软工代码

决策表优化-个人所得税

```
/*
    个人所得说系统
    用决策表来优化
 */

/*
    题目:
        小于 10000 tax is 10%
        大于 10000 小于 20000 的部分 12%
        大于 20000 小于 30000 的部分 15%
        大于 30000 小于 40000 的部分 18%
        大于 40000 的部分 20%
 */
class Main{
    public int calculateTax(int taxable_income){
        int tax = 0;
        if(taxable_income == 0){
            //goto Exit;
            return 0;
        }
        if(taxable_income > 10000){
            tax = tax + 1000;//10000以下的那部分税
        }else{
            tax = tax + taxable_income * 0.1;
            //goto Exit;
            return tax;
        }
        if(taxable_income > 20000){
            tax = tax + 1200;
        }else {//小于20000
            tax += (taxable_income-10000)*0.12;
```

```
            return tax;
        }
        if(taxable_income > 30000){
            tax = tax + 1500;
        }else {//小于 30000
            tax += (taxable_income-20000)*0.15;
            return tax;
        }if(taxable_income > 40000){
            tax = tax + 1800;
            tax += (taxable_income-40000)*0.2;
        }else {
            tax += (taxable_income-30000)*0.18;
            return tax;
        }
        return tax;
    }
}


/*
    修改
 */

int[] percent = {10%,12%,15%,18%,20%};
int[] bracket = {0,10000,20000,30000,40000};
int[] base = {0,1000,1200,2200,3700,5500}

public int calculateTax(int taxable_income){
    int level = 1;
    for(int i=0;i<5;i++){
        if(taxable_income > bracket[i]){
            level++;
        }
    }
    int tax = base[level] +(taxable_income - bracket[level])*percent[level];
    return tax;
}

/*
    表驱动编程  P307
 */

int[] prePoint = {1000,2000,5000};
int[] postPoint = {1000,2000,5000};
int[] levelArray = {1,2,3};
public int calculateLavel(int prePoint ,int postPoint){
    for(int i=0;i<3;i++){
        if(prePoint < prePoint[i] && postPoint >= postPoint[i]){
            return levelArray[i];
```

```
        }
    }
}
```

面向对象优化--android&ios

```
/*
 * 面向对象设计原则
 */

public class Application{
    private String applicationName;
    private float averageRate;
    private ArrayList<NewFeature> newFeatureItems = new ArrayList<NewFeature>();

    public static String getDescriptionForIOS(){
        StringBuffer result = new StringBuffer();
        result.append("This is "+ this.applicationName + " for IOS platform\n");
        for(int i= 0;i<newFeatureItems.size();i++){
            result.append(newFeatureItems.get(i).getDescription);
        }
        result.append("Average Rate from APP Store\n");
        result.append(String.valueOf(avarageRate));
        return result.toString();
    }

    public static String getDescriptionForAndriod(){
        StringBuffer result = new StringBuffer();
        result.append("This is "+ this.applicationName + " for Andriod platform\n");
        for(int i= 0;i<newFeatureItems.size();i++){
            result.append(newFeatureItems.get(i).getDescription);
        }
        result.append("Average Rate from Google Play\n");
        result.append(String.valueOf(avarageRate));
        return result.toString();
    }
}

/*
 * 违反了OCP和 Do not repeat
 * 修改
 */

public class Application{
    private String applicationName;
    private float averageRate;
    private ArrayList<NewFeature> newFeatureItems = new ArrayList<NewFeature>();
```

```
    private Phone phone;
    Phone = new AndriodPhone();
    //Phone = new Iphone();

    public static String getDescriptionForAndriod(){
        StringBuffer result = new StringBuffer();
        result.append("This is "+ this.applicationName + " for" +
phone.getPlatformName()+ " platform\n");


        for(int i= 0;i<newFeatureItems.size();i++){
            result.append(newFeatureItems.get(i).getDescription);
        }
        // for 循环初选隐式方法，应该使用迭代器
        Iterator<NewFeature> it = newFeature.iterator();
        while(it.hasNext()){
            result.append(it.getDescription);
        }


        result.append("Average Rate from "+ phone.getSourceName()+"\n");
        result.append(String.valueOf(avarageRate));
        return result.toString();
    }
}
public interface Phone{
    public String getSourceName();
    public String getPlatformName();
}
public class AndriodPhone implements Phone{
    private String sourceName = "Google Play";
    private String platformName = "Andriod";
    public String getSourceName(){
        return this.sourceName;
    }
    public String getPlatformName(){
        return this.platformName;
    }
}
public class Iphone implements Phone{
    private String sourceName = "APP Store";
    private String platformName = "IOS";
    public String getSourceName(){
        return this.sourceName;
    }
    public String getPlatformName(){
        return this.platformName;
    }
}
```

```
/*
 * 使用策略模式
 */

public class TestCase{
    public void test(){
        Application andriodApp = new Application();
        andriodApp.setApplicationName("淘宝");
        andriodApp.setAverageRate(0.7);
        andriodApp.setNewFeatureItems(new ArrayList<NewFeature>());

        Device andriod = new Andriod();
        andriodApp.setDevice(andriod);
        andriodApp.getDescription();

    }
}

public class Application{
    private String appicationName;
    private float averageRate;
    private ArrayList<NewFeature> newFeatureItems = new ArrayList<NewFeature>();

    private Device device;

    public void setDevice(Device device){
        this.device = device;
    }

    public String getSourceName(){
        device.getSourceName();
    }
    public String getPlatformName(){
        device.getPlatformName();
    }
    public String getDescription(){
        StringBuffer result = new StringBuffer();
        result.append("This is "+ this.applicationName + " for"+
device.getPlatformName() +" platform\n");
        for(int i= 0;i<newFeatureItems.size();i++){
            result.append(newFeatureItems.get(i).getDescription);
        }
        result.append("Average Rate from " + device.getSourceName()+ "\n");
        result.append(String.valueOf(avarageRate));
        return result.toString();
    }
}
public interface Device{
```

```
    public String getSourceName();
    public String getPlatformName();
}
public class Andriod implements Device{
    public String getSourceName(){
        return "Google Play";
    }
    public String getPlatformName(){
        return "Andriod";
    }
}
public class IOS implements Device{
    public String getSourceName(){
        return "App Store";
    }
    public String getPlatformName(){
        return "IOS";
    }

}
/*
 使用工厂模式
 */

interface App{
    public String getDescription();
}
public class IOS implements App{
    public String getDescription(){

    }
}
public class Andriod implements App{
    public String getDescription(){

    }
}
interface Factory{
    public App createApp();
}
public class iosApp implements Factory{
    public App createApp{
        return new IOS();
    }
}
public class andriodApp implements Factory{
    public App createApp(){
        return new Andriod();
    }
```

```
}
```

