가톨릭대학교 THE CATHOLIC UNIVERSITY OF KOREA Computer SCIENCE

컴퓨터정보공학부- 201921725 안성현

프로그래밍언어론 실습과제 2

<2022/12/07>

예외처리 프로그래밍

① 프로그램 개요

학생 점수를 관리하는 프로그램이다. 관리자가 학생의 이름과 점수를 저장하면, 학생의 이름이나 번호를 통해서 해당 점수를 검색할 수 있다.

② 유즈케이스

1. 로그인 기능
- 로그인 성공
- 로그인 실패
- 로그인 실패 (비밀번호가 0) -> ArithmeticException 발생

2. 점수 저장
- 점수 저장 성공
- 점수 저장 실패 (학생 수가 음수) -> NegativeArraySizeException 발생
- 점수 저장 실패 (점수가 음수) -> NegativeScoreException (사용자 정의 예외) 발생

3. 이름으로 학생 점수 확인
- 이름으로 학생 점수 확인 성공
- 이름으로 학생 점수 확인 실패 (해당 이름이 없음)
- 이름으로 학생 점수 확인 실패 (저장된 학생이 없음) -> NullPointerException 발생

4. 번호로 학생 점수 확인
- 번호로 학생 점수 확인 성공
- 번호로 학생 점수 확인 실패 (해당 번호가 없음) ->ArrayIndexOutofBoundsException 발생
- 번호로 학생 점수 확인 (저장된 학생이 없음) -> NullPointerException 발생

< 예외사항 목록 >

1. ArithmeticException
   0으로 나누는 경우에 발생하는 예외

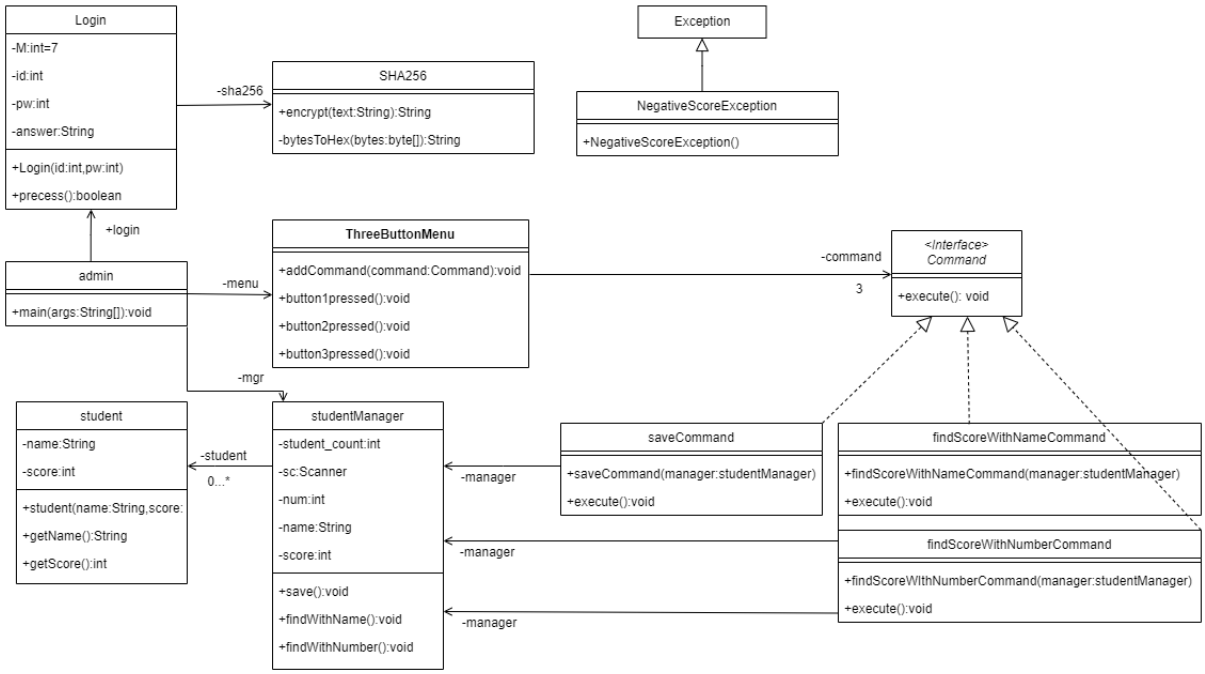
2. NegativeArraySizeException
   배열의 크기가 음수로 된 경우에 발생하는 예외

3. NegativeScoreException (사용자 정의 예외)
   학생의 점수가 음수로 입력됐을 때 발생하는 예외

4. NullPointerException
   생성되지 않은 객체를 이용해서 객체의 멤버를 접근하는 경우에 발생하는 예외

5. ArrayIndexOutofBoundsException
   배열의 크기보다 크거나 인덱스가 음수인 원소를 접근하려고 할 때 발생하는 예외

③ 클래스 다이어그램



1. admin

main메소드가 위치한 클래스로, studentManager, ThreeButtonMenu, Login 객체를 만든 뒤 프로그램을 시작한다. 사용자가 1번을 입력하면 menu의 button1pressed메소드를 호출함으로써 기능을 처리한다.

