院系：信息学院
专业：计算机科学与技术
学号：37220222203675
姓名：李哲彦
时间：2024年4月19日

题目内容

1、 Date, Random, Point2D综合应用

（1） 解题思路

先用 $Date$ 类自带的方法直接求出日期，然后用 $String$ 类的格式函数直接生成标准格式日期。最后按照题意模拟，按照 $Keans$ 算法正常做法即可。

(2) 核心代码解析

生成 $date$ 类，然后获取格式化的日期。

```
Date d = new Date();

String formatted_time = String.format("%04d-%02d-%02d-%02d:%02d", 1900 +
d.getYear(), d.getMonth() + 1, d.getDate(), d.getHours(), d.getMinutes());
System.out.println(formatted_time);
```

按照要求生成随机数。

```
int a = random.nextInt(), b = random.nextInt();
System.out.printf("%d %d\n", a, b);
int[] x = new int[11];
int[] y = new int[11];

for(int i = 1; i <= 10; i++){
    x[i] = random.nextInt(10);
    y[i] = random.nextInt(10); // 生成小数字方便检查
}
```

$Kmeans$ 核心代码,在随机预处理中心点后，不断更新当前点的最近中心点。为了方便聚类数量的变化，使用 $list$ 类来存储每一类的具体点。

```
while(true){
//            cnt++;
            L = new List[k+1];
            for(int i = 1; i <= k; i++)
                L[i] = new ArrayList<Point2D>();
            for(int i = 1; i <= 10; i++){
                double Minn = 1e18;
                int kind_index = 0;
                for(int j = 1; j <= k; j++){
                    double now_Dis = points[i].distance(centres[j]);
                    if(now_Dis < Minn){
                        Minn = now_Dis;
                        kind_index = j;
                    }
                }

                L[kind_index].add(points[i]);

            }
```

接着更新中心点，直到中心点不再变化为止。

```
// 更新中心点
Refresh_centre(centres, L, k);
boolean flag = false;
for(int i = 1; i <= k; i++){
    if(centres[i] != Lstcentres[i]){
        flag = true;
        break;
    }
}
```

下面详细介绍如何更新中心点。对于当前聚类，先求出其 x 和 y 的平均值，然后在各自聚类中寻找距离其最近的点，让它成为新的中心点，然后更新即可。

```
private static void Refresh_centre(Point2D[] centres, List<Point2D>[] L, int
k){
    for(int i = 1; i <= k; i++){
        double tmpx = 0, tmpy = 0;
        for(Point2D now : L[i]){
            tmpx += now.getX();
            tmpy += now.getY();
        }
        tmpx /= (double) L[i].size();
```

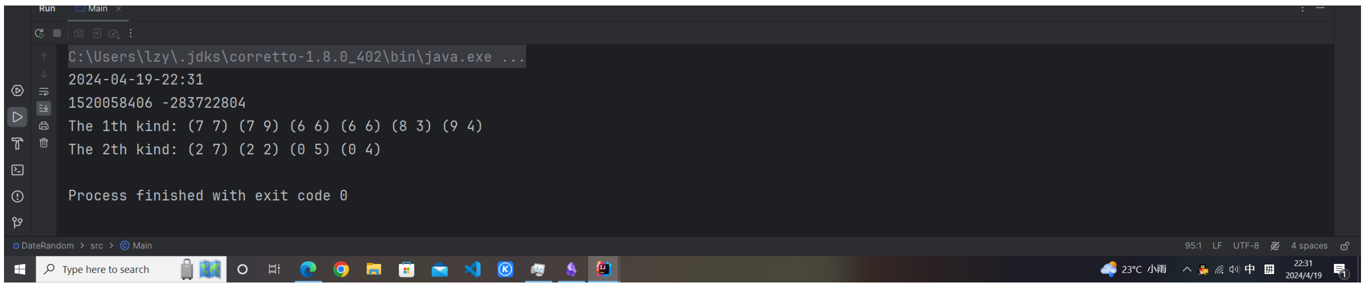
```java
            tmpy /= (double) L[i].size();

            Point2D tmp_centre = new Point2D.Double(tmpx, tmpy);
            Point2D new_centre = new Point2D.Double(0, 0);
            double min_dis = (double)1e18;
            for(Point2D now : L[i]){
                if(now.distance(tmp_centre) < min_dis){
                    min_dis = now.distance(tmp_centre);
                    new_centre = now;
                }
            }
            centres[i] = new_centre;
        }
        return ;
    }
```

