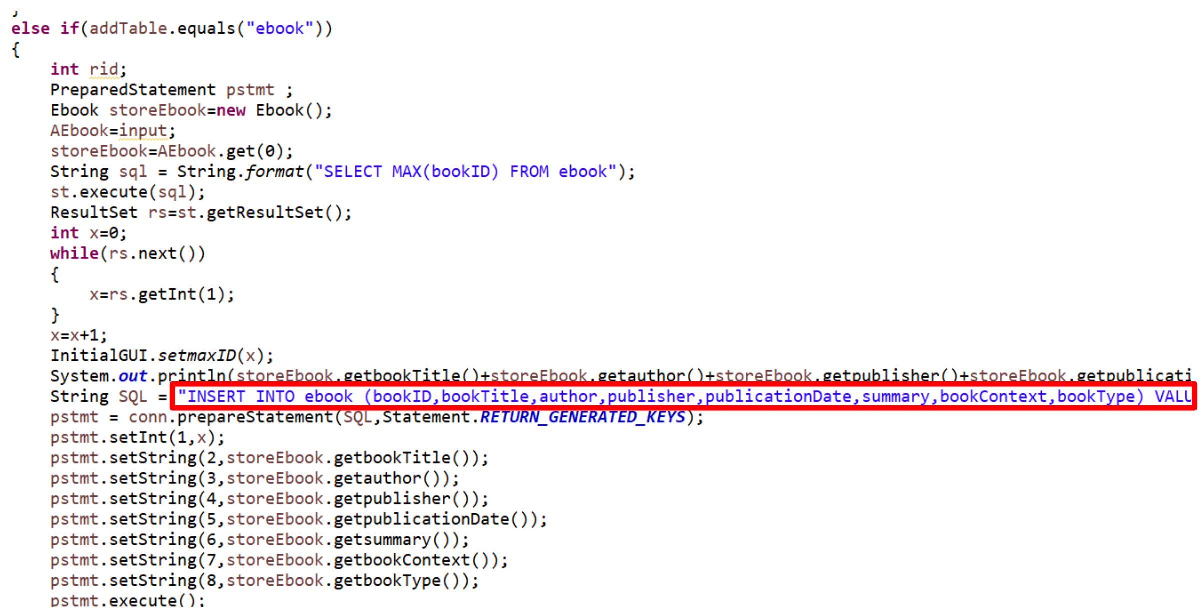
```
public static void addData(ArrayList input,String addTable)throws Exception
{
    ArrayList<Librarian> ALibrarian=new ArrayList<Librarian>();
    ArrayList<Member> AMember=new ArrayList<Member>();
    ArrayList<Ebook> AEbook=new ArrayList<Ebook>();
    ArrayList<PaperBook> APaperBook=new ArrayList<PaperBook>();
    Connection conn = null;
        try {
                Class.forName("com.mysql.jdbc.Driver");
                String datasource="jdbc:mysql://localhost/library?user=kendy&password=ken033580964";
                conn = DriverManager.getConnection(datasource);
                System.out.println("成功");
                Statement st = conn.createStatement();
                if(addTable.equals("librarian"))
                {
                    Librarian storeLibrarian=new Librarian();
                    ALibrarian=input;
                    storeLibrarian=ALibrarian.get(0);
                    String SQL = String.format("INSERT INTO librarian VALUES ('%s', '%s')",storeLibrarian.getlibrairanID(),s
                    st.execute(SQL);
                    st.close();
                }
                else if(addTable.equals("member"))
                {
                    Member storeMember=new Member();
                    AMember=input;
                    storeMember=AMember.get(0);
                    boolean i=storeMember.getright();
                    int x;
                    if(i)
```



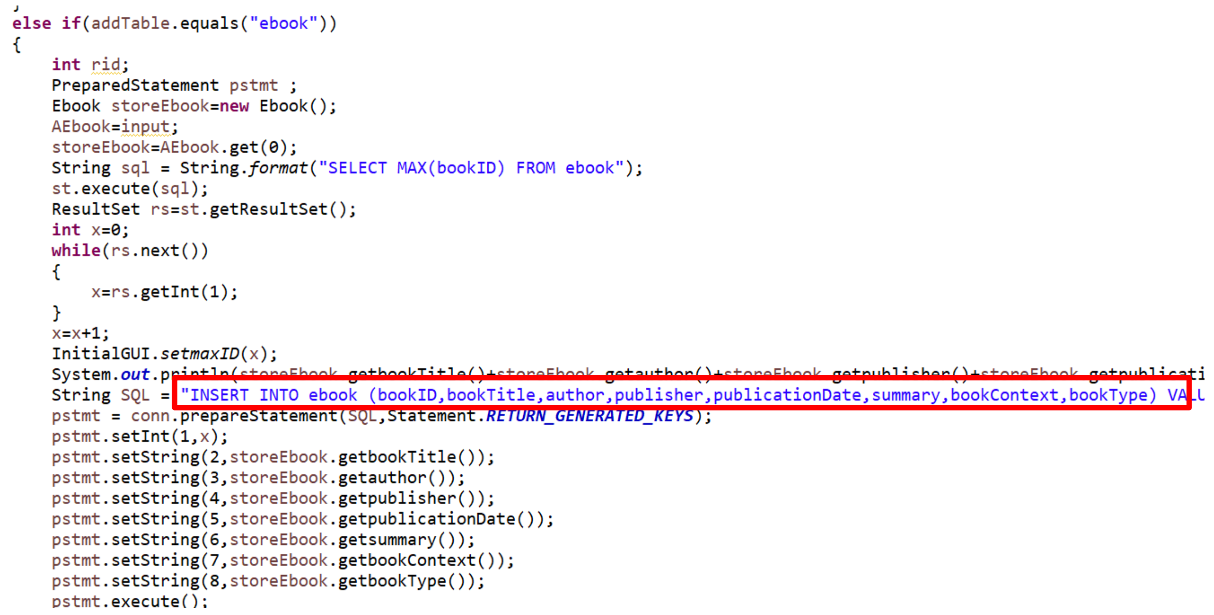
```
        else if(addTable.equals("member"))
        {
            Member storeMember=new Member();
            AMember=input;
            storeMember=AMember.get(0);
            boolean i=storeMember.getright();
            int x;
            if(i)
            {
                x=1;
            }
            else
            {
                x=0;
            }
            System.out.println(storeMember.getmemberID()+storeMember.getmemberPassword()+storeMember.getmemberemail()+storeMembe
            String SQL = String.format("INSERT INTO member (memberID,memberPassword,memberName,memberRepublicofChinaNationalID,e
                    ,storeMember.getmemberID()
                    ,storeMember.getmemberPassword()
                    ,storeMember.getmemberName()
                    ,storeMember.memberRepublicofChinaNationalID()
                    ,storeMember.getmemberemail()
                    ,storeMember.getnumberOfBorrowBook()
                    ,storeMember.getnumberOfOverdueBook()
                    ,storeMember.getnumberOfNoticeBook()
                    ,x
                    );
            st.execute(SQL);
            st.close();
        }
        else if(addTable.equals("paperbook"))
```



```
            st.execute(SQL);
            st.close();
        }
        else if(addTable.equals("paperbook"))
        {
            PaperBook storePaperBook=new PaperBook();
            APaperBook=input;
            storePaperBook=APaperBook.get(0);
            int x=0;
            String sql = String.format("SELECT MAX(bookID) FROM paperbook");
            st.execute(sql);
            System.out.println("執行成功");
            ResultSet rs=st.getResultSet();
            System.out.println(rs);
            while(rs.next())
            {
                x=rs.getInt(1);
            }
            x=x+1;
            InitialGUI.setmaxID(x);
            PreparedStatement pstmt;
            String SQL = "INSERT INTO paperbook (bookID,bookTitle,author,publisher,publicationDate,summary,state,bo
            pstmt = conn.prepareStatement(SQL,Statement.RETURN_GENERATED_KEYS);
            pstmt.setInt(1,x);
            pstmt.setString(2,storePaperBook.getbookTitle());
            pstmt.setString(3,storePaperBook.getauthor());
            pstmt.setString(4,storePaperBook.getpublisher());
            pstmt.setString(5,storePaperBook.getpublicationDate());
            pstmt.setString(6,storePaperBook.getsummary());
```



```
else if(addTable.equals("ebook"))
{
    int rid;
    PreparedStatement pstmt ;
    Ebook storeEbook=new Ebook();
    AEbook=input;
    storeEbook=AEbook.get(0);
    String sql = String.format("SELECT MAX(bookID) FROM ebook");
    st.execute(sql);
    ResultSet rs=st.getResultSet();
    int x=0;
    while(rs.next())
    {
        x=rs.getInt(1);
    }
    x=x+1;
    InitialGUI.setmaxID(x);
    System.out.println(storeEbook.getbookTitle()+storeEbook.getauthor()+storeEbook.getpublisher()+storeEbook.getpublicati
    String SQL = "INSERT INTO ebook (bookID,bookTitle,author,publisher,publicationDate,summary,bookContext,bookType) VALU
    pstmt = conn.prepareStatement(SQL,Statement.RETURN_GENERATED_KEYS);
    pstmt.setInt(1,x);
    pstmt.setString(2,storeEbook.getbookTitle());
    pstmt.setString(3,storeEbook.getauthor());
    pstmt.setString(4,storeEbook.getpublisher());
    pstmt.setString(5,storeEbook.getpublicationDate());
    pstmt.setString(6,storeEbook.getsummary());
    pstmt.setString(7,storeEbook.getbookContext());
    pstmt.setString(8,storeEbook.getbookType());
    pstmt.execute();
```



```
else if(addTable.equals("ebook"))
{
    int rid;
    PreparedStatement pstmt ;
    Ebook storeEbook=new Ebook();
    AEbook=input;
    storeEbook=AEbook.get(0);
    String sql = String.format("SELECT MAX(bookID) FROM ebook");
    st.execute(sql);
    ResultSet rs=st.getResultSet();
    int x=0;
    while(rs.next())
    {
        x=rs.getInt(1);
    }
    x=x+1;
    InitialGUI.setmaxID(x);
    System.out.println(storeEbook.getbookTitle()+storeEbook.getauthor()+storeEbook.getpublisher()+storeEbook.getpublicati
    String SQL = "INSERT INTO ebook (bookID,bookTitle,author,publisher,publicationDate,summary,bookContext,bookType) VALU
    pstmt = conn.prepareStatement(SQL,Statement.RETURN_GENERATED_KEYS);
    pstmt.setInt(1,x);
    pstmt.setString(2,storeEbook.getbookTitle());
    pstmt.setString(3,storeEbook.getauthor());
    pstmt.setString(4,storeEbook.getpublisher());
    pstmt.setString(5,storeEbook.getpublicationDate());
    pstmt.setString(6,storeEbook.getsummary());
    pstmt.setString(7,storeEbook.getbookContext());
    pstmt.setString(8,storeEbook.getbookType());
    pstmt.execute();
```

