Java 工程师实例参考手册

(V1.0)

整理：廖清远

湖南商务职业技术学院

目录

一 基础程序练习

【程序 1】古典兔子问题

题目：古典问题：有一对兔子，从出生后第3个月起每个月都生一对兔子，小兔子长到第三个月后每个月又生一对兔子，假如兔子都不死，问每个月的兔子总数为多少？
1.程序分析： 兔子的规律为数列1,1,2,3,5,8,13,21....
*/

```
package cn.com.flywater.FiftyAlgorthm;
public class FirstRabbit {
public static final int MONTH = 15;
public static void main(String[] args) {
    long f1 = 1L, f2 = 1L;
    long f;
    for(int i=3; i<MONTH; i++) {
     f = f2;
     f2 = f1 + f2;
     f1 = f;
     System.out.print("第" + i +"个月的兔子对数: ");
     System.out.println(" " + f2);
    }
}
}
```

【程序 2】素数输出

题目：判断101-200之间有多少个素数，并输出所有素数。
1.程序分析：判断素数的方法：用一个数分别去除2到 sqrt(这个数)，如果能被整除，
则表明此数不是素数，反之是素数。 */

```
package cn.com.flywater.FiftyAlgorthm;
public class SecondPrimeNumber {
public static int count = 0;
 public static void main(String[] args) {
      for(int i=101;i<200;i++){
           boolean b = true;//默认此数就素数
           for(int j=2;j<=Math.sqrt(i);j++){
                if(i%j==0){
                     b = false; //此数不是素数
                     break;
```

```
                }
            }
            if(b){
                count++;
                System.out.print(i+" ");
            }
        }
        System.out.println("\n 素数的个数："+count);
    }
}
```

【程序 3】水仙花数

题目：打印出所有的"水仙花数(narcissus number)"，所谓"水仙花数"是指一个三位数，
其各位数字立方和等于该数本身。例如：153是一个"水仙花数"，因为153=1的三次方＋5的三次方＋3的三次方。
1.程序分析：利用 for 循环控制100-999个数，每个数分解出个位，十位，百位。 */

```
package cn.com.flywater.FiftyAlgorthm;
public class ThirdNarcissusNum {
static int b, bb, bbb;
public static void main(String[] args) {

   for(int num=101; num<1000; num++) {
    ThirdNarcissusNum tnn = new ThirdNarcissusNum();
    tnn.f(num);
   }
}
public void f(int m) {
   bbb = m / 100;
   bb = (m % 100) / 10;
   b = (m % 100) % 10;
   if((bbb * bbb * bbb + bb * bb * bb + b * b * b) == m) {
    System.out.println(m);
   }
}
}
```

【程序 4】正整数分解质因数

题目：将一个正整数分解质因数。例如：输入90,打印出90=2*3*3*5。
程序分析：对 n 进行分解质因数，应先找到一个最小的质数 k，然后按下述步骤完成：
(1)如果这个质数恰等于 n，则说明分解质因数的过程已经结束，打印出即可。
(2)如果 n>k，但 n 能被 k 整除，则应打印出 k 的值，并用 n 除以 k 的商,作为新的正整数你 n,重复执行第一步。
(3)如果 n 不能被 k 整除，则用 k+1作为 k 的值,重复执行第一步。 */

```
package cn.com.flywater.FiftyAlgorthm;
import java.util.Scanner;
public class FourthPrimeFactor {
static int n, k = 2;
public static void main(String[] args) {
    Scanner s = new Scanner(System.in);
    n = s.nextInt();
    System.out.print(n + "=" );
    FourthPrimeFactor fpf = new FourthPrimeFactor();
    fpf.f(n);
}
public void f(int n) {
    while(k <= n) {
     if(k == n) {
      System.out.println(n);
      break;
     } else if(n > k && n % k == 0) {
      System.out.print(k + "*");
      n = n / k;
      f(n);
      break;
     } else if(n > k && n % k != 0) {
      k++;
      f(n);
      break;
     }
    }
}
}
```

【程序 5】成绩分段表示

题目：利用条件运算符的嵌套来完成此题：学习成绩>=90分的同学用 A 表示，60-89分之间的用 B 表示，60分以下的用 C 表示。
1.程序分析：(a>b)?a:b 这是条件运算符的基本例子。 */

```
package cn.com.flywater.FiftyAlgorthm;
import java.util.Scanner;
public class FifthCondition {
//public static final int S1 = 90;
//public static final int S2 = 60;
static int grade;
public static void main(String[] args) {
    Scanner str = new Scanner(System.in);
    int s = str.nextInt();
    FifthCondition fc = new FifthCondition();
    grade = fc.compare(s);
    if(grade == 1) {
     System.out.print('A');
    } else if(grade == 2) {
     System.out.print('B');
    } else {
     System.out.println('C');
    }
}
public int compare(int s) {
    return s > 90 ? 1
      : s > 60 ? 2
      :3;
}
}
```

【程序 6】最大公约数和最小公倍数

题目：输入两个正整数 m 和 n，求其最大公约数和最小公倍数。
1.程序分析：利用辗除法。 */

```
/*
* 在循环中，只要除数不等于0，用较大数除以较小的数，将小的一个数作为下一轮循环的大数，取得的余数作为下一轮循环的较小的数，如此循环直到较小的
```

```
数的值为0，返回
* 较大的数，此数即为最小公约数，最小公倍数为两数之积除以最小公倍数。
* */
package cn.com.flywater.FiftyAlgorthm;
import java.util.Scanner;
public class SixthCommonDiviser {
public static void main(String[] args) {
    int a, b;
    Scanner s1 = new Scanner(System.in);
    Scanner s2 = new Scanner(System.in);
    a = s1.nextInt();
    b = s2.nextInt();
    SixthCommonDiviser scd = new SixthCommonDiviser();
    int m = scd.division(a, b);
    int n = a * b / m;
    System.out.println("最大公约数: " + m);
    System.out.println("最小公倍数: " + n);
}
public int division(int x, int y) {
    int t;
    if(x < y) {
     t = x;
     x = y;
     y = t;
    }

    while(y != 0) {
     if(x == y) return 1;
     else {
      int k = x % y;
      x = y;
      y = k;
     }
    }
    return x;
}
}
```

【程序 7】字符统计

题目：输入一行字符，分别统计出其中英文字母、空格、数字和其它字符的个数。
1.程序分析：利用 while 语句,条件为输入的字符不为 '\n '. */

```
package cn.com.flywater.FiftyAlgorthm;
import java.util.*;
public class SeventhCharacterStatistics {
static int digital = 0;
static int character = 0;
static int other = 0;
static int blank = 0;
public static void main(String[] args) {
    char[] ch = null;
    Scanner sc = new Scanner(System.in);
    String s = sc.nextLine();
    ch = s.toCharArray();


    for(int i=0; i<ch.length; i++) {
     if(ch[i] >= '0' && ch[i] <= '9') {
      digital ++;
     } else if((ch[i] >= 'a' && ch[i] <= 'z') || ch[i] > 'A' && ch[i] <= 'Z')
{
      character ++;
     } else if(ch[i] == ' ') {
      blank ++;
     } else {
      other ++;
     }

    }
    System.out.println("数字个数: " + digital);
    System.out.println("英文字母个数: " + character);
    System.out.println("空格个数: " + blank);
    System.out.println("其他字符个数:" + other );
}

}
```

【程序 8】求 s=a+aa+aaa+aaaa+aa...a 的值

题目：求 s=a+aa+aaa+aaaa+aa...a 的值，其中 a 是一个数字。例如 2+22+222+2222+22222(此时共有5个数相加)，几个数相加有键盘控制。

```
*/
/*
* 算法： 定义一个变量 b， 赋初值为0；定义一变量 sum， 赋初值为0，
* 进入循环后，将 a + b 的值赋给 b，将 sum + b 的值赋给 sum；
```

```
* 同时，将 a 增加十倍， ++ i； 继续循环；
* 循环结束后，输出 sum 的值。
*/
package cn.com.flywater.FiftyAlgorthm;
import java.util.Scanner;
public class EightPlus {
static long a = 2, b = 0;
public static void main(String[] args) {
    Scanner s = new Scanner(System.in);
    int n = s.nextInt();
    int i = 0;
    long sum = 0;
    while(i < n) {
     b = b + a;
     sum = sum + b;
     a = a * 10;
     ++ i;
    }
    System.out.println("input number: " + n);
    System.out.println(sum);
}
}
```

【程序 9】求完数

题目：一个数如果恰好等于它的因子之和，这个数就称为 "完数 "。例如6=1＋2＋3.编程 找出1000以内的所有完数。

```
*/
package cn.com.flywater.FiftyAlgorthm;
public class NinthWanshu {
public static void main(String[] args) {

    System.out.println("1到1000的完数有： ");
    for(int i=1; i<1000; i++) {
     int t = 0;
     for(int j=1; j<= i/2; j++) {
      if(i % j == 0) {
       t = t + j;
      }
     }
     if(t == i) {
      System.out.print(i + " ");
     }
```

```
    }
}
}
```

【程序 10】球的自由落体运算

题目：一球从100米高度自由落下，每次落地后反跳回原高度的一半；再落下，求它在 第10次落地时，共经过多少米？第10次反弹多高？

```
*/
package cn.com.flywater.FiftyAlgorthm;
public class TenthTreeFall {
static double height = 100;
static double distance = 100;
public static void main(String[] args) {
    for(int i=1; i<10; i++) {
     distance = distance + height;
     height = height / 2;
    }

    System.out.println("路程：" + distance);
    System.out.println("高度：" + height / 2);
}
}
```

【程序 11】排列组合

题目：有1、2、3、4个数字，能组成多少个互不相同且无重复数字的三位数？都是多少？

1.程序分析：可填在百位、十位、个位的数字都是1、2、3、4。组成所有的排列后再去 掉不满足条件的排列。

```
*/
/*算法：3个 for 循环加一个 if 语句；
*
*/
package cn.com.flywater.FiftyAlgorthm;
public class EleventhNumberRange {
public static void main(String[] args) {
    int count = 0;
    for(int x=1; x<5; x++) {
     for(int y=1; y<5; y++) {
```

```
      for(int z=1; z<5; z++) {
       if(x != y && y != z && x != z) {
        count ++;
        System.out.print(x*100 + y*10 + z + "    ");
        if(count % 4 == 0) {
         System.out.println();
        }
       }
      }
     }
    }
    System.out.println("共有" + count + "个三位数");
}
}
```

【程序 12】 利润提成

题目：企业发放的奖金根据利润提成。利润(I)低于或等于10万元时，奖金可提10%；
利润高于10万元，低于20万元时，低于10万元的部分按10%提成，高于10万元的部分，
可可提成7.5%；20万到40万之间时，高于20万元的部分，
可提成5%；40万到60万之间时高于40万元的部分，可提成3%；
60万到100万之间时，高于60万元的部分，可提成1.5%，高于100万元时，超过100万元的部分按1%提成，
从键盘输入当月利润 I，求应发放奖金总数？
1.程序分析：请利用数轴来分界，定位。注意定义时需把奖金定义成长整型。
*/
/*注意： 要精确到小数点后多少位，用 DecimalFormat df = new DecimalFormat("#0.0000");
*
*/

```
package cn.com.flywater.FiftyAlgorthm;
import java.text.DecimalFormat;
import java.util.*;
public class TwelfthProfitAward {
static double profit = 0;
static double award = 0;
public static void main(String[] args) {
    Scanner s = new Scanner(System.in);
    profit = s.nextInt();
    System.out.println("输入的利润是" + profit + "万");
```

```
    if(profit > 0 && profit <= 10) {
     award = profit * 0.1;
    } else if(profit > 10 && profit <= 20) {
     award = 10 * 0.1 + (profit - 10) * 0.075;
    } else if(profit > 20 && profit <= 40) {
     award = 10 * 0.1 + 10 * 0.075 + (profit - 20) * 0.05;
    } else if(profit > 40 && profit <= 60) {
     award = 10 * 0.1 + 10 * 0.075 + 20 * 0.05 + (profit - 40) * 0.03;
    } else if(profit > 60 && profit <= 100) {
     award = 20 * 0.175 + 20 * 0.05 + 20 * 0.03 + (profit - 60) *
0.015;
    } else if(profit > 100) {
     award = 20 * 0.175 + 40 * 0.08 + 40 * 0.015 + (profit - 100) *
0.01;
    }
    DecimalFormat df = new DecimalFormat("#0.00000");

    System.out.println("应该提取的奖金是 " + df.format(award) + "万");
}
}
```