（3）调试过程

本题目大部分过程只要根据题目要求以及 $Kmeans$ 算法实现即可。但是对于 $date$ 类获取日期的函数需要注意。其年份返回的是距离 1900 年的差值，输出时必须要加上 1900 才是正确的年份。月份也要注意，是从 0 下标开始的，输出时需要 $+1$ 才是正确的月份。

（4）本题输出结果



```
C:\Users\lzy\.jdks\corretto-1.8.0_402\bin\java.exe ...
2024-04-19-22:31
1520058406 -283722804
The 1th kind: (7 7) (7 9) (6 6) (6 6) (8 3) (9 4)
The 2th kind: (2 7) (2 2) (0 5) (0 4)

Process finished with exit code 0
```

2、汽车租赁程序

（1）解题思路

先建立 Car 类，按照要求设立所有函数。然后设立 $Rental_company$，按照题意模拟即可。

(2) 核心代码解析

在 Car 类之中，题目要求有两个构造函数。其中一个带参，同时也要自己定义不带参的构造函数。

```java
Car(String carRenter, int carNum, String carName, String carType, double rate,
int days){
    this.carRenter = carRenter;
    this.carNum = carNum;
    this.carName = carName;
    this.carType = carType;
    this.rate = rate;
    this.days = days;
}

Car(){
    this.carRenter = "";
    this.carNum = 0;
    this.carName = "";
    this.carType = "";
    this.rate = 0;
    this.days = 0;
}
```

对于 *toString* 函数，直接使用字符串拼接方法，方便返回汽车的所有信息。

```
public String toString() {
    System.out.println("The following is the car's information: ");
    String str = "";
    str = "The car renter: " + this.carRenter + "\n" + "The car number: " +
this.carNum + "\n" + "The car type: " + this.carType + "\n" + "The rate: " +
this.rate + "\n" + "The rented days: " + this.days ;
    System.out.println(str);
    return str;
}
```

对于 *Rental_Company* 类，其构造函数定义与上类似。

```
RentalCarCompany(String Company_name){
    this.Company_name = Company_name;
}

RentalCarCompany(){
    this.Company_name = "";
    this.theCars = new Car[101];
    this.totDays = 0;
    this.totRate = 0;
    this.rentCnt = 0;
}
```

添加一个新的汽车订单时，一方面更新 $theCars$ 数组，一方面也要更新总租赁天数和总租赁价格。

```java
public void addReservation(String renter, int carNum, String carName, String
carType, double rate, int days){
    this.theCars[++this.rentCnt] = new Car(renter, carNum, carName, carType,
rate, days);
    this.totDays += days;
    this.totRate += rate;
    return ;
}
```

对于查询某个汽车信息的操作，直接使用 for 循环遍历所有汽车，如果相等，则直接输出即可。

```java
void findReservation(int x){
    boolean flag = false;
    for(int i = 1; i <= rentCnt; i++){
        if(theCars[i].getCarNum() == x){
            theCars[i].toString();
            flag = true;
            break;
        }
    }
    if(!flag){
        System.out.printf("Could not find reservation for this car number
%d\n", x);
    }
    return ;
}
```

对于测试类，直接按照要求输入即可。需要注意的是输入 int 类型时，其会在行末缓冲区留下一个换行符，会影响之后读入字符串。所以选择额外调用一次 $nextLine$ 方法或者用字符串转换为 int 类型即可。

```java
public class RentalCarTester{
    RentalCarCompany company = new RentalCarCompany();

    public void test(){
        Scanner input = new Scanner(System.in);
        System.out.print("Rental Car Company: ");
        String compnay_name = input.nextLine(); // 先输入一次租车公司名称
        company.setCompany_name(compnay_name);
        String opt = "-1";
        while(!opt.equals("0")){
            System.out.println("1 to add Car, 2 to find car number, 3 to ask for average data, 0 to stop! ");
            opt = input.nextLine();  // 输入每次的操作选择
            if(opt.equals("0")) break;
            if(opt.equals("1")){  // 进行对应操作即，见完整代码
               /****/

            }
            if(opt.equals("2")){
                /*****/
            }
            if(opt.equals("3")){
                /*****/
            }

        }
        return ;
    }
}
```

对于主函数，直接调用测试类的 $test$ 函数。

```
public class Main {

    static RentalCarTester Tester = new RentalCarTester();

    public static void main(String[] args) {
        Tester.test();
        return ;
    }
}
```