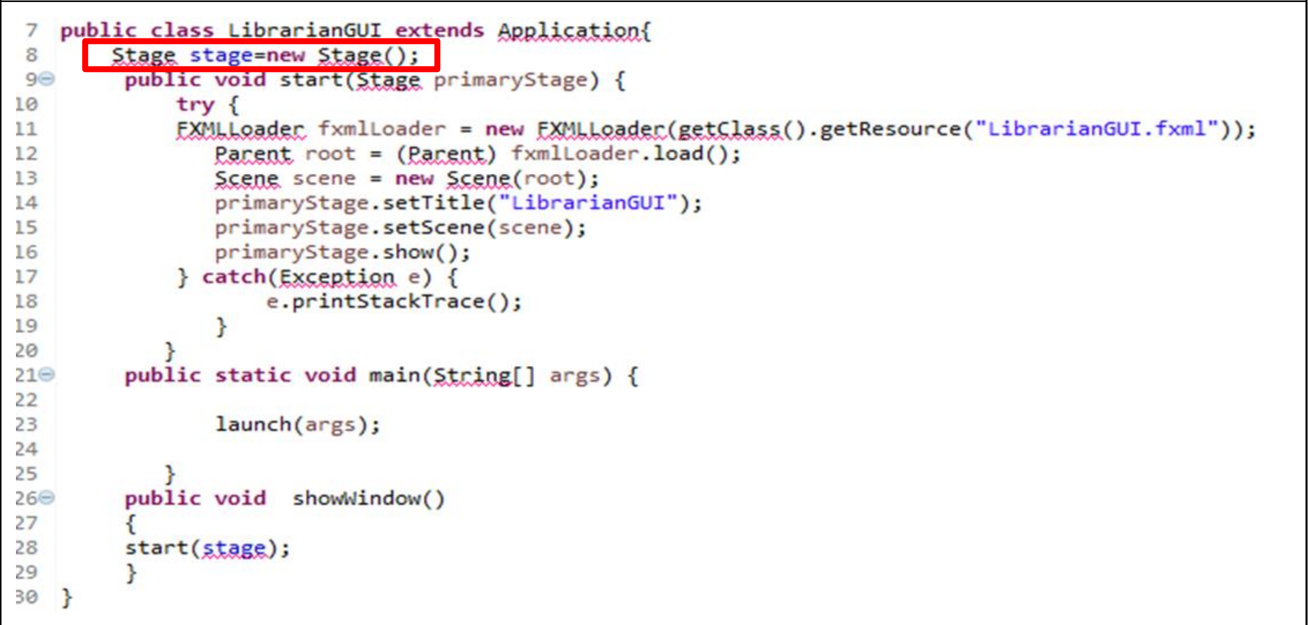
4) Connascence generalized the ideas of cohesion and coupling, use three pieces of your project to describe what types of the connascence they belong to.

1. Type of Class Connascence

If a class has an attribute of type A, it is tied to the type of the attribute. If the type of the attribute changes, the attribute declaration will have to be changed.

For example, if Stage class changes, LibrarianGUI class's Stage() method will also be changed.

class

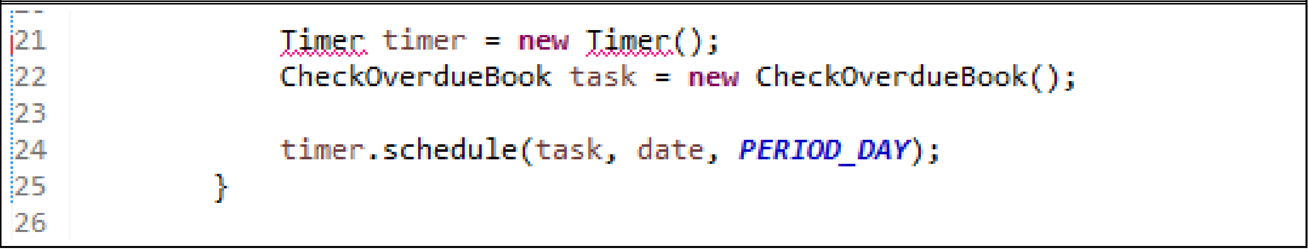


```java
public class LibrarianGUI extends Application{
    Stage stage=new Stage();
    public void start(Stage primaryStage) {
        try {
            FXMLLoader fxmlLoader = new FXMLLoader(getClass().getResource("LibrarianGUI.fxml"));
                Parent root = (Parent) fxmlLoader.load();
                Scene scene = new Scene(root);
                primaryStage.setTitle("LibrarianGUI");
                primaryStage.setScene(scene);
                primaryStage.show();
        } catch(Exception e) {
                e.printStackTrace();
            }
        }
    public static void main(String[] args) {

            launch(args);

        }
    public void  showWindow()
    {
    start(stage);
    }
}
```

2. Name Connascence

If a method refers to an attribute, it is tied to the name of the attribute. If the attribute's names changes, the content of the method will have to be changed.

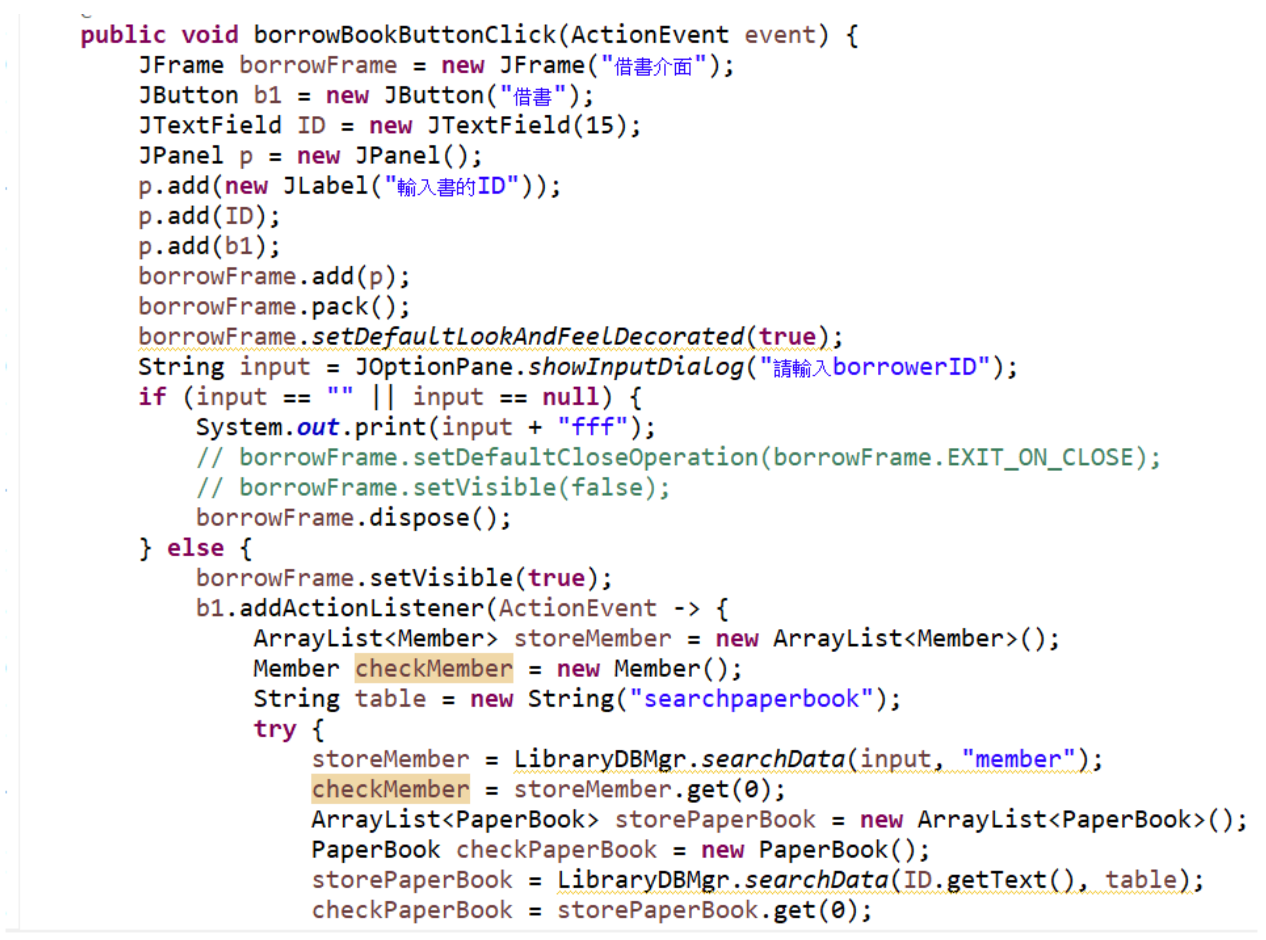
class Time



```java
            Timer timer = new Timer();
            CheckOverdueBook task = new CheckOverdueBook();

            timer.schedule(task, date, PERIOD_DAY);
        }
```

3. Convention Connascence

If the value’s range changes, every method that used the value would have to be modified.

class


```
public void borrowBookButtonClick(ActionEvent event) {
    JFrame borrowFrame = new JFrame("借書介面");
    JButton b1 = new JButton("借書");
    JTextField ID = new JTextField(15);
    JPanel p = new JPanel();
    p.add(new JLabel("輸入書的ID"));
    p.add(ID);
    p.add(b1);
    borrowFrame.add(p);
    borrowFrame.pack();
    borrowFrame.setDefaultLookAndFeelDecorated(true);
    String input = JOptionPane.showInputDialog("請輸入borrowerID");
    if (input == "" || input == null) {
        System.out.print(input + "fff");
        // borrowFrame.setDefaultCloseOperation(borrowFrame.EXIT_ON_CLOSE);
        // borrowFrame.setVisible(false);
        borrowFrame.dispose();
    } else {
        borrowFrame.setVisible(true);
        b1.addActionListener(ActionEvent -> {
            ArrayList<Member> storeMember = new ArrayList<Member>();
            Member checkMember = new Member();
            String table = new String("searchpaperbook");
            try {
                storeMember = LibraryDBMgr.searchData(input, "member");
                checkMember = storeMember.get(0);
                ArrayList<PaperBook> storePaperBook = new ArrayList<PaperBook>();
                PaperBook checkPaperBook = new PaperBook();
                storePaperBook = LibraryDBMgr.searchData(ID.getText(), table);
                checkPaperBook = storePaperBook.get(0);
```

