Chapter 1. Iterator(반복자)
하나씩 열거하면서 처리

교재: 자바언어로배우는디자인패턴입문(개정판)/YukiHiroshi저/김윤정역/영진닷컴

덕성여자대학교 컴퓨터학과
최 승 훈

01. Iterator 패턴

- for 문의 루프 변수 i의 역할

```
for ( int i = 0  ;  i < array.length ;  i++  ) {
     System.out.println(array[i]);
}
```

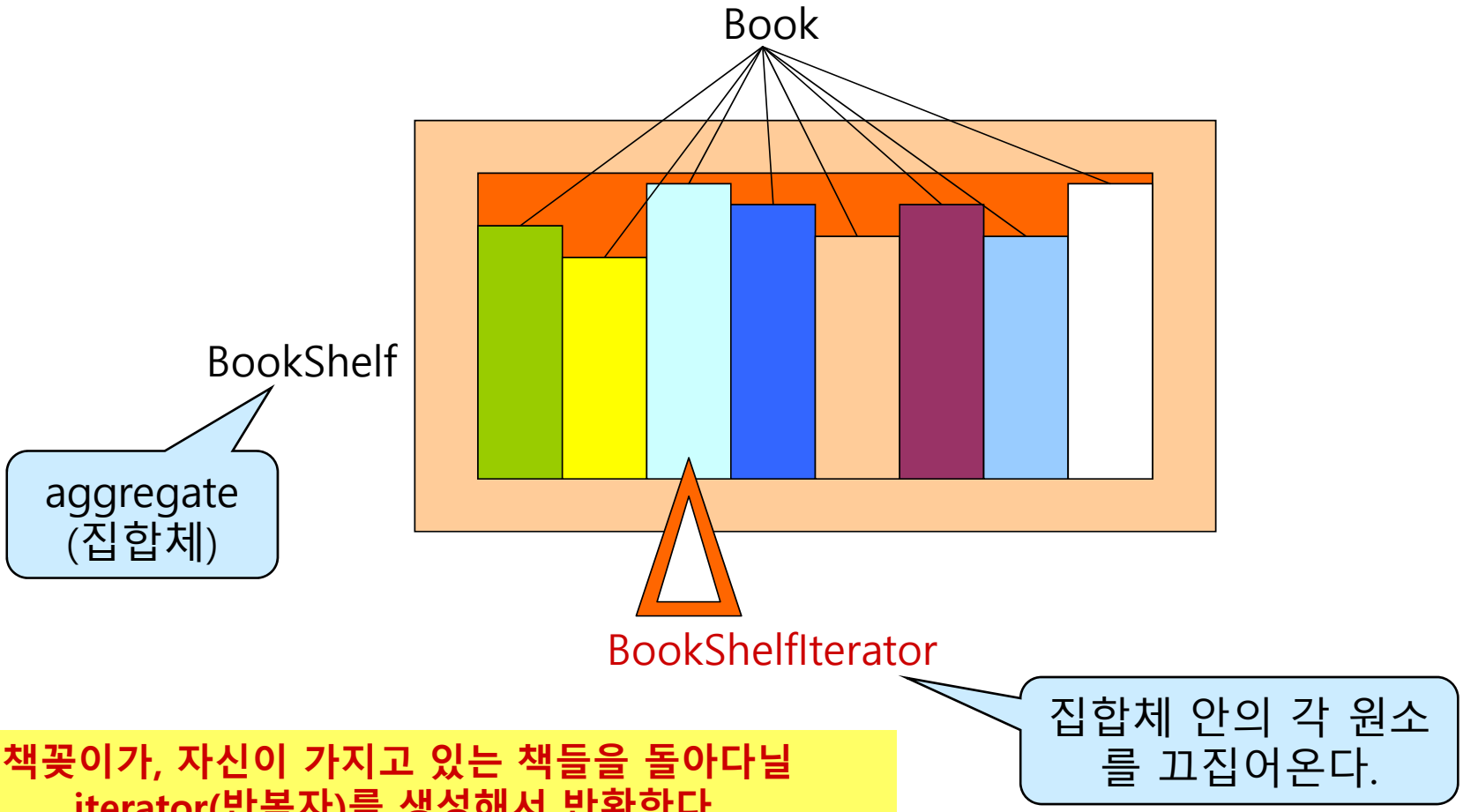
  - 변수 i: 여러 원소가 모여있는 배열(array)의 각 원소를 차례대로 선택하는데 사용된다.

