

廈門大學



信息学院软件工程系

《JAVA 程序设计》实验报告

实验七

姓名：庾晓萍

学号：20420192201952

学院：信息学院

专业：软件工程

完成时间：2022/4/8

一、实验目的及要求

（一）实验目的

- 1、多态
- 2、熟悉异常处理

（二）实验要求

- 1、按照题目要求写代码和实验报告，并上传到 FTP

二、实验题目及实现过程

一、基本题目：

题目 1: Payroll System Modification

（一）实验环境

操作系统：Windows 10;

IDE：Eclipse Java 2018-12

编程语言：Java;

（二）实现过程

（1）设计思路

①修改工资系统，增加一个 `Employee` 子类 `PieceWorker`，这个员工的工资是根据生产商品的数量决定的。`PieceWorker` 类包含 `private` 实例变量 `wage`（工

资)和 pieces (存储生产的件数量)。提供 earnings 方法的具体实现来计算工资 (生产的数量乘以每件工资)。

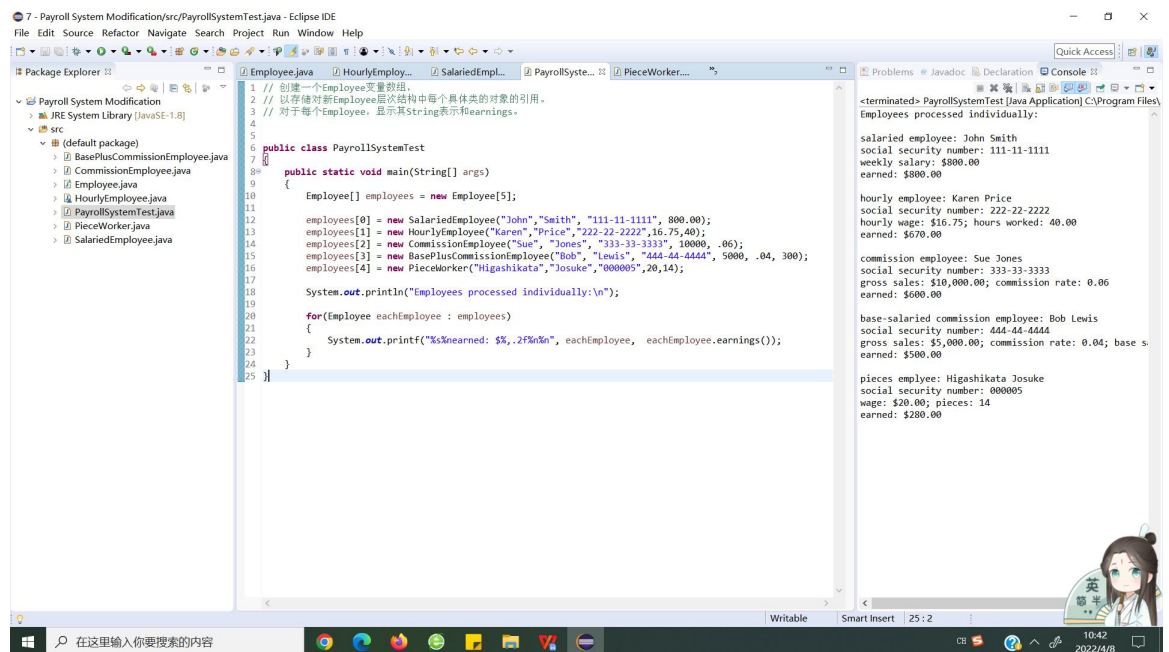
②在 PayrollSystemTest 类中创建一个 Employee 变量数组, 对 Employee 层次结构中每个具体类的对象引用。对于每个 Employee, 通过动态绑定, 显示不同的 String 表示和 earnings。

```
Employee[] employees = new Employee[5];

employees[0] = new SalariedEmployee("John", "Smith", "111-11-1111", 800.00);
employees[1] = new HourlyEmployee("Karen", "Price", "222-22-2222", 16.75, 40);
employees[2] = new CommissionEmployee("Sue", "Jones", "333-33-3333", 10000, .06);
employees[3] = new BasePlusCommissionEmployee("Bob", "Lewis", "444-44-4444", 5000, .04, 300);
employees[4] = new PieceWorker("Higashikata", "Josuke", "000005", 20, 14);
```

