64 Java의 정석定石 3판 - 연습문제 풀이

# Chapter

예외처리

Exception Handling

## [ 연습문제 ]

[8-1] 예외처리의 정의와 목적에 대해서 설명하시오.

[8-2] 다음은 실행도중 예외가 발생하여 화면에 출력된 내용이다. 이에 대한 설명 중 옳 지 않은 것은?

java.lang.ArithmeticException : / by zero

at ExceptionEx18.method2(ExceptionEx18.java:12) at ExceptionEx18.method1(ExceptionEx18.java:8) at ExceptionEx18.main(ExceptionEx18.java:4)

1. 위의 내용으로 예외가 발생했을 당시 호출스택에 존재했던 메서드를 알 수 있다.
2. 예외가 발생한 위치는 method2 메서드이며, ExceptionEx18.java파일의 12번째 줄이다.
3. 발생한 예외는 ArithmeticException이며, 0으로 나누어서 예외가 발생했다.
4. method2메서드가 번째 줄이다.

method1메서드를 호출하였고 그 위치는

ExceptionEx18.java파일의 8

[8-3] 다음 중 오버라이딩이 잘못된 것은? (모두 고르시오)

void add(int a, int b)

throws InvalidNumberException, NotANumberException {}

class NumberException extends Exception {}

class InvalidNumberException extends NumberException {} class NotANumberException extends NumberException {}

1. void add(int a, int b) throws InvalidNumberException, NotANumberException {}
2. void add(int a, int b) throws InvalidNumberException {}
3. void add(int a, int b) throws NotANumberException {}
4. void add(int a, int b) throws Exception {}
5. void add(int a, int b) throws NumberException {}

[8-4] 다음과 같은 메서드가 있을 때, 예외를 잘못 처리한 것은? (모두 고르시오)

void method() throws InvalidNumberException, NotANumberException {} class NumberException extends RuntimeException {}

class InvalidNumberException extends NumberException {} class NotANumberException extends NumberException {}

1. try {method();} catch(Exception e) {}
2. try {method();} catch(NumberException e) {} catch(Exception e) {}
3. try {method();} catch(Exception e) {} catch(NumberException e) {}
4. try {method();} catch(InvalidNumberException e) {

} catch(NotANumberException e) {}

1. try {method();} catch(NumberException e) {}
2. try {method();} catch(RuntimeException e) {}

[8-5] 아래의 코드가 수행되었을 때의 실행결과를 적으시오.

class Exercise8\_5 {

static void method(boolean b) { try {

System.out.println(1);

if(b) throw new ArithmeticException(); System.out.println(2);

} catch(RuntimeException r) { System.out.println(3); return;

} catch(Exception e) { System.out.println(4); return;

} finally {

System.out.println(5);

}

System.out.println(6);

}

public static void main(String[] args) { method(true);

method(false);

} // main

}

[연습문제]**/ch8/Exercise8\_5.java**

[8-6] 아래의 코드가 수행되었을 때의 실행결과를 적으시오.

class Exercise8\_6 {

public static void main(String[] args) { try {

method1();

} catch(Exception e) { System.out.println(5);

}

}

static void method1() { try {

method2(); System.out.println(1);

} catch(ArithmeticException e) { System.out.println(2);

} finally {

System.out.println(3);

}

System.out.println(4);

} // method1()

static void method2() {

throw new NullPointerException();

}

}

[연습문제]**/ch8/Exercise8\_6.java**

[8-7] 아래의 코드가 수행되었을 때의 실행결과를 적으시오.

class Exercise8\_7 {

static void method(boolean b) { try {

System.out.println(1); if(b) System.exit(0); System.out.println(2);

} catch(RuntimeException r) { System.out.println(3); return;

} catch(Exception e) { System.out.println(4); return;

} finally {

System.out.println(5);

}

System.out.println(6);

}

public static void main(String[] args) { method(true);

method(false);

} // main

}

[연습문제]**/ch8/Exercise8\_7.java**

[8-8] 다음은 1~100사이의 숫자를 맞추는 게임을 실행하던 도중에 숫자가 아닌 영문자를 넣어서 발생한 예외이다. 예외처리를 해서 숫자가 아닌 값을 입력했을 때는 다시 입력을 받도록 보완하라.