(3) 调试过程

本题的大部分实现都较为基础，但本人在调试函数的输入处发现了知识盲区。由于输入的要求，我们在输入时，字符串、整数、浮点数都需要读入。但是，如果在读入整数之后直接调用 $next\_Line$ 方法，会出现输入错误的情况。后来经过查证和实验发现，是因为 $next\_int$ 方法会在读入后忽略行末的换行符，则换行符还会处于系统的缓冲区之中。此时，如果我们直接使用 $next\_line$ 方法，那么该方法就会读入一个换行符，而不是我们期望的下一行的字符串操作，导致后续进行操作时运行错误。解决办法是，在读取数量之后立即额外调用一次 $next\_line$ 方法，或者是使用字符串格式读入整型和浮点型，使用方法转换为整型或浮点型即可。

3、智能钱包

(1) 解题思路

在钱包中使用一个 num 数组记录所有钱的数量。大部分操作较为简单，而对于取钱操作，从小到大遍历所有的钱，每次先尝试取最大数量，如果超过，则利用除法向下取整，看是否能恰好取完即可。

(2) 核心代码

定义钱包中钱的数量和具体价值。其中 $praice$ 数组为常量。

```java
int[] number = new int[6];
final int[] price = {0, 1, 5, 10, 20, 100};
```

对于取钱操作，先直接从小到大遍历所有的钱，尝试一次性取完。如果超过上限，则利用差值进行除法，看是否能够刚好取完。同时记录 del 数组，看当前的钱可以取出来多少张，方便最后进行更新操作。

```java
@ParameterizedTest
@CsvSource({"1"})
public int withdraw(int amount){
    int sum = 0;
    int[] del = new int[6];
    boolean flag = true;
    for(int i = 1; i <= 5; i++){
        if(sum == amount) break;
        if(sum + number[i] * price[i] <= amount){
            del[i] = number[i];
            sum = sum + number[i] * price[i];
        }
        else{
            int tmp = (amount - sum) / price[i];
            if(sum + tmp * price[i] < amount){
                flag = false;
                sum = sum + tmp * price[i];
                break;
            }
            else{
                sum = sum + tmp * price[i];
                del[i] = tmp;
            }
        }
    }
    if(flag){
        System.out.printf("Successfully withdraw %d yuan! \n", sum);
        for(int i = 1; i <= 5; i++)
            number[i] -= del[i];
    }
    else System.out.printf("Can only withdraw %d yuan!\n", sum);
    return sum;
}
```

对于 *toString* 方法，通过字符串串联操作，遍历所有的钱，将它们合并到同一个字符串中，且按照固定格式，方便输出。

```
public String toString(){
    String str = "balance: ";
    for(int i = 1; i <= 5; i++){
        str = str + "¥" + price[i] + " * " + number[i] + " ";
    }
    str = str + "total: ¥" + balance();
    return str;
}
```

对于测试类，此处给出两种方法。

第一种是普通常规测试，即进行手动输入样例。由于输入操作前的字符串长度可以预先得知，故在按照固定格式输入时，可以判断首字母，然后选择对应操作。注意，此处操作应严格按照题目样例输入。

```
public void Normal_test(){
    Scanner input = new Scanner(System.in);
    String opt;
    System.out.println("Input exit to stop! ");
    opt = input.nextLine();
    while(true){
        if(opt.charAt(0) == 'e') break;
        if(opt.charAt(0) == 'c'){
            int a = input.nextInt();
            int b = input.nextInt();
            int c = input.nextInt();
            int d = input.nextInt();
            int e = input.nextInt();
            wallet = new Wallet(a, b, c, d, e);
            System.out.println(wallet.toString());
            input.nextLine();
        }
        else if(opt.charAt(0) == 'd'){
            int x = 0;
            int index = 8;
            while(index < opt.length() && opt.charAt(index) >= '0' &&
opt.charAt(index) <= '9'){
                x = x * 10 + opt.charAt(index) - '0';
                index++;
            }
            wallet.deposit(x);
            System.out.println(wallet.toString());
        }
        else{
            int x = 0, index = 16;
            while(index < opt.length() && opt.charAt(index) >= '0' &&
opt.charAt(index) <= '9'){
                x = x * 10 + opt.charAt(index) - '0';
                index++;
            }
            wallet.withdraw(x);
            System.out.println(wallet.toString());
        }
        opt = input.nextLine();
    }

}
```

接下来给出利用 $Junit$ 进行测试的方法。利用 $CsvSource$ 关键字，我们创建一个 $testDeposit$ 方法，每次都先存入 a, 然后尝试取出 b, 可以按照样例测试多组。

```
@ParameterizedTest
@CsvSource(value = {"5 25", "5 22"})
public void testDeposit(int a, int b){
    Wallet wallet = new Wallet();
    wallet.deposit(a);
    System.out.println(wallet.toString());
    wallet.withdraw(b);
    System.out.println(wallet.toString());
    return ;
}
```

