자바 프로그래밍 포트폴리오 시험 평가

a **시험 시간: 50분**

프로젝트 주제: 도서 관리 시스템

1. 요구사항 분석 및 설계 (10점)

1. **요구사항 분석** (5점)
    - 도서 관리 시스템의 요구사항을 분석하고, 다음과 같은 기능들을 포함하도록 시스템을 설계하시오.
        - 도서 등록, 조회, 수정, 삭제
        - 회원 등록, 조회, 수정, 삭제
        - 도서 대여 및 반납
        - 도서 검색 (제목, 저자, 카테고리 등으로 검색)
2. **클래스 정의** (5점)
    - `Person`, `Member`, `Book`, `BorrowRecord` 클래스를 정의하고, 이들 간의 상속 및 연관 관계를 설계하세요.

2. 클래스 구현 (20점)

1. **기본 클래스 구현** (10점)
    - `Person` 클래스를 작성하고, 이를 상속하는 `Member` 클래스를 작성하시오. 각 클래스는 다음과 같은 필드와 메서드를 가져야 합니다.
        - `Person`: `String name`, `String email`, getter와 setter 메서드, `toString` 메서드
        - `Member`: `String memberId`, getter와 setter 메서드, `toString` 메서드 package javaprogrammingunitexam;

public class Person { private String name; private String email;

```
public Person(String name, String email)
{
    this.name = name;
    this.email = email;
}

public String getName() {
    return name;
}

public void setName(String name) {
    this.name = name;
}

public String getEmail() {
    return email;
```

```
    }

    public void setEmail(String email) {
        this.email = email;
    }

    public String tostring()
    {
        return this.toString();
    }

    @Override
    public String toString() {
        return "Person [name=" + name + ", email=" + email + "]";
    }
```

}

package javaprogrammingunitexam;

public class Member extends Person { private String memberId;

```
    public Member(String name, String email, String memberId) {
        super(name, email);
        this.memberId = memberId;
        // TODO Auto-generated constructor stub
    }

    public String getMemberId() {
        return memberId;
    }
    public void setMemberId(String memberId) {
        this.memberId = memberId;
    }


    public String tostring (Member m)
    {
        return m.toString();
    }
```

}

2. **도서 및 대여 기록 클래스 구현** (10점)
    - `Book` 클래스를 작성하시오. 이 클래스는 `String bookId`, `String title`, `String author`, `String category`, `boolean isAvailable` 필드를 가져야 합니다.
    - `BorrowRecord` 클래스를 작성하시오. 이 클래스는 `Book book`, `Member member`, `LocalDate borrowDate`, `LocalDate returnDate` 필드를 가져야 합니다.

package javaprogrammingunitexam;

public class Book {

```
private String bookId;
private String title;
private String author;
private String category;
private boolean isAvailable;

public Book(String bookId, String title, String author,String category, boolean
isAvailable)
{
    this.bookId = bookId;
    this.title = title;
    this.author = author;
    this.category = category;
    this.isAvailable = isAvailable;
}

public String getBookId() {
    return bookId;
}

public void setBookId(String bookId) {
    this.bookId = bookId;
}

public String getTitle() {
    return title;
}

public void setTitle(String title) {
    this.title = title;
}

public String getAuthor() {
    return author;
}

public void setAuthor(String author) {
    this.author = author;
}

public String getCategory() {
    return category;
}

public void setCategory(String category) {
    this.category = category;
```

```
    }

    public boolean isAvailable() {
        return isAvailable;
    }

    public void setAvailable(boolean isAvailable) {
        this.isAvailable = isAvailable;
    }

    @Override
    public String toString() {
        return "Book [bookId=" + bookId + ", title=" + title + ", author=" + author +
", category=" + category
                + ", isAvailable=" + isAvailable + "]";
    }
```

}

package javaprogrammingunitexam;

import java.time.LocalDate;

public class BorrowRecord { private Book book; private Member member; private LocalDate borrowDate; private LocalDate returnDate;

```
    public BorrowRecord(Book book, Member member, LocalDate borrowDate, LocalDate
returnDate) {
        super();
        this.book = book;
        this.member = member;
        this.borrowDate = borrowDate;
        this.returnDate = returnDate;
    }




    public Book getBook() {
        return book;
    }




    public void setBook(Book book) {
        this.book = book;
    }
```

```
public Member getMember() {
    return member;
}



public void setMember(Member member) {
    this.member = member;
}



public LocalDate getBorrowDate() {
    return borrowDate;
}



@Override
public String toString() {
    return "BorrowRecord [book=" + book + ", member=" + member + ", borrowDate=" +
borrowDate + ", returnDate="
            + returnDate + "]";
}



public void setBorrowDate(LocalDate borrowDate) {
    this.borrowDate = borrowDate;
}



public LocalDate getReturnDate() {
    return returnDate;
}



public void setReturnDate(LocalDate returnDate) {
    this.returnDate = returnDate;
}
```