5) **Use one class from your project that can create a set of invariants and add them to the CRC card or the class diagram.**

- **CRC Card**

Front

Class name: LibrarianController	**ID:** 1	**Type:** Concrete, Domain

Description:	**Association Use Case:**
This class provides librarian to save and edit data of memberships, paper book, and e-book. It also can help librarian search book information and provide book service.	Manage Paper Book Manage E-Book Manage Member Borrow Book Return Book Search Book
Responsibilities:	**Collaborators:**
addMemberButtonClick	PaperBook
addEbookButtonClick	Ebook
addBookButtonClick	Member
editMemberButtonClick	LibrarianGUI
editEbookButtonClick	SendEmail
editBookButtonClick	LibraryDBMgr
deleteMemberButtonClick	Search
deleteEbookButtonClick	
searchBookButtonClick	
borrowBookButtonClick	
returnBookButtonClick	
updatePaperBookStateButtonClick	

Back

Attributes:			
addMemberButton	(1..1)	(Button)	
addEbookButton	(1..1)	(Button)	
addBookButton	(1..1)	(Button)	
editMemberButton	(1..1)	(Button)	
editEbookButton	(1..1)	(Button)	
editBookButton	(1..1)	(Button)	
deleteMemberButton	(1..1)	(Button)	
deleteEbookButton	(1..1)	(Button)	
searchBookButton	(1..1)	(Button)	
borrowBookButton	(1..1)	(Button)	
returnBookButton	(1..1)	(Button)	
updatePaperBookStateButton	(1..1)	(Button)	
f	(0..1)	(File)	{f = (File) Actionevent.getNewValue()}
Relationships:			
Generalization(a-kind-of):			
Aggregation(has-parts):			
Other Associations:			
Manage Paper Book Manage E-Book Manage Member Borrow Book Return Book Search Book			

Text File

LibrarianController class invariants: F = (File) Actionevent.getNewValue()

6) Use a method of a class from your project that can create a contract and describe its algorithm specification. Specify the pre- or post- condition and use both Structured English and an activity diagram to specify the algorithm.

● **Contract**

Method Name:	run()	**Class Name:**	CheckOverdueBook	**ID:**	1
Client(consumers):	Time				
Associated Use Case: Member					
Description of Responsibilities: We use class checkOverdueBook to calculate if the book that borrowed by member is overdue or not.					
Arguments Received: day:long					
Pre-Conditions: day=(now.getTime() - beginDate.getTime())/(24*60*60*1000)					

Post-Conditions:
if(day<=3)
bookState = ("notice");
setnumberOfNoticeBook -1
getnumberOfNoticeBook +1
 	
else if(day<0)
bookState = ("overdue");
getnumberOfNoticeBook -1
getnumberOfOverdueBook +1
setright = (false)					

● Method Specification

Method Name: Time()	**Class Name:** Time	**ID:**
Contract ID:	**Programmer:** Kendy	Data Due: 05/28/2018

Programming Language:

Java

Triggers/Events:

CheckOverdueBook task = new CheckOverdueBook();

Arguments Received: Data Type:	**Notes:**
long	Borrowed day minus today

Messages Sent & Argument Passed: ClassName.MethodName:	**Data Type:**	**Notes:**

Arguments Returned: Data Type:	**Notes:**
void	

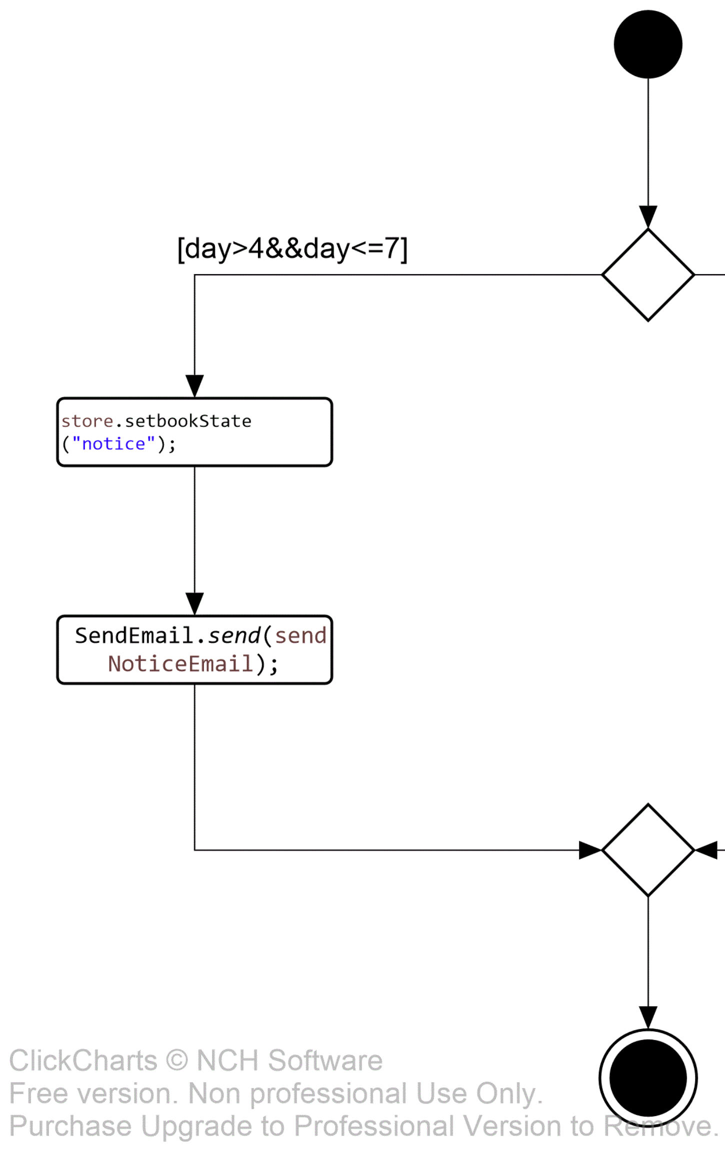
Algorithm Specification:

```
  if(day<=3)
              {
               store.setbookState("notice");

               OM.setnumberOfNoticeBook(OM.getnumberOfBorrowBook()-1);
               OM.setnumberOfNoticeBook(OM.getnumberOfNoticeBook()+1);
               SEP.add(store);
               }
   else if(day<0)
              {
               store.setbookState("overdue");

               OM.setnumberOfNoticeBook(OM.getnumberOfNoticeBook()-1);
               OM.setnumberOfNoticeBook(OM.getnumberOfOverdueBook()+1);
               OM.setright(false);
              }
```

Misc.Notes:

None

- **Activity Diagram**

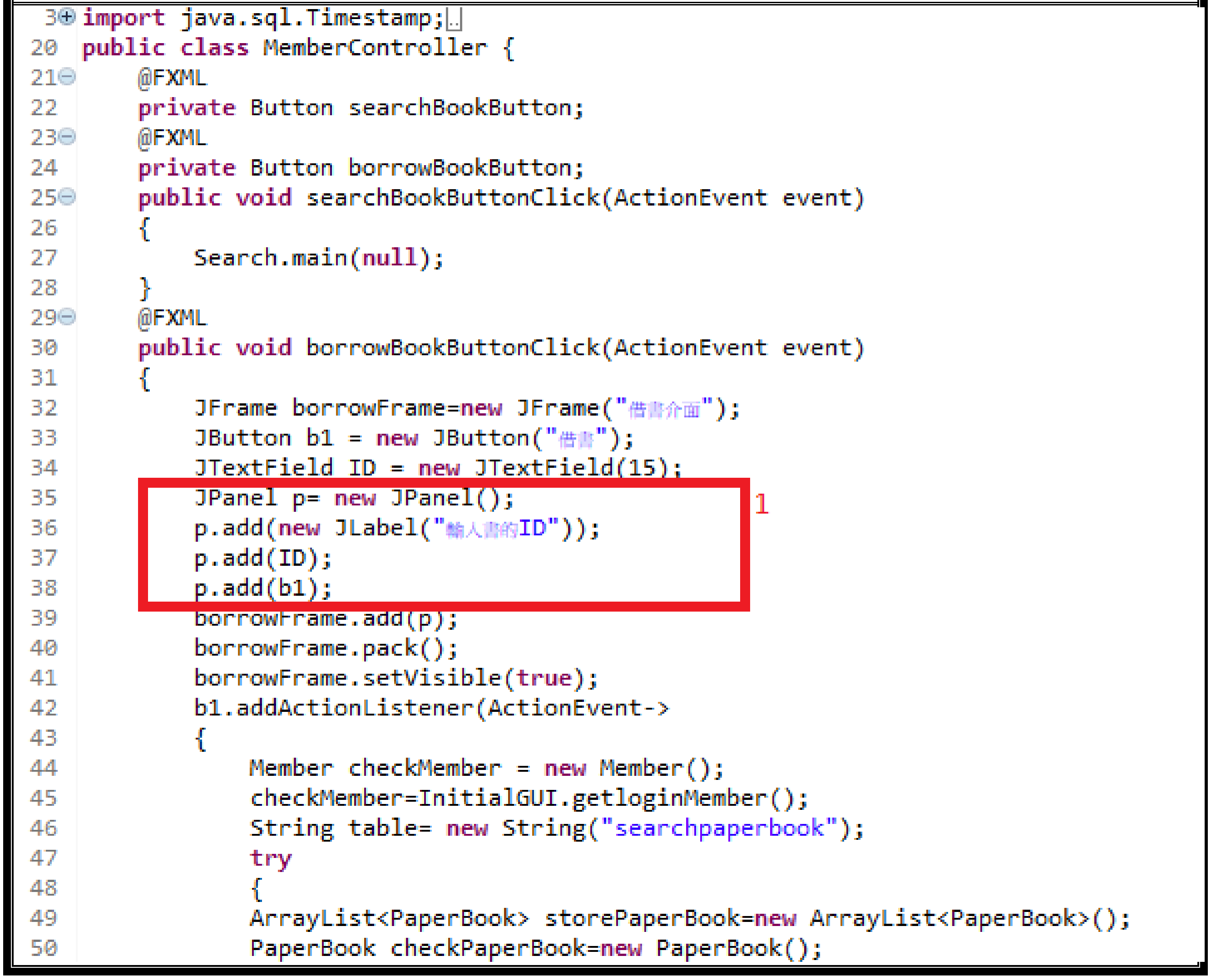

[day>4&&day<=7]
[day>7]
store.setbookState ("notice");
SendEmail.send(sendNoticeEmail);
store.setbookState ("overdue");
OM.setright(false);
SendEmail.send(sendNoticeEmail);
ClickCharts © NCH Software
Free version. Non professional Use Only.
Purchase Upgrade to Professional Version to Remove.

