Java07

Task1

1. 1 Exception

- ArrayIndexOutOfBoundsException

这是我最早接触数组时遇到的异常 导致给数组中的数赋值时索引会超过数组大小界限 例如:

```
package error;
import java.util.Scanner;

public class Test {

        public static void main(String[] args) {
                Scanner in = new Scanner(System.in);
                String line = in.nextLine();
                String[] words = line.split(" ");
                System.out.println(words[1]);
        }

}
```

> 我第一次见到这种数组大小不是一开始就固定的时候感觉挺新奇的 印象比较深 举了这个例子

这里通常有一个解决方案就是

```
                if(words.length > 1) {
                        System.out.println(words[1]);
                }
```

- NullPointerException

这是指访问了值null的变量 例如:

```
package error;

public class Test {

        public static void main(String[] args) {
                String a = null;
                System.out.println(a);
        }

}
```

> 对于Exception 通常是用try-catch和throw来解决

1. 2 Error

- OutOfMemoryError

指的是超出了系统准许的最大内存 例如数组的元素个数超出了最大限制:

```
package error;

public class Test {

        public static void main(String[] args) {
                int[] b = new int[Integer.MAX_VALUE];
        }

}
```

- StackOverflowError

这个指栈溢出 常见于无限递归中 例如:

```
package error;

public class Test {

	public static void f() {
		f();
	}

	public static void main(String[] args) {
		f();
	}

}
```

这个运行起来就会有一大堆的错误提示 显然这个f()会一直不断运行无法结束 最后导致突破限制

> 无论是OutOfMemoryError还是StackOverflowError 都是因为系统资源限制等原因而出错 所以为了避免过度内存占用和性能损耗 通常我们要考虑修改代码 消除这些error 而不是catch他们

2.

- unchecked异常 顾名思义 编译时不会检测的异常 上述Exception的两个例子都是unchecked异常 敲代码的时候无需声明抛出异常 编译器并不会给你报错 运行时则会出现异常 你可以选择使用try-catch处理或者更正代码
- checked异常 相对应 就是编译时会检测的异常 必须要提前声明 否则编译器不给你通过 通常为可预见的异常 例如我们可以自己创建一个由Exception派生出的子类:

```
package error;

class MyException extends Exception{

}

public class Test {

        public static void f() throws MyException {
                System.out.println("hello");
                throw new MyException();
        }

        public static void main(String[] args) {
                try {
                        f();
                } catch (MyException e) {
                        System.out.println("caught!");
                }
        }

}
```

你都已经声明了f()会抛出MyException那系统就认为它会抛出 不管在它内部有没有return new MyException() 都必须在f()处用try-catch捕捉或者在main后面抛出异常 否则编译是不会通过的 而在unchecked异常中编译却是可以通过的

Task2

3. 在new一个对象给account时 初始金额是0到200的随机数 一共两种情况

- 第一种 初始金额大于等于150 会先调用getBalance 输出当前余额 然后进入withdraw 不做if里面的语句 把150从余额里减去 取款成功 程序结束
- 第二种 初始金额不足150 依旧会调用getBalance 输出当前金额 然后进入withdraw 进到if里面的语句 throw出一个新new的InsufficientFundsException类对象 并初始化了它的message 程序在出现异常的地方 即withdraw中止 不做后面语句 跳到异常被捕捉到到的地方catch 做catch后的语句 输出错误:message 程序结束