2. Login

RSA 암호화의 아이디어를 착안하였다. RSA에서는 공개키(n,e)와 메시지(M)가 주어질 때, $C = M^e \bmod n$ 공식을 통해 암호문(C)을 제작한다. 필자는 이 암호화 방식으로 로그인 로직을 설계하였다. 사용자가 정수인 id와 pw를 입력하고, M은 기본적으로 7로 주어진다. $C = M^{id} \bmod pw$를 통해 C를 계산한다. 이후 Hash(C)와 프로그램에 저장된 answer를 비교해서 값이 같은지 확인한다. 값이 같으면 True를 반환하고, 다르면 False를 반환한다. 또한 pw가 0으로 주어지면, 0으로 나누었기 때문에 ArithmeticException이 발생한다. 참고로 해시 기법은 SHA256을 이용하였으며, 해커는 C의 정답이 무엇인지 모르므로 pw를 추정할 수 없다.

3. SHA256

해시함수를 사용하기 위한 클래스이다.

4. ThreeButtonMenu

커맨드 패턴에서 Invoker의 역할을 하는 클래스이다. 커맨드들을 저장한 뒤, 각 버튼에 맞게 세팅하였다. 버튼1은 '점수 저장' 기능을 한다. 버튼2는 '이름으로 점수 확인' 기능을 한다. 버튼3은 '번호로 점수 확인' 기능을 한다.

5. Command

커맨드(버튼을 누를 때 실행되는 기능)의 인터페이스이다.

6. saveCommand
'점수 저장'에 대한 커맨드이다.

7. findScoreWithNameCommand
'이름으로 학생 점수 확인'에 대한 커맨드이다.

8. findScoreWIthNumberCommand
'번호로 학생 점수 확인'에 대한 커맨드이다.

9. studentManager
커맨드 패턴에서 Receiver의 역할을 하는 클래스이다. 커맨드의 실제 기능이 이 클래스에 담겨 있다. 점수 저장을 할 때는 사용자가 입력한 학생의 수를 크기로 갖는 students 배열을 제작한다. 이 때 학생의 수가 음수이면 NegativeArraySizeException이 발생한다. 또한 점수를 입력하는 과정에서 음수가 들어오면 NegativeScoreException이 발생한다. 이름이나 번호로 학생 점수를 확인할 때는 반드시 학생 점수가 미리 저장되어 있어야만 한다. 그렇지 않으면 students 배열이 NULL이므로 NullPointerException이 발생한다. 또한 번호로 학생 점수를 확인할 때, 해당 번호가 없을 때는 students배열에 값이 저장되기 전이므로 ArrayIndexOutofBoundsException이 발생한다.

10. student
학생 클래스이다. 학생의 이름과 점수를 필드로 갖는다.

11. Exception
예외처리의 상위 클래스이다.

12. NegativeScoreException
Exception을 상속받는 클래스로, 관리자가 학생 점수를 입력할 때 음수를 입력하면 발생하는 예외 처리 클래스이다.

④ 정상흐름

1. 로그인
ID는 2019, PW는 21725로 설정하였다.

```
== 로그인 ==

* ID:2019
* Password:21725

== 학생 점수 관리 프로그램 ==

1. 점수 저장
2. 이름으로 점수 확인
3. 번호로 점수 확인

* 선택해주세요:
```

2. 점수 저장

저장할 학생 수를 입력하고 학생 이름과 점수를 차례차례 입력하며 저장한다.

```
== 학생 점수 관리 프로그램 ==

1. 점수 저장
2. 이름으로 점수 확인
3. 번호로 점수 확인

* 선택해주세요:1

저장할 학생 수를 입력하시오:2
1번째 학생 이름을 입력하시오:김하나
1번째 학생 점수를 입력하시오:80
2번째 학생 이름을 입력하시오:김두리
2번째 학생 점수를 입력하시오:90
저장 완료!
```

