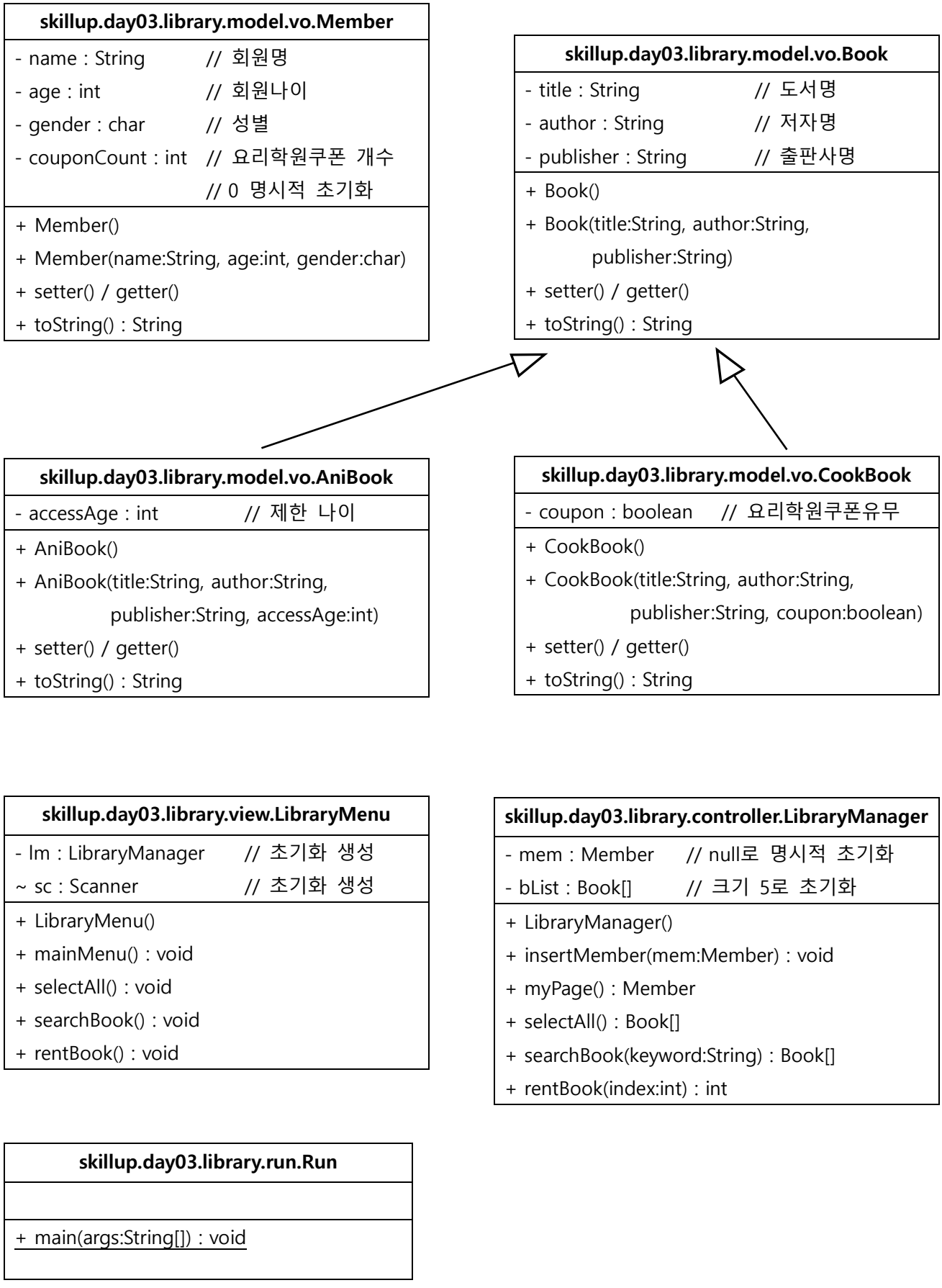
1. 구현 클래스 다이어그램 (Class Diagram)



**** 문제 설명 ****

회원이 만화책 또는 요리책을 빌리려고 한다.

먼저 프로그램이 실행되면 회원의 인적 사항을 입력 받고 그 정보로 회원등록을 해준다. 그 다음에 메인 메뉴가 출력되면서 마이 페이지, 도서전체조회, 도서검색, 도서대여 기능을 할 수 있다. 도서를 대여할 때 해당 도서가 만화책일 경우 나이 제한이 있기 때문에 회원의 나이와 만화책의 제한 나이를 비교해야 되고, 대여할 도서가 요리책일 경우 해당 도서에 요리학원 쿠폰이 있으면 회원에게 쿠폰이 발급된다.