7) Please evaluate any piece of your project in terms of cohesion, coupling, and connascence perspective.

1. Coupling(Data Coupling)

Method borrowBookButton() creates an object p, then uses it to read inputted book ID to change book data to borrowed.

class MemberController

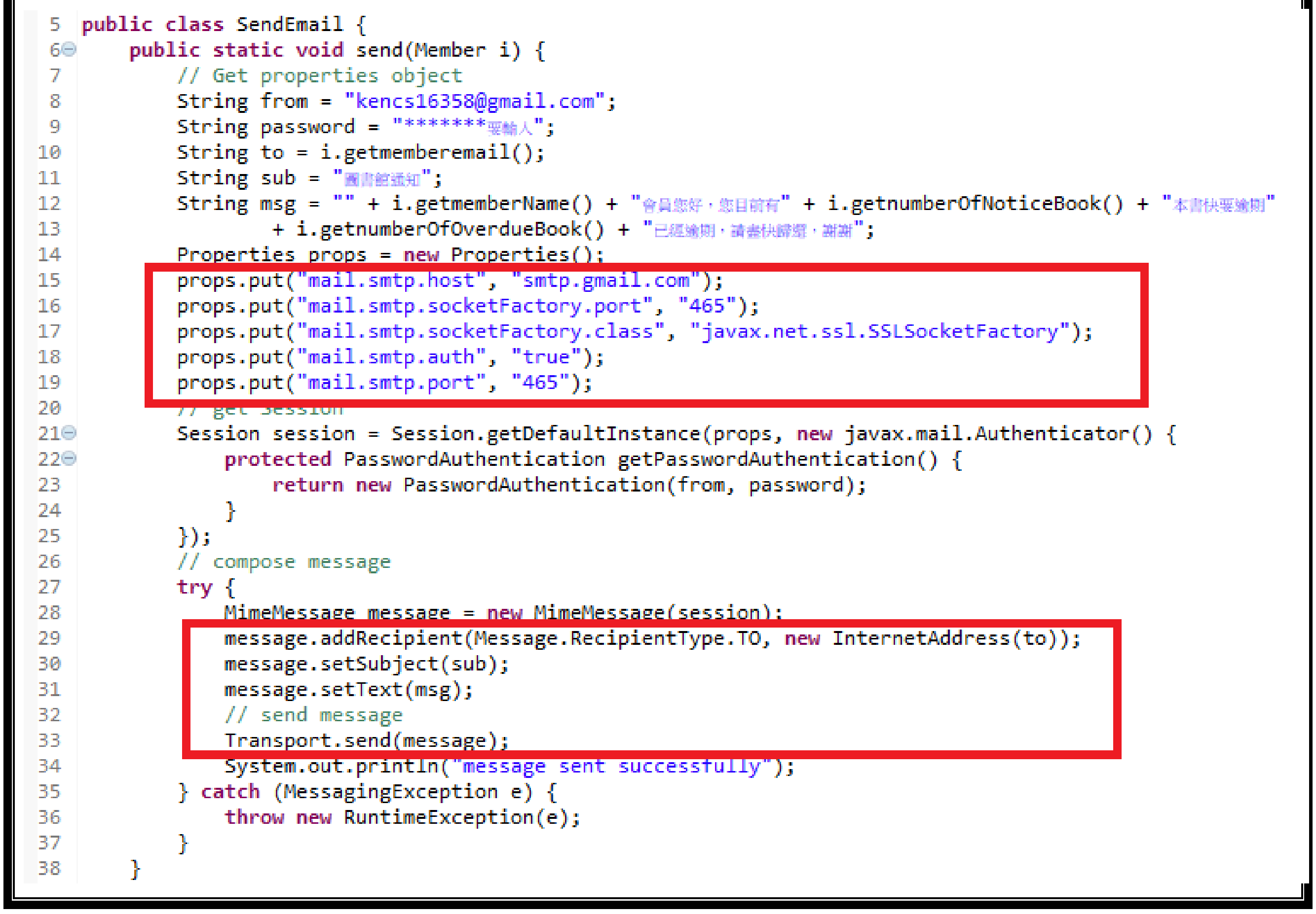


```
import java.sql.Timestamp;
public class MemberController {
    @FXML
    private Button searchBookButton;
    @FXML
    private Button borrowBookButton;
    public void searchBookButtonClick(ActionEvent event)
    {
        Search.main(null);
    }
    @FXML
    public void borrowBookButtonClick(ActionEvent event)
    {
        JFrame borrowFrame=new JFrame("借書介面");
        JButton b1 = new JButton("借書");
        JTextField ID = new JTextField(15);
        JPanel p= new JPanel();
        p.add(new JLabel("輸入書的ID"));
        p.add(ID);
        p.add(b1);
        borrowFrame.add(p);
        borrowFrame.pack();
        borrowFrame.setVisible(true);
        b1.addActionListener(ActionEvent->
        {
            Member checkMember = new Member();
            checkMember=InitialGUI.getloginMember();
            String table= new String("searchpaperbook");
            try
            {
            ArrayList<PaperBook> storePaperBook=new ArrayList<PaperBook>();
            PaperBook checkPaperBook=new PaperBook();
```

2. Cohesion(Function Cohesion)

If member's returning book day is close, system will send e-mail to notify member. System put the properties information at first, set subject and text to the e-mail and send it.

class SendEmail

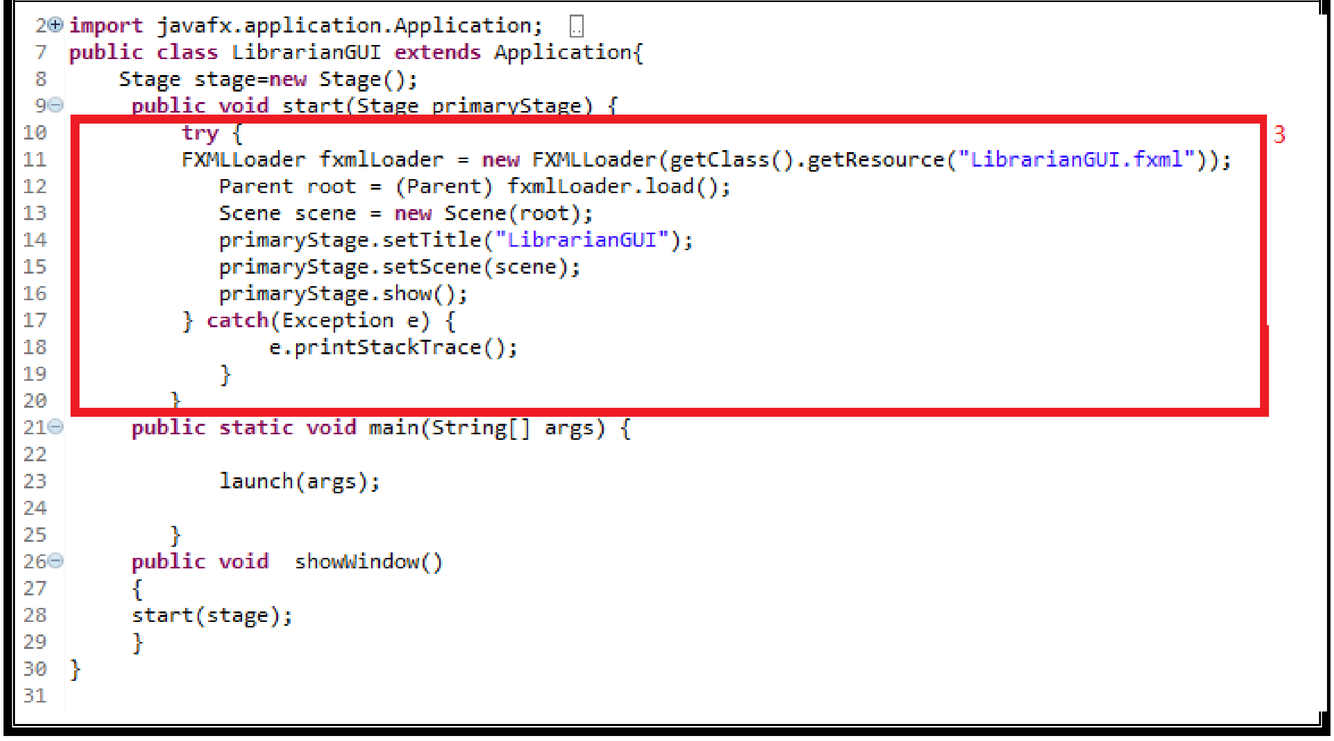


```java
public class SendEmail {
    public static void send(Member i) {
        // Get properties object
        String from = "kencs16358@gmail.com";
        String password = "*******要輸入";
        String to = i.getmemberemail();
        String sub = "圖書館通知";
        String msg = "" + i.getmemberName() + "會員您好，您目前有" + i.getnumberOfNoticeBook() + "本書快要逾期"
                + i.getnumberOfOverdueBook() + "已經逾期，請盡快歸還，謝謝";
        Properties props = new Properties();
        props.put("mail.smtp.host", "smtp.gmail.com");
        props.put("mail.smtp.socketFactory.port", "465");
        props.put("mail.smtp.socketFactory.class", "javax.net.ssl.SSLSocketFactory");
        props.put("mail.smtp.auth", "true");
        props.put("mail.smtp.port", "465");
        // get session
        Session session = Session.getDefaultInstance(props, new javax.mail.Authenticator() {
            protected PasswordAuthentication getPasswordAuthentication() {
                return new PasswordAuthentication(from, password);
            }
        });
        // compose message
        try {
            MimeMessage message = new MimeMessage(session);
            message.addRecipient(Message.RecipientType.TO, new InternetAddress(to));
            message.setSubject(sub);
            message.setText(msg);
            // send message
            Transport.send(message);
            System.out.println("message sent successfully");
        } catch (MessagingException e) {
            throw new RuntimeException(e);
        }
    }
```

