REPORT

Library system

목 차

과목명: 객체지향 프로그래밍

학과: 컴퓨터공학과

학번:

이름: 오지현 양서희 문규인 황지상

1. 문제 명세

2. 문제 분석

3. 설계

3-1 UML 다이어그램

3-2 클래스 설명

문제 명세

Library System

1. 책

1. 도서 관리(조회, 삽입, 삭제 기능)
2. 도서 대여. 회원 등록, 로그인(회원 구분: 학생, 대학원생, 교수 등)
3. 도서 검색 (카테고리, 제목, 저자명, 고유번호)

4-1. 도서 검색 후 책 정보 출력(저자, 출판사, 장르, 리뷰 등)

2. 회원

1. 회원 등록, 로그인(회원 구분: 학생, 대학원생, 교수 등)
2. 연체 상태 보이기
3. 읽었던 책의 별점과 리뷰 등록
4. 대여했던 도서 기록

3. 관리자

1. 회원 관리(조회, 삭제 기능)

2. 도서 관리(추가, 삭제 기능)

문제 분석

클래스 계층구조:

텍스트, 스크린샷, 도표, 직사각형이(가) 표시된 사진

자동 생성된 설명

설계

텍스트, 스크린샷, 도표, 폰트이(가) 표시된 사진

자동 생성된 설명

Member 클래스:

도서관 회원을 나타내며, 아이디, 비밀번호, 이름을 필드로 갖고 있습니다. 또한 회원이 대여한 책의 정보를 저장하기 위한 Book 클래스 배열과 최대 저장 가능한 책의 수, 책 대여 가능 기간을 포함합니다. 생성자, 접근자, 설정자를 포함합니다. 정보를 보여주는 display 메서드가 있습니다.

Book 클래스:

책의 정보를 담는 클래스로, 제목, 저자, 출판사, 장르, 고유번호 등의 필드를 포함합니다. 생성자, 접근자, 설정자를 포함합니다. 정보를 보여주는 display 메서드가 있습니다

Member\_App 클래스:

Member\_App 이라는 클래스는 로그인 회원가입과 같이 회원에 관련하여 화면에 보이고 정보를 입력 받기 위해 만든 클래스이다.

System: MemberSystem 객체

Run(): 사용자가 원하는 메뉴를 실행시킴

Showmenu():메서드는 어떤 메뉴가 있는지 보임

getChoice(): 메서드는 선택한 메뉴 번호 값을 반환

Login (): 메서드는 로그인 과정을 보임

signup(): 클래스는 회원가입 과정을 보임

MemberSystem 클래스:

MemberSystem 클래스는 입력받은 정보를 처리하기 위해서 만들어진 클래스이다.

첫번째로는 members라는 Member(회원)객체 배열을 가지고 있다. public에는 signup(회원가입),login(로그인) 기능의 메서드를 가지고 있다.

또한 멤버 정보를 가지고 있는 파일을 읽어들이거나 저장하는 load\_member\_file, save\_member\_file 함수를 system을 상속받아 가지고 있다.

signup(도서 검색) 함수 :

1. 시퀀스 다이어그램

회원가입 요청

Member\_system

Member\_app

USER

입력 받은 회원정보 멤버 객체화

입력 받은 회원의 정보

파일에 멤버 정보 저장

회원가입 결과 반환

결과 출력

1. 알고리즘 서술

signup(Member& member){

members.push\_back(member);

save\_Member\_File();

}

입력 받은 정보를 member 객체화 시켜 memberSystem에 있는 members 객체 배열에 저장

그 이후 파일에 입력

# login(도서 검색) 함수 :

1. 시퀀스 다이어그램

로그인 요청

Member\_system

Member\_app

USER

입력 받은 로그인정보 멤버 객체화

입력 받은 로그인 정보

로그인 정보랑 회원 정보랑 비교

로그인 결과 반환

결과 출력

2)알고리즘 서술

bool login(const string& id, const string& password) {

for const Member& member : members {

if (member.getId() == id && member.getPassword() == password) then{

출력 << "로그인 성공!"

return true;

}

}

출력<< "로그인 실패. 아이디나 비밀번호가 일치하지 않습니다."

return false;

}

BookApp 클래스:

Book\_App 이라는 클래스는 로그인 회원가입과 같이 회원에 관련하여 화면에 보이고 정보를 입력 받기 위해 만든 클래스이다.