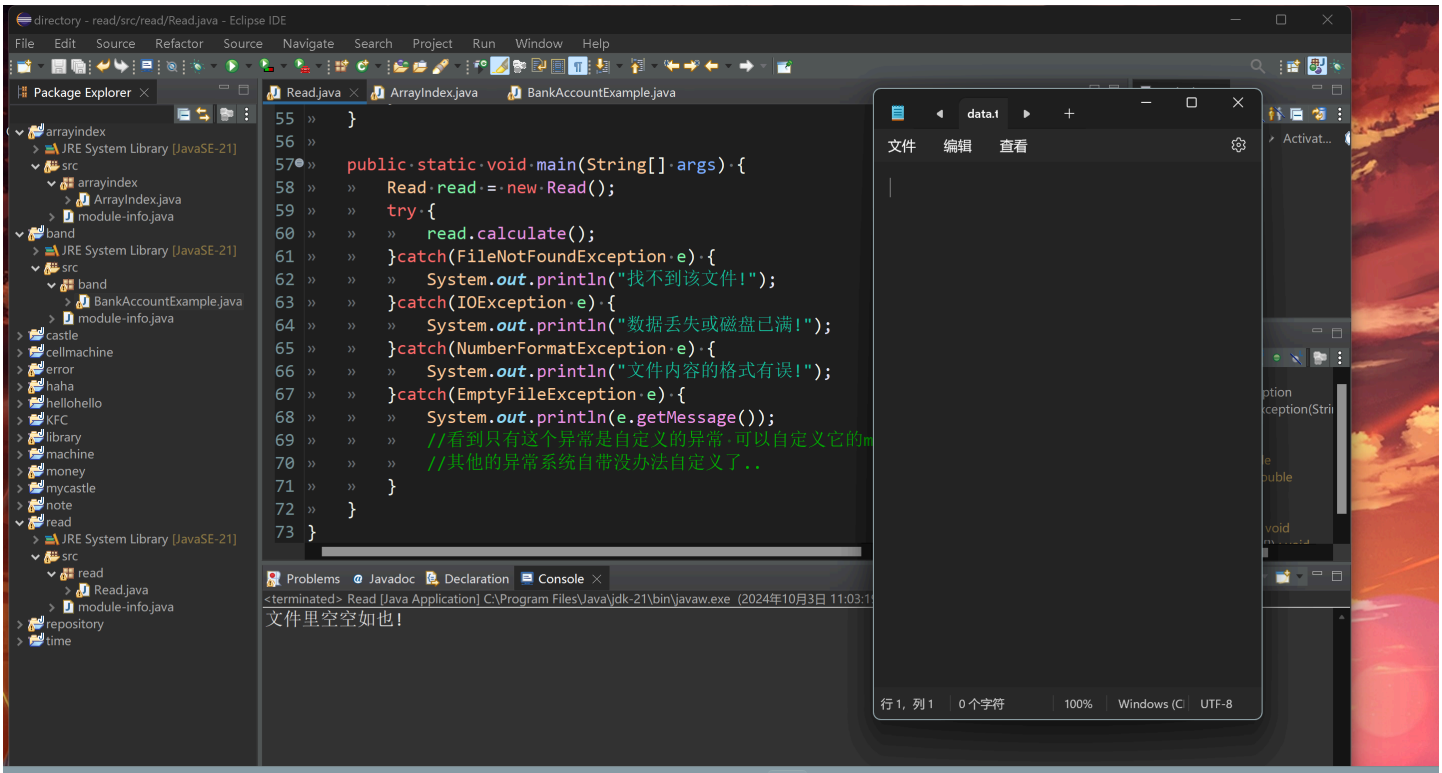
4. 这个题搞了好久 琢磨半天才搞明白怎么读取文件的 涉及到很多我没见过的类 而且题目中要求的自定义异常有两个是系统库中自带的异常 当你在调用一些方法或者创建一些类时可能会抛出这些异常 对于checked异常我们又必须要处理 提前声明 并用catch捕捉 因此对于这些异常我就直接处理系统会抛出的异常了 没有额外再自定义