}

```
```

3. 제너릭 및 컬렉션 사용 (10점)

1. **제너릭 클래스 작성** (5점)
    - 다양한 타입의 데이터를 저장할 수 있는 `Library` 클래스를 작성하시오. 이 클래스는 제너릭 타입 `T`를 사용하여 도서 및 대여 기록을 저장할 수 있어야 합니다.
    - 도서 리스트에서 특정 조건에 맞는 도서를 검색하는 메서드를 작성하시오. 조건으로는 제목, 저자, 카테고리 등이 포함될 수 있습니다.
    - 람다 표현식을 사용하여 검색 조건을 처리하시오.

package javaprogrammingunitexam;

import java.util.*; import java.util.function.Predicate;

public class Library { private List items;

```
 public Library (List<T> item)
 {
     items = item;
 }

 public void plusitems(T t)
 {
     items.add(t);
 }

 public void remocveitem(T t)
 {

     items.remove(t);
}

 public T getitem(int a)
 {
     return items.get(a);
 }
```

// public List searchItems(Predicate condition) { // // return // // code를 작성하세요 // }

```
 @Override
public String toString() {
```

```
    return "Library [items=" + items + "]";
}

public String tostring()
 {
     return items.toString();
 }
```

}

package javaprogrammingunitexam;

public class Main {

```
public static void main(String[] args) {
    // TODO Auto-generated method stub



}
```

}

4. 사용자 인터페이스 구현 (10점)

1. **콘솔 기반 사용자 인터페이스 구현** (10점)
    - 사용자로부터 입력을 받아 도서, 회원, 대여 기록을 등록, 조회, 수정, 삭제할 수 있는 콘솔 기반의 인터페이스를 구현하시오.
    - 도서 검색 기능을 추가하시오. (제목, 저자, 카테고리 등으로 검색)
    - 각 기능이 정상적으로 동작하는지 테스트하시오.

import java.util.*; import java.time.LocalDate;