01. Iterator 패턴

- Iterator 패턴
  - 변수 i의 기능을 추상화해서 일반화 시켰음
  - 무엇인가 많이 모여있는 것 중에서 하나씩 끄집어내어 열거하면서 전체를 처리하는 일을 할 때 이 패턴을 적용한다.

02. 예제 프로그램

- 책꽂이(BookShelf)에 책(Book)을 넣은 후, 순서대로 하나씩 다시 끄집어 내서 책 이름을 표시하는 프로그램



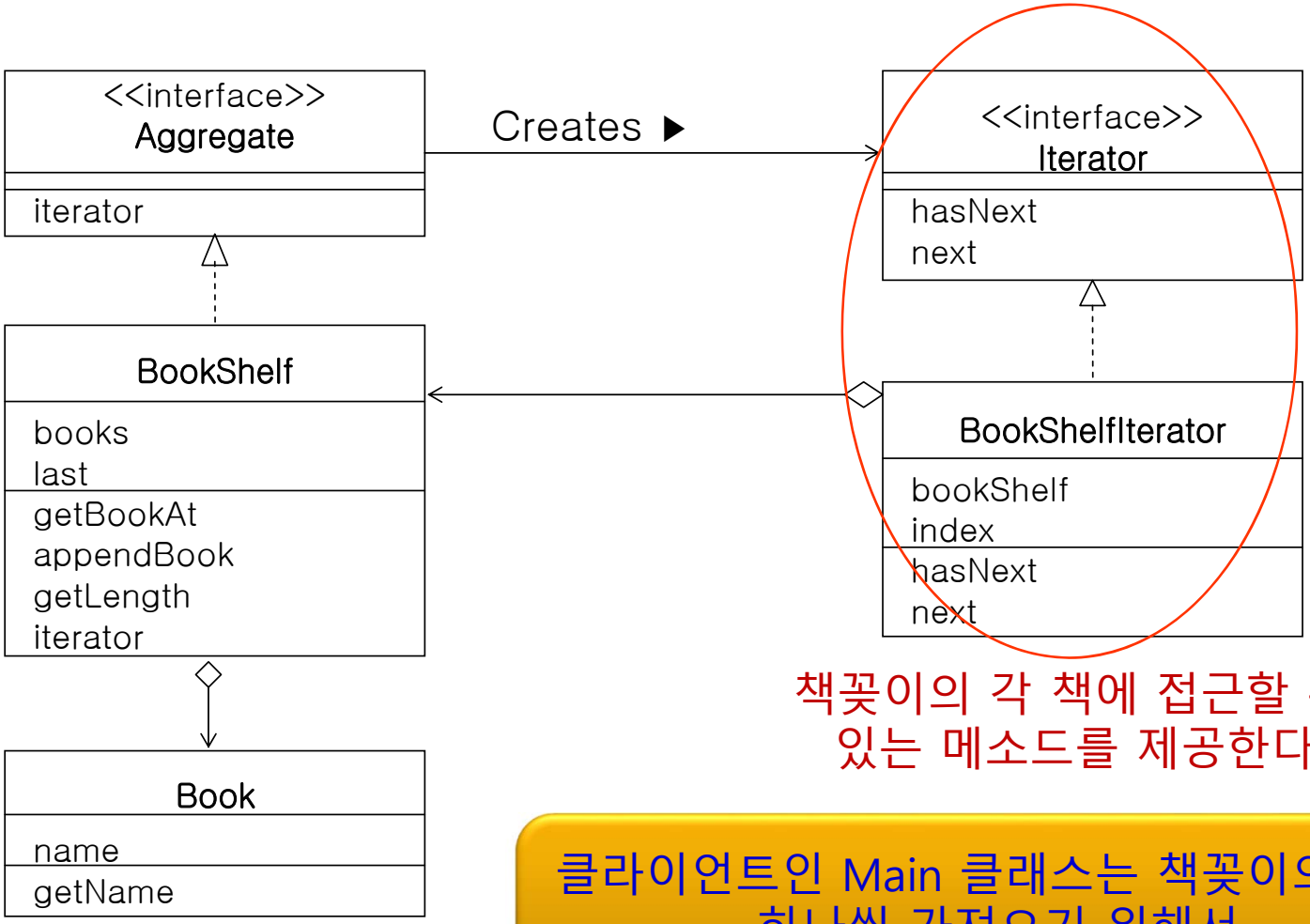
책꽂이가, 자신이 가지고 있는 책들을 돌아다닐 iterator(반복자)를 생성해서 반환한다.

