Java 第六章 异常处理上机练习

22009200601 汤栋文

2023 年 5 月 19 日

目录

1. 个人信息

姓名：汤栋文
学号：22009200601
日期：2023 年 5 月 19 日

2. 任务重述

1. 练习 PPT 中的全部小练习，尝试对小练习中各部分进行修改，并观察修改后的执行效果。
2. 编写一个程序，理解异常的效果。
3. 编写一个程序，实践自定义异常的使用。

3. PPT 练习

（此部分不要求在报告中呈现）

1. 简单描述一下异常处理的流程：执行 try 中可能引发异常的语句。如果没有异常，则完成 try 中所有语句的执行，然后执行 finally 中的语句；如果有异常，则在 catch 中依次匹配异常，如果匹配到了，就执行对应 except 中的语句，最后执行 finally 中的语句。
2. 这里面有几种极端的情况：
   1). 如果在 try 或 catch 语句中执行了 System.exit()
   2). 在 finally 之前死循环
   3). 在 finally 之前 JVM 崩溃
   4). 电源断电
   这些情况下 finally 不会被执行。
3. 另外在 try 或者 catch 中有 return 语句时，会在执行到 return 时
   即 return 的返回值已经确定了，但是还没退出这个函数时，去执行 finally。

4. 编写一个程序实践自定义异常的使用

4.1 实现过程

（因为任务本身比较简单，就把第二个任务和第三个任务合并完成了）

其实在之前的程序里面我已经用到过异常处理，在写复数类的 valueOf 函数时判断输入格式是否正确，如果正确返回对应的复数，如果不正确就抛出一个异常。

原来我使用了 Java 自带的 IllegalArgumentException 这里我自己又写了一个 IllegalComplexNumberException 类，来给出更详细的错误信息。

```
class IllegalComplexNumberException extends IllegalArgumentException{
    public IllegalComplexNumberException() {
        super("Inputed an illegal Complex, please use \"a+bi\" as input."); }
    public IllegalComplexNumberException(String message) {
        super(message+" is an illegal Complex, please use \"a+bi\" as input."); }
}
```

这里的名字与题目要求的 getInstance 不一样，但是实现的功能其实是相同的，作为一个复数类，我觉得还是 valueOf 更合适一点。

```
public static Complex valueOf(String str) {
    // Verify input
    if (str == null) { throw new NullPointerException("Input string is null"); }
    str = str.trim(); // Remove leading and trailing white spaces
    if (str.isEmpty()) { throw new IllegalComplexNumberException("Empty"); }
    if (!str.matches( regex: "-?\\d+(\\.\\d+)?([+-]\\d+(\\.\\d+)?i)?")) {
        throw new IllegalComplexNumberException(str); }
    // Remove all white spaces and split string by "+" or "-"
    String[] parts = str.replaceAll( regex: "\\s",  replacement: "").split( regex: "(?=[+-])");
    double real = 0;
    double imag = 0;
    for (String part : parts) {
        if (part.endsWith("i")) {
            // imaginary part
            part = part.substring(0, part.length() - 1); // remove "i" suffix
            if (part.equals("+") || part.equals("-")) { part += "1"; }
            imag += Double.parseDouble(part);
        } else { // real part
            real += Double.parseDouble(part); }
    }
    return new Complex(real, imag);
}
```

由此实现在构造函数中调用该静态方法，就可以实现由字符串构造复数对象。并且在输入字符串格式不正确时抛出异常。

```
public final class Complex extends Number implements Comparable<Complex>{
    // Construct
    private double real=0;
    private double imag=0;
    public Complex(double real, double imag){
        this.real = real;
        this.imag = imag;
    }
    public Complex(String string){
        Complex new_complex = valueOf(string);
        this.real = new_complex.real;
        this.imag = new_complex.imag;
    }
```

4.2 结果展示

```
public class Main {
    public static void main(String[] args) {
        Complex test = new Complex( string: "3+4j");
        System.out.println(test);
    }
```

```
Main ×
"C:\Program Files\Java\jdk-19\bin\java.exe" "-javaagent:D:\
Exception in thread "main" IllegalComplexNumberException:
3+4j is an illegal Complex, please use "a+bi" as input.
    at Complex.valueOf(Complex.java:41)
    at Complex.<init>(Complex.java:12)
    at Main.main(Main.java:4)

Process finished with exit code 1
```

```
public class Main {
    public static void main(String[] args) {
        try {
            Complex test = new Complex( string: "3+4j");
        } catch (IllegalComplexNumberException e) {
            System.out.println(e.getMessage());
        } finally {
            System.out.println("Finished!");
        }
    }
```

```
Main ×
"C:\Program Files\Java\jdk-19\bin\java.exe" "-javaagent:D:\

3+4j is an illegal Complex, please use "a+bi" as input.
Finished!
```

5. 总结

革命尚未成功，同志仍需努力