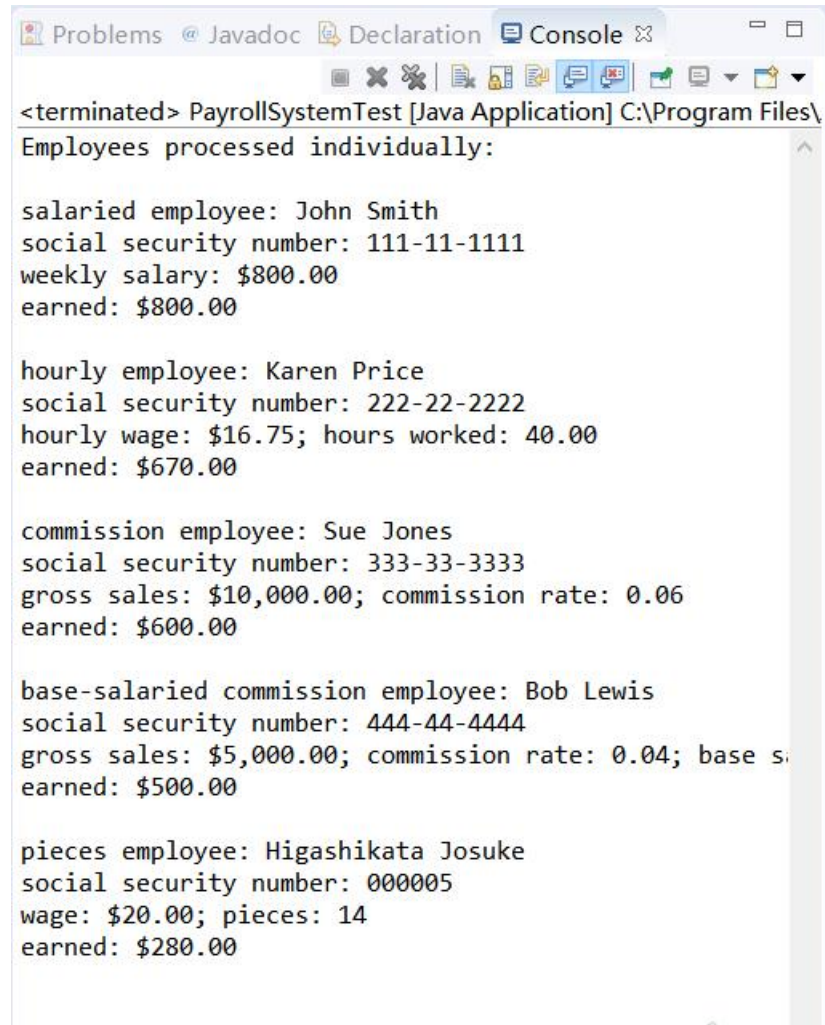
(三) 过程截图

(1) 全屏截图



(2) 运行结果

1. 创建了新类 pieces employee, 打印 Employee 变量数组, 得到结果。



```
<terminated> PayrollSystemTest [Java Application] C:\Program Files\
Employees processed individually:

salaried employee: John Smith
social security number: 111-11-1111
weekly salary: $800.00
earned: $800.00

hourly employee: Karen Price
social security number: 222-22-2222
hourly wage: $16.75; hours worked: 40.00
earned: $670.00

commission employee: Sue Jones
social security number: 333-33-3333
gross sales: $10,000.00; commission rate: 0.06
earned: $600.00

base-salaried commission employee: Bob Lewis
social security number: 444-44-4444
gross sales: $5,000.00; commission rate: 0.04; base s
earned: $500.00

pieces employee: Higashikata Josuke
social security number: 000005
wage: $20.00; pieces: 14
earned: $280.00
```

题目 2: Accounts Payable System Modification

(一) 实验环境

操作系统: Windows 10;

IDE: Eclipse Java 2018-12

编程语言: Java;