【程序 13】 完全平方数

题目：一个整数，它加上100后是一个完全平方数，再加上168又是一个完全平方数，请问该数是多少？
1.程序分析：在10万以内判断，先将该数加上100后再开方，再将该数加上268后再开方，
如果开方后的结果满足如下条件，即是结果。请看具体分析：

```
*/
package cn.com.flywater.FiftyAlgorthm;
public class ThirteenthTwiceSqrt {
public static void main(String[] args) {
    for(long l=1L; l<100000; l++) {
     if(Math.sqrt((long)(l+100)) % 1 == 0) {
      if(Math.sqrt((long)(l+268)) % 1 == 0) {
       System.out.println(l + "加100是一个完全平方数，再加168又是一个
完全平方数");
      }
     }
    }
}
}
```

【程序 14】 判断某一天是这一年的第几天

题目：输入某年某月某日，判断这一天是这一年的第几天？
1.程序分析：以3月5日为例，应该先把前两个月的加起来，
然后再加上5天即本年的第几天，特殊情况，闰年且输入月份大于3时需考虑多加一天。
*/

```
package cn.com.flywater.FiftyAlgorthm;
import java.util.Scanner;
import java.io.*;
public class FourteenthYearMonthDay {
public static void main(String[] args) {
    int year, month, day;
    int days = 0;
    int d = 0;

    FourteenthYearMonthDay fymd = new FourteenthYearMonthDay();

    System.out.print("Input the year:");
    year =fymd.input();
    System.out.print("Input the month:");
    month = fymd.input();
    System.out.print("Input The Day:");
    day = fymd.input();


    if (year < 0 || month < 0 || month > 12 || day < 0 || day > 31) {
     System.out.println("Input error, please run this program again!");
     System.exit(0);
    }
    for (int i=1; i <month; i++) {
     switch (i) {
     case 1:
     case 3:
     case 5:
     case 7:
     case 8:
     case 10:
     case 12:
      days = 31;
```

```
     //d += days;
     break;
    case 4:
    case 6:
    case 9:
    case 11:
     days = 30;
     //d += days;
     break;
    case 2:
     if ((year % 400 == 0) || (year % 4 == 0 && year % 100 != 0))
{
      days = 29;
     } else {
      days = 28;
     }
     //d += days;
     break;
    }

    d += days;

   }
   System.out.println(year + ":" + month + ":" + day + "是今年的第"
+ (d+day) + "天。");
}
public int input() {
   int value = 0;
   Scanner s = new Scanner(System.in);
   value = s.nextInt();
   return value;
}
}
```

【程序 15】三个数排序

题目：输入三个整数 x,y,z，请把这三个数由小到大输出。
1.程序分析：我们想办法把最小的数放到 x 上，先将 x 与 y 进行比较，如果 x> y 则将 x 与 y 的值进行交换，然后再用 x 与 z 进行比较，如果 x> z 则将 x 与 z 的值进行交换，这样能使 x 最小。
*/

```
package cn.com.flywater.FiftyAlgorthm;
```

```
import java.util.*;
public class FifteenthNumberCompare {
public static void main(String[] args) {
    FifteenthNumberCompare fnc = new FifteenthNumberCompare();
    int a, b, c;

    System.out.println("Input 3 numbers:");
    a = fnc.input();
    b = fnc.input();
    c = fnc.input();
//
//    fnc.compare(a, b);//方法调用不能通过改变形参的值来改变实参的值
//    fnc.compare(b, c);// 这种做法是错的
//    fnc.compare(a, c);
    //System.out.println("result:" + a +" " + b + " " + c);// 没有改变

    if(a > b) {
     int t = a;
     a = b;
     b = t;
    }

    if(a > c) {
     int t = a;
     a = c;
     c = t;
    }

    if(b > c) {
     int t = b;
     b = c;
     c = t;
    }
    System.out.println( a + " " + b + " " + c);
}
public int input() {
    int value = 0;
    Scanner s = new Scanner(System.in);
    value = s.nextInt();
    return value;
}
public void compare(int x, int y) {//此方法没用
    if(x > y) {
     int t = x;
```

```
    x = y;
    y = t;
   }
}
}
```

【程序 16】9*9 口诀

```
*题目：输出9*9口诀。
*1.程序分析：分行与列考虑，共9行9列，i 控制行，j 控制列。
**/
package cn.com.flywater.FiftyAlgorthm;
public class SixteenthMultiplicationTable {
public static void main(String[] args) {
   for(int i=1; i<10; i++) {
    for(int j=1; j<=i; j++) {
     System.out.print(j + "*" + i + "=" + j*i + " " );
    }
    System.out.println();
   }
}
}
```

【程序 17】猴子吃桃问题

```
//题目：猴子吃桃问题：猴子第一天摘下若干个桃子，当即吃了一半，还不瘾，
//又多吃了一个 第二天早上又将剩下的桃子吃掉一半，又多吃了一个。
//以后每天早上都吃了前一天剩下 的一半零一个。到第10天早上想再吃时，
//见只剩下一个桃子了。求第一天共摘了多少。
//1.程序分析：采取逆向思维的方法，从后往前推断。
package cn.com.flywater.FiftyAlgorthm;
public class SeventeenthMonkeyPeach {
public static void main(String[] args) {
   int lastdayNum = 1;
   for(int i=2; i<=10; i++) {
    lastdayNum = (lastdayNum+1) * 2;
   }
   System.out.println("猴子第一天摘了 " + lastdayNum + " 个桃子");
}
```

```
}
```

【程序 18】找出三队赛手的名单

题目：两个乒乓球队进行比赛，各出三人。甲队为 a,b,c 三人，乙队为 x,y,z 三人。
已抽签决定比赛名单。有人向队员打听比赛的名单。a 说他不和 x 比，c 说他不和 x,z 比，请编程序找出三队赛手的名单。

```
*/
/*
* 这个程序写得很不好，是知道结果后拼凑起来的，还不如直接写输出语句加上结果来的好。
*/
package cn.com.flywater.FiftyAlgorthm;
public class EighteenthPingpong {
static char[] m = { 'a', 'b', 'c' };
static char[] n = { 'x', 'y', 'z' };
public static void main(String[] args) {
    for (int i = 0; i < m.length; i++) {
     for (int j = 0; j < n.length; j++) {
      if (m[i] == 'a' && n[j] == 'x') {
       continue;
      } else if (m[i] == 'a' && n[j] == 'y') {
       continue;
      } else if ((m[i] == 'c' && n[j] == 'x')
        || (m[i] == 'c' && n[j] == 'z')) {
       continue;
      } else if ((m[i] == 'b' && n[j] == 'z')
        || (m[i] == 'b' && n[j] == 'y')) {
       continue;
      } else
       System.out.println(m[i] + " vs " + n[j]);
     }
    }
}
}
```

【程序 19】打印出如下图案

题目：打印出如下图案（菱形）

```
   *
  ***
 *****
*******
 *****
  ***
   *
```

1.程序分析：先把图形分成两部分来看待，前四行一个规律，后三行一个规律，利用双重 for 循环，
第一层控制行，第二层控制列。
*/

```
/*上半部分循环变量的控制方法是
* for(int i=0; i<(HEIGHT+1) / 2; i++) {
    for(int j=1; j<WIDTH/2-i; j++) {
    for(int k=1; k<(i+1)*2; k++) {

     下半部分循环变量的控制方法是
for(int i=1; i<=HEIGHT/2; i++) {
    for(int j=1; j<=i; j++) {
*     for(int k=1; k<=WIDTH-2*i-1; k++) {
*/
package cn.com.flywater.FiftyAlgorthm;
public class NineteenthPrintRhombic {
static final int HEIGHT = 7;
static final int WIDTH = 8;
public static void main(String[] args) {
    for(int i=0; i<(HEIGHT+1) / 2; i++) {
     for(int j=1; j<WIDTH/2-i; j++) {
      System.out.print(" ");
     }
     for(int k=1; k<(i+1)*2; k++) {
      System.out.print('*');
     }
     System.out.println();
    }

    for(int i=1; i<=HEIGHT/2; i++) {
     for(int j=1; j<=i; j++) {
      System.out.print(" ");
     }
```

```
     for(int k=1; k<=WIDTH-2*i-1; k++) {
      System.out.print('*');
     }
     System.out.println();
    }
}
}
```

上半部分第二重循环应改为： for(int j=0; j<WIDTH/2-i; j++)
上半部分第三重循环应改为： for(int k=1; k<=WIDTH-2*i; k++)

【程序 20】求数列的前 20 项之和

题目：有一分数序列：2/1，3/2，5/3，8/5，13/8，21/13...求出这个数列的前20项之和。
1.程序分析：请抓住分子与分母的变化规律。

```
*/
package cn.com.flywater.FiftyAlgorthm;
import java.text.DecimalFormat;
public class TwentiethFractionSum {
public static void main(String[] args) {
    int x = 2, y = 1, t;
    double sum = 0;

    DecimalFormat df = new DecimalFormat("#0.0000");

    for(int i=1; i<=20; i++) {
     sum += (double)x / y;
     t = y;
     y = x;
     x = y + t;
     System.out.println("第 " + i + " 次相加，和是 " + df.format(sum));
    }
}
```

【程序 21】 求 1+2!+3!+...+20!的和

题目：求1+2!+3!+...+20!的和
1.程序分析：此程序只是把累加变成了累乘。

```
*/
package cn.com.flywater.FiftyAlgorthm;
public class Twenty_firstFactorialSum {
static long sum = 0;
static long fac = 0;
public static void main(String[] args) {
    long sum = 0;
    long fac = 1;
    for(int i=1; i<=10; i++) {
     fac = fac * i;
     sum += fac;
    }
    System.out.println(sum);
}
}
```

【程序 22】利用递归方法求 5!

