I/O （100%）

第一题：（15%）

A. Java 中存在两种类型的异常，运行时异常和非运行时异常，简要说明这两种异常发生的场景，并举例说明。（简单代码表示）

答案摘要：

运行时异常继承于 RuntimeException。Java 编译器允许程序不对它们做出处理，如空指针异常，数组越界异常等。代码示例：

```
class ExceptionDemo{
public static void main( String args[ ] ){
    int a = 0;
    System.out.println( 5/a );
}
}
```

非运行时异常除了运行时异常外的其他由 Exception 继承来的异常类。Java 编译器要求程序必须捕获或者声明抛出这种异常。如 IO 异常等

```
import java.io.*;
Class ExceptionDemo2{
public static void main(String args[]) throws FileNotFoundException
{
    FileInputStream fis = new FileInputStream( "test" );
}
}
```

B.在 try .. catch .. finally 结构中，简要说明 finally 的作用，并举例说明。（简单代码表示）

答案摘要：捕获异常的最后一步是通过 finally 语句为异常处理提供一个统一的出口，使得在控制流转到程序的其它部分以前，能够对程序的状态作统一的管理。不论在 try 代码块中是否发生了异常事件，finally 块中的语句都会被执行。finally 块经常用于释放资源。

```
public static final String test() {
        String t = "";
        try {
            t = "try";
```

```
            return t;
        } catch (Exception e) {
            // result = "catch";
            t = "catch";
            return t;
        } finally {
            t = "finally";
        }
    }
```

第二题：（15%）

```
import java.io.*;
public class Quiz1{
    public static void main(String arg[]){
    int i;
    System.out.print("Go ");
    try{
        System.out.print("in ");
        i=System.in.read();
        if (i=='0') {throw new MyException();}
        System.out.print("this ");
    }
    catch(IOException e){}
    catch(MyException e){
        System.out.print("that ");
    }
    System.out.print("way.\n");
    }
}
class MyException extends Exception{}
```

运行该程序后输入字符'0',请问运行结果为何？

（A）Go in this way

（B）Go in that this way

（C）Go in that

（D）Go in that way

第三题（提交代码）：（20%）

代码填充：对任意指定的文件，按照指定字符集读取文件，在控制台中按照原文件格式

输出文件所有内容。

使用示例：　　readFile(“./hello.txt”,”utf8”)

一种实现：

```
public void readFile(String fileName，String charSet)
{
    StringBuffer strBuf = new StringBuffer();

        BufferedReader br = null;
        try
        {
            br = new BufferedReader(new InputStreamReader(new FileInputStream(
                    new File(fileName)),charSet));

            String temp = null;

            while (true)
            {
                temp = br.readLine();

                if (temp == null)
                {
                    break;
                }
                strBuf.append(temp + "\n");
            }
        } catch (FileNotFoundException e)
        {
            e.printStackTrace();
        } catch (IOException e)
        {
            e.printStackTrace();
        } finally
        {
            if (br != null)
            {
                try
                {
                    br.close();
                } catch (IOException e)
                {
                    e.printStackTrace();
                }
```

```
            }
        }
        return strBuf.toString();
}
```

第四题（提交代码）：（20%）

A.编写一个程序，用于从键盘读入信息，并以字符串形式（明文形式，即打开文件后可以看到写入的内容）存储到当前路径下 info.txt 中。

要求：

1. 以行的方式分别读入姓名和学号信息，例如：张三 20071215

在 cmd 中输入示例

> enter StudentName：（输入提示）
>张三（键盘输入）
>enter StudentNo:
>20071215（键盘输入）
>ok,info :张三 20071215 written successfully!
>continue? (y or n)
>y
>enter StudentName:
.....

2.循环读入，每次输入完成后询问是否继续，输入 y 继续，n 退出程序。

3.文件保存在当前程序运行目录下，若文件不存在则创建,若文件存在则向文件追加内容。

4.在整个上述过程中，要做例外处理

5.StudentName 为随意字符串（2<=长度<=20），对 StudentNo 要进行非数字字符的辨 别处理（学号长度规定为 8），如 2h071215 被认为错误输入，要求重新输入。

简单参考：

键盘输入： BufferedReader br= **new** BufferedReader(**new** InputStreamReader(System.*in*));

当前目录：“.”， new File(“./info.txt”) new File(“info.txt”) 均表示当前目录下的 info.txt

循环输入： while+循环结束条件

B.利用第三题代码，将 info.txt 内容读出来。

第五题（提交代码）：（30%）

编写程序，列出当前目录（包含其任意深度的子目录）下的文件名称中包含某一字符串的所有文件,要求显示完整路径，大小写不敏感。

示例：cmd
> search content: (输入提示)
>he（键盘输入）
>ok.
Moto.he ./Moto.he
Hello.txt ./ss/Hello.txt
sshe.oo ./ufo/big/sshe.oo
.....
>continue?(yes or no)
>yes
>search content:
>.....

>no
>exit

部分程序代码参考：

```
//递归遍历文件夹并搜索指定内容
public static  void search(File path,String info)
{
    if(info==null||path==null||path.isFile())
    {
        return;
    }
    File[] files=path.listFiles();
    if(files.length==0)
    {
        return;
    }

    for(File f:files)
    {
        if (f.isFile())
        {
            String fileName=f.getName();
```

```
                //也可根据需要将下列判断条件改为使用正则表达式匹配
                if(fileName.toLowerCase().contains(info.toLowerCase()))
                {
                    System.out.println("\t"+f.getName()+"\t\t"+f);
                }
            }else
            {
            search(f,info);
            }
        }
}
```