3. Connascence(Convention Connascence)

If the value’s range changes, every method that used the value would have to be modified.

class PaperBook



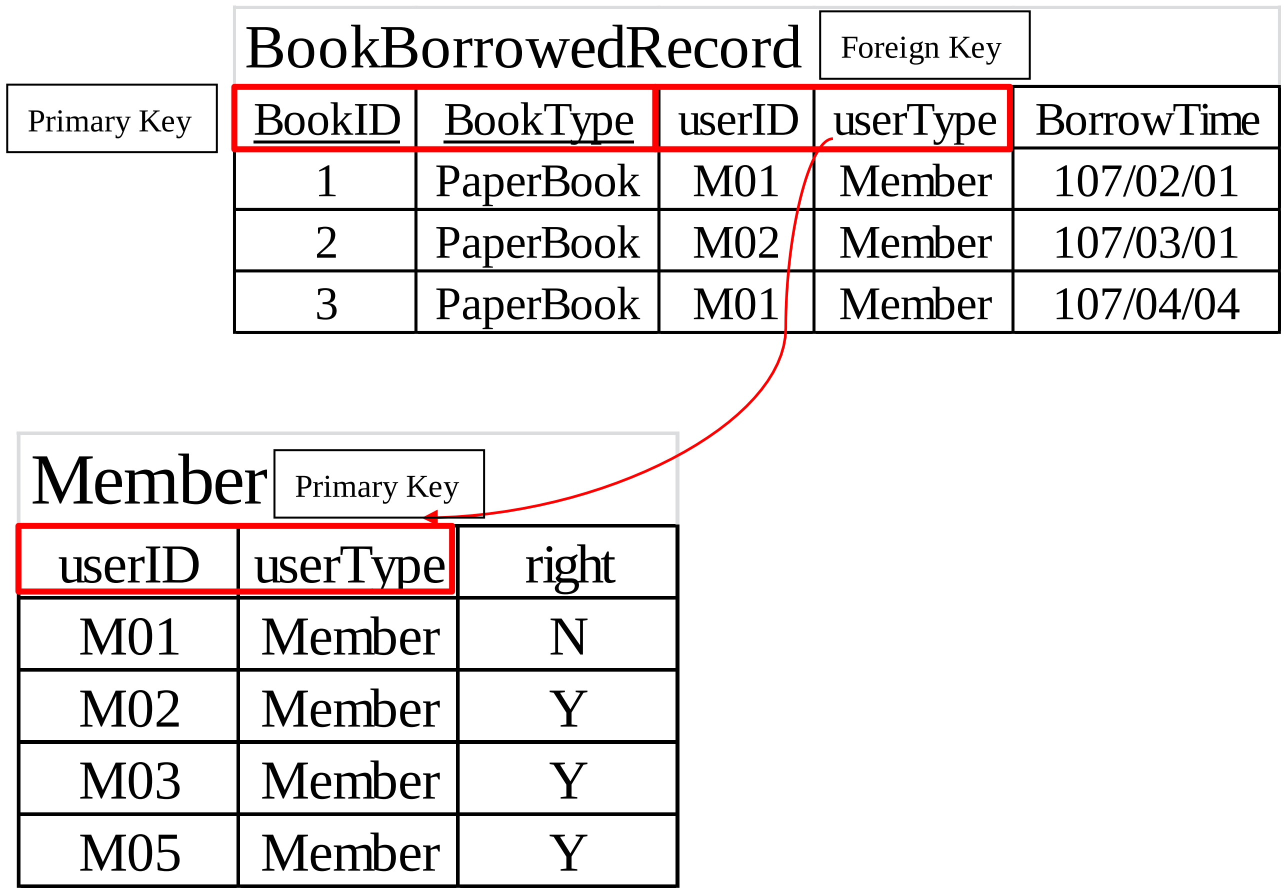
```
import javafx.application.Application;
public class LibrarianGUI extends Application{
    Stage stage=new Stage();
     public void start(Stage primaryStage) {
         try {
         FXMLLoader fxmlLoader = new FXMLLoader(getClass().getResource("LibrarianGUI.fxml"));
            Parent root = (Parent) fxmlLoader.load();
            Scene scene = new Scene(root);
            primaryStage.setTitle("LibrarianGUI");
            primaryStage.setScene(scene);
            primaryStage.show();
         } catch(Exception e) {
                e.printStackTrace();
            }
        }
     public static void main(String[] args) {

            launch(args);

        }
     public void  showWindow()
     {
     start(stage);
     }
}
```

8) Assume that you are going to adopt RDBMs to your project, please describe the referential integrity.

Table BookBorrowedRecord’s primary key is bookID and bookType, and foreign key is userID and userType which is used to record the book is borrowed by whom. userID and userType are primary key of table Member (userID and userType).

Foreign key’s value can be null because the book may not be borrowed. But if foreign key’s value isn’t null, the value must be Member table’s primary key – user ID’s value. Then table BookBorrowedRecord and table Member can refer each other.

If foreign key’s value isn’t member ID’s value, data will not refer. Refer will be wrong.



BookBorrowedRecord

BookID	BookType	userID	userType	BorrowTime
1	PaperBook	M01	Member	107/02/01
2	PaperBook	M02	Member	107/03/01
3	PaperBook	M01	Member	107/04/04

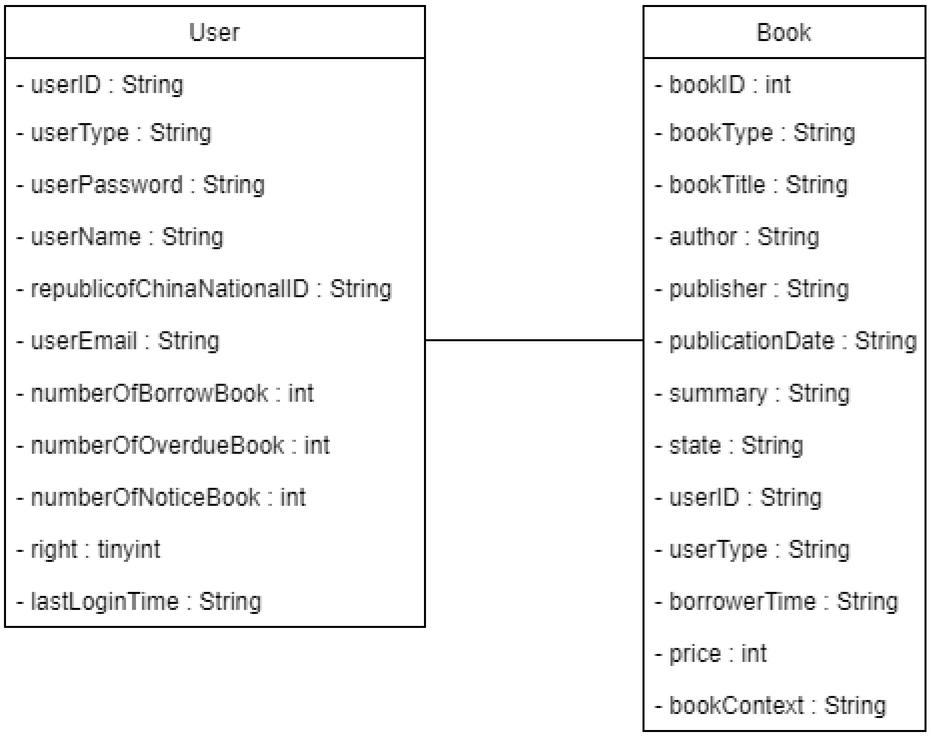
Member

userID	userType	right
M01	Member	N
M02	Member	Y
M03	Member	Y
M05	Member	Y

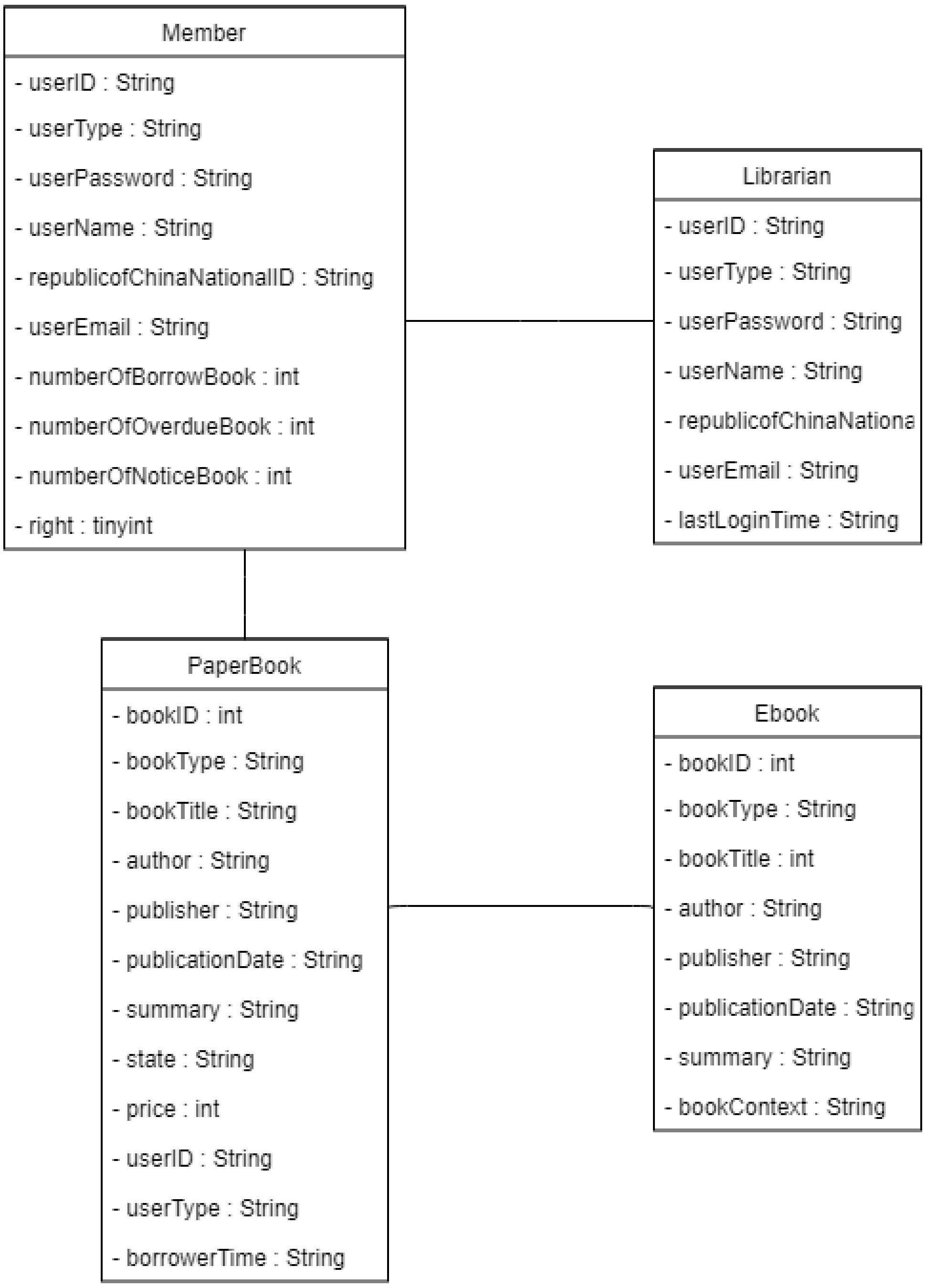
9) **Using the steps of normalization, create a model that represents the file of your project in third normal form. Please make necessary assumptions to explain why the tables are related.**