(二) 实现过程

(1) 设计思路

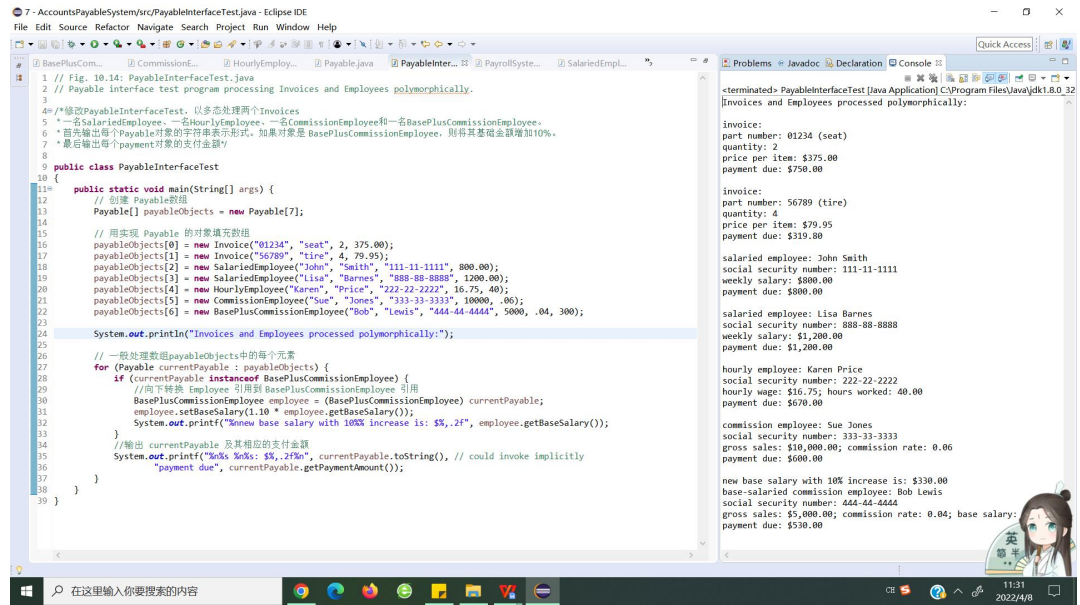
①修改 accounts payable application, 以包括工资单申请的完整功能。应用程序仍然应该处理两个 Invoice 对象, 但现在应该处理四个 Employee 子类中每个对象中的一个对象。如果当前正在处理的对象是 Base-PlusCommissionEmployee, 则应用程序应该将 Base-PlusCommissionEmployee 的 basesalary 增加 10%。最后, 应用程序应该输出每个对象的支付金额。

(2) 设计类

- ① 修改 HourlyEmployee 和 CommissionEmployee, 将它们作为实现 Payable 的 Employee 版本的子类, 放置在 Payable 层次结构中。将子类中的方法 earnings 的名称更改为 getPaymentAmount, 以便该类满足其继承的与接口 Payable 的约定。
- ② 修改类 BasePlusCommissionEmployee, 以便扩展①部分中创建的 CommissionEmployee 类的版本。
- ③ 修改 PayableInterfaceTest, 以多态处理两个 Invoices, 一名 SalariedEmployee、一名 HourlyEmployee、一名 CommissionEmployee 和一名 BasePlusCommissionEmployee。首先输出每个 Payable 对象的字符串表示形式。接下来, 如果一个对象是 BasePlusCommissionEmployee, 则将其基础金额增加 10%。最后, 输出每个 payment 对象的支付金额。

(三) 过程截图

(1) 全屏截图



```
1 // Fig. 10.14: PayableInterfaceTest.java
2 // Payable interface test program processing Invoices and Employees polymorphically.
3
4 /*修改PayableInterfaceTest, 以多态处理两个Invoices
5 * 一名SalariedEmployee, 一名HourlyEmployee, 一名CommissionEmployee和一名BasePlusCommissionEmployee.
6 * 首先输出每个Payable对象的字符串表示形式, 如果对象是BasePlusCommissionEmployee, 则将其基础金额增加10%.
7 * 最后输出每个payment对象的支付金额*/
8
9 public class PayableInterfaceTest
10 {
11     public static void main(String[] args) {
12         // 创建 Payable数组
13         Payable[] payableObjects = new Payable[7];
14
15         // 用实现 Payable 的对象填充数组
16         payableObjects[0] = new Invoice("01234", "seat", 2, 375.00);
17         payableObjects[1] = new Invoice("56789", "tire", 4, 79.95);
18         payableObjects[2] = new SalariedEmployee("John", "Smith", "111-11-1111", 800.00);
19         payableObjects[3] = new SalariedEmployee("Lisa", "Barnes", "888-88-8888", 1200.00);
20         payableObjects[4] = new HourlyEmployee("Karen", "Price", "222-22-2222", 16.75, 40);
21         payableObjects[5] = new CommissionEmployee("Sue", "Jones", "333-33-3333", 10000, .06);
22         payableObjects[6] = new BasePlusCommissionEmployee("Bob", "Lewis", "444-44-4444", 5000, .04, 300);
23
24         System.out.println("Invoices and Employees processed polymorphically.");
25
26         // 一般处理数组payableObjects中的每个元素
27         for (Payable currentPayable : payableObjects) {
28             if (currentPayable instanceof BasePlusCommissionEmployee) {
29                 // 向下转换 Employee 引用到 BasePlusCommissionEmployee 引用
30                 BasePlusCommissionEmployee employee = (BasePlusCommissionEmployee) currentPayable;
31                 employee.setBaseSalary(1.10 * employee.getBaseSalary());
32                 System.out.printf("New base salary with 10%% increase is: $%,.2f", employee.getBaseSalary());
33             }
34             // 输出 currentPayable 及其相应的支付金额
35             System.out.printf("\n%s: $%,.2f", currentPayable.toString(), // could invoke implicitly
36                               "payment due", currentPayable.getPaymentAmount());
37         }
38     }
39 }
```

