

```

package team3.assignment;

import javax.swing.*.*;
import java.awt.*.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.io.*;
import java.util.HashMap;
import java.util.Map.Entry;
import java.util.Scanner;

/*
1- 손명진 : 사용할 기능 입력 및 도서 목록 조회
2- 안상완 : 도서 검색
3- 신현빈 : 도서 대여 및 반납
4- 서유빈 : 새로운 책 등록 및 삭제
5- 서준호 : 보고서 보완 및 스윙 디자인 다듬기
*/

public class Bookshelf extends JFrame {
    private HashMap<String, book> dataHashMap;
    private Font font = new Font("함초롱바탕", Font.BOLD, 17); // 폰트와 글씨
크기를 새로 지정하는 폰트 삽입
    private JButton searchBook, showBooks, rentalBook, returnBook,
wearBook, deleteBook, exit;
    private JPanel mainPage;
    // Color buttonColor = new Color(255, 158, 0);
    // JButton[] jButtons = {searchBook, showBooks, rentalBook, returnBook,
wearBook, deleteBook, exit};
    public static void main(String[] args) {
        new Bookshelf();
    }
    public Bookshelf() {
        GUISetUp();
    }
    public void GUISetUp(){

        initiateSetting(); // HashMap 생성 및 초기화
        initiateComponentCondition(); // Bookshelf 클래스의 필드 버튼들 생성 및
설정
        ComponentListener(); // 컴포넌트 (버튼들) 에 리스너 지정

        setTitle("도서 관리 프로그램");
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

        Container contentPane = getContentPane();
        contentPane.setLayout(new BorderLayout());

        JPanel introPage = new JPanel();
        contentPane.add(introPage, BorderLayout.NORTH); // 여기까지
컨텐츠페인(JFrame 에 JLabel 을 넣은 JPanel 을 NORTH 에 삽입)

```

```

        mainPage = new JPanel(); // JFrame 의 contentPane 의 가운데에 들어갈
JPanel
        contentPane.add(mainPage, BorderLayout.CENTER);

        JPanel menus = new JPanel(new GridLayout(7, 1, 5, 5)); //
contentPane 의 좌측에 삽입될, 버튼을 모아놓은 JPanel

        contentPane.add(menus, BorderLayout.WEST);
        menus.add(searchBook);
        menus.add(showBooks);
        menus.add(rentalBook);
        menus.add(returnBook);
        menus.add(wearBook);
        menus.add(deleteBook);
        menus.add(exit);

        setSize(700, 800);
        setVisible(true);
    }

    public void initiateComponentCondition(){
        searchBook = new JButton("도서 검색");
        showBooks = new JButton("도서 목록 출력");
        rentalBook = new JButton("도서 대여하기");
        returnBook = new JButton("도서 반납하기");
        wearBook = new JButton("입고 도서 등록"); //입고 도서 등록
        deleteBook = new JButton("기존 도서 삭제"); //기존 도서 삭제
        exit = new JButton("시스템 종료");

        JButton[] jButtons = {searchBook, showBooks, rentalBook, returnBook,
wearBook, deleteBook, exit};

        for(JButton buttonComponent : jButtons){
            buttonComponent.setFont(font);
            buttonComponent.setBorderPainted(false);
            buttonComponent.setOpaque(true);
            buttonComponent.setBackground(new Color(255, 158, 0));
        }
    }

//    public void setButtonCondition(JButton btn){ 메소드 폐기
//        btn.setFont(font);
//        btn.setBorderPainted(false);
//        btn.setOpaque(true);
//        btn.setBackground(new Color(255, 158, 0));
//    }

    public void ComponentListener(){
        searchBook.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                mainPage.removeAll();
                JPanel jPanel = new JPanel();
                jPanel.setLayout(new BoxLayout(jPanel, BoxLayout.Y_AXIS));

                JTextField jTextField = new JTextField("찾는 책의 이름을
입력하세요", 30);