当然，我们也可以在主函数中调用这个方法，手动输入多个测试样例。

```
public class Main {

    // static TestWallet tw = new TestWallet();
    static Find_wallet fW = new Find_wallet();  // 测试类
    @ParameterizedTest
    @CsvSource(value = {"5 25", "5 22"})  // 利用 Junit 测试
    public static void Test(int a, int b){
        tw.testDeposit(a, b);
    }

    public static void main(String[] args) {

//          tw.Normal_test();
        fW.Test();
    }
}
```

(3) 调试过程

本题目中，我在安装 $Junit$ 时遇到了一些困难，后面通过网上查阅教程，发现安装 $Maven$ 后，使用其管理 $Java$ 项目可以非常方便地安装 $Junit$ 。只需要在 $pom.xml$ 中添加依赖项便可以自动下载安装，非常方便。同时，$Junit$ 也非常方便，可以直接单独测试某一个方法，而不需要编译整个项目来进行调试，非常有利于在代码量较大的时候锁定问题来源，及时进行排查调整，是一个很好的编程利器。

4、找出智能钱包中的纸币种类和数目

(1) 解题思路

从小到大一个一个取，比如每次只取 1 块钱，或只取 5 块钱，直到当前数目无法取出为止。可以知道，此时和更大面额的钱无关，不会取出更大面额的钱币。如果当前面额的钱没有取完，那么本次取钱一定会直接取出一张当前面额的钱币。以此类推，我们一直取钱，到无法取出为止所花的次数，就是该面额的总数。之后记录总数即可。

(2) 核心代码

本题目核心代码只有下面一段。首先读入钱包参数并创建，然后从小到大枚举面额。接着依次取出，直到无法取出为止。同时记录总数和操作次数。注意之后通过 for 循环再插回所有的钱币。

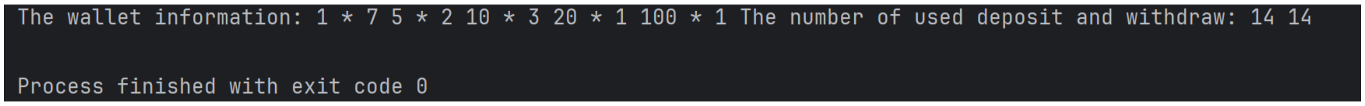
```
Wallet wallet = new Wallet(a, b, c, d, e);

int num1 = 0, num2 = 0;
for(int i = 1; i ≤ 5; i++){
    int denomination = price[i];
    int count = 0;
        while (wallet.balance() > 0 && wallet.withdraw(denomination) ==
denomination){
                count++;
        num1++;
    }
    num[i] = count;
}
for(int i = 1; i ≤ 5; i++){
    for(int j = 1; j ≤ num[i]; j++){
        wallet.deposit(price[i]);
        num2++;
    }
}
System.out.printf("The wallet information: ");
for(int i = 1; i ≤ 5; i++){
    System.out.printf("%d * %d ", price[i], num[i]);
}

System.out.print("The number of used deposit and withdraw: ");
System.out.printf("%d %d\n", num1, num2);
```

输出

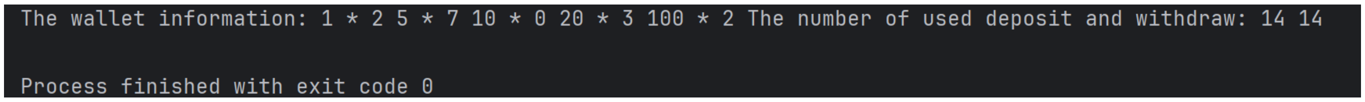
样例 1 输出:



```
The wallet information: 1 * 7 5 * 2 10 * 3 20 * 1 100 * 1 The number of used deposit and withdraw: 14 14

Process finished with exit code 0
```

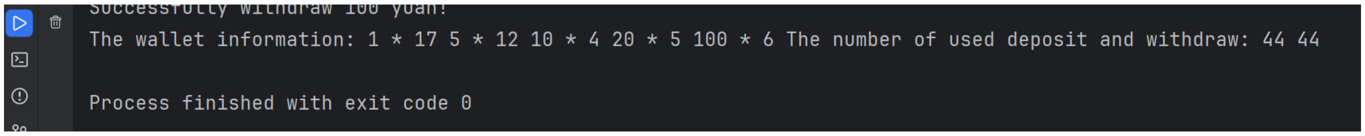
样例 2 输出:



```
The wallet information: 1 * 2 5 * 7 10 * 0 20 * 3 100 * 2 The number of used deposit and withdraw: 14 14

Process finished with exit code 0
```