terminated> PayableInterfaceTest [Java Application] C:\Program Files\Java\jdk1.8.0_32
Invoices and Employees processed polymorphically:

invoice:
part number: 01234 (seat)
quantity: 2
price per item: \$375.00
payment due: \$750.00

invoice:
part number: 56789 (tire)
quantity: 4
price per item: \$79.95
payment due: \$319.80

salaried employee: John Smith
social security number: 111-11-1111
weekly salary: \$800.00
payment due: \$800.00

salaried employee: Lisa Barnes
social security number: 888-88-8888
weekly salary: \$1,200.00
payment due: \$1,200.00

hourly employee: Karen Price
social security number: 222-22-2222
hourly wage: \$16.75; hours worked: 40.00
payment due: \$670.00

commission employee: Sue Jones
social security number: 333-33-3333
gross sales: \$10,000.00; commission rate: 0.06
payment due: \$600.00

new base salary with 10% increase is: \$330.00
base-salaried commission employee: Bob Lewis
social security number: 444-44-4444
gross sales: \$5,000.00; commission rate: 0.04; base salary:
payment due: \$530.00

(2) 运行结果

- ① 运行 PayableInterfaceTest, Invoices 和 Employees 多态处理。首先一般处理数组 payableObjects 中的每个元素, 这个时候可以隐式调用。输出 currentPayable 及其相应的支付金额。当处理的元素类型是 BasePlusCommissionEmployee 时, 向下转换 Employee 引用到 BasePlusCommissionEmployee 引用, 将其基础金额增加 10%。

Invoices and Employees processed polymorphically:

invoice:
part number: 01234 (seat)
quantity: 2
price per item: \$375.00
payment due: \$750.00

invoice:
part number: 56789 (tire)
quantity: 4
price per item: \$79.95
payment due: \$319.80

salaried employee: John Smith
social security number: 111-11-1111
weekly salary: \$800.00
payment due: \$800.00

salaried employee: Lisa Barnes
social security number: 888-88-8888
weekly salary: \$1,200.00
payment due: \$1,200.00

hourly employee: Karen Price
social security number: 222-22-2222
hourly wage: \$16.75; hours worked: 40.00
payment due: \$670.00

commission employee: Sue Jones
social security number: 333-33-3333
gross sales: \$10,000.00; commission rate: 0.06
payment due: \$600.00

new base salary with 10% increase is: \$330.00
base-salaried commission employee: Bob Lewis
social security number: 444-44-4444
gross sales: \$5,000.00; commission rate: 0.04; base salary: \$330.00
payment due: \$530.00

题目 3: CarbonFootprint Interface: Polymorphism

(一) 实验环境

操作系统: Windows 10;

IDE: Eclipse Java 2018-12

编程语言: Java;