- **Class Diagram**

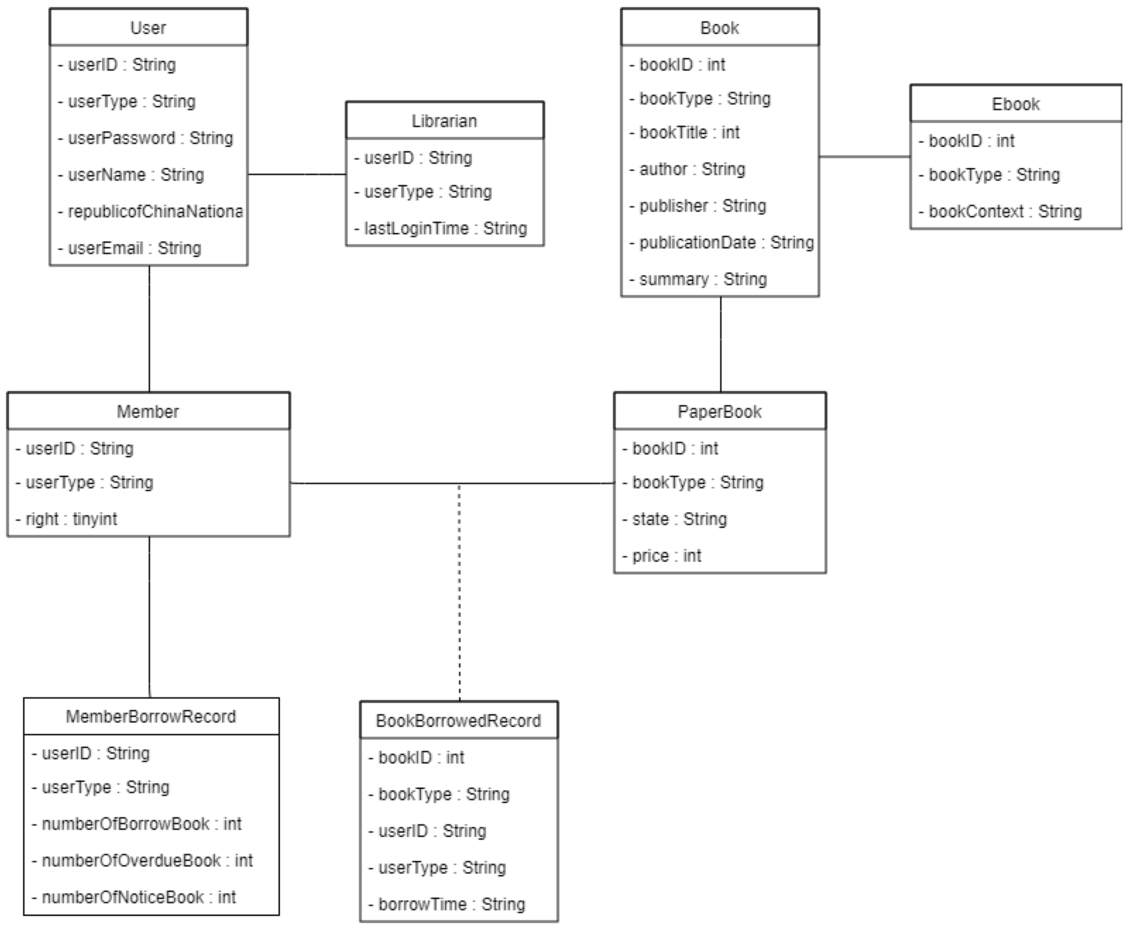
First Normal Form Class Diagram



Second Normal Form Class Diagram



Third Normal Form Class Diagram



● Zero Normal Form

Book

bookID	bookType	bookTitle	author	publisher	publicationDate	summary	state	price	borrowerID	borrowerType	borrowerTime	bookContext
1	PaperBook	I'm No.1	oneno	Red	107/01	this book makes you become no.1	Overdue	111	M01	Member	107/02/01	
2	PaperBook	Second not bad	twowt	BANANA	107/02	don't always want be no.1, I'll tell you advantage of second	Borrowed	222	M02	Member	107/03/01	
1	Ebook	Third what ever	threerht	Zebra	107/03	third means you just behind two people, don't think to much						Third is good, don't mind.Third is good, don't mind.Third is good, don't mind…
3	PaperBook	Forth you better relex	fouruof	OK	107/04	If you want do more better but always get forth, you must get too much pressure to yourself. Try to relax.	Borrowed	333	M01	Member	107/04/04	
4	PaperBook	No fifth	fivevif	BANANA	107/01	No one care about fifth, just practice harder.	Availabe	444				
5	PaperBook	Sixth give up	sixis	BANANA	107/03	six is not a lucky number, give up will give you happy life	Missing	555				
2	Ebook	Lucky seven	oneno	Zebra	107/05	you must a luck guy to get this number, let me aupluse to you						Seven is a good number.Seven is a good number.Seven is a good number…
6	PaperBook	Super eight	sixis	OK	107/02	lying eight is unlimit, you are superman	Unavailable	121				
7	PaperBook	number nine	ninenin	Red	107/03	no no no no, just nine just a number, No MORE	Damaged	232				
8	PaperBook	Top ten fact	seveneves	BANANA	107/01	Fact no.1：If you want to know, borrow me first	Repaired	343				
9	PaperBook	Uncountable	twowt	OK	107/01	I can't count anymore, don't ask me the number behind ten.	Deregistered	10				

User

userID	userType	userPassword	userName	republicofChinaNationalID	userEmail	lastLoginTime	numberOfBorrowBook	numberOfOverdueBook	numberOfNoticeBook	right
M01	Member	1234	Lynn	A111333555	Lynn@mail		2	1	0	N
M02	Member	5678	Tim	R333555777	Tim@mail		1	0	1	Y
M01	Member	1234	Lynn	A111333555	Lynn@mail		2	1	0	N
M03	Member	8901	Ken	P555777999	Ken@mail		0	0	0	Y
M05	Member	2345	Smitevevejagermanjason	L777999111	Smit@mail		0	0	0	Y
L01	Librarian	9876	Omaiwa	I222444666	Omai wa@mail	107/02/02 02:02				
L02	Librarian	5432	Mo sinde	U444666888	Mo sinde@mail	107/01/01 01:01				
L03	Librarian	1098	Iru	T666888000	Iru@mail	106/06/06 06:06				
L04	Librarian	7654	Nani	W888000222	Nani@mail	107/05/05 05:05				

● First Normal Form

Book

bookID	bookType	bookTitle	author	publisher	publicationDate	summary
1	PaperBook	I'm No.1	oneno	Red	107/01	this book makes you become no.1
2	PaperBook	Second not bad	twowt	BANANA	107/02	don't always want be no.1, I'll tell you advantage of second
1	Ebook	Third what ever	threerht	Zebra	107/03	third means you just behind two people, don't think to much
3	PaperBook	Forth you better relex	fouruof	OK	107/04	If you want do more better but always get forth, you must get too much pressure to yourself. Try to relax.
4	PaperBook	No fifth	fivevif	BANANA	107/01	No one care about fifth, just practice harder.
5	PaperBook	Sixth give up	sixis	BANANA	107/03	six is not a lucky number, give up will give you happy life
2	Ebook	Lucky seven	oneno	Zebra	107/05	you must a luck guy to get this number, let me aupluse to you
6	PaperBook	Super eight	sixis	OK	107/02	lying eight is unlimit, you are superman
7	PaperBook	number nine	ninenin	Red	107/03	no no no no, just nine just a number, No MORE
8	PaperBook	Top ten fact	seveneves	BANANA	107/01	Fact no.1：If you want to know, borrow me first
9	PaperBook	Uncountable	twowt	OK	107/01	I can't count anymore, don't ask me the number behind ten.

state	price	borrowerID	borrwerType	borrowerTime	bookContext
Overdue	111	M01	Member	107/02/01	
Borrowed	222	M02	Member	107/03/01	
					Third is good, don't mind.Third is good, don't mind.Third is good, don't mind…
Borrowed	333	M01	Member	107/04/04	
Availabe	444				
Missing	555				
					Seven is a good number.Seven is a good number.Seven is a good number…
Unavailable	121				
Damaged	232				
Repaired	343				
Deregistered	10				

User

userID	userType	userPassword	userName	republicofChinaNationalID	userEmail	lastLoginTime	numberOfBorrowBook	numberOfOverdueBook	numberOfNoticeBook	right
M01	Member	1234	Lynn	A111333555	Lynn@mail		2	1	0	N
M02	Member	5678	Tim	R333555777	Tim@mail		1	0	1	Y
M03	Member	8901	Ken	P555777999	Ken@mail		0	0	0	Y
M05	Member	2345	Smitevevejagermanjason	L777999111	Smit@mail		0	0	0	Y
L01	Librarian	9876	Omai wa	I222444666	Omai wa@mail	107/02/02 02:02				
L02	Librarian	5432	Mo sinde	U444666888	Mo sinde@mail	107/01/01 01:01				
L03	Librarian	1098	Iru	T666888000	Iru@mail	106/06/06 06:06				
L04	Librarian	7654	Nani	W888000222	Nani@mail	107/05/05 05:05				