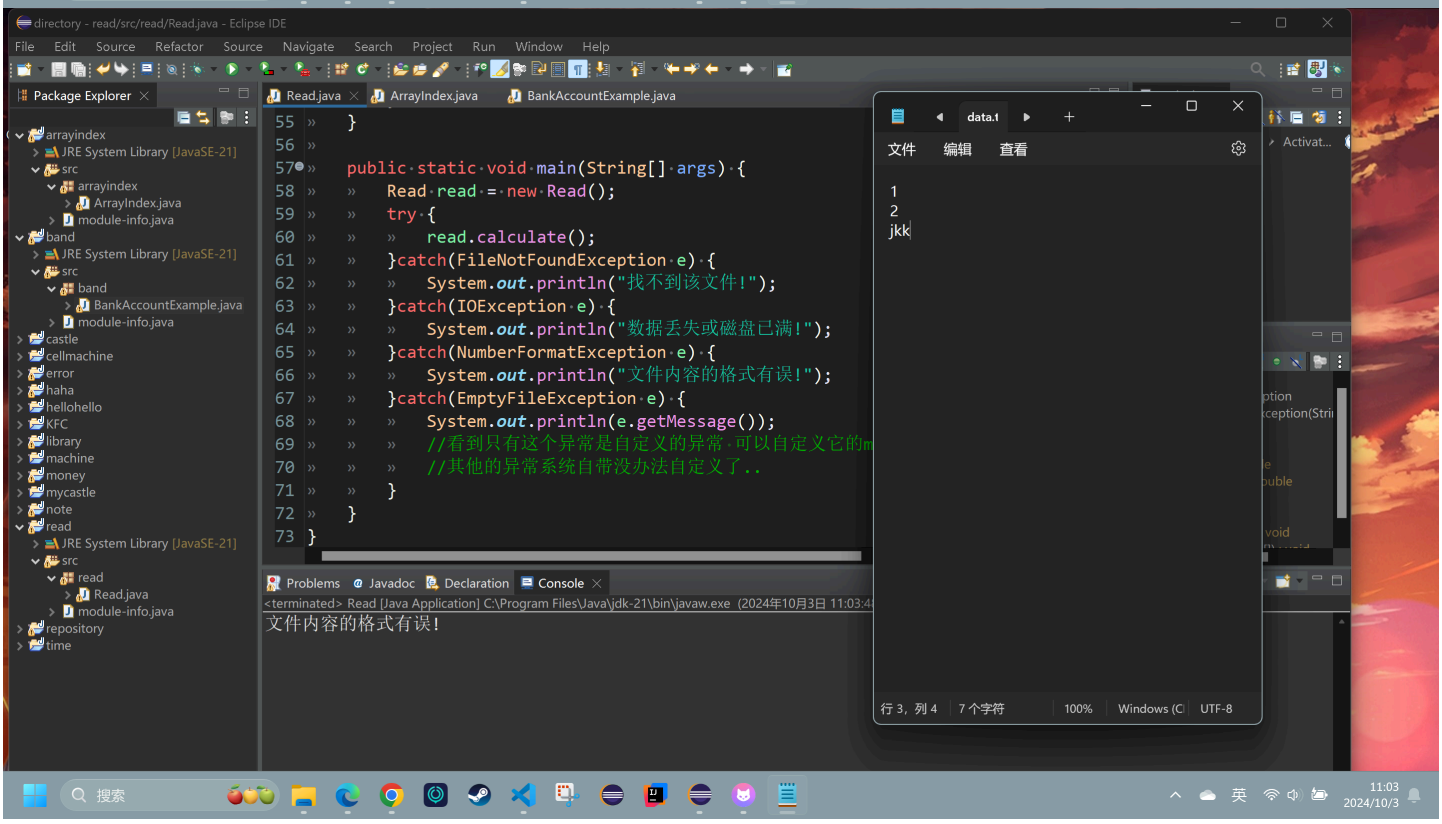
```java
package read;

import java.io.File;
import java.io.FileNotFoundException;
import java.io.FileReader;
import java.io.BufferedReader;
import java.io.IOException;

//class FileNotFoundException extends Exception {
//      public FileNotFoundException(String message) {
//              super(message);
//      }
//}
//这里注释掉了 系统库里有这个异常 因为是checked异常 必须要处理 那就不额外再自定义了

class EmptyFileException extends Exception {
        public EmptyFileException(String message) {
                super(message);
        }
}

//class NumberFormatException extends Exception {
//      public NumberFormatException(String message) {
//              super(message);
//      }
//}
//这个也是系统库自带的异常
// 因为我用了readLine读取到的是字符串 把他转化成double加到sum上可能会抛这个异常
//是unchecked异常 我也拿来用了 但下面有声明和catch

public class Read{

        private int cnt = 0;
        private double sum = 0;
        private double average = 0;
        private String line;

        private File file = new File("data.txt");

    //下面我选择用try-with-resources语句
        public void calculate() throws FileNotFoundException,NumberFormatException,IOException
                try(FileReader filereader = new FileReader(file);//可能会抛FileNotFoundExceptio
                        BufferedReader bufferedreader = new BufferedReader(filereader)){//可能
                        boolean isEmpty = true;
                        while((line = bufferedreader.readLine()) != null) {//可能会抛IOExceptio
                                sum += Double.parseDouble(line);//可能会抛NumberFormatException
```

