## 第十一章练习题

### 11.8

package work7;  
  
import java.util.Date;  
import java.util.ArrayList;  
  
public class TestAccout {  
 public static void main(String[] args) {  
 // 用于为账户存储交易  
 // ArrayList<Object> transactions=new ArrayList<>();  
  
 Account object = new Account("George", 1122, 1000);  
 object.setAnnualInterestRate(0.015);  
  
 object.deposit(30);  
 object.deposit(40);  
 object.deposit(50);  
  
 object.withDraw(5);  
 object.withDraw(4);  
 object.withDraw(2);  
  
 System.out.println("账户持有者名字： " + object.getName()  
 + "\n利率: " + object.getAnnualInterestRate()  
 + "\n收支额： " + object.getBalance()  
 + "\n所有的交易:" + object.toString());  
 }  
}  
  
class Account {  
 private String name;  
  
 // 用于为账户存储交易  
 private ArrayList<Object> transactions = new ArrayList<>();  
  
 private int id = 0;// 用户名  
 private double balance = 0;// 余额  
 private double annualInteresRate = 0;// 当前利率  
 private Date dateCreated;// 存储开户日期  
  
 // 无参构造方法  
 public Account() {  
 Date dateCreated = new Date();  
 this.dateCreated = dateCreated;  
  
 }  
  
 // 有参构造方法  
 public Account(int id, double balance) {  
 Date dateCreated = new Date();  
 this.dateCreated = dateCreated;  
 this.id = id;  
 this.balance = balance;  
 }  
  
 // 一个新的有参构造方法  
 public Account(String name, int id, double balance) {  
 Date dateCreated = new Date();  
 this.dateCreated = dateCreated;  
 this.id = id;  
 this.balance = balance;  
 this.name = name;  
 }  
  
 public void setId(int id) {  
 this.id = id;  
 }  
  
 public void setBalance(double balance) {  
 this.balance = balance;  
 }  
  
 public void setAnnualInterestRate(double annualInterestRate) {  
 this.annualInteresRate = annualInterestRate;  
 }  
  
 public int getId() {  
 return id;  
  
 }  
  
 public double getBalance() {  
 return balance;  
  
 }  
  
 public String getName() {  
 return name;  
  
 }  
  
 public double getAnnualInterestRate() {  
 return annualInteresRate;  
  
 }  
  
 public double getMonthlyInterestRate() {  
 return annualInteresRate / 12;  
  
 }  
  
 public String getDateCreated() {  
 return dateCreated.toString();  
  
 }  
  
 // 取钱  
 public double withDraw(double withDarw) {  
  
 // 创建一笔取钱交易  
 Transaction withDrawTransacte = new Transaction('W', withDarw, (this.balance = this.balance - withDarw),  
 "取款：" + withDarw + "美元");  
 transactions.add(withDrawTransacte.getDescription());  
  
 return this.balance;  
  
 }  
  
 // 存钱  
 public double deposit(double deposit) {  
 // 创建一笔存钱交易  
 Transaction depositTransacte = new Transaction('D', deposit, (this.balance = this.balance + deposit),  
 "存款：" + deposit + "美元");  
 transactions.add(depositTransacte.getDescription());  
  
 return this.balance;  
 }  
  
 public String toString() {  
 return "\n日期： " + getDateCreated() + "\n" + transactions.toString();  
  
 }  
}  
  
class Transaction {  
 // 交易日期  
 private Date date;  
  
 // 交易类型，例如'W','D'  
 private char type;  
  
 // 交易量  
 private double amount;  
  
 // 交易后的新余额  
 private double balance;  
  
 // 交易描述  
 private String description;  
  
 // 一个有参构造方法  
 public Transaction(char type, double amount, double balance, String description) {  
 this.type = type;  
 this.amount = amount;  
 this.balance = balance;  
 this.description = description;  
  
 // 创建一个交易日期  
 this.date = new Date();  
 }  
  
 // 获得交易日期  
 public String getDate() {  
 return date.toString();  
  
 }  
  
 // 设置交易日期  
 public void setDate(long eclapseTime) {  
 date.setTime(eclapseTime);  
 }  
  
 // 交易类型：‘W’——取款，‘D’——存款  
 public char getType() {  
 return type;  
  