2. 구현 클래스 설명

Package명	Class명	Method	설명
skillup.day03. library.model.vo	Member	+ Member()	기본 생성자
		+ Member(name:String, age:int, gender:char)	3개의 초기값을 받는 생성자 (이름, 나이, 성별)
		+ toString() : String	Member 클래스의 모든 필드 값들을 합쳐 String형으로 반환
	Book	+ Book()	기본 생성자
		+ Book(title:String, author:String, publisher:String)	3개의 초기값을 받는 생성자
		+ toString() : String	Book 클래스의 모든 필드 값들을 합쳐 String 형으로 반환
	AniBook	+ AniBook()	기본 생성자
		+ AniBook(title:String, author:String, publisher:String, accessAge:int)	4개의 초기값을 받는 생성자 super 생성자를 이용하여 부모 초기값 3개는 넘김
		+ toString() : String	super의 toString에 AniBook 클래스만의 필드 값을 합쳐 String형으로 반환
	CookBook	+ CookBook()	기본 생성자
		+ CookBook(title:String, author:String, publisher:String, coupon:boolean)	4개의 초기값을 받는 생성자 super 생성자를 이용하여 부모 초기값 3개는 넘김
		+ toString() : String	super의 toString에 CookBook 클래스의 필드 값을 합쳐 String형으로 반환

* class 명과 method 명은 변경하지 않는다.

Package명	Class명	Method	설명
skillup.day03. library.view	LibraryMenu	+ LibraryMenu()	기본 생성자
		+ mainMenu() : void	회원 이름, 나이, 성별을 입력 받아 Member 객체를 생성하고 lm의 insertMember() 메소드로 생성된 회원 객체의 주소 값 전달, 무한 반복 메뉴 출력하여 각 메뉴 버튼 클릭 시 해당하는 메소드 호출 (클래스 구조 참고)
		+ selectAll() : void	lm의 selectAll() 메소드 호출 → 결과 값을 Book[] 자료형으로 받아준 뒤 for문을 이용하여 도서 전체 목록 출력
		+ searchBook() : void	검색할 키워드 값을 입력 받고 그 입력 받은 키워드를 lm의 searchBook() 메소드로 전달 → 결과 값을 Book[] 자료형으로 받아 for문을 이용하여 출력
		+ rentBook() : void	대여할 도서 인덱스를 입력 받아 lm의 rentBook() 메소드로 전달 → 결과 값을 result로 받아 result가 0일 경우, 1일 경우, 2일 경우 각각에 해당하는 출력문 출력 (클래스 구조 참고)
skillup.day03. library.controller	LibraryManager	+ LibraryManager()	기본 생성자
		+ insertMember (m:Member) : void	전달 받은 m 주소 값을 this.mem(회원)에 대입
		+ myPage() : Member	회원 레퍼런스(mem) 주소 값 리턴
		+ selectAll() : Book[]	도서배열 레퍼런스(bList) 주소 값 리턴
		+ searchBook(keyword:String) : Book[]	검색 결과 여러 개의 도서가 검색될 수도 있으니 검색된 도서를 담아줄 Book객체 배열 새로 생성, for문을 이용하여 bList 안의 도서들과 전달받은 keyword를 비교하여 제목에 해당 keyword를 포함하고 있는 경우 새로운 배열에 차곡차곡 담기 → 그 배열 주소 값 리턴

		+ rentBook(index:int) : int	대여 결과 값을 받아줄 int 변수 result 선언 (0으로 초기화) 전달받은 인덱스의 도서가 AniBook일 경우 회원 나이와 해당 도서의 제한 나이를 비교하여 회원 나이가 더 적을 경우(대여 불가) result 값 1로 초기화 CookBook일 경우 해당 도서에 요리쿠폰이 있을 경우(대여 성공 및 쿠폰 발급) result 2로 초기화 후 회원의 couponCount 수 1 증가