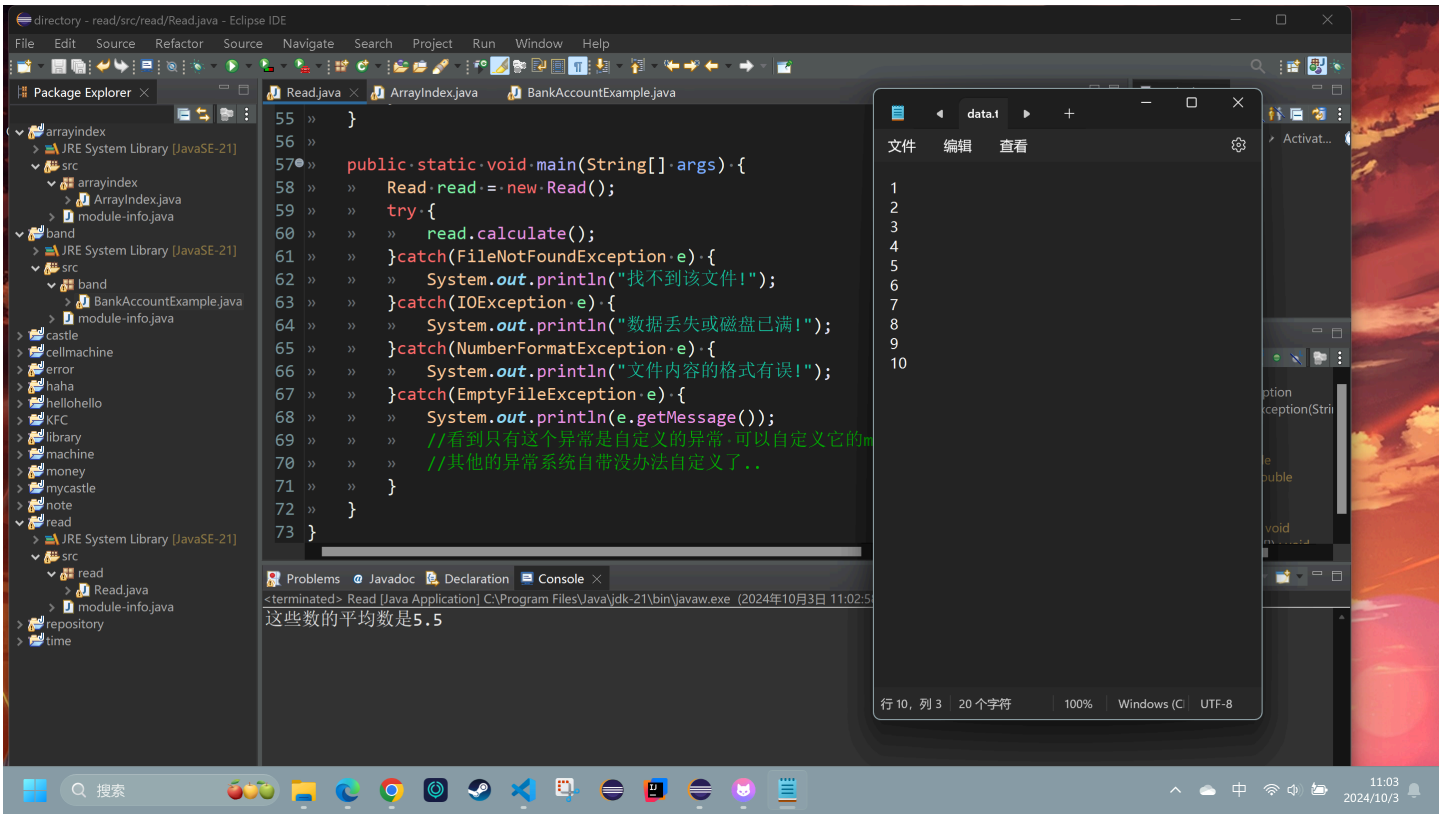
```
                                cnt++;
                                isEmpty = false;
                        }
                        if(isEmpty) {
                                throw new EmptyFileException("文件里空空如也!");//抛EmptyFileEx
                        }else {
                                average = sum / cnt;
                                System.out.println("这些数的平均数是"+average);
                        }
                }
        }

        public static void main(String[] args) {
                Read read = new Read();
                try {
                        read.calculate();
                }catch(FileNotFoundException e) {
                        System.out.println("找不到该文件!");
                }catch(IOException e) {
                        System.out.println("数据丢失或磁盘已满!");
                }catch(NumberFormatException e) {
                        System.out.println("文件内容的格式有误!");
                }catch(EmptyFileException e) {
                        System.out.println(e.getMessage());
                        //看到只有这个异常是自定义的异常 可以自定义它的message
                        //其他的异常系统自带没办法自定义了..
                }
        }
}
```