System: BookSystem 객체

Run(): 사용자가 원하는 메뉴를 실행시킴

Showmenu(): 어떤 메뉴가 있는지 보임

getChoice(): 선택한 메뉴 번호 값을 반환

rentBook() : 책을 빌리는 과정을 보임

returnBook(): 책을 반납하는 과정을 보임

BookSystem 클래스:

Rent\_book(도서 대여) 함수 :

1)시퀀스 다이어그램

BookSystem

Book\_App

USER

도서 대여 요청

입력 받은 도서 정보

입력 받은 도서가 대여 중인지 확인 후 대여

대여 결과 반환

결과 출력

1. 알고리즘 기술

bool rentBook(const string& memberId, const string& bookName) {

auto it = find\_if(members.begin(), members.end(), [&](const Member& member) {

return member.getId() == memberId;

});

if (it != members.end()) then {

it->addRentedBook(bookName);

save\_Member\_File();

출력: bookName "이(가) 대여되었습니다."

return true;

}

else then {

출력: "해당 아이디의 회원을 찾을 수 없습니다."

return false;

}

}

Return\_book(도서 반납) 함수 :

1)시퀀스 다이어그램

BookSystem

Book\_App

USER

도서 반납 요청

입력 받은 도서 정보

해당 도서를 빌린 회원이 맞는지 확인후 반납

반납 결과 반환

결과 출력

2)알고리즘 기술

bool returnBook(const string& memberId, const string& bookName) {

auto it = find\_if(members.begin(), members.end(), [&](const Member& member) {

return member.getId() == memberId;

});

if (it != members.end()) then{

it->removeRentedBook(bookName);

save\_Member\_File();

출력: bookName "이(가) 반납되었습니다." ;

return true;

}

else then{

출력:"해당 아이디의 회원을 찾을 수 없습니다." ;

return false;

}

}

Search\_book(도서 검색) 함수 :

1)시퀀스 다이어그램

BookSystem

Book\_App

USER

도서 검색 요청

입력 받은 도서 정보

입력 받은 정보와 책 정보들 비교

알맞은 책의 정보 가져옴

해당 도서 정보 출력

2)알고리즘 기술

void searchBooks(const string& searchKeyword) {

bool found = false;

cout << "\"" << searchKeyword << "\"에 대한 검색결과" << endl;

for (const Book& book : books)

if (book.getBookName().find(searchKeyword) != string::npos || book.getWriter().find(searchKeyword) != string::npos) then {

출력: 제목, 저자 출판사 ,장르.고유번호,페이지수,대여여부;

found = true;

}

}

if (!found) then

cout << "해당 검색어의 결과를 찾을 수 없습니다." << endl;

}

}

LibrarianApp 클래스:

Book\_App 이라는 클래스는 로그인 회원가입과 같이 회원에 관련하여 화면에 보이고 정보를 입력 받기 위해 만든 클래스이다.

System: BookSystem 객체

Run(): 사용자가 원하는 메뉴를 실행시킴

Showmenu(): 어떤 메뉴가 있는지 보임

getChoice(): 선택한 메뉴 번호 값을 반환

removeBook() : 책을 파일에서 제거하는 과정

AddBook(): 책을 파일에 추가하는 과정

removeMember(): 멤버를 파일에서 제거하는 과정

displayMember(): 멤버를 보이고 조회하는 과정

BookSystem 클래스:

add\_book(도서 추가) 함수 :

1)시퀀스 다이어그램

librarian\_System

librarian\_App

USER

도서 추가 요청

입력 받은 도서 정보

입력 받은 도서 파일에 추가

도서 추가 결과 반환

결과 출력

2)알고리즘 기술

void add\_Book(const Book& book) {

books.push\_back(book);

save\_Book\_File();

}

remove\_book(도서 삭제) 함수 :

1)시퀀스 다이어그램

librarian\_System

librarian\_App

USER

도서 삭제 요청

입력 받은 도서 정보

입력 받은 도서 파일에 있는 도서 정보와 비교 후 삭제

도서 삭제 결과 반환

삭제 출력

2)알고리즘 기술

void remove\_Book(const string& book\_Name) {

auto it = std::find\_if(books.begin(), books.end(), [&](const Book& book) {

return book.getBookName() == book\_Name;

});

if (it != books.end()) then{

books.erase(it);

save\_Book\_File();

출력: book\_Name "이 삭제 되었습니다.";

}

else then

출력: book\_Name "이라는 도서가 없습니다.";

}

deleteMember(회원 삭제) 함수 :

1. 시퀀스 다이어그램

회원 삭제 요청

librarian\_system

librarian\_app

USER

입력 받은 회원 아이디 정보

회원 아이디 정보와 회원들 정보 비교 후 삭제

삭제 결과 반환

결과 출력

1. 알고리즘 기술

bool deleteMember(const string& id) {

auto it = find\_if(members.begin(), members.end(), [&](const Member& member) {

return member.getId() == id;

});

if (it != members.end()) then{

members.erase(it);

Load\_Member\_File();

출력: id " 회원이 삭제되었습니다.";

return true;

}

else then

출력: "해당 아이디의 회원을 찾을 수 없습니다.";

return false;

}

displayMembers(회원 조회) 함수 :

1. 시퀀스 다이어그램

회원 조회 요청

librarian\_system

librarian\_app

USER

입력 받은 회원 아이디 정보

회원 아이디 정보와 회원들 정보 비교 후 조회

조회 결과 반환

결과 출력

2)알고리즘 기술

void displayMembers() {

if (members.empty ) then {

출력:"회원 목록이 비어 있습니다.";

}

else {

출력:"회원 목록: ";

for (size\_t i = 0; i < members.size(); ++i) {

출력: "회원 번호: " << i + 1 ;

members[i].display();

}

}

}