**객지\_텀프로젝트\_도서검색 및 출력**  양서희

* 맡은 부분: 2번

|  |
| --- |
| 3. 도서 검색 (카테고리, 제목, 저자명)  4. 검색하면 책 정보 간략하게 뜨기 (저자, 출판사, 장르, 페이지 수 등), 책 검색하고 선택하면 리뷰 보이게 |

1. 문제 명세

도서관에서 사용할 수 있는 시스템 만들기

* 두 가지 측면에서 생각 가능

1. 도서관 관리(관리자 로그인으로 접근) – 새로운 책을 삽입, 사용 불가능 책을 삭제
2. 도서관 이용(회원 로그인으로 접근) – 로그인(회원-학부생, 대학원생, 교수), 책 정보 검색(제목, 저자, 출판사, 분야, 고유번호, 페이지 수, 대여 여부, 평점, 리뷰), 책 대여, 책 반납
3. 분석(클래스 계층 구조, 클래스 다이어그램)
4. 시퀀스 다이어그램

library

검색이 유효하면

Book

USER

파일이 열리면

리뷰 파일 오픈

리뷰 불러옴

파일이 열리지 않으면 오류 출력

해당 도서 정보 출력

1. UML 다이어그램

|  |
| --- |
| Book class |
| * bookname : string * writer : string * publisher :string * genre : string * numbering : float * star\_point : float * review : string * page : int * rent : bool |
| 생성자, 접근자, 설정자 생략  + saveReviewToFile() : void  + readReviewFile() : void |

전역함수 AddBookToLibrary() / searchBooks()

1. 설계(클래스 설계, 함수 알고리즘)

\*Vector로 책 정보를 저장해둠. -> #include <vector> 추가

\*파일 읽어옴 -> #include <fstream> 추가

\*이름공간 std사용

\*void Book::readReviewFile() 에서 예외 처리 사용

사용자가 검색어를 입력하면 해당 검색어와 제목이나 저자가 일치하는 도서의 정보를 출력함. 일부만 동일하여도 출력되게 함.

리뷰는 사용자가 등록할 때 파일에 저장하였다고 가정 -> 검색하는 도서의 리뷰 파일에서 불러와 도서 정보를 출력할 때 함께 출력

[코드]

* search.h

|  |
| --- |
| #pragma once  #ifndef SEARCH\_H  #define SEARCH\_H  #include<iostream>  #include<string>  #include <vector> //추가  #include <fstream> //추가  using namespace std; //추가  class Book {  private:  string bookname; // 책이름  string writer; // 저자  string publisher; // 출판사  string genre; // 책 장르(분야)  float numbering; // 책 고유번호  float star\_point; // 별점  string review; // 리뷰  int page; // 페이지 수  bool rent; // 대여 여부  public:  // 생성자  Book(const std::string& bookName, const std::string& writer, const std::string& publisher, const std::string& genre, float numbering, float starPoint, const std::string& review, int page, bool rent);  // 접근자 (Getter)  string getBookName() const;  string getWriter() const;  string getPublisher() const;  string getGenre() const;  float getNumbering() const;  float getStarPoint() const;  string getReview() const;  int getPage() const;  bool isRented() const;  // 수정자 (Setter)  void setBookName(const std::string& bookName);  void setWriter(const std::string& writer);  void setPublisher(const std::string& publisher);  void setGenre(const std::string& genre);  void setNumbering(float numbering);  void setStarPoint(float starPoint);  void setReview(const std::string& review);  void setPage(int page);  void setRentStatus(bool rent);  //추가 함수  void saveReviewToFile() const;  void readReviewFile();  };  #endif  #define BOOK\_H  #include<iostream>  #include<string>  #include <vector> //추가  #include <fstream> //추가  using namespace std; //추가  class Book {  private:  string bookname; // 책이름  string writer; // 저자  string publisher; // 출판사  string genre; // 책 장르(분야)  float numbering; // 책 고유번호  float star\_point; // 별점  string review; // 리뷰  int page; // 페이지 수  bool rent; // 대여 여부  public:  // 생성자  Book(const std::string& bookName, const std::string& writer, const std::string& publisher, const std::string& genre, float numbering, float starPoint, const std::string& review, int page, bool rent);  // 접근자 (Getter)  string getBookName() const;  string getWriter() const;  string getPublisher() const;  string getGenre() const;  float getNumbering() const;  float getStarPoint() const;  string getReview() const;  int getPage() const;  bool isRented() const;  // 수정자 (Setter)  void setBookName(const std::string& bookName);  void setWriter(const std::string& writer);  void setPublisher(const std::string& publisher);  void setGenre(const std::string& genre);  void setNumbering(float numbering);  void setStarPoint(float starPoint);  void setReview(const std::string& review);  void setPage(int page);  void setRentStatus(bool rent);  //추가 함수  void saveReviewToFile() const;  void readReviewFile();  };  #endif |