(二) 实现过程

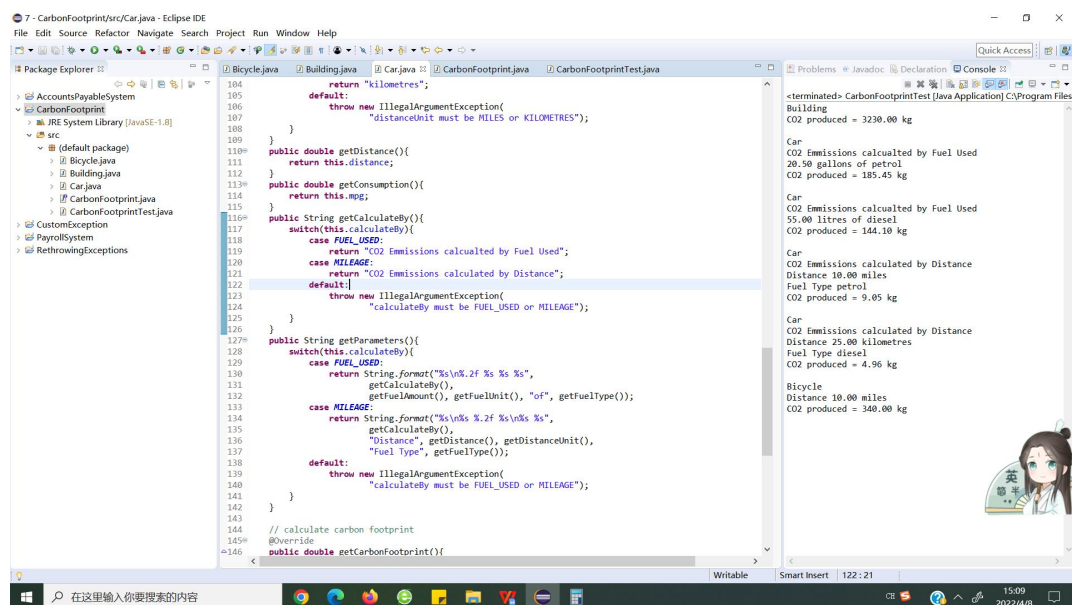
(1) 设计思路

① 使用接口，可以为可能不同的类指定类似的行为。创建三个与继承类不相关的小类，**Building**, **Car** 和 **Bicycle**。给每个类一些与其他类没有共同点的独特的适当属性和行为。编写一个拥有 `getCarbonFootprint` 方法的接口 `CarbonFootprint`。让每个类都实现该接口，`getCarbonFootprint` 方法为该类计算适当的碳足迹。

② 编写应用程序，创建三个类的每个对象，将这些对象的应用放在 `ArrayList<CarbonFootprint>>` 中，然后遍历 `ArrayList<CarbonFootprint>`，多态调用每个对象的 `getCarbonFootprint` 方法。对于每个对象，打印识别信息和对象的碳足迹。