```

```

        JTextArea jTextArea = new JTextArea("", 10, 30);
        jTextField.setFont(font);
        jTextArea.setFont(font);
        jPanel.add(jTextField);
        jPanel.add(new JScrollPane(jTextArea));
        mainPage.add(jPanel);
        jTextField.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                String temp = ((JTextField) e.getSource()).getText();
                book book = dataHashMap.get(temp);
                if (book == null) {
                    jTextArea.setText("해당하는 책이 없습니다.");
                } else {
                    jTextArea.setText(book.toString());
                }
            }
        });
        revalidate();
        repaint();
    }
});

showBooks.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        mainPage.removeAll();
        JPanel jPanel = new JPanel();
        jPanel.setLayout(new BoxLayout(jPanel, BoxLayout.Y_AXIS));

        JLabel temp;
        temp = new JLabel("도서명 | 저자 | 출판사 | 대여가능여부");
        temp.setAlignmentX(CENTER_ALIGNMENT);
        jPanel.add(temp);
        temp.setFont(font);

        try (FileReader fReader = new FileReader("bookData.txt");
            Scanner bookReader = new Scanner(fReader)) {

            while (bookReader.hasNext()) {
                temp = new JLabel(bookReader.nextLine());
                temp.setFont(font);
                temp.setAlignmentX(CENTER_ALIGNMENT);
                jPanel.add(temp);
            }

        } catch (FileNotFoundException ept) {
            ept.printStackTrace();
        } catch (IOException ept) {
            ept.printStackTrace();
        }
        mainPage.add(jPanel);
        revalidate();
        repaint();
    }
});

rentalBook.addActionListener(new ActionListener() {

```

```

@Override
public void actionPerformed(ActionEvent e) {

    mainPage.removeAll();
    JTextField jtf = new JTextField("대여하실 책의 이름을 입력하세요",
35);

    mainPage.add(jtf);
    jtf.addActionListener(new ActionListener() {
        @Override
        public void actionPerformed(ActionEvent e) {
            String temp = ((JTextField) e.getSource()).getText();
            book book = dataHashMap.get(temp);
            if (book == null) {
                jtf.setText("해당하는 책이 없습니다.");
            } else {
                if (book.isrental) jtf.setText("이미 대여중인
책입니다.");

                else {
                    book.setRental(true);
                    saveData();
                    jtf.setText("대여완료되었습니다.");
                }
            }
        }
    });
    revalidate();
    repaint();
}

returnBook.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        mainPage.removeAll();
        JTextField jtf = new JTextField("반납하실 책의 이름을 입력하세요",
35);

        mainPage.add(jtf);
        jtf.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                String temp = ((JTextField) e.getSource()).getText();
                book book = dataHashMap.get(temp);
                if (book == null) {
                    jtf.setText("해당하는 책이 없습니다.");
                } else {
                    if (!book.isrental) jtf.setText("대출한적이 없는
책입니다.");

                    else {
                        book.setRental(false);
                        saveData();
                        jtf.setText("반납완료되었습니다.");
                    }
                }
            }
        });
    }
});

```

```

        revalidate();
        repaint();
    }
});

wearBook.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        mainPage.removeAll();

        JTextField jtf = new JTextField("입고된 책의 제목, 저자, 출판사를
차례로 입력하세요", 35);
        mainPage.add(jtf);
        jtf.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                String temp = ((JTextField) e.getSource()).getText() +
",대여가능"; //책 제목, 저자, 출판사 입력값 가져오기
                String[] splitTemp = temp.split(",");
                try(FileWriter fWriter = new FileWriter("bookData.txt",
true)){
                    if(dataHashMap.containsKey(splitTemp[0])){ // Index
에러에 취약.

                        jtf.setText("이미 있는 책입니다 : 제목,저자,출판사를
차례대로 입력하세요");
                    }else {
                        try { //bookData 에 새로운 책 등록
                            fWriter.write("\n" +temp);
                            fWriter.flush();
                            book b = new book(splitTemp[0], splitTemp[1],
splitTemp[2], false);

                            dataHashMap.put(splitTemp[0], b);
                        } catch (IOException e1) {
                            e1.printStackTrace();
                        }
                    }
                } catch (IndexOutOfBoundsException e2){
                    jtf.setText("형식을 지켜서 입력하세요 >
제목,저자,출판사");
                } catch (IOException e2){
                    jtf.setText("IOException 발생 다시 > 제목,저자,출판사");
                }
            }
        });
        revalidate();
        repaint();
    }
});

deleteBook.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        mainPage.removeAll();

        JTextField jtf = new JTextField("삭제하실 책의 제목을 입력하세요",