LSP-长方形-正方形

```
/*
    长方形和正方形
 */
class Rectangle{
    int length;
    int width;

    public int area(){
        return length*width;
    }

    public int getLength() {
        return length;
    }

    public void setLength(int length) {
        this.length = length;
    }

    public int getWidth() {
        return width;
    }

    public void setWidth(int width) {
        this.width = width;
    }
}

/*
    组合代替继承
 */

class Square{
    Rectangle rectangle;
    int edge;
    public Square(int edge){
        rectangle = new Rectangle(edge,edge);
    }
    public void setEdge(int edge) {
        this.rectangle.setWidth(edge);
        this.rectangle.setWidth(edge);
    }
    public int area(){
        return rectangle.area();
```

```
    }
}
```

迪米特法则-- 电影光盘租赁

```
public class Customer{
    Rental rental;
    int getNewRentPoint(){
        Movie m = rental.getMovieRented();
        if((m.getPriceCode == Movie.NEW_RELEASE)&& rental.getDaysRented()>1){
            return 2;
        }else {
            return 1;
        }
    }
}
public  class Rental{
    private int daysRented;
    private Movie movieRented;


    private int getDaysRented(){
        return daysRented;
    }
    public Movie getMovieRented{
        return movieRented;
    }
}
public class Movie {
    private int priceCode;
    public static final int CHILDRENS = 2;
    public static final int REGUALR = 20;
    public static final int NEW_RELEASE = 1;

    public int getPriceCode{
        return priceCode;
    }
}


/*
 违反了迪米特法则
 */


public class Customer{
```

```
    Rental rental;
    int getNewRentPoint(){
        rental.getMovieRentPoint();
    }
}
public  class Rental{
    private int daysRented;
    private Movie movieRented;

    public int getDaysRented(){
        return daysRented;
    }

    public int getMovieRentPoint{
        if(movieRented.getPriceCode == Movie.NEW_RELEASE && this.getDaysRented()){
            return 2;
        }else{
            return 1;
        }
    }
}
public class Movie {
    private int priceCode;
    public static final int CHILDRENS = 2;
    public static final int REGUALR = 20;
    public static final int NEW_RELEASE = 1;

    public int getPriceCode{
        return priceCode;
    }
}
```

前50名成绩

```
public class Grade{
    public float averageGradefroTop50(ArrayList<Student>allStudent){
        ArrayList<Student> sortedStudent = allStudent;
        int totalGrade = 0;
        for(int i=0;i<50;i++){
            totalGrade += sortedStudent.get(i).getGrade();
        }
        double averageGrade = totalGrade/50.0;
        return allStudent;
    }
}

/*
    方法一：  利用组合
```

```
 */
public class Grade{
    ArrayList<Student>sortedStudet;

    public ArrayList<Student> getSortedStudet() {
        return sortedStudet;
    }

    public void setSortedStudet(ArrayList<Student> sortedStudet) {
        this.sortedStudet = sortedStudet;
    }

    public float averageGradefroTop50(){
        int totalGrade = 0;
        for(int i=0;i<50;i++){
            totalGrade += this.sortedStudent.get(i).getGrade();
        }
        double averageGrade = totalGrade/50.0;
        return averageGrade;
    }
}
/*
    另一种方法：迭代器
 */
public class Grade{
    public float averageGradefroTop50(ArrayList<Student>allStudent){
        ArrayList<Student> sortedStudent = allStudent;
        Iterator<Student> iterator = sortedStudent.iterator();
        int totalGrade = 0;
        for(int i=0;i<50;i++){
            Student student = iterator.hasNext();
            totalGrade += student.getGrade();
        }
        double averageGrade = totalGrade/50.0;
        return allStudent;
    }
}
```

决策表优化 --- 计算每月天数

```
class main{
    int[] days = {31,28,31,30,31,30,31,31,30,31,30,31}
    int getDaysofMonth(int month){
        return days[i-1];
    }
}
```

迪米特法则 -- 计算购物金额

```
class Sales{
    public int getSubtotal(int commodityId){
        SalesLineItem salesLineItem = SalesLineItemMap.get(commodityId);
        return salesLineItem.getSubtoal();
    }
}
class SalesLineItem{
    Commodity commodity;
    int nums;
    public getSubtotal(){
        return commodity.getPrice * nums;
    }
}
class Commodity{
    int price;

    public int getPrice() {
        return price;
    }

    public void setPrice(int price) {
        this.price = price;
    }
}
```