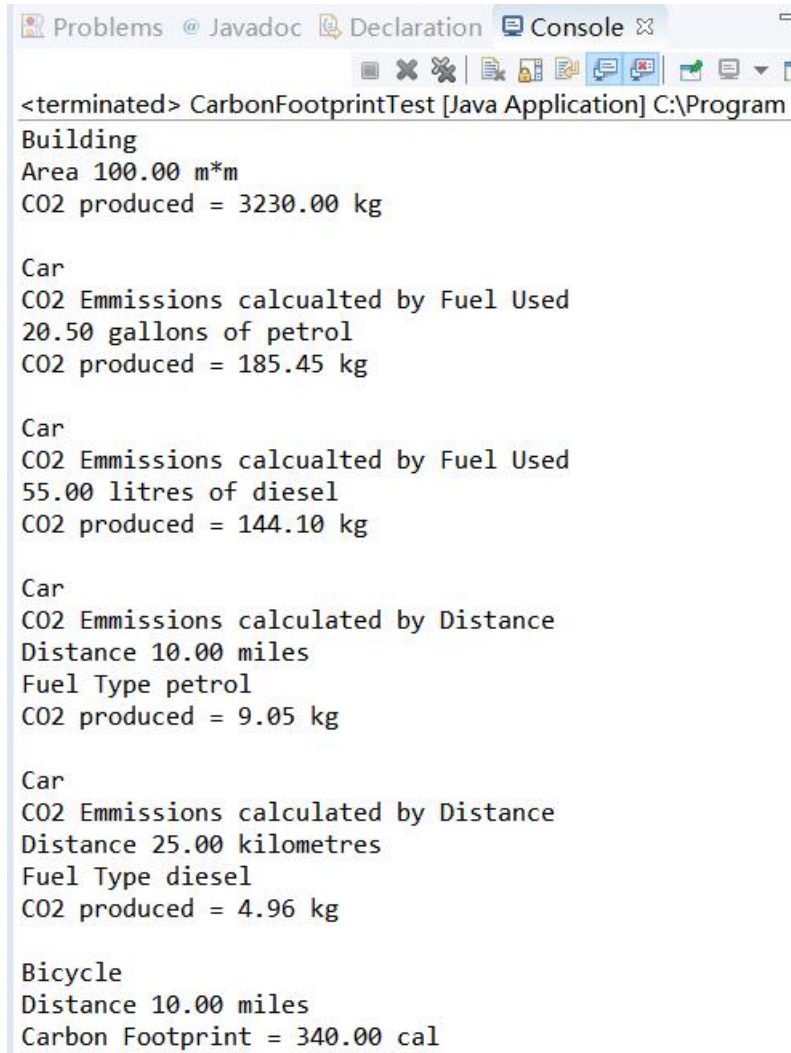
(三) 过程截图

(1) 全屏截图



(2) 运行结果

- ① 查找资料，对于 Building 类，中国现有城镇总建筑存量约 650 亿平方米，这些建筑在使用过程中排放了约 21 亿吨二氧化碳，平均每平方米排放约 32.3kg 的二氧化碳。通过构造函数参数对建筑的面积和高度进行初始化，并计算总排放。
- ② 对于 Car 类，每升汽油的二氧化碳排放是 2.39kg，每升柴油的二氧化碳排放量是 2.62kg，通过构造函数参数确定汽车使用燃油的单位（是加仑还是升），燃油的类别（是柴油还是汽油），燃油量。距离的单位，距离，燃油类别，每加仑可以行驶多少英里。
- ③ 对于自行车，通过构造函数参数初始化骑行距离，利用卡路里计算公式算出碳排放。



```
<terminated> CarbonFootprintTest [Java Application] C:\Program
Building
Area 100.00 m*m
CO2 produced = 3230.00 kg

Car
CO2 Emmissions calcualted by Fuel Used
20.50 gallons of petrol
CO2 produced = 185.45 kg

Car
CO2 Emmissions calcualted by Fuel Used
55.00 litres of diesel
CO2 produced = 144.10 kg

Car
CO2 Emmissions calculated by Distance
Distance 10.00 miles
Fuel Type petrol
CO2 produced = 9.05 kg

Car
CO2 Emmissions calculated by Distance
Distance 25.00 kilometres
Fuel Type diesel
CO2 produced = 4.96 kg

Bicycle
Distance 10.00 miles
Carbon Footprint = 340.00 cal
```

题目 4：自定义异常的定义、抛出和捕获

（一）实验环境

操作系统：Windows 10;

IDE：Eclipse Java 2018-12

编程语言：Java;

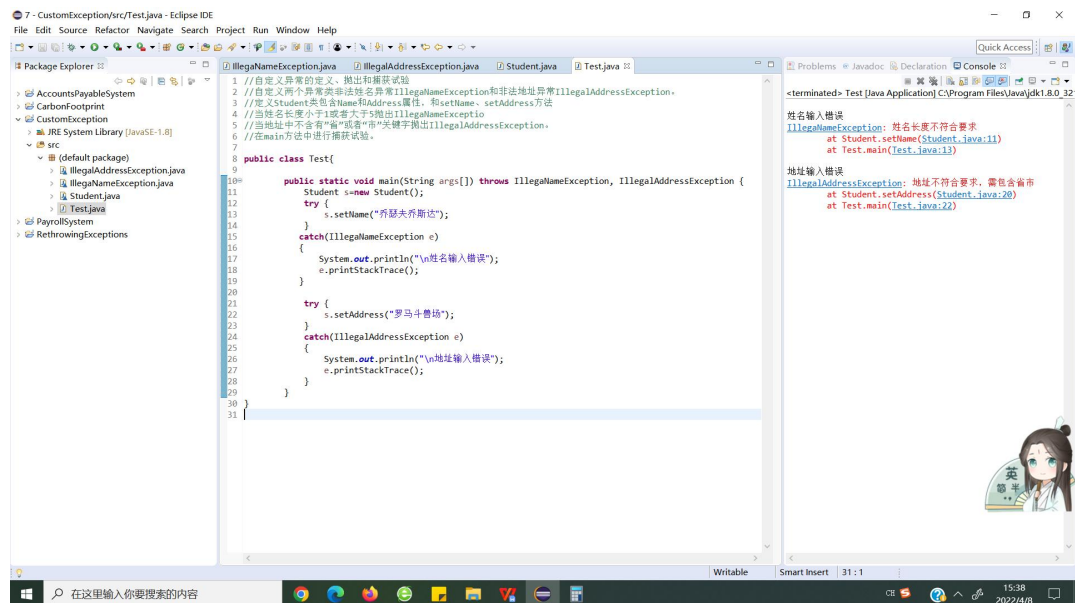
（二）实现过程

（1）设计思路

- ① 自定义两个异常类：非法姓名异常 `IllegalNameException` 和非法地址异常 `IllegalAddressException`。
- ② 定义 `Student` 类包含 `name` 和 `address` 属性，和 `setName`、`setAddress` 方法，当姓名长度小于 1 或者大于 5 时抛出 `IllegalNameException`，当地址中不含有“省”或者“市”关键字时抛出 `IllegalAddressException`。
- ③ 编写程序抛出这两种异常，在 `main` 方法中进行捕获并合理地处理。

（三）过程截图

（1）全屏截图



（2）运行结果

```
public static void main(String args[]) throws IllegalArgumentException, IllegalAddressException {
    Student s=new Student();
    try {
        s.setName("乔瑟夫乔斯达");
    }
    catch(IllegalArgumentException e)
    {
        System.out.println("\n姓名输入错误");
        e.printStackTrace();
    }

    try {
        s.setAddress("罗马斗兽场");
    }
    catch(IllegalAddressException e)
    {
        System.out.println("\n地址输入错误");
        e.printStackTrace();
    }
}
```