样例 3 输出:



```
Successfully withdraw 100 yuan!
The wallet information: 1 * 17 5 * 12 10 * 4 20 * 5 100 * 6 The number of used deposit and withdraw: 44 44

Process finished with exit code 0
```

(3) 调试过程

本人一开始并没有使用下标遍历面额，而是直接使用 $foreach$ 方法便利所有面额。 但是这遇到了一个问题，我们在记录面额数量时，需要知道当前面额的大小，然后映射至 1~5 上，这需要耗费较多精力与代码进行分类讨论。直接使用下标，然后赋值给变量 $denomination$ 会更加方便。

实验总结

通过本次实验，我加深了对 $Java$ 各预设类的熟练程度，对字符串处理方面有了更深的见解。值得一提的是，在输入方面，以前我从未遇到过同时读入 $String$ 类型和 $Integer$ 整型以及浮点型的情况。起初我直接使用 $nextint$ 和 $nextLine$ 方法，这导致我在之后进行输入测试的时候吃了不少苦头，好在通过上网查询，得知了末尾换行符这一问题。

其次是对于 $Junit$ 的使用。这款功能强大的插件，可以让我们在不编译整个项目的情况下，单独测试某个方法的正确性，极大地提升了检查代码的效率。同时，为了安装这款插件，我也安装了 $Maven$ 的 $Java$ 管理方法。根据介绍，这种管理方法有很多优点，值得我以后继续深入探索。

在第一个实验中，我也独自完成了 $Kmeans$ 聚类算法。作为一款经典的分类算法，在模式识别课上我对其早有耳闻。这次我也是详细地研究了其算法流程：先随机选择 k 个中心点，然后对每个点寻找最近的中心，接着求出某个类的平均值，更新最新点，直到无变化为止。同时，为了实现这个算法，方便在各个方法之间进行传输，我也使用了 $List$ 类。这有些类似 $C++$ 中的 $Vector$ 类，可以方便地动态扩展，有利于这种长度不确定但总量只有一定数量的情况。

总之，通过这次实验，我掌握了新插件和新类的用法，日后还要继续巩固。

完整代码

1、

```
import java.awt.*;
import java.awt.geom.Point2D;
import java.time.LocalDate;
import java.time.LocalDateTime;
import java.time.format.DateTimeFormatter;
import java.util.ArrayList;
import java.util.Date;
import java.util.List;
import java.util.Random;

//TIP To <b>Run</b> code, press <shortcut actionId="Run"/> or
// click the <icon src="AllIcons.Actions.Execute"/> icon in the gutter.
public class Main {

    public static void main(String[] args) {
        Date d = new Date();

        String formatted_time = String.format("%04d-%02d-%02d-%02d:%02d", 1900
+ d.getYear(), d.getMonth() + 1, d.getDate(), d.getHours(), d.getMinutes());
        System.out.println(formatted_time);

        Random random = new Random(d.getTime());
        int a = random.nextInt(), b = random.nextInt();
        System.out.printf("%d %d\n", a, b);
        int[] x = new int[11];
        int[] y = new int[11];

        for(int i = 1; i <= 10; i++){
            x[i] = random.nextInt(10);
            y[i] = random.nextInt(10); // 生成小数字方便检查
        }
        Point2D[] points = new Point2D[11];
        for(int i = 1; i ≤ 10; i++){
            points[i] = new Point2D.Double(x[i], y[i]);
//            points[i].setLocation(x[i], y[i]);
        }

        int k = 2; // 可以任选k的大小 (分多少类)
        List<Point2D> ans[] = Kmeans(points, k);

        for(int i = 1; i ≤ k; i++){
            System.out.printf("The %dth kind: ", i);
            for(Point2D now : ans[i]){
                System.out.printf("(%d %d) ", (int)now.getX(),
(int)now.getY());
            }
            System.out.println();
        }


        return ;
    }

    private static List<Point2D>[] Kmeans(Point2D[] points, int k){ // k: 分多
少类
        List<Point2D>[] L = new List[k+1]; // 返回不同类的结果
        Point2D[] centres = new Point2D[k+1]; // 当前的中心点下标
        Point2D[] Lstcentres = new Point2D[k+1]; // 上一次的中心点坐标
        for(int i = 1; i ≤ k; i++){
            centres[i] = points[i]; // 初始化中心点
            Lstcentres[i] = centres[i];
        }
        int cnt = 0;
        while(true){
//            cnt++;
            L = new List[k+1];
            for(int i = 1; i ≤ k; i++)
                L[i] = new ArrayList<Point2D>();
            for(int i = 1; i ≤ 10; i++){
                double Minn = 1e18;
                int kind_index = 0;
                for(int j = 1; j ≤ k; j++){
                    double now_Dis = points[i].distance(centres[j]);
                    if(now_Dis < Minn){
                        Minn = now_Dis;
                        kind_index = j;
                    }
                }

                L[kind_index].add(points[i]);

            }

            // 更新中心点
            Refresh_centre(centres, L, k);
            boolean flag = false;
            for(int i = 1; i <= k; i++){
                if(centres[i] != Lstcentres[i]){
                    flag = true;
                    break;
                }
            }
            Lstcentres = centres;
            if(!flag) break;
        }
        return L;
    }

    private static double Cal_distance(Point2D a, Point2D b){
        double x1 = a.getX();
        double x2 = b.getX();
        double y1 = a.getY();
        double y2 = b.getY();
        return Math.sqrt((x1 - x2) * (x1 - x2) + (y1 - y2) * (y1 - y2));
    }

    private static void Refresh_centre(Point2D[] centres, List<Point2D>[] L,
int k){
        for(int i = 1; i <= k; i++){
            double tmpx = 0, tmpy = 0;
            for(Point2D now : L[i]){
                tmpx += now.getX();
                tmpy += now.getY();
            }
            tmpx /= (double) L[i].size();
            tmpy /= (double) L[i].size();

            Point2D tmp_centre = new Point2D.Double(tmpx, tmpy);
            Point2D new_centre = new Point2D.Double(0, 0);
            double min_dis = (double)1e18;
```

```java
            for(Point2D now : L[i]){
                if(now.distance(tmp_centre) < min_dis){
                    min_dis = now.distance(tmp_centre);
                    new_centre = now;
                }
            }
            centres[i] = new_centre;
        }
        return ;
    }

}
```