其实这个FileNotFoundException也是IOException中的一部分 当文件不存在时会自动抛IOException 但这里我为了让FileNotFoundException起到作用 把它的catch放到了前面 如果出现这个异常就会先被FileNotFoundException的catch捕捉到

下面是运行成功的截图:


directory - read/src/read/Read.java - Eclipse IDE
File Edit Source Refactor Source Navigate Search Project Run Window Help
Package Explorer
Read.java
ArrayIndex.java
BankAccountExample.java
55 }
56
57 public static void main(String[] args) {
58 Read read = new Read();
59 try {
60 read.calculate();
61 }catch(FileNotFoundException e) {
62 System.out.println("找不到该文件!");
63 }catch(IOException e) {
64 System.out.println("数据丢失或磁盘已满!");
65 }catch(NumberFormatException e) {
66 System.out.println("文件内容的格式有误!");
67 }catch(EmptyFileException e) {
68 System.out.println(e.getMessage());
69 //看到只有这个异常是自定义的异常,可以自定义它的m
70 //其他的异常系统自带没办法自定义了..
71 }
72 }
73 }
Problems Javadoc Declaration Console
<terminated> Read [Java Application] C:\Program Files\Java\jdk-21\bin\javaw.exe (2024年10月3日 11:03:1
文件里空空如也!
data.1
文件 编辑 查看
行 1，列 1 0 个字符 100% Windows (C UTF-8
搜索
11:03
2024/10/3


directory - read/src/read/Read.java - Eclipse IDE
File Edit Source Refactor Source Navigate Search Project Run Window Help
Package Explorer
Read.java
ArrayIndex.java
BankAccountExample.java
55 }
56
57 public static void main(String[] args) {
58 Read read = new Read();
59 try {
60 read.calculate();
61 }catch(FileNotFoundException e) {
62 System.out.println("找不到该文件!");
63 }catch(IOException e) {
64 System.out.println("数据丢失或磁盘已满!");
65 }catch(NumberFormatException e) {
66 System.out.println("文件内容的格式有误!");
67 }catch(EmptyFileException e) {
68 System.out.println(e.getMessage());
69 //看到只有这个异常是自定义的异常,可以自定义它的m
70 //其他的异常系统自带没办法自定义了..
71 }
72 }
73 }
Problems Javadoc Declaration Console
<terminated> Read [Java Application] C:\Program Files\Java\jdk-21\bin\javaw.exe (2024年10月3日 11:03:4
文件内容的格式有误!
data.1
文件 编辑 查看
1
2
jkk
行 3，列 4 7 个字符 100% Windows (C UTF-8
搜索
11:03
2024/10/3


directory - read/src/read/Read.java - Eclipse IDE
File Edit Source Refactor Source Navigate Search Project Run Window Help
Package Explorer
Read.java
ArrayIndex.java
BankAccountExample.java
55 }
56
57 public static void main(String[] args) {
58 Read read = new Read();
59 try {
60 read.calculate();
61 }catch(FileNotFoundException e) {
62 System.out.println("找不到该文件!");
63 }catch(IOException e) {
64 System.out.println("数据丢失或磁盘已满!");
65 }catch(NumberFormatException e) {
66 System.out.println("文件内容的格式有误!");
67 }catch(EmptyFileException e) {
68 System.out.println(e.getMessage());
69 //看到只有这个异常是自定义的异常,可以自定义它的m
70 //其他的异常系统自带没办法自定义了..
71 }
72 }
73 }
Problems Javadoc Declaration Console
<terminated> Read [Java Application] C:\Program Files\Java\jdk-21\bin\javaw.exe (2024年10月3日 11:02:5
这些数的平均数是5.5
data.1
文件 编辑 查看
1
2
3
4
5
6
7
8
9
10
行 10，列 3 20 个字符 100% Windows (C UTF-8
搜索
11:03
2024/10/3