public class LibraryManagementSystem { private Library bookLibrary; private Library borrowLibrary; private List members; private Scanner scanner;

```
Book mbooks = new Book("javabook", "aaa", "bbb", "cccc", false);
// 각 필드 초기화 생성자 구현
// code를 작성하세요

public static void main(String[] args) {
LibraryManagementSystem system = new LibraryManagementSystem();
system.run();
}
public void run() {
while (true) {
System.out.println("메뉴:");
System.out.println("1. 도서 등록");
```

```
    System.out.println("2. 도서 조회");
    System.out.println("3. 도서 수정");
    System.out.println("4. 도서 삭제");
    System.out.println("5. 회원 등록");
    System.out.println("6. 회원 조회");
    System.out.println("7. 회원 수정");
    System.out.println("8. 회원 삭제");
    System.out.println("9. 도서 대여");
    System.out.println("10. 도서 반납");
    System.out.println("11. 도서 검색");
    System.out.println("0. 종료");
    System.out.print("선택: ");
    int choice = scanner.nextInt();
    scanner.nextLine(); // 줄바꿈 문자 소비
    switch (choice) {
    case 1:
    addBook();
    break;
    case 2:
    viewBook();
    break;
    case 3:
    updateBook();
    break;
    case 4:
    deleteBook();
    break;
    case 5:
    addMember();
    break;
    case 6:
        viewMember();
        break;
        case 7:
        updateMember();
        break;
        case 8:
        deleteMember();
        break;
        case 9:
        borrowBook();
        break;
        case 10:
        returnBook();
        break;
        case 11:
        searchBooks();
        break;
        case 0:
        System.out.println("프로그램을 종료합니다.");
        return;
```

```
        default:
        System.out.println("잘못된 선택입니다. 다시 시도하세요.");
        }
        }
        }

    private void addBook() {
        System.out.print("도서ID: ");
        String bookId = scanner.nextLine();
        System.out.print("제목: ");
        String title = scanner.nextLine();
        System.out.print("저자: ");
        String author = scanner.nextLine();
        System.out.print("카테고리: ");
        String category = scanner.nextLine();
        mbook
        System.out.println("도서가 등록되었습니다.");
        }
        private void viewBook() {
        System.out.print("도서ID: ");
        String bookId = scanner.nextLine();
        // 위 bookId 로컬 변수로 bookLibrary에서 검색하는 코드를 작성하세요.
        // code 를 작성하세요
        if (books.isEmpty()) {
        System.out.println("해당 도서ID의 도서가 없습니다.");
        } else {
        books.forEach(System.out::println);

        }
        }
        private void updateBook() {
        System.out.print("도서ID: ");
        String bookId = scanner.nextLine();
        // 위 bookId 로컬 변수로 bookLibrary에서 검색하는 코드를 작성하세요.
        // code를 작성하세요
        if (books.isEmpty()) {
        System.out.println("해당 도서ID의 도서가 없습니다.");
        return;
        }
        Book book = books.get(0);
        System.out.print("새 제목: ");
        String title = scanner.nextLine();
        System.out.print("새 저자: ");
        String author = scanner.nextLine();
        System.out.print("새 카테고리: ");
        String category = scanner.nextLine();
        // 해당 도서의 정보를 수정하는 코드를 작성하세요
        // code를 작성하세요
        System.out.println("도서 정보가 수정되었습니다.");
        }
        private void deleteBook() {
```

```
        System.out.print("도서ID: ");
        String bookId = scanner.nextLine();
        // 위 bookId 로컬 변수로 bookLibrary에서 검색하는 코드를 작성하세요.
        // code를 작성하세요

        if (books.isEmpty()) {
        System.out.println("해당 도서ID의 도서가 없습니다.");
        return;
        }
        // 해당 도서를 삭제하는 코드를 작성하세요
        // code를 작성하세요
        System.out.println("도서가 삭제되었습니다.");
        }
        private void addMember() {
        System.out.print("이름: ");
        String name = scanner.nextLine();
        System.out.print("이메일: ");
        String email = scanner.nextLine();
        System.out.print("회원ID: ");
        String memberId = scanner.nextLine();
        members.add(new Member(name, email, memberId));
        System.out.println("회원이 등록되었습니다.");
        }
        private void viewMember() {
        System.out.print("회원ID: ");
        String memberId = scanner.nextLine();
        // 해당 멤버를 검색하는 코드를 작성하세요
        // code를 작성하세요

        System.out.println("해당 회원ID의 회원이 없습니다.");
        }
        private void updateMember() {
        System.out.print("회원ID: ");
        String memberId = scanner.nextLine();
        for (Member member : members) {
        // code를 작성하세요
        }
        System.out.println("해당 회원ID의 회원이 없습니다.");
        }
        private void deleteMember() {
        System.out.print("회원ID: ");
        String memberId = scanner.nextLine();
        for (Member member : members) {
        // code를 작성하세요
        }
        System.out.println("해당 회원ID의 회원이 없습니다.");
        }
        private void borrowBook() {
        System.out.print("회원ID: ");
        String memberId = scanner.nextLine();
        Member member = null;
```

```
        for (Member m : members) {
        if (m.getMemberId().equals(memberId)) {
        member = m;
        break;
        }
        }
        if (member == null) {
        System.out.println("해당 회원ID의 회원이 없습니다.");
        return;
        }
        // code를 작성하세요
        System.out.println("도서가 대여되었습니다.");
        }
        private void returnBook() {
        System.out.print("도서ID: ");
        String bookId = scanner.nextLine();

        List<BorrowRecord> records = borrowLibrary.searchItems(record ->
        record.getBook().getBookId().equals(bookId) && record.getReturnDate() ==
        null);
         if (records.isEmpty()) {
         System.out.println("해당 도서ID의 대여 기록이 없습니다.");
         return;
         }
         // code를 작성하세요
         System.out.println("도서가 반납되었습니다.");
         }
         private void searchBooks() {
         System.out.println("검색 조건을 선택하세요:");
         System.out.println("1. 제목");
         System.out.println("2. 저자");
         System.out.println("3. 카테고리");
         System.out.print("선택: ");
         int choice = scanner.nextInt();
         scanner.nextLine(); // 줄바꿈 문자 소비
         switch (choice) {
         case 1:
         System.out.print("제목: ");
         String title = scanner.nextLine();
         List<Book> booksByTitle = bookLibrary.searchItems(book ->
        book.getTitle().equalsIgnoreCase(title));
         booksByTitle.forEach(System.out::println);
         break;
         case 2:
         System.out.print("저자: ");
         String author = scanner.nextLine();
         List<Book> booksByAuthor = bookLibrary.searchItems(book ->
        book.getAuthor().equalsIgnoreCase(author));
         booksByAuthor.forEach(System.out::println);
         break;
         case 3:
```

```
        System.out.print("카테고리: ");
        String category = scanner.nextLine();
        List<Book> booksByCategory = bookLibrary.searchItems(book ->
      book.getCategory().equalsIgnoreCase(category));
        booksByCategory.forEach(System.out::println);
        break;
        default:
        System.out.println("잘못된 선택입니다.");
        }
        }
      }
```

각 항목에 대해 요구되는 내용을 충실히 작성하고 제출하시면 됩니다. 프로젝트는 클래스 설계, 구현, 그리고 테스트로 구성되어 있으며, 각 단계별로 점수가 부여됩니다. 성실하게 작성해 주시기 바랍니다.