第 5 章

异常处理与输入输出流

要点导读

本章内容需要与配套的主教材《Java 语言程序设计》(第 3 版)第 5 章配合学习。

主教材第 5 章介绍了异常处理与输入输出流。异常(Exception)又称为例外,是特殊的运行错误对象。Java 提供了异常处理机制,在 Exception 类中定义了程序产生异常的条件。对待异常通常并不是简单地结束程序,而是转去执行某段特殊代码处理这个异常,设法恢复程序继续执行。Java 中预定义的常用异常有 ArithmeticException、NullPointerException、ArrayIndexOutOfBoundsException、NegativeArraySizeException、FileNotFoundException、IOException 等。对于异常,可以声明抛出,由调用方法处理,也可以捕获并处理。抛出异常使用 throws 子句实现,捕获和处理异常使用 try、catch 和 finally 语句。程序员也可以自定义异常类,表示某种异常。

在 Java 中输入/输出操作通常都是通过输入/输出流来实现的。一个流就是一个从源流向目的地的数据序列。输入/输出流可以与各种数据源和目标相连。Java 所有的输入和输出操作都要通过 IO 包中的一些流类的方法来实现。从流的方向划分,可以分为输入流和输出流;从流的分工划分,可以分为节点流和处理流;从流的内容划分,可以分为面向字符的流和面向字节的流。

java.io 包中提供了用于读写文本文件和二进制文件的类和方法,另外,对于大文件的读写,可以使用缓冲流提高效率。File 类是 IO 包中唯一表示磁盘文件信息的对象,它定义了一些与平台无关的方法来操纵文件。java.util.zip 包中提供了一些类,可以以压缩格式对流进行读写。java.io 包中提供了专门用于对象信息存储和读取的输入输出流类 ObjectInputStream 和 ObjectOutputStream。java.io 包提供了 RandomAccessFile 类用于随机文件的创建和访问。

实验 5　异常处理与输入输出流

一、实验目的

(1) 了解 Java 的异常处理机制,会编写异常处理程序。

(2) 理解输入输出(I/O)流的概念,掌握其分类。

(3) 掌握文本文件读写、二进制文件读写、处理流类的概念和用法、对象序列化。

(4) 掌握 File 类、压缩流类、随机读写流类。

(5) 遇到 I/O 方面的问题,能够自行查阅 API 文档解决。

二、实验任务

1. 函数返回及异常处理

目测以下程序代码能编译通过吗? 不能的话有什么错误? 若有错误则尝试在不改变原程序语义的情况下,给出两种解决方案并分别实现在 Calculator1.java 和 Calculator2.java 中,解决针对 Exception 的错误。找错误时先不使用 IDE 辅助,尽可能找出所有错误。

```
public class Calculator {

    public float getValue(String type) throws Exception {
        //获得产品单价
        if (type.equals("cookie")) {
            return 1.11f;
        } else if (type.equals("pie")) {
            return 5.5f;
        }
    }

    public int getValue(String type) throws Exception {
        //获得产品数量
        if (type.equals("cookie")) {
            return 10;
        } else {
            return 20;
        }
    }

    public float calculate() {
        float price = getValue("cookie");
        int amount = getValue("cookie");
        return price * amount;
    }
}
```

2. 商品找零

从键盘上输入一件商品的价格(0.01～5.00 元)。如果支付 5 元,给出一种找零方案,使得所找纸币及硬币的个数最少。例如,输入物品价格 1.68 元,给出找零方案 2 元 1 张,1 元 1 张,2 角 1 个,1 角 1 个,2 分 1 个。可找币种类包括所有发行的纸币和硬币。

如果输入在 0.01～5.00 范围内则给出方案并继续等待下个输入;如果输入 0,则程序退出。如果输入 3 位(含)以上的小数,例如 3.618,则抛出自定义异常 WrongFormatException,并让用户继续重新输入;如果输入负数或者大于 5 的数值,则抛出自定义异常

InputOutOfRangeException，并且程序终止。

3. 各种输入/输出流的特点

简要回答问题：FileReader 和 FileInputStream 有什么共同点？有什么区别？

读写文件时，可直接使用 FileReader/FileWriter，FileInputStream/FileOutputStream。也可以在其上套上一层 BufferedReader/BufferedWriter，BufferedInputStream/BufferedOutputStream。这两种方法有什么区别？一般读写较大文件时应该使用哪种方法？