● Second Normal Form

PaperBook

bookID	bookType	bookTitle	author	publisher	publicationDate	summary	state	price	borrowerID	borrowerType	borrowerTime
1	PaperBook	I'm No.1	oneno	Red	107/01	this book makes you become no.1	Overdue	111	M01	Member	107/02/01
2	PaperBook	Second not bad	twowt	BANANA	107/02	don't always want be no.1, I'll tell you advantage of second	Borrowed	222	M02	Member	107/03/01
3	PaperBook	Forth you better relex	fouruof	OK	107/04	If you want do more better but always get forth, you must get too much pressure to yourself. Try to relax.	Borrowed	333	M01	Member	107/04/04
4	PaperBook	No fifth	fivevif	BANANA	107/01	No one care about fifth, just practice harder.	Availabe	444			
5	PaperBook	Sixth give up	sixis	BANANA	107/03	six is not a lucky number, give up will give you happy life	Missing	555			
6	PaperBook	Super eight	sixis	OK	107/02	lying eight is unlimit, you are superman	Unavailable	121			
7	PaperBook	number nine	ninenin	Red	107/03	no no no no, just nine just a number, No MORE	Damaged	232			
8	PaperBook	Top ten fact	seveneves	BANANA	107/01	Fact no.1：If you want to know, borrow me first	Repaired	343			
9	PaperBook	Uncountable	twowt	OK	107/01	I can't count anymore, don't ask me the number behind ten.	Deregistered	10			

Ebook

bookID	bookType	bookTitle	author	publisher	publicationDate	summary	bookContext
1	Ebook	Third what ever	threerht	Zebra	107/03	third means you just behind two people, don't think to much	Third is good, don't mind.Third is good, don't mind.Third is good, don't mind…
2	Ebook	Lucky seven	oneno	Zebra	107/05	you must a luck guy to get this number, let me aupluse to you	Seven is a good number.Seven is a good number.Seven is a good number…

Member

userID	userType	userPassword	userName	republicofChinaNationalID	userEmail	numberOfBorrowBook	numberOfOverdueBook	numberOfNoticeBook	right
M01	Member	1234	Lynn	A111333555	Lynn@mail	2	1	0	N
M02	Member	5678	Tim	R333555777	Tim@mail	1	0	1	Y
M03	Member	8901	Ken	P555777999	Ken@mail	0	0	0	Y
M05	Member	2345	Smitevevejagermanjason	L777999111	Smit@mail	0	0	0	Y

Librarian

userID	userType	userPassword	userName	republicofChinaNationalID	userEmail	lastLoginTime
L01	Librarian	9876	Omai wa	I222444666	Omai wa@mail	107/02/02 02:02
L02	Librarian	5432	Mo sinde	U444666888	Mo sinde@mail	107/01/01 01:01
L03	Librarian	1098	Iru	T666888000	Iru@mail	106/06/06 06:06
L04	Librarian	7654	Nani	W888000222	Nani@mail	107/05/05 05:05

● Third Normal Form

Book

bookID	bookType	bookTitle	author	publisher	publicationDate	summary
1	PaperBook	I'm No.1	oneno	Red	107/01	this book makes you become no.1
2	PaperBook	Second not bad	twowt	BANANA	107/02	don't always want be no.1, I'll tell you advantage of second
3	PaperBook	Forth you better relex	fouruof	OK	107/04	If you want do more better but always get forth, you must get too much pressure to yourself. Try to relax.
4	PaperBook	No fifth	fivevif	BANANA	107/01	No one care about fifth, just practice harder.
5	PaperBook	Sixth give up	sixis	BANANA	107/03	six is not a lucky number, give up will give you happy life
6	PaperBook	Super eight	sixis	OK	107/02	lying eight is unlimit, you are superman
7	PaperBook	number nine	ninenin	Red	107/03	no no no no, just nine just a number, No MORE
8	PaperBook	Top ten fact	seveneves	BANANA	107/01	Fact no.1：If you want to know, borrow me first
9	PaperBook	Uncountable	twowt	OK	107/01	I can't count anymore, don't ask me the number behind ten.
1	Ebook	Third what ever	threerht	Zebra	107/03	third means you just behind two people, don't think to much
2	Ebook	Lucky seven	oneno	Zebra	107/05	you must a luck guy to get this number, let me aupluse to you

Member

userID	userType	right
M01	Member	N
M02	Member	Y
M03	Member	Y
M05	Member	Y

User

userID	userType	userPassword	userName	republicofChinaNationalID	userEmail
M01	Member	1234	Lynn	A111333555	Lynn@mail
M02	Member	5678	Tim	R333555777	Tim@mail
M03	Member	8901	Ken	P555777999	Ken@mail
M05	Member	2345	Smitevevejagermanjason	L777999111	Smit@mail
L01	Librarian	9876	Omai wa	I222444666	Omai wa@mail
L02	Librarian	5432	Mo sinde	U444666888	Mo sinde@mail
L03	Librarian	1098	Iru	T666888000	Iru@mail
L04	Librarian	7654	Nani	W888000222	Nani@mail

PaperBook

bookID	bookType	state	price
1	PaperBook	Overdue	111
2	PaperBook	Borrowed	222
3	PaperBook	Borrowed	333
4	PaperBook	Availabe	444
5	PaperBook	Missing	555
6	PaperBook	Unavailable	121
7	PaperBook	Damaged	232
8	PaperBook	Repaired	343
9	PaperBook	Deregistered	10

Ebook

bookID	bookType	bookContext
1	Ebook	Third is good, don't mind.Third is good, don't mind.Third is good, don't mind…
2	Ebook	Seven is a good number.Seven is a good number.Seven is a good number…

Librarian

userID	userType	lastLoginTime
L01	Librarian	107/02/02 02:02
L02	Librarian	107/01/01 01:01
L03	Librarian	106/06/06 06:06
L04	Librarian	107/05/05 05:05

BookBorrowedRecord

BookID	BookType	userID	userType	BorrowTime
1	PaperBook	M01	Member	107/02/01
2	PaperBook	M02	Member	107/03/01
3	PaperBook	M01	Member	107/04/04

MemberBorrowedRecord

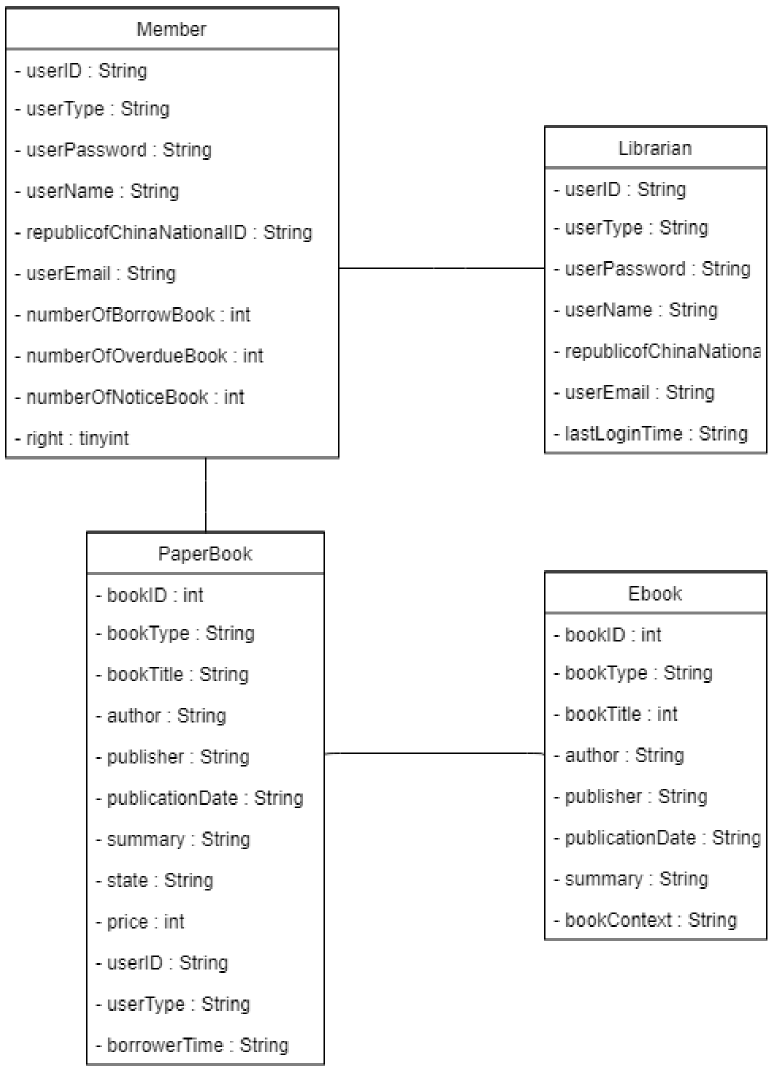
userID	userType	numberOfBorrowBook	numberOfOverdueBook	numberOfNoticeBook
M01	Member	2	1	0
M02	Member	1	0	1
M03	Member	0	0	0
M05	Member	0	0	0

10) Describe how you would denormalize the model that you created in question 9. Draw the new class diagram based on your suggested changes.