题目 5: Rethrowing Exceptions

(一) 实验环境

操作系统: Windows 10;

IDE: Eclipse Java 2018-12

编程语言: Java;

（二）实现过程

（1）设计思路

① 编写一个说明重新抛出异常的程序。定义方法 `someMethod` 和 `someMethod2`。方法 `someMethod2` 最初引发异常。方法 `someMethod` 调用 `someMethod2`，捕获异常并重新抛出它。从方法 `main` 调用 `someMethod`，并捕获重新抛出的异常。打印此异常的堆栈跟踪。

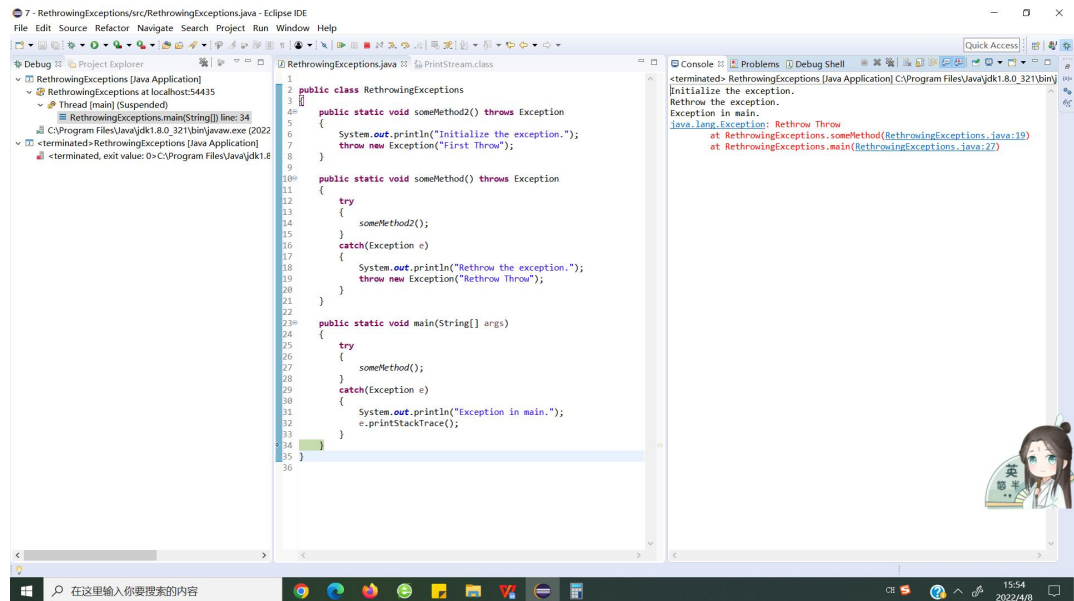
```
public class RethrowingExceptions
{
    public static void someMethod2() throws Exception
    {
        System.out.println("Initialize the exception.");
        throw new Exception("First Throw");
    }

    public static void someMethod() throws Exception
    {
        try
        {
            someMethod2();
        }
        catch(Exception e)
        {
            System.out.println("Rethrow the exception.");
            throw new Exception("Rethrow Throw");
        }
    }

    public static void main(String[] args)
    {
        try
        {
            someMethod();
        }
        catch(Exception e)
        {
            System.out.println("Exception in main.");
            e.printStackTrace();
        }
    }
}
```

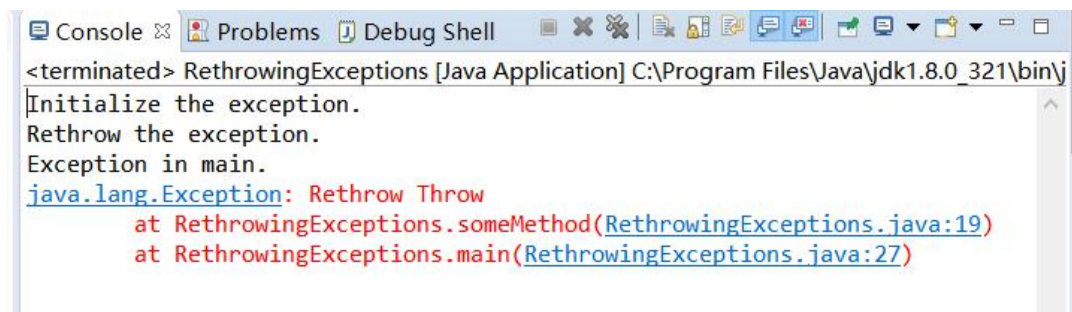
（三）过程截图

(1) 全屏截图



(2) 运行结果

方法 `someMethod2` 最初引发异常。方法 `someMethod` 捕获异常并重新抛出它。从方法 `main` 捕获重新抛出的异常。打印此异常的堆栈跟踪。