题目：利用递归方法求5!。
1.程序分析：递归公式：fn=fn_1*4!

```
*/
package cn.com.flywater.FiftyAlgorthm;
import java.util.Scanner;
public class Twenty_secondFactorialRecursion {
public static void main(String[] args) {
    Scanner s = new Scanner(System.in);
    int n = s.nextInt();
    Twenty_secondFactorialRecursion tfr = new
Twenty_secondFactorialRecursion();
    System.out.println(tfr.recursion(n));
}
public long recursion(int n) {
    long value = 0 ;
    if(n ==1 || n == 0) {
     value = 1;
    } else if(n > 1) {
     value = n * recursion(n-1);
    }
    return value;
}
}
```

【程序 23】第五个人多大

题目：有5个人坐在一起，问第五个人多少岁？他说比第4个人大2岁。
问第4个人岁数，他说比第3个人大2岁。问第三个人，又说比第2人大两岁。
问第2个人，说比第一个人大两岁。最后问第一个人，他说是10岁。请问第五个人多大？
1.程序分析：利用递归的方法，递归分为回推和递推两个阶段。
要想知道第五个人岁数，需知道第四人的岁数，依次类推，推到第一人（10岁），再往回推。
**/

```
package cn.com.flywater.FiftyAlgorthm;
public class Twenty_thirdPeopleAge {
public static void main(String[] args) {
    int age = 10;

    for(int i=2; i<=5; i++) {
     age += 2;
    }
    System.out.println(age);
}
}
```

【程序 24】逆序打印

题目：给一个不多于5位的正整数，
要求：一、求它是几位数，二、逆序打印出各位数字。
说明： 这个算法实现虽然实现了这个功能，但不健壮，当输入字符是，会出现异常。
*/

```
package cn.com.flywater.FiftyAlgorthm;
import java.util.Scanner;
public class Twenty_fourthNumber {
public static void main(String[] args) {

    Twenty_fourthNumber tn = new Twenty_fourthNumber();
    Scanner s = new Scanner(System.in);
    long a = s.nextLong();
```

```
    if(a < 0 || a > 100000) {
     System.out.println("Error Input, please run this program Again");
     System.exit(0);
    }

    if(a >=0 && a <=9) {
     System.out.println( a + "是一位数");
     System.out.println("按逆序输出是" + '\n' + a);
    } else if(a >= 10 && a <= 99) {
     System.out.println(a + "是二位数");
     System.out.println("按逆序输出是" );
     tn.converse(a);
    } else if(a >= 100 && a <= 999) {
     System.out.println(a + "是三位数");
     System.out.println("按逆序输出是" );
     tn.converse(a);
    } else if(a >= 1000 && a <= 9999) {
     System.out.println(a + "是四位数");
     System.out.println("按逆序输出是" );
     tn.converse(a);
    } else if(a >= 10000 && a <= 99999) {
     System.out.println(a + "是五位数");
     System.out.println("按逆序输出是" );
     tn.converse(a);
    }
}
public void converse(long l) {
    String s = Long.toString(l);
    char[] ch = s.toCharArray();
    for(int i=ch.length-1; i>=0; i--) {
     System.out.print(ch[i]);

    }
}
}
```

这个算法实在太土了，也许只有我若水飞天才会这样写，
下面写一个优雅一点的

```
import java.util.Scanner;
public class Twenty_fourthNumber {
public static void main(String[] args) {

    Twenty_fourthNumber tn = new Twenty_fourthNumber();
    Scanner s = new Scanner(System.in);
    long a = s.nextLong();
```

```
        String s = Long.toString(l);
    char[] ch = s.toCharArray();
System.out.println(a + "是" + ch.length + “位数”);
    for(int i=ch.length-1; i>=0; i--) {
     System.out.print(ch[i]);
}
}
}
```

【程序 25】回文数

题目：一个5位数，判断它是不是回文数。即12321是回文数，个位与万位相同，十位与千位相同。

```
package cn.com.flywater.FiftyAlgorthm;
import java.util.Scanner;
public class Twenty_fifthPalindrom {
static int[] a = new int[5];
static int[] b = new int[5];
public static void main(String[] args) {

    boolean is =false;
    Scanner s = new Scanner(System.in);
    long l = s.nextLong();

    if (l > 99999 || l < 10000) {
     System.out.println("Input error, please input again!");
     l = s.nextLong();
    }

    for (int i = 4; i >= 0; i--) {
     a[i] = (int) (l / (long) Math.pow(10, i));
     l =(l % ( long) Math.pow(10, i));
    }
    System.out.println();
    for(int i=0,j=0; i<5; i++, j++) {
      b[j] = a[i];
    }


    for(int i=0,j=4; i<5; i++, j--) {
```

```
     if(a[i] != b[j]) {
      is = false;
      break;
     } else {
      is = true;
     }
    }
    if(is == false) {
     System.out.println("is not a Palindrom!");
    } else if(is == true) {
     System.out.println("is a Palindrom!");
    }
}
}
```

这个更好，不限位数

```
public static void main(String[] args) {
    Scanner s = new Scanner(System.in);
    System.out.print("请输入一个正整数：");
    long a = s.nextLong();
    String ss = Long.toString(a);
     char[] ch = ss.toCharArray();
     boolean is =true;
    int j=ch.length;
    for(int i=0; i<j/2; i++) {
    if(ch[i]!=ch[j-i-1]){is=false;}
    }
    if(is==true){System.out.println("这是一个回文数");}
      else {System.out.println("这不是一个回文数");}
     }
```

【程序 26】星期几

题目：请输入星期几的第一个字母来判断一下是星期几，
如果第一个字母一样，则继续 判断第二个字母。
1.程序分析：用情况语句比较好，如果第一个字母一样，
则判断用情况语句或 if 语句判断第二个字母。
此程序虽然实现了基本功能，但必须严格按照题目的要求输入，否则程序无法执行

```
**/
package cn.com.flywater.FiftyAlgorthm;
import java.util.Scanner;
public class Twenty_sixthWeek {
Scanner s = new Scanner(System.in);
```

```
public static void main(String[] args) {
    Twenty_sixthWeek tw = new Twenty_sixthWeek();
    char ch = tw.getChar();
    switch(ch) {
     case 'M':
      System.out.println("Monday");
      break;
     case 'W':
      System.out.println("Wednesday");
      break;
     case 'F':
      System.out.println("Friday");
      break;
     case 'T': {
      System.out.println("please input the second letter!");
      char ch2 = tw.getChar();
      if(ch2 == 'U') {System.out.println("Tuesday"); }
      else if(ch2 == 'H') {System.out.println("Thursday"); }

     };
      break;
     case 'S': {
      System.out.println("please input the scecond letter!");
      char ch2 = tw.getChar();
      if(ch2 == 'U') {System.out.println("Sunday"); }
      else if(ch2 == 'A') {System.out.println("Saturday"); }

     };
      break;
    }
}
public char getChar() {
    String str = s.nextLine();
    char ch = str.charAt(0);
    if(ch<'A' || ch>'Z') {
     System.out.println("Input error, please input a capital letter");
     getChar();
    }
    return ch;
}
}
```

【程序 27】素数

题目：求100之内的素数
1.程序分析：判断素数的方法：用一个数分别去除2到 sqrt(这个数)，
如果能被整除， 则表明此数不是素数，反之是素数。

```
package cn.com.flywater.FiftyAlgorthm;
public class Twenty_seventhPrimeNumber {
public static void main(String[] args) {
    boolean b =false;
    int count = 0;
    for(int i=2; i<100; i+=1) {
     for(int j=2; j<=Math.sqrt(i); j++) {
      if(i % j == 0) {
       b = false;
       break;
      } else{
       b = true;
      }
     }

     if(b == true) {
      count ++;
      System.out.print(i + " ");
     }
    }
    System.out.println('\n' + "The number of PrimeNumber is :" +
count);
}
}
```

【程序 28】排序

题目：对10个数进行排序
1.程序分析：可以利用选择法，即从后9个比较过程中，
选择一个最小的与第一个元素交换， 下次类推，
即用第二个元素与后8个进行比较，并进行交换。

```
**/
package cn.com.flywater.FiftyAlgorthm;
import java.util.Scanner;
```

```
public class Twehty_eighthNumberSort {
public static void main(String[] args) {
    Scanner s = new Scanner(System.in);
    int[] a = new int[10];
    for(int i=0; i<10; i++) {
     a[i] = s.nextInt();
    }
    for(int i=0; i<10; i++) {
     for(int j=i+1; j<10; j++) {
      if(a[i] > a[j]) {
       int t = a[i];
       a[i] = a[j];
       a[j] = t;
      }
     }
    }

    for(int i=0; i<10; i++) {
     System.out.print(a[i] + " ");
    }

}
}
```

【程序 29】求主对角线的和

题目：求一个3*3矩阵对角线元素之和
1.程序分析：利用双重 for 循环控制输入二维数组，再将 a[i][i]累加后输出。
**/

```
package cn.com.flywater.FiftyAlgorthm;
import java.util.Scanner;
public class Twenty_ninthCrossSum {
public static void main(String[] args) {
    Scanner s = new Scanner(System.in);
    int[][] a = new int[3][3];

    for(int i=0; i<3; i++) {
     for(int j=0; j<3; j++) {
      a[i][j] = s.nextInt();
     }
    }
```

```
    System.out.println("输入的3 * 3 矩阵是:");
    for(int i=0; i<3; i++) {
     for(int j=0; j<3; j++) {
      System.out.print(a[i][j] + " ");
     }
     System.out.println();
    }

    int sum = 0;
    for(int i=0; i<3; i++) {
     for(int j=0; j<3; j++) {
      if(i == j) {
       sum += a[i][j];
      }
     }
    }
    System.out.println("对角线和是 " + sum);
}
}
```

【程序 30】有序插入

题目：有一个已经排好序的数组。现输入一个数，要求按原来的规律将它插入数组中。

1. 程序分析：首先判断此数是否大于最后一个数，
然后再考虑插入中间的数的情况，插入后此元素之后的数，依次后移一个位置。

```
**/
package cn.com.flywater.FiftyAlgorthm;
import java.util.Scanner;
public class ThirtiethInsert {
public static void main(String[] args) {

    int[] a = new int[]{1, 2, 3, 4, 5, 6, 7};
    int[] b = new int[a.length+1];
    int t1 =0, t2 = 0;
    int i =0;
    Scanner s= new Scanner(System.in);
    int num = s.nextInt();

    /*
    * 定义两个数组 a，b，一个 a 的长度比另一个 b 大1， a 看做是
```

```
    * 已经排好序的。
    * 接下来的过程是
    * 1: 如果 num 比最后一个数大，把 num 赋值给数组 b 的最后一个数
    *     再按顺序把 a 的每个元素赋给 b
    * 2: 否则（num 不比 a 的最后一个数大），
    *      如果 a 的元素比 num 小，则将这些元素按顺序赋给 b
    *      将 num 赋给比 num 大的 b 数组的元素，
    *      跳出第一个 for 循环。
    * 3: 定义一个循环控制变量，从 num 传给数组后 num 的下标值加一开始；
    *     直到 b 的结尾，将剩下的 a 的值赋给 b,赋值的过程是 b[j] = a[i-1];
    *
    */

    if(num >= a[a.length-1]) {
     b[b.length-1] = num;
     for(i=0; i<a.length; i++) {
      b[i] = a[i];
     }
    } else {
     for(i=0; i<a.length; i++) {
      if(num >= a[i]) {
       b[i] = a[i];
      } else {
       b[i] = num;
       break;
      }
     }
     for(int j=i+1; j<b.length; j++) {
      b[j] = a[j-1];
     }
    }

    for (i = 0; i < b.length; i++) {
     System.out.print(b[i] + " ");
    }
}

}
```

【程序 31】求 1+2!+3!+...+20!的和

题目：求1+2!+3!+...+20!的和
1.程序分析：此程序只是把累加变成了累乘。
*/
```
package cn.com.flywater.FiftyAlgorthm;
public class Twenty_firstFactorialSum {
static long sum = 0;
static long fac = 0;
public static void main(String[] args) {
    long sum = 0;
    long fac = 1;
    for(int i=1; i<=10; i++) {
     fac = fac * i;
     sum += fac;
    }
    System.out.println(sum);
}
}
```

【程序 32】取一个整数的某一位

题目：取一个整数 a 从右端开始的4～7位。
程序分析：可以这样考虑：
(1)先使 a 右移4位。
(2)设置一个低4位全为1,其余全为0的数。可用~(~0 < <4)
(3)将上面二者进行&运算。
**/
```
/*这个题我不会做，如有高手路过，还望指点
*
*/
package cn.com.flywater.FiftyAlgorthm;
public class Thirty_secondFS {
public static void main(String[] args) {

}
}
```

