08_1 Exception Class

Object-Oriented Programming

Errors and Exceptions

- Errors: Serious errors that **cannot be handled** by program code
  - Compile-time Error
  - Run-time Error
  - Logic Error
- Exceptions
  - Minor errors that **can be handled** by program code
- Exception Handling
  - Preventing abnormal termination of the program **by coding**
  - Maintaining normal execution

Types of Exception

- General Exception (= compile-time checked exception)
  - Compiler checks for existence of appropriate exception handling code
  - If there is no exception handling code, a compile error occurs

- Runtime Exception (= compile-time unchecked exception)
  - Does not check for exception handling code at compile time
  - But, to handle the exception, exception-handling code must be written by programmer
  - If runtime exception occurs
    - If the program has exception-handling code, then the code is executed
    - If no exception-handling code, the program stops immediately

Exception Classes

- java.lang.Exception
  - java.lang.ClassNotFoundException
  - java.io.IOException
  - ….
  - java.lang.RuntimeException
    - java.lang.ArithmeticException
    - java.lang.ClassCastException
    - java.lang.NullPointerException
    - java.lang.IndexOutOfBoundsException
    - …

} General Exception
(Compile-time checked Exception)

} Runtime Exception
(Compile-time unchecked Exception)

General Exception: ClassNotFoundException (1/3)

```java
class TempClass0 { }

public class ClassClass0 {
    public static void main(String[] args) {
        TempClass0 t0 = new TempClass0();
        System.out.println("t0.getClass() returns: " + t0.getClass());
    }
}
```

```
OUTPUT: t0.getClass() returns: class TempClass0
```

General Exception: ClassNotFoundException (2/3)

```
class TempClass0 { }

public class ClassClass0 {
    public static void main(String[] args) {
        TempClass0 t0 = new TempClass0();
        System.out.println("t0.getClass() returns: " + t0.getClass());
        Class classInfo;
        classInfo = Class.forName("TempClass0");
        System.out.println("classInfo: " + classInfo);
        System.out.println("classInfo.getName() returns: " + classInfo.getName());
    }
}
```

```
java: unreported exception java.lang.ClassNotFoundException; must be caught or declared
to be thrown
```

General Exception: ClassNotFoundException (3/3)

```
class TempClass1 { }

public class ClassClass {
    public static void main(String[] args) {
        TempClass1 t1 = new TempClass1();
        Class classInfo;
        System.out.println("t1.getClass() returns: " + t1.getClass());
        try {
            classInfo = Class.forName("TempClass1");
            System.out.println("classInfo: " + classInfo);
            System.out.println("classInfo.getName() returns: " + classInfo.getName());
        } catch (ClassNotFoundException e) {
            System.out.println("Class not found \"TempClass1\"");
        }
    }
}
```

```
t1.getClass() returns: class TempClass1
classInfo: class TempClass1
classInfo.getName() returns: TempClass1
```

General Exception: java.io.IOException

```java
public class IOExceptionExample {
    public static void main(String[] args) {
        BufferedReader reader = null;
        try {
            reader = new BufferedReader(new FileReader("example.txt"));
            String line;
            while ((line = reader.readLine()) != null) {
                System.out.println(line);
            }
        }
        catch (IOException e) {
            System.out.println("파일을 read 중에 오류가 발생: " + e.getMessage());
            e.printStackTrace();
        }
```

```
파일을 read 중에 오류가 발생: example.txt (No such file or directory)
java.io.FileNotFoundException: example.txt (No such file or directory)
	at java.base/java.io.FileInputStream.open0(Native Method)
	at java.base/java.io.FileInputStream.open(FileInputStream.java:213)
	at java.base/java.io.FileInputStream.<init>(FileInputStream.java:152)
	at java.base/java.io.FileInputStream.<init>(FileInputStream.java:106)
	at java.base/java.io.FileReader.<init>(FileReader.java:60)
	at IOExceptionExample.main(IOExceptionExample.java:9)
```