```

```

35);

    mainPage.add(jtf);
    jtf.addActionListener(new ActionListener() {
        @Override
        public void actionPerformed(ActionEvent e) {
            String temp = ((JTextField) e.getSource()).getText();
            book book = dataHashMap.get(temp);
            if (book == null) {
                jtf.setText("해당하는 책이 없습니다.");
            } else {
                jtf.setText(book.toString() + "를 삭제합니다.");
                try {
                    InputStream is = new
FileInputStream("bookData.txt");
                } catch (FileNotFoundException e3) {
                    e3.printStackTrace();
                }
                String result = "";
                try {
                    BufferedReader br = new BufferedReader(new
FileReader(new File("bookData.txt")));
                    BufferedWriter bw = new BufferedWriter(new
FileWriter(new File("newbookData.txt")));

                    while ((result = br.readLine()) != null) {
                        if (result.startsWith(temp)) {
                            result = result.replace(book.toString(),
""); //행 삭제
                        }
                        bw.write(result);
                        bw.flush();
                    }
                    bw.close();
                    br.close();

                    File file1 = new File("bookData.txt"); //원래 파일
                    File file2 = new File("newbookData.txt"); //책을
삭제한 후의 파일

                    file1.delete(); //원래 파일 삭제
                    file2.renameTo(new File("bookData.txt")); //파일
이름 변경

                } catch (FileNotFoundException e1) {
                    e1.printStackTrace();
                } catch (IOException e2) {
                    e2.printStackTrace();
                }
                jtf.setText("삭제되었습니다.");
            }
        }
    });
    revalidate();
    repaint();

```

```

    }
    });

    exit.addActionListener(new ActionListener() { // 서준호 메소드 보완,
종료 기능 구현
        @Override
        public void actionPerformed(ActionEvent e) {
            System.exit(0);
        }
    });
}

public void saveData() { // 신현빈 제작 메소드, 텍스트파일 변경
    try (FileWriter fileWriter = new FileWriter("bookData.txt")) {
        PrintWriter printWriter = new PrintWriter(fileWriter);
        for (Entry<String, book> entrySet : dataHashMap.entrySet()) {
            printWriter.print(entrySet.getValue().toString() + "\n");
        }
    } catch (FileNotFoundException ept) {
        ept.printStackTrace();
    } catch (IOException ept) {
        ept.printStackTrace();
    }
}

private void initiateSetting() { // 안상완 제작 메소드, 시작시에 파일로부터
데이터를 읽고, 해당 데이터를 분류하여 HsahMap에 저장한다.
    dataHashMap = new HashMap<String, book>();
    try (FileReader fReader = new FileReader("bookData.txt");
        Scanner bookReader = new Scanner(fReader)) {

        while (bookReader.hasNext()) {
            String[] temp = bookReader.nextLine().split(",");
            book b = new book(temp[0], temp[1], temp[2],
(temp[3].equals("대여중")) ? true : false);
            dataHashMap.put(temp[0], b);
        }
    } catch (FileNotFoundException ept) {
        ept.printStackTrace();
    } catch (IOException ept) {
        ept.printStackTrace();
    }
}

class book{
    private String name, writer, publisher;
    Boolean Isrental;

    public book(String name, String writer, String publisher, Boolean b) {
        this.name = name;
        this.writer = writer;
        this.publisher = publisher;
        this.Isrental = b;
    }

    public void setRental(Boolean b){
        Isrental = b;
    }
}

```

```
}

@Override
public String toString(){
    String s = name + "," + writer + "," + publisher + ",";
    if(Isrental) s += "대여중";
    else s += "대여가능";
    return s;
}
}
```