 }  
  
 public void setType(char type) {  
 this.type = type;  
 }  
  
 public double getAmount() {  
 return amount;  
  
 }  
  
 public void setAmount(double amount) {  
 this.amount = amount;  
 }  
  
 public double getBalance() {  
 return balance;  
  
 }  
  
 public void setBalance(double balance) {  
 this.balance = balance;  
 }  
  
 public String getDescription() {  
 return description;  
  
 }  
  
 public void setDescription(String description) {  
 this.description = description;  
 }  
}

### 11.16

package work7;  
  
import java.util.Scanner;  
import java.util.ArrayList;  
  
public class RepeatAddtionQuiz {  
 public static void main(String[] args) {  
 int number1 = (int) (Math.random() \* 10);  
 int number2 = (int) (Math.random() \* 10);  
  
 Scanner input = new Scanner(System.in);  
  
 ArrayList<Integer> list = new ArrayList<Integer>();  
  
 int answer = -1;  
 while (answer != number1 + number2) {  
 System.out.print("What is " + number1 + " + " + number2 + "? ");  
 answer = input.nextInt();  
 if (answer == number1 + number2) {  
 continue;  
 }  
 if (list.contains(answer)) {  
 System.out.println("You already entered " + answer + ".");  
 } else {  
 System.out.print("Wrong answer. Try again. ");  
 list.add(answer);  
 }  
 }  
 System.out.println("You got it!");  
  
 input.close();  
 }  
}

### 11.17

package work7;  
  
import java.util.Scanner;  
  
public class work11\_17 {  
 public static void main(String[] args) {  
 Scanner input = new Scanner(System.in);  
  
 System.out.print("Enter an integer m: ");  
  
 int m = input.nextInt();  
  
 int[] array = new int[1000];  
 int p = 2;  
 int temp = m;  
 while (temp != 1) {  
 if (temp % p == 0) {  
 temp = temp / p;  
 array[p]++;  
 } else {  
 p++;  
 }  
 }  
 int ans = 1;  
 for (int i = 2; i < array.length; i++) {  
 if ((array[i] & 1) == 1) {  
 ans \*= i;  
 }  
 }  
 int ans2 = ans \* m;  
  
 System.out.println("The smallest number for m \* n to be a perfect square is " + ans);  
 System.out.println("m \* n is " + ans2);  
  
 input.close();  
 }  
}

### 11.19

package work7;
  
  
import java.util.Scanner;
  
import java.util.ArrayList;
  
  
public class work11\_19 {
  
 public static void main(String[] args) {
  
 Scanner input = new Scanner(System.in);
  
  
 System.out.print("Enter the number of objects: ");
  
 int num = input.nextInt();
  
 System.out.print("Enter the weights of the objects: ");
  
  
 ArrayList<Integer> weights = new ArrayList<Integer>();
  
 for (int i = 0; i < num; i++) {
  
 weights.add(input.nextInt());
  
 }
  
  
 ArrayList<ArrayList<Integer>> ans = new ArrayList<ArrayList<Integer>>();
  
 int[] V = new int[100];
  
 for (int i = 0; i < 100; i++) {
  
 V[i] = 10;
  
 }
  
 int cnt = 0, pos = 0;
  
 while (pos < weights.size()) {
  
 boolean flag = true;
  
 for (int i = 0; i < cnt; i++) {
  
 if (weights.get(pos) <= V[i]) {
  
 V[i] -= weights.get(pos);
  
 ans.get(i).add(weights.get(pos));
  
 flag = false;
  
 break;
  
 }
  
 }
  
 if (flag) {
  
 cnt++;
  
 ans.add(new ArrayList<Integer>());
  
 ans.get(cnt - 1).add(weights.get(pos));
  
 V[cnt - 1] -= weights.get(pos);
  
 }
  
 ++pos;
  
 }
  
  
 for (int i = 0; i < ans.size(); i++) {
  
 System.out.printf("Container %d contains objects with weight", i + 1);
  
 for (int j = 0; j < ans.get(i).size(); j++) {
  
 System.out.print(" " + ans.get(i).get(j));
  
 }
  
 System.out.println("");
  
 }
  
 input.close();
  
 }
  
}