我没有用提示的方法，采用了字串截取。

```
public static void main(String[] args) {
    Scanner s = new Scanner(System.in);
    boolean is =true;
    System.out.print("请输入一个7位以上的正整数：");
    long a = s.nextLong();
    String ss = Long.toString(a);
    char[] ch = ss.toCharArray();
    int j=ch.length;
    if (j<7){System.out.println("输入错误！");}
    else {
    System.out.println(" 截 取 从 右 端 开 始 的 4 ～ 7 位 是 ："
"+ch[j-7]+ch[j-6]+ch[j-5]+ch[j-4]);
    }
    }
```

【程序 33】杨辉三角形

题目：打印出杨辉三角形（要求打印出10行如下图）

1.程序分析：

```
         1
        1 1
       1 2 1
      1 3 3 1
     1 4 6 4 1
   1 5 10 10 5 1
*/
/*
* 网上千篇一律是这种写法，我也没有什么创新，
* a[i][j]=a[i-1][j]+a[i-1][j-1] 就是这个程序的核心
* 定义的是二维数组，为了使输出的结果看起来漂亮一点
* 可以用 for（int k=0; k<2*(10-i)-1; k++）控制输出的空格
* 这个循环是在控制行数的循环里面，控制列数的循环外面。
* 记得在输出菱形时为了控制下半部分的输出，在下拼命的写出
* for(int k=1; k<=WIDTH-2*i-1; k++) 才算了事。
*/
package cn.com.flywater.FiftyAlgorthm;
public class Thirty_thirdYangTriangle {
public static void main(String[] args) {

    int[][] a = new int[10][10];
```

```
    for(int i=0; i<10; i++) {
     a[i][i] = 1;
     a[i][0] = 1;
    }
    for(int i=2; i<10; i++) {
     for(int j=1; j<i; j++) {
      a[i][j] = a[i-1][j-1] + a[i-1][j];
     }
    }

    for(int i=0; i<10; i++) {
     for(int k=0; k<2*(10-i)-1; k++) {
      System.out.print(" ");
     }
     for(int j=0; j<=i; j++) {
      System.out.print(a[i][j] + "    ");
     }
     System.out.println();
    }
}
}
```

【程序 34】排序

题目：输入3个数 a,b,c，按大小顺序输出。
1.程序分析：利用指针方法。
*/

```
/*
* 可惜，Java 好像没有指针
*/
package cn.com.flywater.FiftyAlgorthm;
import java.util.Scanner;
public class Thirty_forthCompare {
public static void main(String[] args) {
    Scanner s = new Scanner(System.in);
    int a = s.nextInt();
    int b = s.nextInt();
    int c = s.nextInt();

    if(a < b) {
     int t = a;
```

```
    a = b;
    b = t;
   }

   if(a < c) {
    int t = a;
    a = c;
    c = t;
   }

   if(b < c) {
    int t = b;
    b = c;
    c = t;
   }

   System.out.println("从大到小的顺序输出:");
   System.out.println(a + " " + b + " " + c);
}

}
```

【程序 35】数组交换

题目：输入数组，最大的与第一个元素交换，最小的与最后一个元素交换，输出数组。

```
package cn.com.flywater.FiftyAlgorthm;
import java.util.Scanner;
public class Thirty_fifthSwop {
static final int N = 8;
public static void main(String[] args) {

   int[] a = new int [N];
   Scanner s = new Scanner(System.in);
   int index1 = 0, index2 = 0;

   System.out.println("please input numbers");
   for(int i=0; i<N; i++) {
    a[i] = s.nextInt();
    System.out.print(a[i] + " ");
   }
```

```
    int max =a[0], min = a[0];
    for(int i=0; i<a.length; i++) {
     if(a[i] > max) {
      max = a[i];
      index1 = i;
     }
     if(a[i] < min) {
      min = a[i];
      index2 = i;
     }
    }

    if(index1 != 0) {
     int temp = a[0];
     a[0] = a[index1];
     a[index1] = temp;
    }

    if(index2 != a.length-1) {
     int temp = a[a.length-1];
     a[a.length-1] = a[index2];
     a[index2] = temp;
    }
    System.out.println("after swop:");
    for(int i=0; i<a.length; i++) {
     System.out.print(a[i] + " ");
    }
}
}
```

【程序 36】位置交换

题目：有 n 个整数，使其前面各数顺序向后移 m 个位置，最后 m 个数变成最前面的 m 个数

```
**/
/*
* 这个题不知道有什么好办法，比较直接方法的是把这个数组分成两个数组，
* 再将两个数组合起来，但如果不控制好数组的下标，就会带来很多麻烦。
*/
package cn.com.flywater.FiftyAlgorthm;
```

```
import java.util.Scanner;
public class Thirty_sixthBackShift {
public static final int N =10;
public static void main(String[] args) {
    int[] a = new int[N];
    Scanner s = new Scanner(System.in);
    System.out.println("please input array a, ten numbers:");
    for(int i=0; i<a.length; i++) {
     a[i] = s.nextInt();
    }
    System.out.println("please input m , one number:");
    int m = s.nextInt();

    int[] b = new int[m];
    int[] c = new int[N-m];
    for(int i=0; i<m; i++) {
     b[i] = a[i];
    }

    for(int i=m,j=0; i<N; i++,j++) {
     c[j] = a[i];
    }

    for(int i=0; i<N-m; i++) {
     a[i] = c[i];
    }

    for(int i=m,j=0; i<N; i++,j++) {
     a[i] = b[j];
    }
    for(int i=0; i<a.length; i++) {
     System.out.print(a[i] + " ");
    }
}
}
```

【程序 37】围圈报数出列

题目：有 n 个人围成一圈，顺序排号。从第一个人开始报数（从1到3报数），凡报到3的人退出圈子，问最后留下的是原来第几号的那位。

```
**/
/*
```

```
* 这个程序是完全抄别人的
*/
package cn.com.flywater.FiftyAlgorthm;
import java.util.Scanner;
public class Thirty_sevenCount3Quit {
public static void main(String[] args) {
    Scanner s = new Scanner(System.in);
    int n = s.nextInt();

    boolean[] arr = new boolean[n];
    for(int i=0; i<arr.length; i++) {
     arr[i] = true;//下标为 TRUE 时说明还在圈里
    }

    int leftCount = n;
    int countNum = 0;
    int index = 0;

    while(leftCount > 1) {
     if(arr[index] == true) {//当在圈里时
      countNum ++; //报数递加
      if(countNum == 3) {//报道3时
       countNum =0;//从零开始继续报数
       arr[index] = false;//此人退出圈子
       leftCount --;//剩余人数减一
      }
     }
     index ++;//每报一次数，下标加一

     if(index == n) {//是循环数数，当下标大于 n 时，说明已经数了一圈，
      index = 0;//将下标设为零重新开始。
     }
    }

    for(int i=0; i<n; i++) {
     if(arr[i] == true) {
      System.out.println(i);
     }
    }
}
}
```

【程序 38】求字符串长度

题目：写一个函数，求一个字符串的长度，在 main 函数中输入字符串，并输出其长度。

```
package cn.com.flywater.FiftyAlgorthm;
public class Thirty_eighthStringLength {
public static void main(String[] args) {
    String s = "jdfifdfhfhuififffdfggee";
    System.out.print("字符串的长度是：");
    System.out.println(s.length());
}
}
```

【程序 39】求 1/2+1/4+...+1/n

题目：编写一个函数，输入 n 为偶数时，调用函数求1/2+1/4+...+1/n, 当输入 n 为奇数时，调用函数1/1+1/3+...+1/n(利用指针函数)

```
**/
package cn.com.flywater.FiftyAlgorthm;
import java.text.DecimalFormat;
import java.util.*;
public class Thirty_ninthFactionSum {
public static void main(String[] args) {
    Scanner s = new Scanner(System.in);
    int n = s.nextInt();
    DecimalFormat df = new DecimalFormat("#0.00000");

    System.out.println( n +" **** result " + df.format(sum(n)));

}
public static double sum(int n) {
    double result = 0;
    if(n % 2 == 0) {
     for(int i=2; i<=n; i+=2) {
      result += (double)1 / n;
     }
    } else {
     for(int i=1; i<=n; i+=2) {
      result += (double)1 / i ;
     }
    }
```

```
    return result;
}
}
```

【程序 40】字符串排序

题目：字符串排序。

```
public class lianxi40 {
public static void main(String[] args) {
    int N=5;
    String temp = null;
    String[] s = new String[N];
    s[0] = "matter";
    s[1] = "state";
    s[2] = "solid";
    s[3] = "liquid";
    s[4] = "gas";
    for(int i=0; i<N; i++) {
     for(int j=i+1; j<N; j++) {
      if(compare(s[i], s[j]) == false) {
       temp = s[i];
       s[i] = s[j];
       s[j] = temp;
      }
     }
    }
     for(int i=0; i<N; i++) {
     System.out.println(s[i]);
    }
}
static boolean compare(String s1, String s2) {
    boolean result = true;
    for(int i=0; i<s1.length() && i<s2.length(); i++) {
     if(s1.charAt(i) > s2.charAt(i)) {
      result = false;
      break;
     } else if(s1.charAt(i) <s2.charAt(i)) {
      result = true;
      break;
     } else {
      if(s1.length() < s2.length()) {
       result = true;
      } else {
```

```
      result = false;
     }
    }
   }
   return result;
}
}
```

【程序 41】猴子分桃

题目：海滩上有一堆桃子，五只猴子来分。第一只猴子把这堆桃子凭据分为五份，多了一个，这只猴子把多的一个扔入海中，拿走了一份。第二只猴子把剩下的桃子又平均分成五份，又多了一个，它同样把多的一个扔入海中，拿走了一份，第三、第四、第五只猴子都是这样做的，问海滩上原来最少有多少个桃子？

```
public class lianxi41 {
public static void main (String[] args) {
int i,m,j=0,k,count;
for(i=4;i<10000;i+=4)
   { count=0;
     m=i;
     for(k=0;k<5;k++)
        {
         j=i/4*5+1;
         i=j;
         if(j%4==0)
            count++;
            else break;
       }
    i=m;
if(count==4)
{System.out.println("原有桃子 "+j+" 个");
break;}
}
}
}
```

【程序 42】809*??=800*??+9*??+1

题目：809*??=800*??+9*??+1 其中??代表的两位数,8*??的结果为两位数，9*??的结果为 3 位数。求??代表的两位数，及 809*??后的结果。

```
//题目错了！809x=800x+9x+1 这样的方程无解。去掉那个 1 就有解了。
public class lianxi42 {
public static void main (String[] args) {
int a=809,b,i;
for(i=10;i<13;i++)
{b=i*a ;
if(8*i<100&&9*i>=100)
System.out.println ("809*"+i+"="+"800*"+i+"+"+"9*"+i+"="+b);}
}
}
```

【程序 43】求 0—7 所能组成的奇数个数

题目：求 0—7 所能组成的奇数个数。

```
//组成 1 位数是 4 个。
//组成 2 位数是 7*4 个。
//组成 3 位数是 7*8*4 个。
//组成 4 位数是 7*8*8*4 个。
//......
public class lianxi43 {
public static void main (String[] args) {
int sum=4;
int j;
System.out.println("组成 1 位数是 "+sum+" 个");
sum=sum*7;
System.out.println("组成 2 位数是 "+sum+" 个");
for(j=3;j<=9;j++){
sum=sum*8;
System.out.println("组成"+j+"位数是 "+sum+" 个");
}
}
}
```