* search.cpp

|  |
| --- |
| #include "search.h"  // 생성자 구현  Book::Book(const std::string& bookName, const std::string& writer, const std::string& publisher, const std::string& genre, float numbering, float starPoint, const std::string& review, int page, bool rent)  : bookname(bookName), writer(writer), publisher(publisher), genre(genre), numbering(numbering), star\_point(starPoint), review(review), page(page), rent(rent) {}  // 접근자 메서드 구현  std::string Book::getBookName() const {  return bookname;  }  std::string Book::getWriter() const {  return writer;  }  std::string Book::getPublisher() const {  return publisher;  }  std::string Book::getGenre() const {  return genre;  }  float Book::getNumbering() const {  return numbering;  }  float Book::getStarPoint() const {  return star\_point;  }  std::string Book::getReview() const {  return review;  }  int Book::getPage() const {  return page;  }  bool Book::isRented() const {  return rent;  }  // 수정자 메서드 구현  void Book::setBookName(const std::string& bookName) {  bookname = bookName;  }  void Book::setWriter(const std::string& writer) {  this->writer = writer;  }  void Book::setPublisher(const std::string& publisher) {  this->publisher = publisher;  }  void Book::setGenre(const std::string& genre) {  this->genre = genre;  }  void Book::setNumbering(float numbering) {  this->numbering = numbering;  }  void Book::setStarPoint(float starPoint) {  star\_point = starPoint;  }  void Book::setReview(const std::string& review) {  this->review = review;  }  void Book::setPage(int page) {  this->page = page;  }  void Book::setRentStatus(bool rent) {  this->rent = rent;  }  //-----------코드 추가 부분------------  //도서를 벡터로 저장(코드 합치면 없어질듯)  void AddBookToLibrary(vector<Book>& library, const Book& book) {  library.push\_back(book); //벡터에 객체 저장  }  //도서 검색  void searchBooks(const vector<Book>& library, const string& searchKeyword) {  bool found = false; //부울 변수를 거짓으로 지정. found는 검색어가 유효한지를 나타냄  cout << "\"" << searchKeyword << "\"에 대한 검색결과" << endl;  for (const Book& book : library) { //library안의 객체를 반복  // 검색어가 도서 이름 또는 저자에 포함되는지 확인  if (book.getBookName().find(searchKeyword) != string::npos || book.getWriter().find(searchKeyword) != string::npos) {  //책 제목이나 저자 중에 검색어를 찾지 못한것이 아니라면; 검색어를 찾았다면  cout << "제목: " << book.getBookName() << endl;  cout << "저자: " << book.getWriter() << endl;  cout << "출판사: " << book.getPublisher() << endl;  cout << "장르: " << book.getGenre() << std::endl;  cout << "고유번호: " << book.getNumbering() << endl;  cout << "페이지 수: " << book.getPage() << endl;  cout << "대여 여부: " << (book.isRented() ? "Yes" : "No") << endl; //대여 여부에 따라 다른 출력  cout << "평점: " << book.getStarPoint() << endl;  cout << "리뷰: " << book.getReview() << endl;  found = true; //found를 true로  }  }  if (!found) { //found가 fasle라면; 찾지 못한다면  cout << "해당 검색어의 결과를 찾을 수 없습니다." << endl;  }  }  //도서 리뷰를 파일에 저장(코드 합치면 없어질 듯)  void Book::saveReviewToFile() const {  ofstream file(getBookName() + "\_review.txt"); //책제목\_review.txt 파일을 쓰기 모드로  if (file.is\_open()) { //파일이 열린다면  file << review; //리뷰를 씀  file.close(); //파일을 닫음  }  else //파일이 없으면 오류  cout << "파일 오류 발생" << endl;  }  //리뷰를 파일에서 불러옴(예외처리 사용)  void Book::readReviewFile() {  try { //예외 처리  ifstream file(getBookName() + "\_review.txt"); //책 제목\_reveiw.txt 파일을 읽기 모드로  if (file.is\_open()) { //파일이 열린다면  getline(file, review); //review의 문장을 파일에 저장  file.close(); //파일을 닫음  }  else { //파일이 열리지 않으면 예외 발생, 예외 메세지 출력  throw runtime\_error("파일을 열 수 없습니다.");  }  }  catch (const exception& e) { //예외 발생 시  cerr << "파일 읽기 예외 발생: " << e.what() << endl; //오류메세지 출력  }  }  int main() {  vector<Book> library;  //(코드 합칠 때는 도서 삽입 때 쓴 것을 불러오도록)  Book book1("도둑맞은 집중력", "요한 하리", "어크로스", "인문", 1.0, 4.5, "", 940, false);  //(코드 합칠 때는 다른 코드에서 입력 받은 리뷰 불러오기)  book1.setReview("좋은 책이네요. 추천합니다!"); //리뷰 저장  book1.saveReviewToFile(); //리뷰 불러옴  AddBookToLibrary(library, book1); //해당 도서를 벡터에 저장  Book book2("모순", "양귀자", "쓰다", "소설", 2.0, 3.8, "", 650, true);  book2.setReview("재밌게 읽었어요.");  book2.saveReviewToFile();  AddBookToLibrary(library, book2);  //검색 전에 모든 객체 마다 파일에서 리뷰를 읽어옴  for (Book& book : library) {  book.readReviewFile();  }  string searchKeyword;  cout << "검색 할 도서의 제목이나 저자의 이름을 입력하세요: ";  getline(cin, searchKeyword); //사용자에게 검색어 입력 받음  searchBooks(library, searchKeyword); //검색어로 도서 검색  return 0;  } |