2、

共四个文件。

Main.java

```java
//TIP To <b>Run</b> code, press <shortcut actionId="Run"/> or
// click the <icon src="AllIcons.Actions.Execute"/> icon in the gutter.
public class Main {

    static RentalCarTester Tester = new RentalCarTester();

    public static void main(String[] args) {
        Tester.test();
        return ;
    }
}
```

Car.java

```java
public class Car{
    private String carRenter;
    private String carName;
    private String carType;
    private int carNum;
    private double rate;
    private int days;

    Car(String carRenter, int carNum, String carName, String carType, double
rate, int days){
        this.carRenter = carRenter;
        this.carNum = carNum;
        this.carName = carName;
        this.carType = carType;
        this.rate = rate;
        this.days = days;
    }

    Car(){
        this.carRenter = "";
        this.carNum = 0;
        this.carName = "";
        this.carType = "";
        this.rate = 0;
        this.days = 0;
    }

    public String getCarRenter(){
        return this.carRenter;
    }

    public int getCarNum(){
        return  this.carNum;
    }
    public String getCarName(){
        return this.carName;
    }

    public String getCarType() {
        return this.carType;
    }

    public double getRate(){
        return this.rate;
    }

    public int getDays() {
        return this.days;
    }

    public void setCarRenter(String carRenter) {
        this.carRenter = carRenter;
    }

    public void setCarName(String carName) {
        this.carName = carName;
    }
    public void setCarType(String carType) {
        this.carType = carType;
    }

    public void setDays(int days) {
        this.days = days;
    }

    public void setRate(double rate) {
        this.rate = rate;
    }

    public void setCarNum(int carNum) {
        this.carNum = carNum;
    }

    @Override
    public String toString() {
        System.out.println("The following is the car's information: ");
        String str = "";
        str = "The car renter: " + this.carRenter + "\n" + "The car number: "
+ this.carNum + "\n" + "The car type: " + this.carType + "\n" + "The rate: " +
this.rate + "\n" + "The rented days: " + this.days ;
        System.out.println(str);
        return str;
    }
}
```