- **Denormalization**

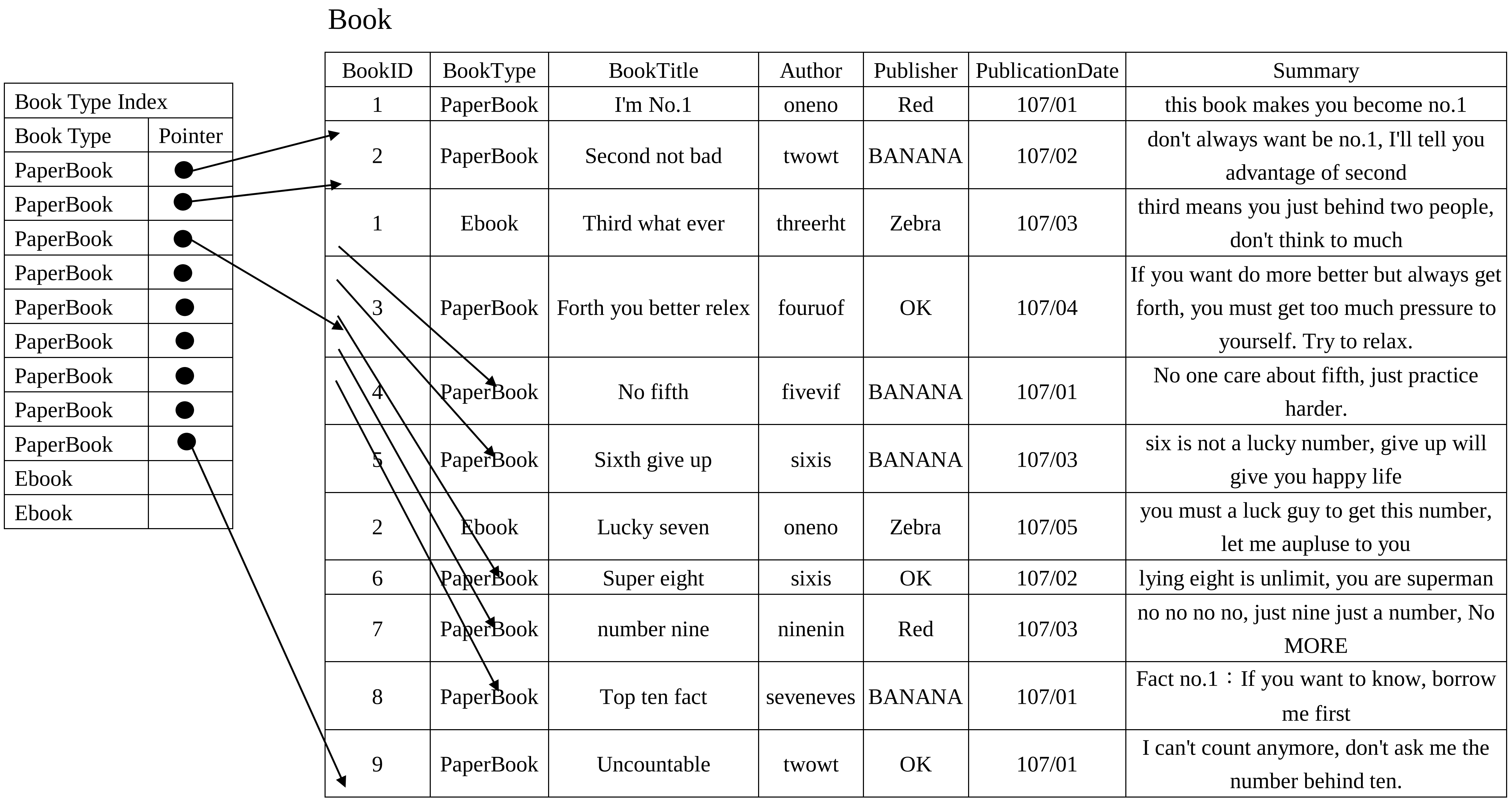
The third normalization has too many tables. First, it will lead to system's running speed slower. Second, these tables don't help us about manage data. So, we denormalize in order to improve system's running speed.

- **Class Diagram**



11) Examine the model that you created in question 10. Develop the inter-file clustering and index strategies. Describe how your clustering strategy will improve the performance of the database. List possible indices you would recommend and describe the reasons.

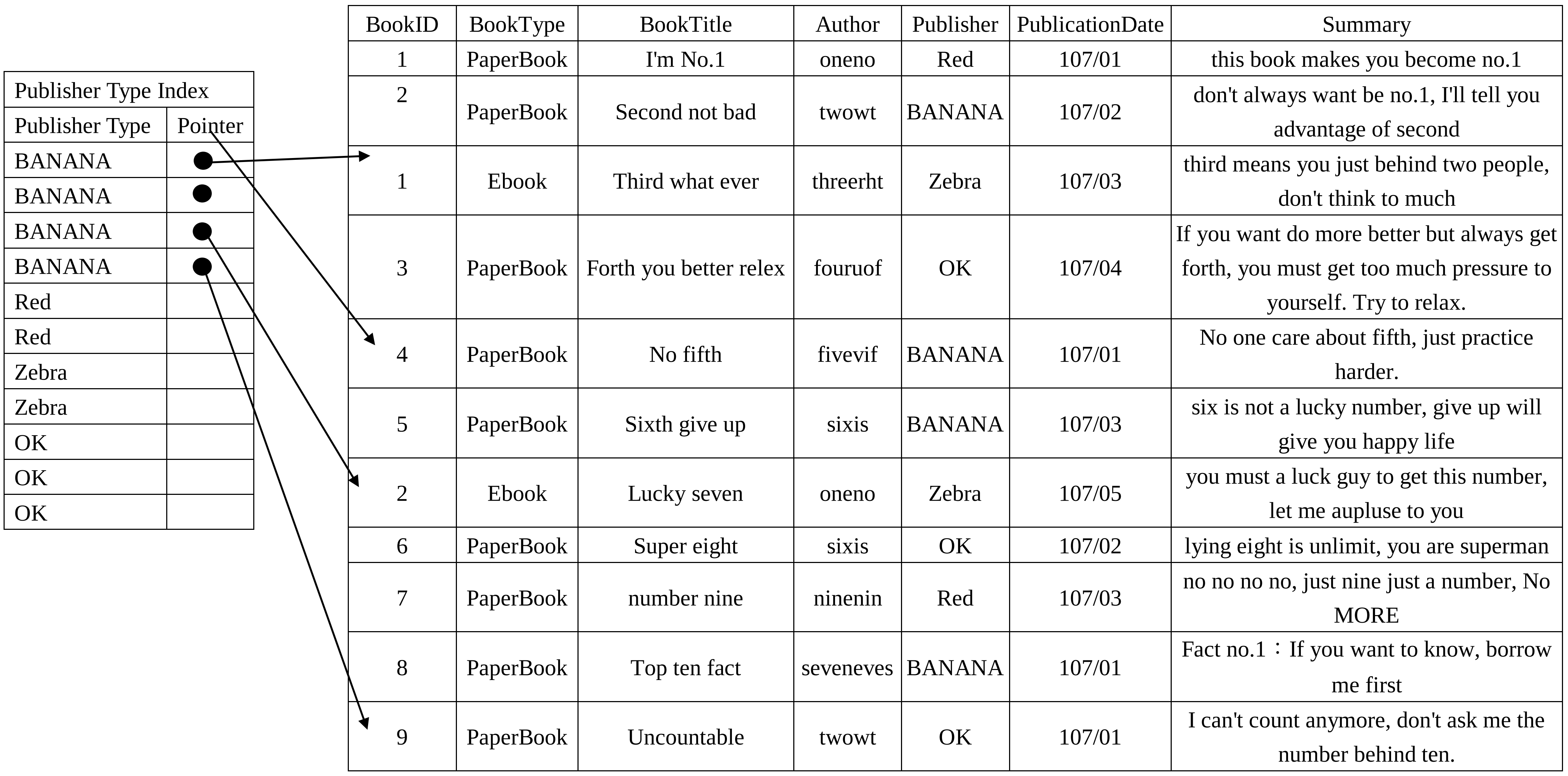
We will create two indexes between Book Type and author. Because if we know which kind of book we need, then we can use index for us to search more quickly. Like, I want to find Harry Potter, if I don’t know which kind of book it is. Then system needs to find every book in system. It will waste too much time. But if we know it is E-book, then the system can just find E-book part. Saving our time and more efficient. So, such as index Book Type, we will create another index about author. Because in some situation, people just know the book’s author. So if we make an index about author. It will take less time than search every book data.



Book

Book Type Index	
Book Type	Pointer
PaperBook	●
PaperBook	●
PaperBook	●
PaperBook	●
PaperBook	●
PaperBook	●
PaperBook	●
PaperBook	●
PaperBook	●
Ebook	
Ebook	

BookID	BookType	BookTitle	Author	Publisher	PublicationDate	Summary
1	PaperBook	I'm No.1	oneno	Red	107/01	this book makes you become no.1
2	PaperBook	Second not bad	twowt	BANANA	107/02	don't always want be no.1, I'll tell you advantage of second
1	Ebook	Third what ever	threerht	Zebra	107/03	third means you just behind two people, don't think to much
3	PaperBook	Forth you better relex	fouruof	OK	107/04	If you want do more better but always get forth, you must get too much pressure to yourself. Try to relax.
4	PaperBook	No fifth	fivevif	BANANA	107/01	No one care about fifth, just practice harder.
5	PaperBook	Sixth give up	sixis	BANANA	107/03	six is not a lucky number, give up will give you happy life
2	Ebook	Lucky seven	oneno	Zebra	107/05	you must a luck guy to get this number, let me aupluse to you
6	PaperBook	Super eight	sixis	OK	107/02	lying eight is unlimit, you are superman
7	PaperBook	number nine	ninenin	Red	107/03	no no no no, just nine just a number, No MORE
8	PaperBook	Top ten fact	seveneves	BANANA	107/01	Fact no.1：If you want to know, borrow me first
9	PaperBook	Uncountable	twowt	OK	107/01	I can't count anymore, don't ask me the number behind ten.



Publisher Type Index	
Publisher Type	Pointer
BANANA	●
BANANA	●
BANANA	●
BANANA	●
Red	
Red	
Zebra	
Zebra	
OK	
OK	
OK	

Book

BookID	BookType	BookTitle	Author	Publisher	PublicationDate	Summary
1	PaperBook	I'm No.1	oneno	Red	107/01	this book makes you become no.1
2	PaperBook	Second not bad	twowt	BANANA	107/02	don't always want be no.1, I'll tell you advantage of second
1	Ebook	Third what ever	threerht	Zebra	107/03	third means you just behind two people, don't think to much
3	PaperBook	Forth you better relex	fouruof	OK	107/04	If you want do more better but always get forth, you must get too much pressure to yourself. Try to relax.
4	PaperBook	No fifth	fivevif	BANANA	107/01	No one care about fifth, just practice harder.
5	PaperBook	Sixth give up	sixis	BANANA	107/03	six is not a lucky number, give up will give you happy life
2	Ebook	Lucky seven	oneno	Zebra	107/05	you must a luck guy to get this number, let me aupluse to you
6	PaperBook	Super eight	sixis	OK	107/02	lying eight is unlimit, you are superman
7	PaperBook	number nine	ninenin	Red	107/03	no no no no, just nine just a number, No MORE
8	PaperBook	Top ten fact	seveneves	BANANA	107/01	Fact no.1：If you want to know, borrow me first
9	PaperBook	Uncountable	twowt	OK	107/01	I can't count anymore, don't ask me the number behind ten.

Participate In Assignments

ID	Name	Participate	Responsibility
A10523006	Maggie	100%	Question3 Question4 Question6
A10523049	Peggy	100%	Question11 PPT
B10423003	Kurumi	100%	Word Question1 Question8 Question10
B10423029	Bean	0%	
B10523020	Kendy	100%	Java Code Question1 Question2 Question6
B10523030	Jerry	100%	Question7
B10523053	Lynn	100%	Question1 Question5 Question9
M10723001	Joe	100%	Java Code Question2 Question9 Question11