【程序 44】一个偶数总能表示为两个素数之和

题目：一个偶数总能表示为两个素数之和。

```
//由于用除 sqrt(n)的方法求出的素数不包括 2 和 3，
//因此在判断是否是素数程序中人为添加了一个 3。
import java.util.*;
public class lianxi44 {
public static void main(String[] args) {
Scanner s = new Scanner(System.in);
int n,i;
do{
     System.out.print("请输入一个大于等于 6 的偶数：");
     n = s.nextInt();
    } while(n<6||n%2!=0);    //判断输入是否是>=6 偶数,不是,重新输入
fun fc = new fun();
    for(i=2;i<=n/2;i++){
    if((fc.fun(i))==1&&(fc.fun(n-i)==1))
    {int j=n-i;
     System.out.println(n+" = "+i+" + "+j);
     } //输出所有可能的素数对
   }
}
}
class fun{
public int fun (int a)    //判断是否是素数的函数
{
int i,flag=0;
if(a==3){flag=1;return(flag);}
for(i=2;i<=Math.sqrt(a);i++){
   if(a%i==0) {flag=0;break;}
      else flag=1;}
return (flag) ;//不是素数,返回 0,是素数,返回 1
}
}
//解法二
import java.util.*;
public class lianxi44 {
public static void main(String[] args) {
Scanner s = new Scanner(System.in);
int n;
do{
     System.out.print("请输入一个大于等于 6 的偶数：");
```

```
     n = s.nextInt();
    } while(n<6||n%2!=0);    //判断输入是否是>=6 偶数,不是,重新输入

   for(int i=3;i<=n/2;i+=2){
    if(fun(i)&&fun(n-i)) {
      System.out.println(n+" = "+i+" + "+(n-i));
      } //输出所有可能的素数对
   }
}
static boolean fun (int a){    //判断是否是素数的函数
boolean flag=false;
if(a==3){flag=true;return(flag);}
for(int i=2;i<=Math.sqrt(a);i++){
   if(a%i==0) {flag=false;break;}
      else flag=true;}
return (flag) ;
}
}
```

【程序 45】判断一个素数能被几个 9 整除

题目：判断一个素数能被几个 9 整除

```
//题目错了吧？能被 9 整除的就不是素数了！所以改成整数了。
import java.util.*;
public class lianxi45 {
public static void main (String[] args) {
   Scanner s = new Scanner(System.in);
   System.out.print("请输入一个整数：");
    int num = s.nextInt();
    int   tmp = num;
    int count = 0;
       for(int i = 0 ; tmp%9 == 0 ;){
           tmp = tmp/9;
            count ++;
          }
     System.out.println(num+" 能够被 "+count+" 个 9 整除。");
     }
}
```

【程序 46】两个字符串连接程序

题目：两个字符串连接程序

```
import java.util.*;
public class lianxi46 {
public static void main(String[] args) {
    Scanner s = new Scanner(System.in);
    System.out.print("请输入一个字符串：");
    String str1 = s.nextLine();
    System.out.print("请再输入一个字符串：");
    String str2 = s.nextLine();
    String str = str1+str2;
    System.out.println("连接后的字符串是："+str);
    }
    }
```

【程序 47】打印*

题目：读取7个数（1—50）的整数值，每读取一个值，程序打印出该值个数的＊。

```
import java.util.*;
public class lianxi47 {
public static void main(String[] args) {
Scanner s = new Scanner(System.in);
int n=1,num;
while(n<=7){
        do{
         System.out.print("请输入一个1--50之间的整数：");
           num= s.nextInt();
         }while(num<1||num>50);
     for(int i=1;i<=num;i++)
     {System.out.print("*");
     }
System.out.println();
n ++;
}
}
}
```

【程序 48】加密计算

题目：某个公司采用公用电话传递数据，数据是四位的整数，在传递过程中是加密的，加密规则如下：每位数字都加上 5,然后用和除以 10 的余数代替该数字，再将第一位和第四位交换，第二位和第三位交换。

```
import java.util.*;
public class lianxi48   {
public static void main(String args[]) {
Scanner s = new Scanner(System.in);
int num=0,temp;
do{
   System.out.print("请输入一个 4 位正整数：");
      num = s.nextInt();
     }while (num<1000||num>9999);
int a[]=new int[4];
a[0] = num/1000; //取千位的数字
a[1] = (num/100)%10; //取百位的数字
a[2] = (num/10)%10; //取十位的数字
a[3] = num%10; //取个位的数字
for(int j=0;j<4;j++)
{
a[j]+=5;
a[j]%=10;
}
for(int j=0;j<=1;j++)
    {
    temp = a[j];
    a[j] = a[3-j];
    a[3-j] =temp;
    }
System.out.print("加密后的数字为：");
for(int j=0;j<4;j++)
System.out.print(a[j]);
}
}
```

【程序 49】算字符串中子串出现的次数

题目：计算字符串中子串出现的次数

```
import java.util.*;
public class lianxi49 {
public static void main(String args[]){
Scanner s = new Scanner(System.in);
    System.out.print("请输入字符串：");
    String str1 = s.nextLine();
    System.out.print("请输入子串：");
    String str2 = s.nextLine();
int count=0;
if(str1.equals("")||str2.equals(""))
   {
   System.out.println("你没有输入字符串或子串,无法比较!");
   System.exit(0);
   }
else
   {
    for(int i=0;i<=str1.length()-str2.length();i++)
     {
     if(str2.equals(str1.substring(i, str2.length()+i)))
      //这种比法有问题，会把"aaa"看成有 2 个"aa"子串。
       count++;
       }
System.out.println("子串在字符串中出现: "+count+" 次");
}
}
}
```

【程序 50】计算平均成绩

题目：有五个学生，每个学生有 3 门课的成绩，从键盘输入以上数据（包括学生号，姓名，三门课成绩），计算出平均成绩，把原有的数据和计算出的平均分数存放在磁盘文件 "stud "中。

```
import java.io.*;
import java.util.*;
public class lianxi50 {
public static void main(String[] args){
```

```
   Scanner ss = new Scanner(System.in);
   String [][] a = new String[5][6];
   for(int i=1; i<6; i++) {
    System.out.print("请输入第"+i+"个学生的学号：");
    a[i-1][0] = ss.nextLine();
    System.out.print("请输入第"+i+"个学生的姓名：");
    a[i-1][1] = ss.nextLine();
    for(int j=1; j<4; j++) {
       System.out.print("请输入该学生的第"+j+"个成绩：");
       a[i-1][j+1] = ss.nextLine();
       }
System.out.println("\n");
   }
//以下计算平均分
float avg;
int sum;
for(int i=0; i<5; i++) {
sum=0;
   for(int j=2; j<5; j++) {
   sum=sum+ Integer.parseInt(a[i][j]);
      }
   avg= (float)sum/3;
   a[i][5]=String.valueOf(avg);
}
//以下写磁盘文件
String s1;
try {
    File f = new File("C:\\stud");
    if(f.exists()){
      System.out.println("文件存在");
      }else{
         System.out.println("文件不存在，正在创建文件");
          f.createNewFile();//不存在则创建
        }
BufferedWriter output = new BufferedWriter(new FileWriter(f));
for(int i=0; i<5; i++) {
for(int j=0; j<6; j++) {
   s1=a[i][j]+"\r\n";
   output.write(s1);
    }
}
output.close();
System.out.println("数据已写入 c 盘文件 stud 中！");
   } catch (Exception e) {
```

```
      e.printStackTrace();
      }
}
}
```

二 面向对象程序练习

【程序 1】protected 方法和友好方法的区别

举例说明 protected 方法和友好方法的区别。

如果子类和父类不在同一个包中,那么,子类继承了父类的 protected、public 成员变量做为子类的成员变量，并且继承了父类的 protected、public 方法为子类的方法。如果子类和父类不在同一个包里，子类不能继承父类的友好变量和友好方法。

下面的例子中，Father 和 Jerry 分别隶属不同的包。

Father.java:

```
package tom.jiafei;
public class  Father
{   int  height;
   protected int money;
   public   int weight;
   public Father(int m) {
   {  money=m;
   }
   protected int getMoney()
   {  return money;
   }
    void setMoney(int newMoney)
   {  money=newMoney;
   }
}
```

Jerry.java:

```
package sun.com;
import tom.jiafei.Father;
public class Jerry extends Father   //Jerry 和 Father 在不同的包中.
```

```
    {  public Jerry()
       {   super(20);
       }
       public static void main(String args[])
       {  Jerry  jerry=new Jerry();
          jerry.height=12;           //非法，因为 Jerry 没有继承友好的
height。
          jerry.weight=200;         //合法。
          jerry.money=800;          //合法。
          int m=jerry.getMoney(); //合法。.
          jerry.setMoney(300);     //非法，因为 Jerry 没有继承友好的方法
setMoney。
          System.out.println("m="+m);
       }
}
```

【程序 2】饲养员给动物喂食物

做一个饲养员给动物喂食物的例子体现 JAVA 中的面向对象思想,接口（抽象类）的用处

```
  package com.softeem.demo;

/**
*@authorleno
*动物的接口
*/
interface Animal
{
   publicvoid eat(Food food);
}
/**
*@authorleno
*一种动物类:猫
*/
class Cat implements Animal
{
   publicvoid eat(Food food)
   {
     System.out.println("小猫吃"+food.getName());
   }
}
/**
```

```
*@authorleno
*一种动物类:狗
*/
class Dog implements Animal
{
    publicvoid eat(Food food)
    {
      System.out.println("小狗啃"+food.getName());
    }
}

/**
*@authorleno
*食物抽象类
*/
abstractclass Food
{
    protected String name;
    public String getName() {
      returnname;
    }

    publicvoid setName(String name) {
      this.name = name;
    }
}

/**
*@authorleno
*一种食物类:鱼
*/
class Fish extends Food
{
    public Fish(String name) {
      this.name = name;
    }
}
/**
*@authorleno
*一种食物类:骨头
*/
class Bone extends Food
{
    public Bone(String name) {
```

```
      this.name = name;
    }
}

/**
*@authorleno
*饲养员类
*
*/
class Feeder
{
    /**
    *饲养员给某种动物喂某种食物
    *@paramanimal
    *@paramfood
    */
    publicvoid feed(Animal animal,Food food)
    {
      animal.eat(food);
    }
}

/**
*@authorleno
*测试饲养员给动物喂食物
*/
publicclass TestFeeder {

    publicstaticvoid main(String[] args) {
      Feeder feeder=new Feeder();
      Animal animal=new Dog();
      Food food=new Bone("肉骨头");
      feeder.feed(animal,food); //给狗喂肉骨头
      animal=new Cat();
      food=new Fish("鱼");
      feeder.feed(animal,food); //给猫喂鱼


    }
}
```

【程序 3】类方法和实例方法

举例说明类方法和实例方法以及类变量和实例变量的区别。

类变量和实例变量的区别：

类体的定义包括成员变量的定义和方法的定义，并且成员变量又分为实例变量和类变量，用 static 修饰的变量是类变量。那么类变量和实例变量有什么区别呢？

我们已经知道：一个类通过使用 new 运算符可以创建多个不同的对象，这些对象将被分配不同的内存空间，说得准确些就是：不同的对象的实例变量将被分配不同的内存空间，如果类中的成员变量有类变量，那么所有对象的这个类变量都分配给相同的一处内存，改变其中一个对象的这个类变量会影响其它对象的这个类变量。也就是说对象共享类变量。

我们知道，当 Java 程序执行时，类的字节码文件被加载到内存，如果该类没有创建对象，类的实例成员变量不会被分配内存。但是，类中的类变量，在该类被加载到内存时，就分配了相应的内存空间。如果该类创建对象，那么不同对象的实例变量互不相同，即分配不同的内存空间，而类变量不再重新分配内存，所有的对象共享类变量，即所有的对象的类变量是相同的一处内存空间，类变量的内存空间直到程序退出运行，才释放所占有的内存。Java 语言允许通过类名直接访问类变量。