02. 예제 프로그램

❑ 각 클래스와 인터페이스 설명

이름	해설
Aggregate	집합체를 나타내는 인터페이스
Iterator	하나씩 열거하면서 스캔을 실행하는 인터페이스
Book	책을 나타내는 클래스
BookShelf	서가를 나타내는 클래스
BookShelfIterator	서가를 검색하는 클래스
Main	동작 테스트용 클래스

02. 예제 프로그램



책꽂이의 각 책에 접근할 수 있는 메소드를 제공한다.

클라이언트인 Main 클래스는 책꽂이의 책을 하나씩 가져오기 위해서 BookShelfIterator 를 이용한다.

02. 예제 프로그램

- Aggregate 인터페이스 (sample/Aggregate.java)
  - iterator( ): 집합체에 대응하는 Iterator 한 개를 생성하는데 사용될 메소드
    - 어떤 집합체의 원소를 하나씩 열거하거나 조사하고자 할 때 이 메소드를 사용해서 Iterator 인터페이스를 구현한 클래스의 인스턴스를 한 개 얻어온다.

02. 예제 프로그램

- Iterator 인터페이스(sample/Iterator.java)
  - 집합체의 원소를 하나하나 끄집어내는 루프 변수와 같은 역할을 한다.
  - hasNext( ): 다음 원소가 존재하는지 조사할 때 사용하는 메소드
    - 반환형은 boolean (마지막 원소에 도달하면 false를 반환함)
  - next( ): 다음 원소를 얻어올 때 사용하는 메소드
    - 반환형은 Object

02. 예제 프로그램

- Book 클래스(sample/Book.java)
  - 책을 나타내는 클래스
  - name 속성: 책이름을 저장하는 변수
  - getName( ): 책의 이름을 얻어올 때 호출하는 메소드

02. 예제 프로그램

- BookShelf 클래스(sample/BookShelf.java)
  - 책꽂이를 나타내는 클래스 = 집합체(aggregate)
  - Aggregate 인터페이스를 구현하였다.
    - BookShelf 클래스는, iterator( ) 메소드의 구현 부분을 제공한다.
    - BookShelf 클래스는, 이 외에 다른 추가의 메소드도 제공한다.
  - books 필드: Book의 배열
    - 이 배열의 크기는, 생성자(BookShelf( )) 호출 시 지정된다.
    - private으로 선언된 이유는, 클래스 외부에서 이 필드를 변경하지 못하게 하기 위해서이다.
  - appendBook( ): 책 한 권을 서가에 추가하는 메소드
  - getLength( ): 현재 책꽂이에 있는 책의 개수를 반환하는 메소드
  - iterator( ): 책꽂이의 책 하나하나를 끄집어내는 일을 하는 BookShelfIterator를 생성하는 메소드