RentalCarCompany.java

```
public class RentalCarCompany{
    private String Company_name;
    private Car[] theCars = new Car[101];
    private int totDays;
    private double totRate;
    private int rentCnt;

    RentalCarCompany(String Company_name){
        this.Company_name = Company_name;
    }

    RentalCarCompany(){
        this.Company_name = "";
        this.theCars = new Car[101];
        this.totDays = 0;
        this.totRate = 0;
        this.rentCnt = 0;
    }

    public Car[] getTheCars(){
        return this.theCars;
    }

    public String getCompany_name(){
        return this.Company_name;
    }

    public void setCompany_name(String company_name) {
        this.Company_name = company_name;
    }

    public void addReservation(String renter, int carNum, String carName,
String carType, double rate, int days){
        this.theCars[++this.rentCnt] = new Car(renter, carNum, carName,
carType, rate, days);
        this.totDays += days;
        this.totRate += rate;
        return ;
    }

    public double getAverageDays(){
        return (double)this.totDays / (double)rentCnt;
    }

    public double getAverageRate(){
        return (double)this.totRate / (double)this.rentCnt;
    }

    public double getRentialincome(){
        double sum = 0;
        for(int i = 1; i <= rentCnt; i++){
            sum += theCars[i].getRate() * theCars[i].getDays();
        }
        return sum;
    }

    void findReservation(int x){
        boolean flag = false;
        for(int i = 1; i <= rentCnt; i++){
            if(theCars[i].getCarNum() == x){
                theCars[i].toString();
                flag = true;
                break;
            }
        }
        if(!flag){
            System.out.printf("Could not find reservation for this car number 
%d\n", x);
        }
        return ;
    }
}
```

RentalCarTester.java

```
import java.util.Scanner;

public class RentalCarTester{
    RentalCarCompany company = new RentalCarCompany();

    public void test(){
        Scanner input = new Scanner(System.in);
        System.out.print("Rental Car Company: ");
        String compnay_name = input.nextLine();
        company.setCompany_name(compnay_name);
        String opt = "-1";
        while(!opt.equals("0")){
            System.out.println("1 to add Car, 2 to find car number, 3 to ask 
for average data, 0 to stop! ");
            opt = input.nextLine();
            if(opt.equals("0")) break;
            if(opt.equals("1")){
                System.out.print("Car renter's name: ");;
                String render = input.nextLine();
                System.out.print("Car number: ");
                String number = input.nextLine();
                System.out.print("Car name: ");
                String carName = input.nextLine();
                System.out.print("Car type: ");
                String carType = input.nextLine();
                System.out.print("Rate: ¥");
                String rate = input.nextLine();
                System.out.print("Rented for(days): ");
                String  days = input.nextLine();
                company.addReservation(render, Integer.parseInt(number),
carName, carType, Double.parseDouble(rate), Integer.parseInt(days));

            }
            if(opt.equals("2")){
                System.out.println("Search for car number: ");
                String x = input.nextLine();
                company.findReservation(Integer.parseInt(x));
            }
            if(opt.equals("3")){
                System.out.printf("Average days rented out: %f\n",
company.getAverageDays());
                System.out.printf("Average rent: %f\n",
company.getAverageRate());
                System.out.printf("Tot rential income is: %f\n",
company.getRentialincome());
            }

        }
        return ;
    }
}
```

3和4

共4个文件。

Main.java

```
package org.example;
import org.testng.annotations.Test;
import org.junit.jupiter.params.ParameterizedTest;
import org.junit.jupiter.params.provider.CsvSource;

public class Main {

    static TestWallet tw = new TestWallet();
    static Find_wallet fW = new Find_wallet();
    @ParameterizedTest
    @CsvSource(value = {"5 25", "5 22"})
    public static void Test(int a, int b){
        tw.testDeposit(a, b);
    }

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

```
//          tw.Normal_test();
        fW.Test();
    }
}
```

Wallet.java

```
package org.example;

import org.junit.jupiter.params.ParameterizedTest;
import org.junit.jupiter.params.provider.CsvSource;

public class Wallet{
    int[] number = new int[6];
    final static int[] price = {0, 1, 5, 10, 20, 100};

    Wallet(){
        for(int i = 0; i < 6; i++)
            number[i] = 0;
    }

    Wallet(int a, int b, int c, int d, int e){
        number[1] = a;
        number[2] = b;
        number[3] = c;
        number[4] = d;
        number[5] = e;
    }

    @ParameterizedTest
    @CsvSource({"1"})
    public int withdraw(int amount){
        int sum = 0;
        int[] del = new int[6];
        boolean flag = true;
        for(int i = 1; i <= 5; i++){
            if(sum == amount) break;
            if(sum + number[i] * price[i] <= amount){
                del[i] = number[i];
                sum = sum + number[i] * price[i];
            }
            else{
                int tmp = (amount - sum) / price[i];
                if(sum + tmp * price[i] < amount){
                    flag = false;
```