下面例子中的梯形对象共享一个下底。

例子

```
class 梯形
{
    float 上底,高;
    static float 下底;                    //类变量。
    梯形(float 上底,float 高)
    {
        this.上底=上底;
        this.高=高;
    }
    float 获取上底()
    {
        return 上底;
    }
    float 获取下底()
    {
        return 下底;
    }
```

```
}
class Example4_5
{
    public static void main(String args[])
    {
        梯形  laderOne,laderTwo;                    //梯形的字节码被加载到内存。
        梯形.下底=60;                               //通过类名操作类变量。
        laderOne=new  梯形(18.0f,20);
        laderTwo=new  梯形(9.0f,10);
        System.out.println("laderOne 的上底:"+laderOne.获取上底());
        System.out.println("laderOne 的下底:"+laderOne.获取下底());
        System.out.println("laderTwo 的上底:"+laderTwo.获取上底());
        System.out.println("laderTwo 的下底:"+laderTwo.获取下底());
    }
}
```

类方法和实例方法区别：

我们已经知道类体中的方法分为实例方法和类方法两种，用 static 修饰的是类方法。二者有什么区别呢？当一个类创建了一个对象后，这个对象就可以调用该类的方法。

当类的字节码文件被加载到内存时，类的实例方法不会被分配入口地址，当该类创建对象后，类中的实例方法才分配入口地址，从而实例方法可以被类创建的任何对象调用执行。需要注意的是，当我们创建第一个对象时，类中的实例方法就分配了入口地址，当再创建对象时，不再分配入口地址，也就是说，方法的入口地址被所有的对象共享，当所有的对象都不存在时，方法的入口地址才被取消。

对于类中的类方法，在该类被加载到内存时，就分配了相应的入口地址。从而类方法不仅可以被类创建的任何对象调用执行，也可以直接通过类名调用。类方法的入口地址直到程序退出才被取消。

类方法在类的字节码加载到内存时就分配了入口地址，因此，Java 语言允许通过类名直接调用类方法，而实例方法不能通过类名调用。在讲述类的时候我们强调过，在 Java 语言中，类中的类方法不可以操作实例变量，也不可以调用实例方法，这是因为在类创建对象之前，实例成员变量还没有分配内存，而且实例方法也没有入口地址。

```
class A
{     int x,y;
     static float f(int a){}
     float g(int x1,int x2){}
```

```
}
class B
{   public static void main(String args[])
    {  A a1=new A();
       A.f(2,3);          //合法。
       a1.f(2,4);         //合法。
       a1.g(2,5);        //合法。
       A.g(3,2);        //非法。
    }
}
```

三 组件练习

【程序 1】文本区事件编程

编写有两个文本区的小应用程序。当我们在一个文本区中输入若干个数时，另一个文本区同时对你输入的数进行求和运算并求出平均值，也就是说随着你输入的变化，另一个文本区不断地更新求和及平均值。

```
import java.util.*;
import java.applet.*;
import java.awt.*;
import java.awt.event.*;
public class MyFrame extends Frame implements TextListener
{
      TextArea text1,text2;
      int count=1;
      double sum=0,aver=0;
      public MyFrame()
      {
         setLayout(new FlowLayout());
         text1=new TextArea(6,20);
         text2=new TextArea(6,20);
         add(text1);
         add(text2);
         text2.setEditable(false);
         text1.addTextListener(this) ;
     }
  public void textValueChanged(TextEvent e)
     {
```

```
        String s=text1.getText();
          sum=0;
          aver=0;
        StringTokenizer fenxi=new StringTokenizer(s," ,'\n'");//空格、回车和逗号做分隔符。
        int n=fenxi.countTokens();
        count=n;
        double a[]=new double[n];
        for(int i=0;i<=n-1;i++)
        {
            String temp=fenxi.nextToken();   //从文本区中取出数据。
            try
                {
                  a[i]=Double.parseDouble(temp);
                  sum=sum+a[i];
                }
            catch(Exception ee)
                {
                  count--;
                }
        }
        aver=sum/count;
        text2.setText(null); //刷新显示。
        text2.append("\n 和:"+sum);
        text2.append("\n 平均值:"+aver);
    }
   public static void main(String args[])
    {
          MyFrame f=new MyFrame();
          f.setBounds(12,12,300,300);
          f.setVisible(true);
          f.validate();
          f. addWindowListener(new WindowAdapter()
                                {
                                      public void windowClosing(WindowEvent e)
                                        {   System.exit(0);
                                        }
                                }
                              );
     }
}
```

【程序 2】简单算术计算

编写一个小应用程序，设计四个按钮，分别命名为“加”、“差”、“积、”、“除”。有三个文本框。单击相应的按钮，将两个文本框的数字做运算，在第三个文本框中显示结果。要求处理 NumberFormatException.

```
import javax.swing.*;
import java.awt.*;
import javax.swing.border.*;
import java.awt.event.*;
class WindowBox extends JFrame    implements ActionListener
{
      Box boxH1,boxH2,boxH3,boxV1,boxV2,baseBox;
      JTextField inputText1,inputText2,resultText;
      JButton buttonAdd,buttonSub,buttonMul,buttonDiv;
      WindowBox()
      {
        inputText1=new JTextField();
        inputText1.setHorizontalAlignment(JTextField.CENTER);
        inputText2=new JTextField();
        inputText2.setHorizontalAlignment(JTextField.CENTER);
        resultText=new JTextField();
        resultText.setHorizontalAlignment(JTextField.CENTER);
        buttonAdd=new JButton("加");
        buttonSub=new JButton("减");
        buttonMul=new JButton("乘");
        buttonDiv=new JButton("除");
        buttonAdd.addActionListener(this);
        buttonSub.addActionListener(this);
        buttonMul.addActionListener(this);
        buttonDiv.addActionListener(this);
        boxV1=Box.createVerticalBox();
        boxV1.add(new JLabel("输入运算数 1"));
        boxV1.add(Box.createVerticalStrut(8));
        boxV1.add(new JLabel("输入运算数 2"));
        boxV1.add(Box.createVerticalStrut(8));
        boxV2=Box.createVerticalBox();
        boxV2.add(inputText1);
```

```
    boxV2.add(Box.createVerticalStrut(8));
    boxV2.add(inputText2);
    boxH1=Box.createHorizontalBox();
    boxH1.add(boxV1);
    boxH1.add(boxV2);
    boxH2=Box.createHorizontalBox();
    boxH2.add(buttonAdd);
    boxH2.add(buttonSub);
    boxH2.add(buttonMul);
    boxH2.add(buttonDiv);
    boxH2.add(resultText);
    boxH3=Box.createHorizontalBox();
    boxH3.add(new JLabel("运算的结果:"));
    boxH3.add(resultText);
    baseBox=Box.createVerticalBox();
    baseBox.add(boxH1);
    baseBox.add(boxH2);
    baseBox.add(boxH3);
    Container con=getContentPane();
    con.setLayout(new FlowLayout());
    con.add(baseBox);
    con.validate();
    setBounds(120,125,200,200);
    setVisible(true);
    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
}
public void actionPerformed(ActionEvent e)//接口方法的实现.
{
    double n;
    if(e.getSource()==buttonAdd)
    {
       double n1,n2;
       try{
            n1=Double.parseDouble(inputText1.getText());
            n2=Double.parseDouble(inputText2.getText());
            n=n1+n2;
            resultText.setText(String.valueOf(n));
           }
       catch(NumberFormatException ee)
           {
             resultText.setText("请输入数字字符");
           }
  }
else if(e.getSource()==buttonSub)
```

```
        {
            double n1,n2;
             try{
                   n1=Double.parseDouble(inputText1.getText());
                   n2=Double.parseDouble(inputText2.getText());
                   n=n1-n2;
                   resultText.setText(String.valueOf(n));
                  }
             catch(NumberFormatException ee)
                  {
                    resultText.setText("请输入数字字符");
                  }
         }
         else if(e.getSource()==buttonMul)
          {
           double n1,n2;
             try{
                   n1=Double.parseDouble(inputText1.getText());
                   n2=Double.parseDouble(inputText2.getText());
                   n=n1*n2;
                   resultText.setText(String.valueOf(n));
                  }
             catch(NumberFormatException ee)
                  {
                    resultText.setText("请输入数字字符");
                  }
          }
          else if(e.getSource()==buttonDiv)
          {
           double n1,n2;
             try{
                   n1=Double.parseDouble(inputText1.getText());
                   n2=Double.parseDouble(inputText2.getText());
                   n=n1/n2;
                   resultText.setText(String.valueOf(n));
                  }
             catch(NumberFormatException ee)
                  {
                    resultText.setText("请输入数字字符");
                  }
          }
    }
}
public class Example
```

```
{
    public static void main(String args[])
    {
       new WindowBox();
    }
}
```

【程序 3】List 组件练习

编写应用程序，有一个窗口对象，该窗口取它的默认布局：BorderLayout 布局，北面添加一个 List 组件，该组件有四个商品名称的选项。中心添加一个文本区，当选择 List 组件中的某个选项后，文本区显示对该商品的价格和产地；当用鼠标双击 List 组件中的某个选项后，文本区显示该商品的明细。

```
import java.applet.*;import java.awt.*;import java.awt.event.*;
public class MyFrame extends Frame implements ItemListener,ActionListener
{
    List list;
    TextArea text1,text2;
    Panel p;
    CardLayout card;
    public MyFrame()
    {
        list=new List(2,false);
        text1=new TextArea();
        text2=new TextArea();
        p=new Panel();
        p.setLayout(new GridLayout(1,2));
        p.add(text1);
        p.add(text2);
        list.add("TCL 王牌彩电");
        list.add("黑妹牙膏");
        list.add("大显手机");
        list.add("联想 5000");
        add(list,BorderLayout.NORTH);
        add(p,BorderLayout.CENTER);
        list.addItemListener(this);
        list.addActionListener(this);
    }
    public void itemStateChanged(ItemEvent e)
    {
          if(list.getSelectedIndex()==0)
           {
              text1.setText("\n"+list.getSelectedItem()+"产品价格和产地介绍
```

```
");
            }
            else if(list.getSelectedIndex()==1)
            {
                text1.setText("\n"+list.getSelectedItem()+"产品价格和产地介绍
");
            }
            else if(list.getSelectedIndex()==2)
            {
                text1.setText("\n"+list.getSelectedItem()+"产品价格和产地介绍
");
            }
            else if(list.getSelectedIndex()==3)
            {
                text1.setText("\n"+list.getSelectedItem()+"产品价格和产地介绍
");
            }
    }
   public void actionPerformed(ActionEvent e)
    {
            if(list.getSelectedIndex()==0)
            {
               text2.setText("\n"+list.getSelectedItem()+"产品明细:");
            }
            else if(list.getSelectedIndex()==1)
            {
               text2.setText("\n"+list.getSelectedItem()+"产品明细:");
            }
            else if(list.getSelectedIndex()==2)
            {
               text2.setText("\n"+list.getSelectedItem()+"产品明细:");
            }
            else if(list.getSelectedIndex()==3)
            {
                text2.setText("\n"+list.getSelectedItem()+"产品明细:");
            }
        }
    public static void main(String args[])
    {
        MyFrame f=new MyFrame();
        f.setBounds(12,12,300,300);
        f.setVisible(true);
        f.validate();
        f. addWindowListener(new WindowAdapter()
```

```
                {
                    public void windowClosing(WindowEvent e)
                    {   System.exit(0);
                    }
                }
            );
    }
}
```

