# Java程序设计实验报告

**学号： 1190201421**

**姓名： 张瑞**

**专业： 工科试验班（计算机与电子通信）**

**班级： 19L0214**

**哈尔滨工业大学**

一、实验内容

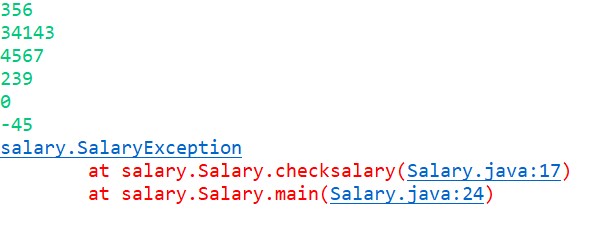
1. **编写一个除法计算器，要求用户输入被除数和除数，程序给出商。要求程序健壮。**
2. **自定义异常类并使用，要求用户创建一个工资类，用以描述员工的工资，同时，在用户输入每个员工工资时，对一些不合事实的输入数据进行处理，例如工资不能为负等。**
3. **编写一个类，实现图片.jpg文件的复制。**
4. **运行并分析一个无聊的城堡文字游戏(castle.zip)，从面向对象设计的角度找出代码中存在问题，并尝试修改。**

二、实验运行结果

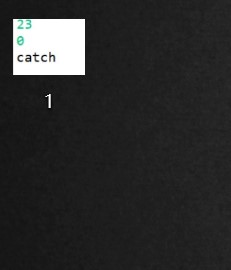
1.



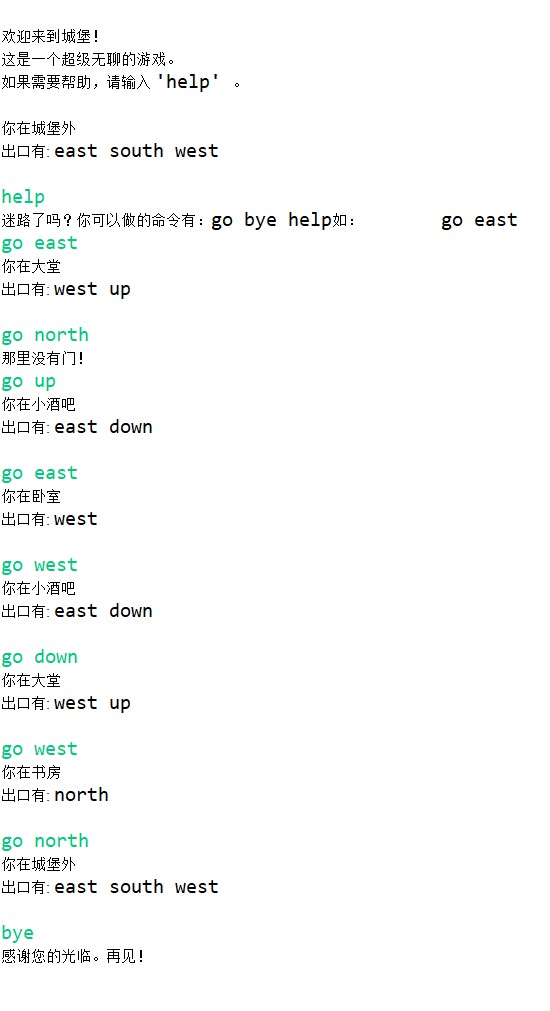
2.



3.

4.



三、程序源代码

**package** devision;

**import** java.util.Scanner;

**public** **class** Devision {

**public** **static** **void** main(String[] args) {

**int** a,b;

Scanner in = **new** Scanner(System.***in***);

a = in.nextInt();

b = in.nextInt();

**try** {

System.***out***.println(a/b);

}**catch**(ArithmeticException e) {

System.***out***.println("catch");//除数为零时，捕捉

}

in.close();

}

}

**package** salary;

**import** java.util.Scanner;

**class** SalaryException **extends** Throwable{//自定义异常类

}

**public** **class** Salary {

**public** **static** **void** checksalary() **throws** SalaryException {

**int**[] salary = **new** **int**[10];//存储工资的数列

Scanner in = **new** Scanner(System.***in***);

**for**(**int** i = 0;i<salary.length;i++) {//依次读入工资

salary[i] = in.nextInt();

**if**(salary[i]<0) {

**throw** **new** SalaryException();//工资为负时，抛出异常

}

}

}

**public** **static** **void** main(String[] args) {

**try** {

*checksalary*();

}**catch**(SalaryException e) {//若捕捉到异常，打印出现异常的位置

e.printStackTrace();

}

}

}

**3.**

**package** copy;

**import** java.io.File;

**import** java.io.FileInputStream;

**import** java.io.FileOutputStream;

**import** java.io.InputStream;

**import** java.io.OutputStream;

**public** **class** Copy {

**public** **static** **void** main(String[] args) **throws** Exception {

File file=**new** File("C:\\Users\\ZR\\Desktop");

String path=file.getAbsolutePath();//获取文件夹的绝对路劲

String isfilepath=path+File.***separator***+"1.jpg";//源图片路径

String osfilepath=path+File.***separator***+"2.jpg";//复制图片路径

InputStream is=**new** FileInputStream(isfilepath);//输入流读图片

OutputStream os=**new** FileOutputStream(osfilepath);//输出流写图片

**byte**[] b=**new** **byte**[1024];

**int** i=0;

**while**((i=is.read(b))!=-1){//循环读取与写

os.write(b,0,i);

}

os.close();

is.close();

}

}

4.

