Java Fundamentals
6-2: Handling Errors
Practice Activities

Lesson Objectives:

- Describe the different kinds of errors that can occur and how they are handled in Java
- Describe what exceptions are used for in Java
- Determine what exceptions are thrown for any foundation class
- Write code to handle an exception thrown by the method of a foundation class

Vocabulary:

Identify the vocabulary word for each definition below.

catch	A keyword in Java that signals the following block of code handles a specified exception.
unchecked exception	An exception that is optional to be handled.
checked exception	An exception that MUST be handled.
errors	Indicates that there is a problem with interpreting your program.
throw	This stops the interpreter from running the rest of the code until it finds a *catch*.
Syntax error	An error that indicates an issue with coding format.
exception	An error that occurs while the program is running, also known as an exception.
logic error	An error that occurs as a result of incorrect programmer logic.
catch block	A block of code that handles exceptions by dealing with the exception if it is thrown.
runtime exceptions	Errors that occur during run-time and can be corrected or *handled* by your code.

Try It/Solve It:

1. Describe the difference between a syntax error, a logic error, and an exception.
   Syntax error is prevents your code from running, like typos, Logic errors run but produce incorrect results and Exceptions are unexpected events during runtime
2. What is wrong with the following code? It should print "Hello World!" four times to the screen.

```
String str = “Hello World”;

for(int i = 0; i < 4; i++);

{

      System.out.println(str);

      str+= ”!”;

}
```

The first problem is the semicolon after the for-loop which doesn't let the code block execute. The 2nd problem is the str += adds an exclamation to each line instead of just the one that each line i supposed to have

Here is the correct code

```
String str = "Hello World!";
for(int i = 0; i < 4; i++) {
    System.out.println(str);
}
```

3. Describe an exception that you have experienced in your program before. Explain how it could be handled with a try/catch block of code.

One common exception is the NullPointerException which happens when you try to access a method or property that hasn't been instantiated. With a try and catch you can attempt the code in the try block and have the catch block catch the error and handle it if it happens.

4. Write a segment of code that has:

   a. A syntax error

   b. A logic error

   c. An exception

```
a. public class SyntaxErrorExample{
        public static void main(String[] args){
            int number = 100
            System.out.println("The number is: "  + number)
}
}
```

5. What is the difference between a checked exception and an unchecked exception?

Checked exceptions are like rules you must follow and handle in your code, as required by Java, while unchecked exceptions are like unexpected problems that pop up and you can choose to deal with them or not

```
b. public class LogicErrorExample {
    public static void main(String[] args) {
        int sum = 0;
        int[] numbers = {1, 2, 3, 4, 5};

        // Intended to calculate the sum of all elements in the array
        for (int i = 0; i <= numbers.length; i++) {
            sum += numbers[i];
        }

        System.out.println("The sum of the numbers is: " + sum);
    }
}
```

```
c. public class DivisionExceptionExample {
    public static void main(String[] args) {
        int numerator = 10;
        int denominator = 0;

        // This line will cause an ArithmeticException: division by zero
        int result = numerator / denominator;

        System.out.println("The result is: " + result);
    }
}
```