3. class 구조

```
public class LibraryMenu{

        public void mainMenu(){

                // "이름을 입력하시오 : "          >> 입력 받음

                // "나이를 입력하시오 : "          >> 입력 받음

                // "성별(F/M)을 입력하시오 : "     >> 입력 받음

                // 입력 받은 이름, 나이, 성별로 Member 매개변수 생성자를 이용하여 생성 후

                // LibraryManager의 insertMember() 메소드에 생성한 객체 주소 값 전달


                // 아래의 메뉴 무한 반복 출력

                    ==== 메뉴 ====

                    1. 마이페이지              // LibraryManager의 myPage() 호출하여 출력

                    2. 도서 전체 조회          // LibraryMenu의 selectAll() 호출

                    3. 도서 검색               // LibraryMenu의 searchBook() 호출

                    4. 도서 대여하기           // LibraryMenu의 rentBook() 호출

                    9. 프로그램 종료하기       // return 처리

        }
```

```
public void selectAll (){
        // LibraryManager의 selectAll() 메소드 호출하여 그 결과값 Book[] 자료형에 담기
        ➔ Book[] bList = lm.selectAll();

        // for문을 이용하여 bList의 모든 도서 목록 출력
        // 단, 도서의 인덱스(도서번호)도 같이 출력
            → 나중에 대여할 때 인덱스(도서번호)를 입력 받기 위해
        ex)   0번 도서 : 백종원의 집밥 / 백종원 / tvN / true
}

public void searchBook(){
        // "검색할 제목의 키워드 입력 : "          >> 입력 받음
        // 입력 받은 키워드를 LibraryManager의 searchBook() 메소드에 전달
        // 그 결과 값을 Book[] 자료형에 담기
        ➔ Book[] searchList = lm.searchBook(keyword);

        // for문을 이용하여 검색 결과의 도서 목록 출력
}

public void rentBook(){
        // "대여할 도서 번호 선택 : "  >> 전체조회를 통해 보여지는 도서번호 입력 받음
        // LibraryManager의 rentBook() 메소드에 입력 받은 도서번호 전달
        // 그 결과 값을 result로 받고 그 result가
        // 0일 경우 → "성공적으로 대여되었습니다."
        // 1일 경우 → "나이 제한으로 대여 불가능입니다."
        // 2일 경우 → "성공적으로 대여되었습니다. 요리학원 쿠폰이 발급되었습니다. 마이페
                        이지에서 확인하세요" 출력
}
```

```
public class LibraryManager{

	private Member mem = null;
	private Book[] bList = new Book[5];

	{
		bList[0] = new CookBook("백종원의 집밥", "백종원", "tvN", true);
		bList[1] = new AniBook("한번 더 해요", "미티", "원모어", 19);
		bList[2] = new AniBook("루피의 원피스", "루피", "japan", 12);
		bList[3] = new CookBook("이혜정의 얼마나 맛있게요", "이혜정", "문학", false);
		bList[4] = new CookBook("최현석 날 따라해봐", "최현석", "소금책", true);
	}

	public void insertMember (Member m){

		// 전달 받은 m을 this의 mem에 대입

	}


	publ          lember myPage(){

		// 회원 레퍼런스(mem) 주소 값 리턴

	}


	public Book[] selectAll(){

		// 도서 객체배열 레퍼런스(bList) 주소 값 리턴

	}


	public Book[] searchBook(String keyword){

		// 검색 결과를 담아줄 새로운 Book 객체 배열 생성(임의로 크기 5로 생성)

		➔ Book[] searchList = new Book[5];



		// for문을 이용하여 bList 도서들 제목 중 전달 받은 키워드를 포함한 도서가 있으면

		// → HINT : 조건식에 String클래스의 contains()사용

		// searchList 객체배열에 담기 (int count; 변수 이용)

		// searchList 주소 값 리턴

	}
```

```
	public int rentBook(int index){
		// int result = 0;

		// 지금부터 instanceof 사용!!
		// 전달 받은 index의 bList객체가 AniBook을 참조하고 있으면
		// 해당 도서의 getAccessAge()와 회원의 나이를 비교하여 회원 나이가 적을 경우
		// result를 1로 초기화 → 나이제한으로 대여 불가

		// 전달 받은 index의 bList객체가 CookBook을 참조하고 있으면
		// 해당 도서의 getCoupon() 값이 true일 경우 즉, 쿠폰이 있는 경우
		// 회원의 couponCount 수 1 증가 처리하고
		// result를 2로 초기화 → 성공적으로 대여 완료, 요리학원 쿠폰 발급 성공

		// result 값 리턴
	}
}
```

```
public class Run {
	public static void main(String[] args){
		new LibraryMenu().mainMenu();
	}
}
```