Java 程序设计 LAB08

1. 简答题

Java 中的检查型异常(checked exception)和非检查型异常(unchecked exception)有什么区别

非检查型异常包括 RuntimeException 及其子类、Error 及其子类。由系统检测，用户的 Java 程序可不做处理，系统将它们交给缺省的异常处理程序。编译器对非检查型异常类不进行检查，即这类异常在会通过编译。

检查型异常包括除了 RuntimeException 及其子类、Error 及其子类以外的其他 Exception 的子类。这些异常在编译时就能被 java 编译器所检测到异常，即不能通过编译。必须采用声明异常或者 try、catch 方式处理异常。

2. 简答题

简述 Java 异常处理中 throws 和 throw 关键字的作用。

throws:throws 语句在方法声明处声明抛出特定异常。声明异常指一个方法不处理它产生的异常，而是沿着调用层次向上传递，由调用它的方法来处理这些异常。throws 语句用来表明一个方法可能抛出的各种异常，并说明该方法会抛出但不捕获异常。
throw:throw 语句在方法中抛出具体的异常。throw 语句可以明确的抛出一个异常。
throw 抛出异常主要用于自定义异常。

3. 简答题

请列出 2 个常见的运行时异常和 2 个非运行时异常。

运行时异常：
ArithmeticsException,NullPointerException,IndexOutOfBoundsException
非运行时异常：IOException,AWTException

4. 改错题-指出下列程序的错误并给出正确的写法

在 catch (RuntimeException re){

```
        re.printStackTrace();
  }
```

之前已经有 catch (Exception ex){

```
            ex.printStackTrace();
        }
```

其中 Exception 是 RuntimeException 的父类，所以如果有 RuntimeException 的异常的异常情况出现，会进入第一个 catch 中进行相应的处理，而不会进入第二个 catch 中。
修改后的文件在：oo/04/p04.java 文件

5. 改错题-指出下列程序的错误并给出正确的写法

1．在父类中声明定义的方法，并声明异常 public void start() throws IOException{

```
    throw new IOException("Unable to start");
}
```

在子类中声明定义的方法，并声明异常 public void start() throws Exception{

```
    throw new Exception("Unable to open file");
}
```

其中在子类中声明的异常 Exception 为父类方法中声明的异常 IOException 的父类，子类重写父类的方法，并且抛出了比被父类方法范围更大的异常类型，这是错的。
2. 子类中的另一个方法 public void open(String fileName){
            FileInputStream fis=new FileInputStream(fileName);
        }
其中可能发生 FileNotFoundException 异常，并且这个异常是受检查异常，必须使用 throws 声明异常或者使用 try、catch 方式处理异常。
修改后的文件在：oo/05/SubClass.java 文件

6. 程序输出题

methodA 抛出一个异常!
执行 methodA 的 finally!
methodB 执行!
执行 methodB 的 finally!

7. 程序输出题

写出程序的输出，试着解释三个函数不同输出的原因

```
-----------------------
error
i in finally block:2
1
-----------------------
error
i in finally:okfinally
ok
-----------------------
error
i in finally:okfinally
okfinally
-----------------------
```

解释：首先执行 get0()方法，在这个方法中声明并赋值了一个 int 类型的变量 i=1，在 try 块中抛出异常，从而进入 catch 块中进行相关的处理，可以看到在 catch 块中出现了 return 语句，即在执行 finally 块前出现 return 语句，那么会把值 i=1 先缓存起来，等执行完 finally 块后，再返回缓存起来的值，即在 finally 块对值做出的改变不会影响返回的值，即在在 finally 块 i=2，但是返回后会输出 1；执行 get1()方法，在这个方法中声明并赋值了一个 String 类型的变量 i="ok"，在 try 块中抛出异常，从而进入 catch 块中进行相关的处理，可以看到在 catch 块中出现了 return 语句，即在执行 finally 块前出现 return 语句，那么会把值 i="ok"先缓存起来，等执行完 finally 块后，再返回缓存起来的值，即在 finally 块对值做出的改变不会影响返回的值，即在在 finally 块 i="okfinally"，但是返回后会输出"ok"；执行 get2()方法，在这个方法中声明并赋值了一个 StringBuilder 类型对象 i，在 try 块中抛出异常，从而进入 catch 块中进行相关的处理，可以看到在 catch 块中出现了 return 语句，即在执行 finally 块前出现 return 语句，那么会把值 i 先缓存起来，等执行完 finally 块后，再返回缓存起来的值，由于 i 是一个对象，在 finally 块 i.append("finally")改变

了原来的对象，即在在 `finally` 块中的改变会影响到缓存起来的 `i`，所以返回后会输出 `"okfinally"`。

8. 编程题-带有异常的整数计算

`oo/08`

9. 编程题-带有异常的 Triangle

`oo/09`