02. 예제 프로그램

- BookShelfIterator 클래스(sample/BookShelfIterator.java)
  - 책꽂이(BookShelf)에 있는 책들을 하나씩 끄집어내는 일을 하는 클래스
  - Iterator 인터페이스를 구현하였다
    - hasNext( )와 next( ) 메소드를 구현함
  - BookShelf 필드: BookShelfIterator가 검색할 책꽂이를 가리키는 변수 (생성자에서 넘겨받은 BookShelf의 인스턴스를 가지고 있음)
  - index 필드: 책꽂이에서의 현재 책을 가리키는 변수
  - hasNext( ): 다음 책이 있으면 true, 없으면 false를 반환함
  - next( ): 현재 가리키고 있는 책을 반환하고, 다음 책을 가리키는 메소드

02. 예제 프로그램

- Main 클래스(sample/Main.java)
  - main( )
    - 1) 책 4권을 책꽂이에 넣는다
    - 2) 책꽂이의 책을 하나씩 끄집어낼 Iterator를 얻는다.
      - Iterator it = bookShelf.iterator( );
    - 3) Iterator의 hasNext( )와 next( ) 메소드를 이용하여 책을 하나씩 끄집어내서 책의 이름을 출력한다.

03. Iterator 패턴에 등장하는 역할

- Iterator(반복자)의 역할
  - 원소를 하나씩 끄집어낼 때 사용할 공통된 메소드를 선언한 인터페이스
  - hasNext( )와 next( )메소드 선언

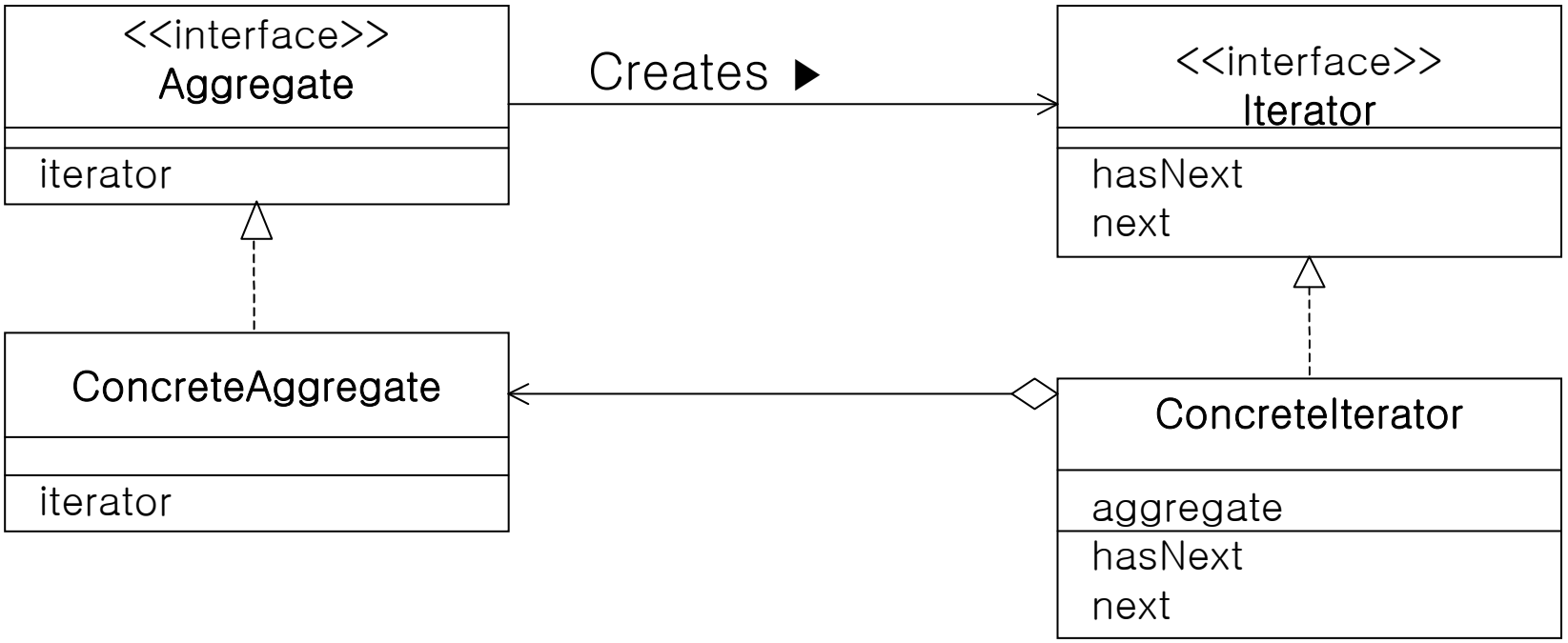
- ConcreteIterator(구체적인 반복자)의 역할
  - Iterator 인터페이스를 실제로 구현하는 클래스
  - BookShelfIterator이 이 역할을 담당하였다.