题目 6：完善上周的模拟考试题目，补充必要的异常处理。

(一) 实验环境

操作系统：Windows 10;

IDE：Eclipse Java 2018-12

编程语言：Java;

（二）实现过程

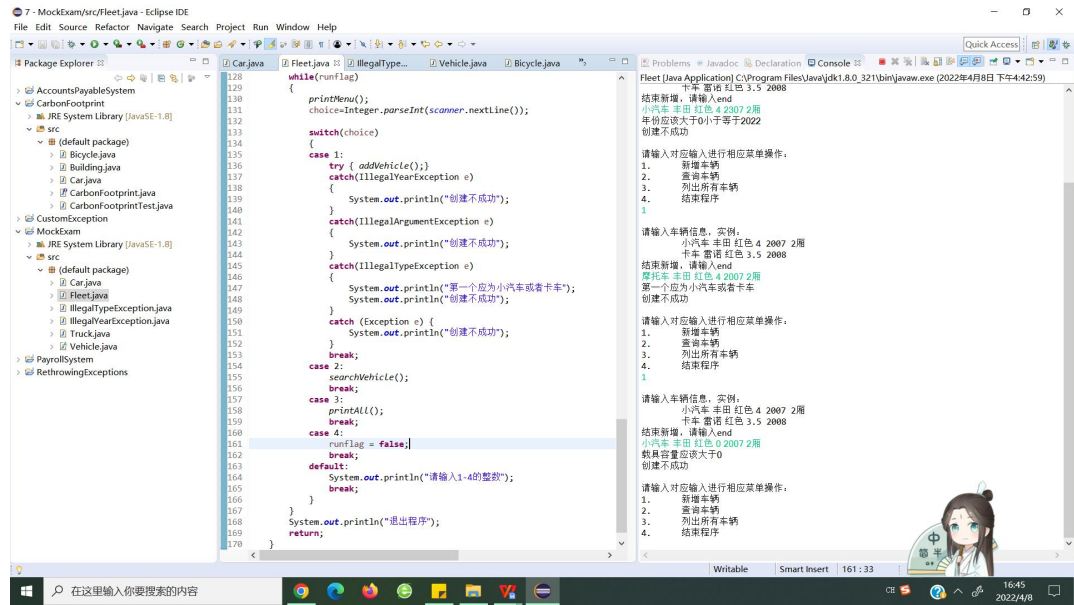
（1）设计思路

① 新增了 Exception 两个子类 `IllegalTypeException`、`IllegalYearException`。前者用来抛出输入车辆类型不正确的错误，后者用来抛出输入年份不正确的错误。同时用 `IllegalArgumentException` 避免输入的载具容量小于或者等于 0。

```
try { addVehicle();}
catch(IllegalYearException e)
{
    System.out.println("创建不成功");
}
catch(IllegalArgumentException e)
{
    System.out.println("创建不成功");
}
catch(IllegalTypeException e)
{
    System.out.println("第一个应为小汽车或者卡车");
    System.out.println("创建不成功");
}
catch (Exception e) {
    System.out.println("创建不成功");
}
```

（三）过程截图

（1）全屏截图



(2) 运行结果

输入小汽车的出厂年份为 2307 年，捕获输入年份异常并输出，输入载具类型为摩托车，捕获输入类型异常并输出，输入小汽车载客量为 0，捕获输入异常并输出。

请输入车辆信息，实例：
小汽车 丰田 红色 4 2007 2厢
卡车 雷诺 红色 3.5 2008
结束新增，请输入end
小汽车 丰田 红色 4 2307 2厢
年份应该大于0小于等于2022
创建不成功

请输入车辆信息，实例：
小汽车 丰田 红色 4 2007 2厢
卡车 雷诺 红色 3.5 2008
结束新增，请输入end
摩托车 丰田 红色 4 2007 2厢
第一个应为小汽车或者卡车
创建不成功


```
请输入车辆信息，实例：
    小汽车 丰田 红色 4 2007 2厢
    卡车 雷诺 红色 3.5 2008
结束新增，请输入end
小汽车 丰田 红色 0 2007 2厢
载具容量应该大于0
创建不成功
```

三、实验总结与心得记录

在本次实验过程中，我练习了多态，熟悉了 java 的语法，熟悉了 java 类的定义，实例化和调用。体会到了 JAVA 语言的优点。