**package** castle;

**import** java.util.HashMap;

**import** java.util.Scanner;

**public** **class** Game {

**private** Room currentRoom;

**private** HashMap<String, Handler> handlers = **new** HashMap<String, Handler>();

**public** Game() {

handlers.put("bye", **new** HandlerBye(**this**));

handlers.put("help", **new** HandlerHelp(**this**));

handlers.put("go", **new** HandlerGo(**this**));

createRooms();

}

**private** **void** createRooms(){

Room outside, lobby, pub, study, bedroom;

// 制造房间

outside = **new** Room("城堡外");

lobby = **new** Room("大堂");

pub = **new** Room("小酒吧");

study = **new** Room("书房");

bedroom = **new** Room("卧室");

// 初始化房间的出口

outside.setExit("east", lobby);

outside.setExit("south", study);

outside.setExit("west", pub);

lobby.setExit("west", outside);

pub.setExit("east", outside);

study.setExit("north", outside);

pub.setExit("east", bedroom);

bedroom.setExit("west", pub);

lobby.setExit("west", study);

lobby.setExit("up", pub);

pub.setExit("down", lobby);

currentRoom = outside; // 从城堡门外开始

}

**private** **void** printWelcome() {

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

System.***out***.println("欢迎来到城堡！");

System.***out***.println("这是一个超级无聊的游戏。");

System.***out***.println("如果需要帮助，请输入 'help' 。");

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

showPrompt();

}

// 以下为用户命令

**public** **void** goRoom(String direction) {

Room nextRoom = currentRoom.getExit(direction);

**if** (nextRoom == **null**) {

System.***out***.println("那里没有门！");

}

**else** {

currentRoom = nextRoom;

showPrompt();

}

}

**public** **void** showPrompt() {

System.***out***.println("你在" + currentRoom);

System.***out***.print("出口有: ");

System.***out***.println(currentRoom.getExitDesc());

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

}

**public** **void** play() {

Scanner in = **new** Scanner(System.***in***);

**while** ( **true** ) {

String line = in.nextLine();

String[] words = line.split(" ");

Handler handler = handlers.get(words[0]);

String value = "";

**if** (words.length>1)

value = words[1];

**if** (handler != **null**) {

handler.doCmd(value);

**if** (handler.isBye()) {

System.***out***.println("感谢您的光临。再见！");

**break**;

}

}

}

in.close();

}

**public** **static** **void** main(String[] args) {

Game game = **new** Game();

game.printWelcome();

game.play();

}

}

**package** castle;

**public** **class** Handler {

**protected** Game game;

**public** Handler(Game game) {

**this**.game = game;

}

**public** **void** doCmd (String word) {

}

**public** **boolean** isBye() {

**return** **false**;

}

}

**package** castle;

**public** **class** HandlerBye **extends** Handler {

**public** HandlerBye(Game game) {

**super**(game);

}

@Override

**public** **boolean** isBye() {

**return** **true**;

}

}

**package** castle;

**public** **class** HandlerGo **extends** Handler {

**public** HandlerGo(Game game) {

**super**(game);

}

@Override

**public** **void** doCmd(String word) {

game.goRoom(word);

}

}

**package** castle;

**public** **class** HandlerHelp **extends** Handler {

**public** HandlerHelp(Game game) {

**super**(game);

}

@Override

**public** **void** doCmd(String word) {

System.***out***.print("迷路了吗？你可以做的命令有：go bye help");

System.***out***.println("如：\tgo east");

}

}

**package** castle;

**import** java.util.HashMap;

**public** **class** Room {

**private** String description;

**private** HashMap<String, Room> exits = **new** HashMap<String, Room>();

**public** Room(String description) {

**this**.description = description;

}

**public** **void** setExit(String dir, Room room) {

exits.put(dir, room);

}

@Override

**public** String toString(){

**return** description;

}

**public** String getExitDesc() {

StringBuffer sb = **new** StringBuffer();

**for** (String dir : exits.keySet()) {

sb.append(dir);

sb.append(' ');

}

**return** sb.toString();

}

**public** Room getExit(String direction) {

**return** exits.get(direction);

}

}