03. Iterator 패턴에 등장하는 역할

- Aggregate(집합체)의 역할
  - Iterator를 만들어내는 인터페이스를 제공함
  - iterator( ): 내가 가지고 있는 각 원소들을 차례로 검색해 줄 사람을 만들어내는 메소드

- ConcreteAggregate(구체적인 집합체)의 역할
  - Aggregate 인터페이스를 구현하는 클래스
  - ConcreteIterator(구체적인 반복자) 객체를 생성한다.
  - BookShelf가 이 일을 담당하였다.

03. Iterator 패턴에 등장하는 역할



04. 독자의 사고를 넓혀주는 힌트

- 왜 Iterator 패턴이 유용한가?
  - 집합체가 원소를 어떻게 유지하고 있든지 상관없이, 집합체의 원소를 차례로 끄집어내고자 하면, Iterator의 hasNext( )와 next( ) 메소드를 사용하면 된다.
  - 예: BookShelf가 Book을 배열에 저장하지 않고, vector에 저장하도록 수정하더라도, Main 클래스(클라이언트)의 main( )메소드의 다음 부분을 변경하지 않아도 된다.

```
while (it.hasNext()) {
        Book book = (Book)it.next();
        System.out.println("" + book.getName());
    }
```

  - 집합체의 각 원소를 끄집어내는 방법이, 집합체 구현과 무관하다.
  - 하나를 수정했을 때, 수정해야 할 부분을 적게 하는 것이 중요하다.

04. 독자의 사고를 넓혀주는 힌트

- 코드의 어떤 부분을 수정했을 때, 그것으로 인해 수정해야 할 코드 부분을 적게 하는 것이 중요하다.

04. 독자의 사고를 넓혀주는 힌트

- 가능한 한 추상 클래스나 인터페이스를 자주 사용해라
  - 구체적인 클래스만으로 프로그래밍하는 것은 좋지 않다.

04. 독자의 사고를 넓혀주는 힌트

- Aggregate과 Iterator의 대응 관계
  - Aggregate 클래스와 Iterator 클래스는 밀접하게 관련이 있다.
  - 예: BookShelf의 getBookFrom( ) 메소드의 이름을 getBookAt( )으로 바꾸면, BookShelfIterator의 next( ) 내부도 수정해야 한다.

04. 독자의 사고를 넓혀주는 힌트

- 여러 종류의 Iterator를 만들 수 있다.
  - 예: 역방향으로 책을 끄집어내는 BookShelfIteratorBackward를 만든 후, BookShelf의 iterator( ) 메소드를 다음과 같이 변경하면, 다른 종류의 Iterator를 생성할 수 있다.

```
return new BookShelfIteratorBackward(this);
```