3. 이름으로 학생 점수 확인

저장한 학생의 이름을 이용해서 학생 점수를 확인할 수 있다.

```
== 학생 점수 관리 프로그램 ==

1. 점수 저장
2. 이름으로 점수 확인
3. 번호로 점수 확인

* 선택해주세요:2

검색할 학생 이름을 입력하시오:김하나
김하나 학생은 80점입니다!
```

4. 번호로 학생 점수 확인

저장한 학생의 번호를 이용해서 학생 점수를 확인할 수 있다.

```
== 학생 점수 관리 프로그램 ==

1. 점수 저장
2. 이름으로 점수 확인
3. 번호로 점수 확인

* 선택해주세요:3

검색할 학생 번호를 입력하시오:1
김하나 학생은 80점입니다!
```

⑤ 예외흐름

1-1. 로그인 실패

잘못된 아이디나 비밀번호를 입력해서 로그인이 실패한 화면이다.

```
== 로그인 ==

* ID:2019
* Password:2019

로그인에 실패하였습니다!
프로그램을 종료합니다!
```

1-2. 로그인 실패 (비밀번호가 0) ArithmeticException

비밀번호를 0으로 입력해서 예외가 발생한 화면이다. 예외가 발생 시 프로그램이 종료된다.

```
== 로그인 ==

* ID:2019
* Password:0

[ERROR] 비밀번호는 0이 될 수 없습니다!

로그인에 실패하였습니다!
프로그램을 종료합니다!
```

2-1. 점수 저장 실패 (학생 수가 음수) NegativeArraySizeException

학생 수를 음수로 입력해서 예외가 발생한 화면이다. 예외가 발생 시 초기화면으로 돌아간다.

```
저장할 학생 수를 입력하시오:-1

[ERROR] 학생 수는 음수일 수 없습니다! 초기화면으로 돌아갑니다!

== 학생 점수 관리 프로그램 ==

1. 점수 저장
2. 이름으로 점수 확인
3. 번호로 점수 확인

* 선택해주세요:
```

2-2. 점수 저장 실패 (점수가 음수) NegativeScoreException

점수를 음수로 입력해서 예외가 발생한 화면이다. 예외가 발생 시 초기화면으로 돌아간다.

```
1번째 학생 점수를 입력하시오:-10

[ERROR] 점수는 음수일 수 없습니다! 초기화면으로 돌아갑니다!

== 학생 점수 관리 프로그램 ==

1. 점수 저장
2. 이름으로 점수 확인
3. 번호로 점수 확인

* 선택해주세요:
```

3-1. 이름으로 학생 점수 확인 실패 (해당 이름이 없음)

입력한 이름이 저장되어 있지 않아 점수 확인에 실패한 화면이다.

```
검색할 학생 이름을 입력하시오:나얼
존재하지 않는 학생입니다! 초기화면으로 돌아갑니다!

== 학생 점수 관리 프로그램 ==

1. 점수 저장
2. 이름으로 점수 확인
3. 번호로 점수 확인

* 선택해주세요:
```

3-2. 이름으로 학생 점수 확인 실패 (저장된 학생이 없음) NullPointerException

저장된 학생이 한 명도 없어 예외가 발생한 화면이다. 예외가 발생 시 초기화면으로 돌아간다.

```
검색할 학생 이름을 입력하시오:김하나

[ERROR] 저장된 학생이 없습니다! 초기화면으로 돌아갑니다!

== 학생 점수 관리 프로그램 ==

1. 점수 저장
2. 이름으로 점수 확인
3. 번호로 점수 확인

* 선택해주세요:
```

4-1. 번호로 학생 점수 확인 실패 (해당 번호가 없음)
ArrayIndexOutofBoundsException

입력한 번호에 해당하는 학생이 존재하지 않아 예외가 발생한 화면이다. 예외가 발생 시 초기화면으로 돌아간다.

```
검색할 학생 번호를 입력하시오:3

[ERROR] 존재하지 않는 번호입니다! 초기화면으로 돌아갑니다!

== 학생 점수 관리 프로그램 ==

1. 점수 저장
2. 이름으로 점수 확인
3. 번호로 점수 확인

* 선택해주세요:
```

4-2. 번호로 학생 점수 확인 실패 (저장된 학생이 없음) NullPointerException

저장된 학생이 한 명도 없어 예외가 발생한 화면이다. 예외가 발생 시 초기화면으로 돌아간다.

```
검색할 학생 번호를 입력하시오:1

[ERROR] 저장된 학생이 없습니다! 초기화면으로 돌아갑니다!

== 학생 점수 관리 프로그램 ==

1. 점수 저장
2. 이름으로 점수 확인
3. 번호로 점수 확인

* 선택해주세요:
```