写一行代码说明如何在 FileReader 上再套一层 BufferedReader。

4. 继续完善记账软件：对象序列化

在实验四的实验任务 3 基础上继续完善记账软件，增加对象序列化功能。

(1) 思考如何给 myaccount.model.Deal 实现对象序列化，序列化一个所有成员变量都不是 null 的 NormalDeal 对象(给它的所有成员变量赋值)，可以序列化成功吗？还需要对其他的类做什么修改？为什么？

(2) 保存二进制文件。

随机生成 10 个不同类型的具体 Deal 对象。要求类型、日期等变量为随机值。将它们以二进制的形式保存到文件中(deal.dat)。

(3) 保存文本文件。

将上题中保存的文件读入，并以一定格式显示到屏幕上，显示的同时将这些文本写入文本文件 dealreport.txt。

(4) 文件打包成 zip 文件。

编程将本任务中生成的 deal.dat 和 dealreport.txt 打包成 deal.zip 文件。

5. 附加题：编码转换

通过字节流类和字符流类，配合 byte[]和 String 类。实现一个文件到另一个文件的编码转换。例如，从 GBK 编码到 UTF-8 编码等的转换。了解 Java 相关的编码转换问题。

三、实验步骤

1. 解决实验任务 1 中的编译错误

(1) 目测题目中代码的编译错误。指出这些错误，并设计出两种异常的处理方法。

(2) 将两种解决方案实现。

2. 验证实验任务 1 的程序

(1) 创建项目 exp5，创建 Java 包，名称为“problem1”。

(2) 在 problem1 包下创建 Calculator.java。将题目中的代码输入，并编译。看看自己是否找到了所有的错误。编译器只显示最高层的错误，将高层的错误修改后可以看到更细节的错误。实际应该找到 3 个不同类型的编译错误。

(3) 在 problem1 包下创建 Calculator1.java 和 Calculator2.java。在不大量改变原程序语义的情况下，修改 Calculator.java 中的错误。分别将两种修改方案在这两个类中实现。

3. 完成实验任务 2 商品找零

(1) 在项目 exp5 中创建新的包和类。创建 Java 包，名称为“problem2”。在 problem2 包下创建 ChangeCalculator.java。

(2) 创建 WrongFormatException.java,InputOutOfRangeException.java。并实现这两个自定义异常类。

(3) 在 ChangeCalculator 类中实现题目中要求的方法。

(4) 对程序进行验证,输入以下数值验证程序的输出。

```
输入: 0          输出: 程序退出
输入: 5.00       输出: 无须找零的方案
输入: 1.68       输出: 2 元 1 张,1 元 1 张,2 角 1 个,1 角 1 个,2 分 1 个的方案
输入: 0.01       输出: 2 元 2 张,5 角 1 个,2 角 2 个,5 分 1 个,2 分 2 个的方案
输入: 2.222      输出: 显示异常 WrongFormatException 并等待重新输入
输入: 6.05       输出: 显示异常 InputOutOfRangeException 并退出
```

注意:此题最容易出现的错误是少找一分钱,例如,输入 0.01 时,输出的不是 2 个 2 分,而是 1 个 2 分和 1 个 1 分。其原因在于浮点数在计算机中的表示是不精确的。此题中,如果要知道某浮点数对应的整数,可以使该浮点数加上一个很小的数,并对结果取整。例如,对浮点数 8.99,如果想得到结果 9,可以使 8.99 加上 0.02,再取整,就可以得到整数 9,即

```
float f = 8.99;
int i = (int)(f + 0.02);
```

此时 i 的值为 9。

4. 完成改进版的记账软件

(1) 复习、理解流的概念、分类、功能,理解流的套接。思考如何给 myaccount.model.Deal 实现对象序列化。

(2) 打开 MyAccount 工程,将题目要求的代码实现在 myaccount.util 包下。

(3) 实现 RandomDealWriter 类。

(4) 实现 TextDealWriter 类。

(5) 实现 DealZipWriter 类。

5. 实现实验任务 5 的编码转换程序(附加,选做)

在目录 exp5 下创建 Java 包,名称为“problem5”。在 problem5 包下创建 EncodeTransfer.java,在其中完成实验任务 5。

(1) 使用 FileInputStream 类从源文件读取内容。

(2) 使用 String 类的 getBytes("GBK")方法将源文件内容解码。

(3) 使用 String 类的编码构造方法 String(xxx, "UTF-8")将内容进行重新编码。

(4) 使用 FileWriter 类将新编码输出到目标文件。

习题解答

1. 什么是异常? 解释抛出异常和捕获异常的含义。

解:

异常(Exception)又称为例外,是特殊的运行错误对象,对应着 Java 语言特定的运行错

误处理机制。抛出是指：不在当前方法内处理异常，而是把异常抛出到调用方法中。捕获是指：使用 try{}catch(){}块，捕获到所发生的异常，并进行相应的处理。

2. 简述 Java 的异常处理机制。

解：

在一个方法的运行过程中，如果发生了异常，则这个方法（或者是 Java 虚拟机）便生成一个代表该异常的对象（包含该异常的详细信息），并将它交给运行时系统，运行时系统查找方法的调用栈，从生成异常的方法开始进行回溯，直到找到包含相应异常处理的方法为止。

3. 系统定义的异常与用户自定义的异常有何不同？如何使用这两类异常？

解：

系统定义的异常是 Java 处理程序错误的一种方式，系统为可能产生非致命性错误的代码段设计错误处理模块，例如，整数除法中，除数为 0 等。用户自定义异常是为了防止程序中断或是出现未知错误。

4. 用户程序如何自定义异常？编程实现一个用户自定义异常。

解：

用户自定义异常类时，只需继承类 Exception 即可。

新建 Exe5_4.java 文件，其内容为：

```
class AgeBelowZeroExeption extends Exception{
    public AgeBelowZeroExeption() {
      super("Age is below zero");
   }
}
public class Exe5_4 {
    private int age;
    public static void setAge(int age) throws AgeBelowZeroExeption{
        if (age < 0)
            throw new AgeBelowZeroExeption();
    }
    public static void main(String[] args) {
        try {
            setAge(-2);
        }
        catch (Exception e) {
            System.out.println(e);
        }
    }
}
```

程序的输出结果如下。

```
AgeBelowZeroExeption: Age is below zero
```

5. 模仿文本文件复制的例题，编写对二进制文件进行复制的程序。

解：

新建 Exe5_5.java 文件，其内容为：

```
import java.io.*;
class BinaryFileCopy {  //声明一个类
    String sourceName, destName;
    FileInputStream source;
    FileOutputStream dest;
    String line;
    //这个私有方法用来打开源文件和目的文件,如无异常则返回 true
    private boolean openFiles() {
        try {
            source = new FileInputStream(sourceName);  //打开源文件
        }
        catch (IOException iox) {
            System.out.println("Problem opening " + sourceName);
                                                        //出现异常显示出错信息
            return false;
        }
        try {
            dest = new FileOutputStream(destName);      //打开目的文件
        }
        catch (IOException iox) {
            System.out.println("Problem opening " + destName);
            return false;
        }
        return true;
    }
    private boolean copyFiles() {  //这个私有方法用来复制文件,如无异常返回 true
        try {
            byte[] buf = new byte[512];
            int num = source.read(buf);         //从源文件读取数据
            while (num > 0) {                   //只要能够读取数据,就继续读
                dest.write(buf, 0, num);        //向目标文件写入
                num = source.read(buf);         //从源文件读取数据
            }
        }
        catch (IOException iox) {
            System.out.println("Problem reading or writing");
            return false;
        }
        return true;
    }
    private boolean closeFiles() {  //此私有方法用来关闭文件,如无异常返回 true
        boolean retVal = true;
```

```
        try {
            source.close();
        }
        catch (IOException iox) {
            System.out.println("Problem closing " + sourceName);
            retVal = false;
        }
        try {
            dest.close();
        }
        catch (IOException iox) {
            System.out.println("Problem closing " + destName);
            retVal = false;
        }
        return retVal;
    }
    public boolean copy(String src, String dst) {
    //这个类中唯一的公有方法,需两个字符串参数
        sourceName = src;
        destName = dst;
        //调用三个私有方法,若都正常返回 true,有问题则返回 false,并显示相应出错信息
        return openFiles() && copyFiles() && closeFiles();
    }
}
public class Exe5_5 {
    public static void main(String[] args) {   //main()函数为程序入口
        if (args.length == 2)       //要求提供两个参数作为源和目标文件名
        {
            new BinaryFileCopy().copy(args[0], args[1]);
        } //新建一个 CopyMaker 类的对象并执行其 copy()方法,参数由命令行提供
        else {
            //如果不是两个参数,则给出提示信息,程序结束
            System.out.println("Please Enter File names");
        }
    }
}
```

6. 创建存储若干随机整数的文本文件,文件名、整数的个数及范围均由键盘输入。

解:

使用 Keyboard 类获得输入,Keyboard 类如下。新建 Keyboard.java 文件,其内容为:

```
import java.util.Scanner;
import java.io.*;
public class Keyboard {
```

```
    static BufferedReader inputStream = new BufferedReader(new
InputStreamReader(System.in));
    public static int getInteger() {
        try {
            return (Integer.valueOf(inputStream.readLine().trim()).intValue());
        } catch (Exception e) {
            e.printStackTrace();
            return 0;
        }
    }
    public static String getString() {
        try {
            return inputStream.readLine();
        } catch (Exception e) {
            e.printStackTrace();
            return null;
        }
    }
}
```

新建 Exe5_6.java 文件，其内容为：

```
import java.io.*;
import java.util.*;
public class Exe5_6 {
    public static void main(String[] args) {
        System.out.println("请输入文件名: ");
        String fileName = Keyboard.getString();
        System.out.println("请输入整数的个数: ");
        int count = Keyboard.getInteger();
        System.out.println("请输入整数范围(最小): ");
        int rangeLow = Keyboard.getInteger();
        System.out.println("请输入整数范围(最大): ");
        int rangeHigh = Keyboard.getInteger();
        File f = new File(fileName);
        try {
            BufferedWriter bw = new BufferedWriter(new FileWriter(f));
            Random r = new Random();
            int range = rangeHigh - rangeLow;
            for(int i = 0; i < count; i++) {
                int number = r.nextInt(range);
                number += rangeLow;
                bw.write(((Integer)number).toString());
                bw.write("\r\n");
            }
```

```
            bw.close();
        }
        catch(Exception e) {
            System.out.println(e);
            System.exit(-1);
        }
    }
}
```

7. 分别使用 FileWriter 和 BufferedWriter 往文件中写入 10 万个随机数，比较用时的多少。

解：

新建 Exe5_7.java 文件，其内容为：

```
import java.io.*;
import java.util.*;
public class Exe5_7 {
    public static void main(String[] args) throws IOException {
        File f1 = new File("D:/random1.txt");
        FileWriter fw = new FileWriter(f1);
        Random r = new Random();
        Calendar start = Calendar.getInstance();
        for (int i = 0; i < 100000; i++) {
            int number = r.nextInt(100);
            fw.write(((Integer) number).toString());
        }
        fw.close();
        Calendar end = Calendar.getInstance();
        long time = end.getTimeInMillis() - start.getTimeInMillis();
        System.out.println("使用 FileWriter 的时间为" + time + "毫秒");
        File f2 = new File("D:/random2.txt");
        BufferedWriter bw = new BufferedWriter(new FileWriter(f2));
        start = Calendar.getInstance();
        for (int i = 0; i < 100000; i++) {
            int number = r.nextInt(100);
            bw.write(((Integer) number).toString());
        }
        bw.close();
        end = Calendar.getInstance();
        time = end.getTimeInMillis() - start.getTimeInMillis();
        System.out.println("使用 BufferedWriter 的时间为" + time + "毫秒");
    }
}
```

程序某次执行的结果为：

```
使用 FileWriter 的时间为 162 毫秒
使用 BufferedWriter 的时间为 15 毫秒
```

8. 用记事本程序创建一篇包含几十个英语单词的小文章，要求从屏幕输出每一个单词。

提示： 查阅 StreamTokenizer、StringTokenizer 类的说明。

解：

新建 Exe5_8.java 文件，其内容为：

```
import java.io.*;
import java.util.*;
public class Exe5_8 {
    public static void main(String[] args) throws IOException {
        File f = new File("D:/word.txt");
        FileReader fr = new FileReader(f);
        char[] buf = new char[(int) f.length()];
        int num = fr.read(buf, 0, buf.length);
        String contents = new String(buf);
        StringTokenizer st = new StringTokenizer(contents);
        while (st.hasMoreTokens()) {
            String s = st.nextToken();
            System.out.println(s);
        }
    }
}
```

9. 从键盘输入一系列字母，将其存储到文件中，对其进行升序排序后，存到另一个文件中，并显示在屏幕上。

解：

新建 Exe5_9.java 文件，其内容为：

```
import java.io.*;
import java.util.*;
public class Exe5_9 {
    public static void main(String[] args) throws IOException{
        System.out.println("请输入字母序列:");
        String s = Keyboard.getString();
        File f1 = new File("D:/a.txt");
        FileWriter fw = new FileWriter(f1);
        fw.write(s);
        byte[] buf = s.getBytes();
        //进行升序排序
         for (int i = 0; i < buf.length - 1; i++) {
            for (int j = i + 1; j < buf.length; j++) {
```