【程序 4】模式对话框练习

编写一个应用程序，用户可以在一个文本框里输入数字字符，按回车后将数字放入一个文本区。当输入的数字大于 1000 时，弹出一个有模式的对话框，提示用户数字已经大于 1000，是否继续将该数字放入文本区。

```
import java.awt.event.*;
import java.awt.*;
import javax.swing.*;
class Dwindow extends Frame implements ActionListener
{
    JTextField inputNumber;
    JTextArea  save;
    Dwindow(String s)
    {
      super(s);
      inputNumber=new JTextField(22);
      inputNumber.addActionListener(this);
      save=new JTextArea(12,16);
      setLayout(new FlowLayout());
      add(inputNumber);
      add(new JScrollPane(save));
      setBounds(60,60,300,300);
      setVisible(true);
      validate();
      addWindowListener(new WindowAdapter()
                {
                    public void windowClosing(WindowEvent e)
                    {   System.exit(0);
                    }
                }
            );
    }
    public void actionPerformed(ActionEvent event)
```

```
    {
        String s=inputNumber.getText();
        double n=0;
        try{
              n=Double.parseDouble(s);
              if(n>1000)
                {
                    int select=JOptionPane.showConfirmDialog(this,"确认正确
吗？","确认对话框",

JOptionPane.YES_NO_OPTION );
                    if(select==JOptionPane.YES_OPTION)
                      {
                          save.append("\n"+s);
                      }
                    else
                      {
                          inputNumber.setText(null);
                      }
                }
              else
                {
                    save.append("\n"+s);
                }
            }
        catch(NumberFormatException e)
            {
                JOptionPane.showMessageDialog(this,"您输入了非法字符","
警告对话框",

JOptionPane.WARNING_MESSAGE);
                inputNumber.setText(null);
            }
    }
}
public class Example
{
    public static void main(String args[])
     {
        new Dwindow("带对话框的窗口");
     }
}
```

四 小应用程序练习

【程序 1】小应用程序练习

编写一个小应用程序，该小应用程序有两个按钮组件和一个窗口。单击其中一个按钮打开窗口、单击另一个按钮关闭窗口。

```
import java.applet.*;
import java.awt.*;
import java.awt.event.*;
class Yourwindow extends Frame
{
   Yourwindow(String s, int a,int b)
   { super(s);
      setLayout(new GridLayout(1,1));
      setSize(a,b);
      setBackground(Color.white);
      setVisible(false);
   }
}
public class Example    extends Applet implements ActionListener
{ Yourwindow window1,window2 ;
   Button button1,button2,button3,button4;
   public void init()
   { button1=new Button("开南窗");
      button2=new Button("开北窗");
      button3=new Button("关南窗");
      button4=new Button("关北窗");
      window1=new Yourwindow("阳光之窗",60,60);
      window2=new Yourwindow("冰雪之窗",70,70);
      button1.addActionListener(this);
      button2.addActionListener(this);
      button3.addActionListener(this);
      button4.addActionListener(this);
      add(button1);
      add(button2);
      add(button3);
      add(button4);
  }
public void actionPerformed(ActionEvent e)
  { if(e.getSource()==button1)
```

```
            { window1.setVisible(true);
            }
        else if(e.getSource()==button3)
            { window1.setVisible(false);
            }
        else if(e.getSource()==button2)
            { window2.setVisible(true);
            }
        else if(e.getSource()==button4)
            { window2.setVisible(false);
            }
      }
}
```

【程序 2】小应用程序布局与事件练习

编写一个小应用程序，该小应用程序中两个 Panel 对象 p1，p2 和两个按钮对象 b1,b2。将 p1 的布局设置为 CardLayout 布局,小应用程序设置为 BorderLyout 布局。p2 添加 b1 和 b2，p1 添加 10 个不同名字的标签，然后将 p1、p2 分别添加到小应用程序的“中心”和“南边”。让小程序作为 b1、 b2 的 ActionEvent 事件监视器，通过单击按钮 b1、b2 实现往后或向前观察 p1 中的标签。

```
import java.applet.*;
import java.awt.*;
import java.awt.event.*;
public class Boy extends Applet implements ActionListener
{
    Panel p1,p2;
    CardLayout card;
    Button button1,button2;
    public void init()
    {
       setLayout(new BorderLayout());
       card=new CardLayout();
       p1=new Panel();
       p1.setLayout(card);
       p1.setBackground(Color.yellow);
       for(int i=1;i<=10;i++)
       {
          p1.add(""+i,new Label("我是第"+i+"个标签",Label.CENTER));
       }
       p2=new Panel();
```

```
        button1=new Button("previous");
        button2=new Button("next");
        p2.add(button1);
        p2.add(button2);
        add(p1,BorderLayout.CENTER);
        add(p2,BorderLayout.NORTH);
        button1.addActionListener(this);
        button2.addActionListener(this);
    }
    public void actionPerformed(ActionEvent e)
    {
        if(e.getSource()==button1)
        {
            card.previous(p1);
        }
        else if(e.getSource()==button2)
        {
            card.next(p1);
        }
    }
}
```

【程序 3】绘图练习（绘制五角星）

编写一个应用程序，绘制五角形，并打印出来。

```
import java.applet.*;
import java.awt.*;
import java.awt.event.*;
class MyCanvas extends Canvas
{
    static int pointX[]=new int[5],
           pointY[]=new int[5];

    public void paint(Graphics g)
    {
        g.translate(200,200) ;//进行坐标变换,将新的坐标原点设置为(200,200)。
        pointX[0]=0;
        pointY[0]=-120;
        double arcAngle=(72*Math.PI)/180;
        for(int i=1;i<5;i++)
        {
```

```
pointX[i]=(int)(pointX[i-1]*Math.cos(arcAngle)-pointY[i-1]*Math.sin(arcAngle));

pointY[i]=(int)(pointY[i-1]*Math.cos(arcAngle)+pointX[i-1]*Math.sin(arcAngle));
            }
            g.setColor(Color.red);
            int
starX[]={pointX[0],pointX[2],pointX[4],pointX[1],pointX[3],pointX[0]};
            int
starY[]={pointY[0],pointY[2],pointY[4],pointY[1],pointY[3],pointY[0]};
            g.drawPolygon(starX,starY,6);
        }
      public static void main(String args[])
        {
            Frame f=new Frame();
            f.setSize(500,450);
            f.setVisible(true);
            MyCanvas canvas=new MyCanvas();
            f.add(canvas,"Center");
            f.validate();
            f.addWindowListener(new WindowAdapter()
                    {
                        public void windowClosing(WindowEvent e)
                        {
                            System.exit(0);
                        }
                    });
            PrintJob p=f.getToolkit().getPrintJob(f,"ok",null);
            Graphics gr=p.getGraphics();
            int
starX[]={pointX[0]+125,pointX[2]+125,pointX[4]+125,pointX[1]+125,
                            pointX[3]+125,pointX[0]+125};
            int starY[]={pointY[0]+125,pointY[2]+125,pointY[4]+125,
                            pointY[1]+125,pointY[3]+125,pointY[0]+125};
            gr.drawPolygon(starX,starY,6);
            p.end();
        }
    }
```

【程序 4】java2D 绘制一条抛物线的一部分

用 java2D 绘制一条抛物线的一部分。

```
import java.awt.*;
import java.applet.*;
import java.awt.geom.*;
public class Boy extends Applet
{   public void paint(Graphics g)
    {
        g.setColor(Color.red) ;
        Graphics2D g_2d=(Graphics2D)g;
        QuadCurve2D quadCurve=
        new   QuadCurve2D.Double(2,10,51,90,100,10);
        g_2d.draw(quadCurve);
        quadCurve.setCurve(2,100,51,10,100,100);
        g_2d.draw(quadCurve);
    }
}
```

【程序 5】java2D 绘制练习

利用 java2D 的平移、缩放、旋转功能绘制一个你喜欢的图形。

```
import java.awt.*;
import java.awt.geom.*;
import java.applet.*;
public class Boy extends Applet
{   public void paint(Graphics g)
    {   Graphics2D   g_2d=(Graphics2D)g;
        //花叶两边的曲线:
        QuadCurve2D
        curve_1=new   QuadCurve2D.Double(200,200,150,160,200,100);
        CubicCurve2D curve_2=
        new   CubicCurve2D.Double(200,200,260,145,190,120,200,100);
        //花叶中的纹线:
        Line2D line=new Line2D.Double(200,200,200,110);
        QuadCurve2D leaf_line1=
        new   QuadCurve2D.Double(200,180,195,175,190,170);
        QuadCurve2D leaf_line2=
        new   QuadCurve2D.Double(200,180,210,175,220,170);
        QuadCurve2D leaf_line3=
        new   QuadCurve2D.Double(200,160,195,155,190,150);
        QuadCurve2D leaf_line4=
        new   QuadCurve2D.Double(200,160,214,155,220,150);
        //利用旋转来绘制花朵:
```

```
        AffineTransform trans=new AffineTransform();
        for(int i=0;i<6;i++)
          {   trans.rotate(60*Math.PI/180,200,200);
              g_2d.setTransform(trans);
              GradientPaint gradient_1=
    new GradientPaint(200,200,Color.green,200,100,Color.yellow);
              g_2d.setPaint(gradient_1);
              g_2d.fill(curve_1);
              GradientPaint gradient_2=new
    GradientPaint(200,145,Color.green,260,145,Color.red,true);
              g_2d.setPaint(gradient_2);
              g_2d.fill(curve_2);
              Color c3=new Color(0,200,0); g_2d.setColor(c3);
              g_2d.draw(line);
              g_2d.draw(leaf_line1); g_2d.draw(leaf_line2);
              g_2d.draw(leaf_line3); g_2d.draw(leaf_line4);
           }
      //花瓣中间的花蕾曲线:
        QuadCurve2D center_curve_1=
    new     QuadCurve2D.Double(200,200,190,185,200,180);
        AffineTransform trans_1=new AffineTransform();
        for(int i=0;i<12;i++)
          {   trans_1.rotate(30*Math.PI/180,200,200);
              g_2d.setTransform(trans_1);
              GradientPaint gradient_3=
              new GradientPaint(200,200,Color.red,200,180,Color.yellow);
                 g_2d.setPaint(gradient_3);
                 g_2d.fill(center_curve_1);
           }
         //再绘制一个 0.4 倍的花朵:
         AffineTransform trans_2=new AffineTransform();
         trans_2.scale(0.4,0.4);
         for(int i=0;i<6;i++)
           {   trans_2.rotate(60*Math.PI/180,200,200);
               g_2d.setTransform(trans_2);g_2d.setColor(Color.pink);
               g_2d.fill(curve_1);g_2d.setColor(Color.green);
               g_2d.fill(curve_2);
           }
      }
}
```

五 线程练习

【程序 1】线程练习（球体运动）

编写一个小应用程序，在小应用程序的主线程中有两个线程，一个负责模仿垂直上抛运动，另一个模仿 45 度的抛体运动。

```
import java.awt.*;import java.awt.event.*;
import java.applet.*;
public class Boy extends Applet        implements Runnable
{   Thread  红色球,兰色球;
    Graphics redPen,bluePen;
    double t=0;
    public void init()
    {   红色球=new Thread(this);
        兰色球=new Thread(this);
      redPen=getGraphics();bluePen=getGraphics();
      redPen.setColor(Color.red); bluePen.setColor(Color.blue);
    }
    public void start()
    {   红色球.start();兰色球.start();
    }
    public void run()
    {   while(true)
        {   t=t+0.2;
            if(Thread.currentThread()==红色球)
                {   if(t>20)    t=0;
                    redPen.clearRect(0,0,38,300);
                    redPen.fillOval(20,(int)(1.0/2*t*t*3.8),16,16);
                    try{   红色球.sleep(50);
                        }
                    catch(InterruptedException e){}
                 }
              else if(Thread.currentThread()==兰色球)
                {   bluePen.clearRect(38,0,500,300);
                    bluePen.fillOval(38+(int)(16*t),(int)(1.0/2*t*t*3.8),16,16);
                    try{   兰色球.sleep(50);
                        }
                    catch(InterruptedException e){}
                }
        }
    }
}
```