```java
                    sum = sum + tmp * price[i];
                    break;
                }
                else{
                    sum = sum + tmp * price[i];
                    del[i] = tmp;
                }
            }
        }
        if(flag){
            System.out.printf("Successfully withdraw %d yuan! \n", sum);
            for(int i = 1; i <= 5; i++)
                number[i] -= del[i];
        }
        else System.out.printf("Can only withdraw %d yuan!\n", sum);
        return sum;
    }

    @ParameterizedTest
    @CsvSource({"1"})
    public void deposit(int val){
        if(val == 1){
            number[1]++;
        }
        if(val == 5){
            number[2]++;
        }
        if(val == 10){
            number[3]++;
        }
        if(val == 20){
            number[4]++;
        }
        if(val == 100){
            number[5]++;
        }
        return ;
    }

    public int balance(){
//        System.out.print("Balance: ");
        int sum = 0;
        for(int i = 1; i <= 5; i++){
//            System.out.printf("¥%d * %d ", number[i], price[i]);
            sum += number[i] * price[i];
        }
```

```
//          System.out.printf("total: ¥%d\n", sum);
        return sum;
    }
    public String toString(){
        String str = "balance: ";
        for(int i = 1; i <= 5; i++){
            str = str + "¥" + price[i] + " * " + number[i] + " ";
        }
        str = str + "total: ¥" + balance();
        return str;
    }


}
```

TestWallet.java

```
package org.example;

import org.junit.Test;
import org.junit.jupiter.params.ParameterizedTest;
import org.junit.jupiter.params.provider.CsvSource;

import java.util.Scanner;

public class TestWallet{

    private Wallet wallet = new Wallet();

    @ParameterizedTest
    @CsvSource(value = {"5 25", "5 22"})
    public void testDeposit(int a, int b){
        Wallet wallet = new Wallet();
        wallet.deposit(a);
        System.out.println(wallet.toString());
        wallet.withdraw(b);
        System.out.println(wallet.toString());
        return ;
    }

    public void Normal_test(){
        Scanner input = new Scanner(System.in);
        String opt;
        System.out.println("Input exit to stop! ");
        opt = input.nextLine();
```

```java
        while(true){
            if(opt.charAt(0) == 'e') break;
            if(opt.charAt(0) == 'c'){
                int a = input.nextInt();
                int b = input.nextInt();
                int c = input.nextInt();
                int d = input.nextInt();
                int e = input.nextInt();
                wallet = new Wallet(a, b, c, d, e);
                System.out.println(wallet.toString());
                input.nextLine();
            }
            else if(opt.charAt(0) == 'd'){
                int x = 0;
                int index = 8;
                while(index < opt.length() && opt.charAt(index) >= '0' &&
opt.charAt(index) <= '9'){
                    x = x * 10 + opt.charAt(index) - '0';
                    index++;
                }
                wallet.deposit(x);
                System.out.println(wallet.toString());
            }
            else{
                int x = 0, index = 16;
                while(index < opt.length() && opt.charAt(index) >= '0' &&
opt.charAt(index) <= '9'){
                    x = x * 10 + opt.charAt(index) - '0';
                    index++;
                }
                wallet.withdraw(x);
                System.out.println(wallet.toString());
            }
            opt = input.nextLine();
        }

    }

}
```

Find_wallet.java

```java
package org.example;

import java.util.Scanner;
```

```
public class Find_wallet{
    private int[] num = new int[6];
    private int[] price = {0, 1, 5, 10, 20, 100};
    public void Test(){
        Scanner input = new Scanner(System.in);
        int a = input.nextInt();
        int b = input.nextInt();
        int c = input.nextInt();
        int d = input.nextInt();
        int e = input.nextInt();
        Wallet wallet = new Wallet(a, b, c, d, e);

        int num1 = 0, num2 = 0;
        for(int i = 1; i <= 5; i++){
            int denomination = price[i];
            int count = 0;
            while (wallet.balance() > 0 && wallet.withdraw(denomination) ==
denomination){
                count++;
                num1++;
            }
            num[i] = count;
        }
        for(int i = 1; i <= 5; i++){
            for(int j = 1; j <= num[i]; j++){
                wallet.deposit(price[i]);
                num2++;
            }
        }
        System.out.printf("The wallet information: ");
        for(int i = 1; i <= 5; i++){
            System.out.printf("%d * %d ", price[i], num[i]);
        }

        System.out.print("The number of used deposit and withdraw: ");
        System.out.printf("%d %d\n", num1, num2);

    }

}
```