Runtime Exception: NullPointerException

- Occurs when the dot (.) operator is used on a reference variable without the object it references
- ex)

```
public class NullPointerExceptionDemo {
    public static void main(String[] args) {
        String str1;
        String str2 = null;
        // System.out.println(str1.toString()); // compile error!!
        System.out.println(str2.toString()); // no compile error
                                             // but runtime exception
    }
}
```

```
Exception in thread "main" java.lang.NullPointerException: Cannot
invoke "String.toString()" because "str2" is null
    at NullPointerExceptionDemo.main(NullPointerExceptionDemo.java:6)
```

Runtime Exception: ArrayIndexOutOfBoundsException

- Occurs when the index range is exceeded in the array

```
public class ArrayIndexOutOfBoundsExceptionDemo {
    public static void main(String[] args) {
        String[] strArray = new String[3];
        strArray[0] = "Korea";
        strArray[3] = "Seoul";
    }
}
```

```
Exception in thread "main" java.lang.ArrayIndexOutOfBoundsException:
Index 3 out of bounds for length 3 at
ArrayIndexOutOfBoundsExceptionDemo.main(ArrayIndexOutOfBoundsExceptionDemo.java:5)
```

Runtime Exception: NumberFormatException

- Occurs when trying to convert a string into a numeric type data, such as int or double, and the string cannot be converted to a number (e.g. "23asdf", "?%^*", ...)

```
public class NumberFormatExceptionDemo {
    public static void main(String[] args) {
        String str1 = "132.68";
        String str2 = "abcde";
        int num1 = Integer.parseInt(str1);
        double num2 = Double.parseDouble(str2);
    }
}

Exception in thread "main" java.lang.NumberFormatException: For input string: "132.68"
    at java.base/java.lang.NumberFormatException.forInputString(NumberFormatException.java:67)
    at java.base/java.lang.Integer.parseInt(Integer.java:588)
    at java.base/java.lang.Integer.parseInt(Integer.java:685)
    at NumberFormatExceptionDemo.main(NumberFormatExceptionDemo.java:5)
```

Runtime Exception: InputMismatchException

```java
import java.util.InputMismatchException;
import java.util.Scanner;

public class InputMismatchExceptionExample {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Input an integer: ");
        int number = scanner.nextInt();  // assume input "abcd"
        System.out.println("Input Integer: " + number);
        scanner.close();
    }
}
```

```
Input an integer: abcd
Exception in thread "main" java.util.InputMismatchException
	at java.base/java.util.Scanner.throwFor(Scanner.java:964)
	at java.base/java.util.Scanner.next(Scanner.java:1619)
	at java.base/java.util.Scanner.nextInt(Scanner.java:2284)
	at java.base/java.util.Scanner.nextInt(Scanner.java:2238)
	at InputMismatchExceptionExample.main(InputMismatchExceptionExample.java:8)
```

Runtime Exception: ArithmeticException

```java
public class ArithmeticExceptionExample {
    public static void main(String[] args) {
        int dividend = 10;
        int divisor = 0;
        int result = dividend / divisor;
        System.out.println("Result: " + result);
    }
}
```

```
Exception in thread "main" java.lang.ArithmeticException: / by zero
	at ArithmeticExceptionExample.main(ArithmeticExceptionExample.java:5)
```

Runtime Exception: ClassCastException (1/2)

- Occurs when type conversion between classes is not possible

```
class Vehicle { };
class Auto extends Vehicle { };
class Bicycle extends Vehicle { };

public class ClassCastExceptionDemo {
    public static void main(String[] args) {
        Vehicle vec1 = new Auto();
        Vehicle vec2 = new Bicycle();

        Auto auto1 = (Auto) vec1;    // OK.. vec1's original class is Auto
        if (vec1 instanceof Auto) {  // preventing wrong conversion
            Auto auto2 = (Auto) vec1;
        }

        Bicycle by = (Bicycle) vec2; // OK.. vec2's original class is Bicycle
        Auto auto3 = (Auto) vec2;    // ClassCastException
                                     // vec2's original class is Bicycle, not Auto
    }
}
```

Runtime Exception: ClassCastException (2/2)

```
Exception in thread "main" java.lang.ClassCastException: class Bicycle cannot be cast to
class Auto (Bicycle and Auto are in unnamed module of loader 'app')
    at ClassCastExceptionDemo.main(ClassCastExceptionDemo.java:16)
```