1과 100사이의 값을 입력하세요 :50

더 작은 수를 입력하세요.

1과 100사이의 값을 입력하세요 :asdf

Exception in thread "main" java.util.InputMismatchException at java.util.Scanner.throwFor(Scanner.java:819)

at java.util.Scanner.next(Scanner.java:1431) at java.util.Scanner.nextInt(Scanner.java:2040) at java.util.Scanner.nextInt(Scanner.java:2000) at Exercise8\_8.main(Exercise8\_8.java:16)

import java.util.\*;

class Exercise8\_8

{

public static void main(String[] args)

{

// 1~100사이의 임의의 값을 얻어서 answer에 저장한다. int answer = (int)(Math.random() \* 100) + 1; int input = 0; // 사용자입력을 저장할 공간

int count = 0; // 시도횟수를 세기 위한 변수

do {

count++;

System.out.print("1과 100사이의 값을 입력하세요 :"); input = new Scanner(System.in).nextInt();

if(answer > input) {

System.out.println("더 큰 수를 입력하세요.");

} else if(answer < input) { System.out.println("더 작은 수를 입력하세요.");

} else {

System.out.println("맞췄습니다."); System.out.println("시도횟수는 "+count+"번입니다."); break; // do-while문을 벗어난다

}

} while(true); // 무한반복문

} // end of main

} // end of class HighLow

[연습문제]**/ch8/Exercise8\_8.java**

1과 100사이의 값을 입력하세요 :50

더 작은 수를 입력하세요.

1과 100사이의 값을 입력하세요 :asdf

유효하지 않은 값입니다. 다시 값을 입력해주세요. 1과 100사이의 값을 입력하세요 :25

더 큰 수를 입력하세요.

[실행결과]

1과 100사이의 값을 입력하세요 :38

더 큰 수를 입력하세요.

1과 100사이의 값을 입력하세요 :44

맞췄습니다. 시도횟수는 5번입니다.

[8-9] 다음과 같은 조건의 예외클래스를 작성하고 테스트하시오.

[참고] 생성자는 실행결과를 보고 알맞게 작성해야한다.

* 클래스명 : UnsupportedFuctionException
* 조상클래스명 : RuntimeException
* 멤버변수 :

이 름 : ERR\_CODE

저장값 : 에러코드 타 입 : int 기본값 : 100

제어자 : final private

* 메서드 :
  1. 메서드명 : getErrorCode

기 능 : 에러코드(ERR\_CODE)를 반환한다. 반환타입 : int

매개변수 : 없음 제어자 : public

* 1. 메서드명 : getMessage

기 능 : 메세지의 내용을 반환한다.(Exception클래스의 getMessage()를 오버라이딩) 반환타입 : String

매개변수 : 없음 제어자 : public

class Exercise8\_9

{

public static void main(String[] args) throws Exception

{

throw new UnsupportedFuctionException("지원하지 않는 기능입니다.",100);

}

}

[연습문제]**/ch8/Exercise8\_9.java**

Exception in thread "main" UnsupportedFuctionException: [100]지원하지 않는 기능 입니다.

at Exercise8\_9.main(Exercise8\_9.java:5)

[실행결과]

[8-10] 아래의 코드가 수행되었을 때의 실행결과를 적으시오.

class Exercise8\_10 {

public static void main(String[] args) { try {

method1(); System.out.println(6);

} catch(Exception e) { System.out.println(7);

}

}

static void method1() throws Exception { try {

method2(); System.out.println(1);

} catch(NullPointerException e) { System.out.println(2);

throw e;

} catch(Exception e) { System.out.println(3);

} finally {

System.out.println(4);

}

System.out.println(5);

} // method1()

static void method2() {

throw new NullPointerException();

}

}

[연습문제]**/ch8/Exercise8\_10.java**