**实验五 异常处理**

**一、 实验目的**

1． 熟悉异常的抛出与捕获的含义；

2． 掌握异常捕获与处理的方法；

3． 能自定义异常。

**二、 实验内容**

**1**. 验证书上例子5，将运行结果保存至实验报告中，并简述Java语言处理异常的过程。

2. 现在多数学校的成绩管理都由计算机进行管理， 这就需要有相应的应用程序。 编写成绩管理应用程序， 其中有成绩录入模块， 成绩录入过程中， 难免出现录入数据的类型出现错误， 或录入的成绩不在合理的范围。 在成绩录入过程中， 若出现上述错误， 程序应该如何处理。

程序框架如下， 将代码补充完整，并调试出结果，并将运行结果保存至实验报告中：

1. 首先定义两个异常类

**负数异常.java文件源代码：**

**class** 负分异常 **extends** Exception //当录入了负分时。 此处应用了自定义异常

{

负分异常(**int** i){

System.***out***.println("分数为负数");

}

}

**高分异常.java文件源代码：**

**class** 高分异常 **extends** Exception //当录入分超过100时。 应用了自定义异常

{

高分异常(**int** i ){

System.***out***.println("分数过高");

}

}

**主类ExceptionDemo.java源文件代码：**

**import** javax.swing.JOptionPane;

**public** **class** ExceptionDemo

{

**static** **final** **int** ***number***=2;

**int** score[]=**new** **int**[***number***];

**public** **void** 检查分数(**int** 分数) **throws** 负分异常,高分异常

//下面方法中判断如果录入的成绩不合理， 则抛出异常， 但本方法并不处理异常， 仅声明了异常

{

**if**(分数 >100) **throw** **new** 高分异常(分数); //通过throw， 人工抛出异常

**if**(分数 <0) **throw** **new** 负分异常(分数);

}

**public** **void** 录入成绩()

{

**int** i;

**for**(i=0;i<***number***;i++)

{

**try**{

score[i]=Integer.*parseInt*(JOptionPane.*showInputDialog*("请输入第"+(i+1)+"个同学的成绩"));

}

**catch**(NumberFormatException e) //在try{…….}catch间的语句中若产生异常， 则捕获异常， 直接进行异常处理

{

【代码1】

}

**try**{

检查分数(score[i]);

//在检查分数模块中， 可能产生高分异常对象， 也可能产生低分异常对象， 根据不同情况，

//进行不同处理

}

**catch**(高分异常 e)

{

System.***out***.println(e);

}

**catch**(负分异常 e)

{

System.***out***.println(e);

}

}

}

**public** **void** 输出成绩()

{

System.***out***.println(score[0]);

System.***out***.println(score[1]);

}

**public** **static** **void** main(String arg[])

{

ExceptionDemo demo = **new** ExceptionDemo();

demo.录入成绩();

demo.输出成绩();

}

}

**3.** 理解 finally 子句的作用

当一个异常被抛出时，程序的执行流程就不再是连续的了，会跳过某些语句，甚至会由于没有与之匹配的catch 子句而过早地返回，结束程序的运行。为了确保一段代码不管发生什么异常都能被执行，可以使用finally 子句。每个try 语句至少都要有一个与之相配的catch 或finally 子句。从一个方法中返回到调用它的另外一个方法，可以通过return 语句或通过一个没有被捕获的异常，但finally 子句总是在返回前被执行。

**创建使用finally 子句的程序文件TestFinally.java。**

（1）程序功能：在类TestFinally 的mathodA 方法中使用try 子句捕获异常，使用finally 子句处理异常。在类TestFinally 的main 方法中使用try 子句捕获mathodA 异常，

class TestFinally{

static void mathodA() {

try {

System.out.println("\nmathodA 抛出一个异常");

throw new RuntimeException();

}finally {

System.out.println("执行 mathodA 的 finally");}

}

static void mathodB() {

try{

System.out.println("mathodB 正常返回");

return;

}finally {

System.out.println("执行 mathodB 的 finally");}

}

public static void main(String args[]) {

try {

mathodA();

} catch (Exception e) {

mathodB(); }

}

}

**编译并运行该程序，将运行结果保存至实验报告，并回答try，catch，和finally，throw，throws五个关键字在异常信息处理中各自的作用。**

**三、提交各实验程序源代码及实验报告**