05. Iterator 패턴과 관련된 패턴

- Visitor 패턴
- Composite 패턴
- Factory Method 패턴

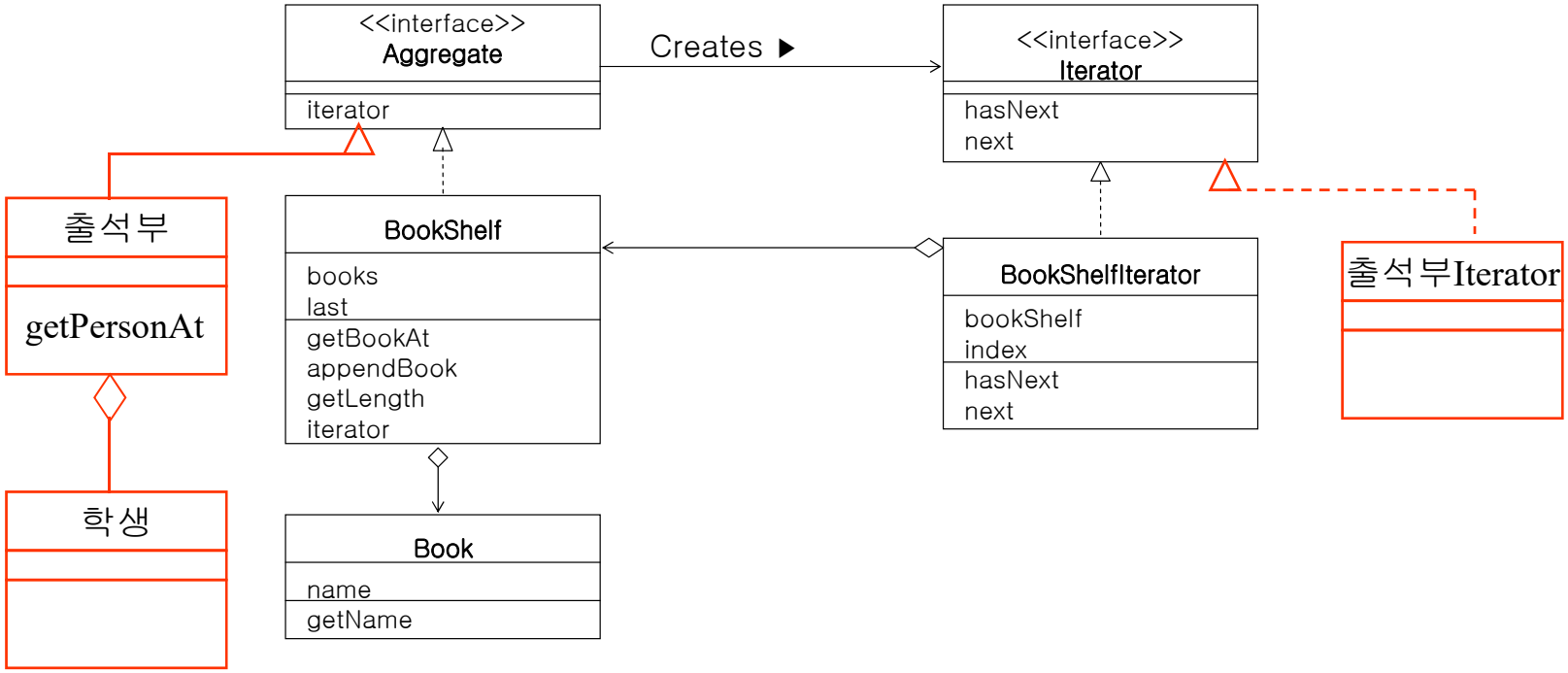
06. 이 장에서 학습한 내용

- 집합체의 원소를 일관된 방법으로 하나하나 세는 Iterator 패턴

연습문제

- BookShelf 클래스에서는 Book을 배열에 저장하기 때문에 책꽂이의 크기를 초과해서 책을 넣을 수 없다.
  - java.util.vector 클래스를 사용하면, 크기 제한 없이 책을 계속해서 넣을 수 있다. 기존의 코드를 이렇게 바꾸시오.

Iterator 보충



클라이언트가 출석부에 있는 학생을 차례대로 끄집어내 오려면,출석부의 iterator() 메소드를 호출하여 출석부Iterator를 얻어온다. 그리고 나서, 출석부Iterator의 hasNext( )와 next( )를 이용하여 각 학생을 가져온다.