【程序 2】线程练习（银行取款问题）

用 JAVA 中的多线程示例银行取款问题

```
packagecom.softeem.demo;

/**
*@authorleno
*账户类
*默认有余额，可以取款
*/
class Account {
    privatefloatbalance = 1000;

    publicfloat getBalance() {
      returnbalance;
    }

    publicvoid setBalance(float balance) {
      this.balance = balance;
    }

    /**
    *取款的方法需要同步
    *@parammoney
    */
    publicsynchronizedvoid withdrawals(float money)
    {
      if(balance>=money)
      {
          System.out.println("被取走"+money+"元!");
          try {
              Thread.sleep(1000);
          } catch (InterruptedException e) {
              // TODO Auto-generated catch block
              e.printStackTrace();
          }
          balance-=money;
      }
      else
      {
          System.out.println("对不起,余额不足!");
```

```
      }
    }

}

/**
*@authorleno
*银行卡
*/
class TestAccount1 extends Thread {

    private Account account;


    public TestAccount1(Account account) {
      this.account = account;
    }


    @Override
    publicvoid run() {
      account.withdrawals(800);
      System.out.println("余额为:"+account.getBalance()+"元!");
    }
}
/**
*@authorleno
*存折
*/
class TestAccount2 extends Thread {

    private Account account;
    public TestAccount2(Account account) {
          this.account = account;
      }
    @Override
    publicvoid run() {
      account.withdrawals(700);
      System.out.println("余额为:"+account.getBalance()+"元!");
    }
}

publicclass Test
{
```

```
    publicstaticvoid main(String[] args) {
      Account account = new Account();
      TestAccount1 testAccount1 = new TestAccount1(account);
      testAccount1.start();
      TestAccount2 testAccount2 = new TestAccount2(account);
      testAccount2.start();
    }
}
```

21.用 JAVA 中的多线程示例火车站售票问题

```
package com.softeem.demo;

/**
*@authorleno
*售票类
*/
class SaleTicket implements Runnable {
    inttickets = 100;

    publicvoid run() {
      while (tickets > 0) {
          sale();
//或者下面这样实现
//          synchronized (this) {
//              if (tickets > 0) {
//                  System.out.println(Thread.currentThread().getName
() + "卖第"
//                          + (100 - tickets + 1) + "张票");
//                  tickets--;
//              }
//          }
      }
    }

    publicsynchronizedvoid sale() {
      if (tickets > 0) {
          System.out.println(Thread.currentThread().getName() + "卖
第"
                  + (100 - tickets + 1) + "张票");
          tickets--;
      }
    }

}
```

```
publicclass TestSaleTicket {

    publicstaticvoid main(String[] args) {
      SaleTicket st = new SaleTicket();
      new Thread(st, "一号窗口").start();
      new Thread(st, "二号窗口").start();
      new Thread(st, "三号窗口").start();
      new Thread(st, "四号窗口").start();

    }
}
```

22.用 JAVA 中的多线程示例生产者和消费者问题

```
package com.softeem.demo;

class Producer implements Runnable
{
private SyncStack stack;

    public Producer(SyncStack stack) {
    this.stack = stack;
}

    publicvoid run() {
      for (int i = 0; i < stack.getProducts().length; i++) {
          String product = "产品"+i;
          stack.push(product);
          System.out.println("生产了: "+product);
          try
          {
            Thread.sleep(200);
          }
          catch(InterruptedException e)
          {
            e.printStackTrace();
          }


      }
    }

}

class Consumer implements Runnable
```

```
{
    private SyncStack stack;

    public Consumer(SyncStack stack) {
    this.stack = stack;
}
    publicvoid run() {
      for(int i=0;i <stack.getProducts().length;i++)
          {
          String product =stack.pop();
          System.out.println("消费了: "+product);
          try
          {
            Thread.sleep(1000);
          }
          catch(InterruptedException e)
          {
            e.printStackTrace();
          }

          }


    }
}

class SyncStack
{
    private String[] products = new String[10];
    privateintindex;
    publicsynchronizedvoid push(String product)
    {
      if(index==product.length())
      {
          try {
              wait();
          } catch (InterruptedException e) {
              // TODO Auto-generated catch block
              e.printStackTrace();
          }
      }
      notify();
      products[index]=product;
      index++;
```

```
    }

    publicsynchronized String pop()
    {
      if(index==0)
      {
          try {
              wait();
          } catch (InterruptedException e) {
              // TODO Auto-generated catch block
              e.printStackTrace();
          }
      }
      notify();
      index--;
      String product = products[index];
      return product;
    }

    public String[] getProducts() {
      returnproducts;
    }


}
publicclass TestProducerConsumer {

    publicstaticvoid main(String[] args) {
      SyncStack stack=new SyncStack();
      Producer p=new Producer(stack);
      Consumer c=new Consumer(stack);

      new Thread(p).start();
      new Thread(c).start();
      }
    }
```

六 数组与集合的练习

【程序 1】创建 Map 对象

```
Map map = new LinkedHashMap();

        // Add some elements
        map.put("1", "value1");
        map.put("2", "value2");
        map.put("3", "value3");
        map.put("2", "value4");

        // List the entries
        for (Iterator it = map.keySet().iterator(); it.hasNext();) {
            Object key = it.next();
            Object value = map.get(key);
        }
        // [1=value1, 2=value4, 3=value3]
```

【程序 2】创建 HashMap 对象

```
import java.util.HashMap;
import java.util.Iterator;
import java.util.Map;
import java.util.TreeMap;

public class Test {
    public static void main(String[] args) {
        Student[] st = { new Student("2006001", "张艺", "男", 20),
                new Student("2006002", "张尔", "男", 22),
                new Student("2006003", "张三", "男", 23),
                new Student("2006004", "张思", "女", 19),
                new Student("2006005", "张武", "男", 25) };

        RearlinkQueue rearlinkQueue = new RearlinkQueue();

        for (Student temp : st) {
            rearlinkQueue.enQueue(temp);
        }
        System.out.println("初始化数据");
```

```
        System.out.println(rearlinkQueue.toString());

        System.out.println("出列第一个数据");
        rearlinkQueue.deQueue();
        System.out.println(rearlinkQueue.toString());
        aa();
    }

    public static void aa() {
        // Create a hash table
        Map map = new HashMap(); // hash table
        map = new TreeMap(); // sorted map

        // Add key/value pairs to the map
        map.put("a", new Integer(1));
        map.put("b", new Integer(2));
        map.put("c", new Integer(3));

        // Get number of entries in map
        int size = map.size(); // 2

        // Adding an entry whose key exists in the map causes
        // the new value to replace the old value
        Object oldValue = map.put("a", new Integer(9)); // 1

        // Remove an entry from the map and return the value of the
removed
        // entry
        oldValue = map.remove("c"); // 3

        // Iterate over the keys in the map
        Iterator it = map.keySet().iterator();
        while (it.hasNext()) {
            // Get key
            Object key = it.next();
        }

        // Iterate over the values in the map
        it = map.values().iterator();
        while (it.hasNext()) {
            // Get value
            Object value = it.next();
        }
```

```
    }
}
```

【程序 3】对数组进行排序

```
int[] intArray = new int[] {4, 1, 3, -23};
        Arrays.sort(intArray);
        // [-23, 1, 3, 4]

        String[] strArray = new String[] {"z", "a", "C"};
        Arrays.sort(strArray);
        // [C, a, z]

        // Case-insensitive sort
        Arrays.sort(strArray, String.CASE_INSENSITIVE_ORDER);
        // [a, C, z]

        // Reverse-order sort
        Arrays.sort(strArray, Collections.reverseOrder());
        // [z, a, C]

        // Case-insensitive reverse-order sort
        Arrays.sort(strArray, String.CASE_INSENSITIVE_ORDER);
        Collections.reverse(Arrays.asList(strArray));
        // [z, C, a]
```

【程序 4】Arrays 类的练习

使用 java.util 包中的 Arrays 类的静态方法:public static void sort(double a[])可以把参数 a 指定的 double 型数组按升序排序。Arrays 类的静态方法:public static void sort(double a[],int start,int end)可以把参数 a 指定的 double 型数组中从位置 statr 到 end 位置的数按升序排序。编写程序，使用 sort 方法对数组排序。

```
import java.util.*;
class E5
{   public static void main(String args[])
    {
        int a[]={9,4,2,12,45};
        double b[]={-3,5,3,45,6,7,405,-12,456};
```

```
        Arrays.sort(a);Arrays.sort(b,1,4);
        for(int i=0;i<a.length;i++)
        {   System.out.print(a[i]+",");
        }
        for(int i=0;i<b.length;i++)
        {   System.out.print(b[i]+",");
        }
    }
}
```

【程序 5】从有序的数组中查找一个元素

```
int index = Arrays.binarySearch(sortedArray, object);
      if (index < 0) {
         // not found
      }
```

【程序 6】数组转换为集合

```
// Fixed-size list
      List list = Arrays.asList(array);

      // Growable list
      list = new LinkedList(Arrays.asList(array));

      // Duplicate elements are discarded
      Set set = new HashSet(Arrays.asList(array));
```

【程序 7】集合转换为数组

```
// Create an array containing the elements in a list
      Object[] objectArray = list.toArray();
         MyClass[]  array  =  (MyClass[])list.toArray(new
MyClass[list.size()]);

      // Create an array containing the elements in a set
      objectArray = set.toArray();
```

```
                    array    =    (MyClass[])set.toArray(new
MyClass[set.size()]);

        // Create an array containing the keys in a map
        objectArray = map.keySet().toArray();
              array   =   (MyClass[])map.keySet().toArray(new
MyClass[set.size()]);

        // Create an array containing the values in a map
        objectArray = map.values().toArray();
              array   =   (MyClass[])map.values().toArray(new
MyClass[set.size()]);
```

【程序 8】创建一个有序的集合 Set

```
// Create the sorted set
        SortedSet set = new TreeSet();

        // Add elements to the set
        set.add("b");
        set.add("c");
        set.add("a");

        // Iterating over the elements in the set
        Iterator it = set.iterator();
        while (it.hasNext()) {
            // Get element
            Object element = it.next();
        }
        // The elements are iterated in order: a, b, c

        // Create an array containing the elements in a set (in this
case a
        // String array).
        // The elements in the array are in order.
        String[]    array    =    (String[])    set.toArray(new
String[set.size()]);
```

七 文件练习

【程序 1】RandomAccessFile 写入练习

使用 RandomAccessFile 对象将信息写入文件的末尾。

```
import java.io.*;
public class Example
{
    public static void main(String args[])
     {
        RandomAccessFile inAndOut=null;
        long length=0;
        try{
              inAndOut=new RandomAccessFile("AAAAA.txt","rw");
              length=inAndOut.length();
            }
        catch(Exception e){}
        try{
              inAndOut.seek(length);
              byte b[]="你好呀".getBytes();
              inAndOut.write(b);
              int n=-1;
              inAndOut.seek(0);
              length=inAndOut.length();
              byte c[]=new byte[(int)length];
              inAndOut.readFully(c);
              System.out.print(new String(c));
              inAndOut.close();
            }
        catch(IOException e){}
     }
}
```

【程序 2】RandomAccessFile 读取练习

使用 RandomAccessFile 对象读取文件的后 5 行。

```
import java.io.*;
public class Example
{
```

```
    public static void main(String args[])
     {
        RandomAccessFile random=null;
        long length=0;
        try{
              random=new RandomAccessFile("Example.java","rw");
              length=random.length();
              //找倒数第 6 行：
              random.seek(length);
              long lastFifthEndPosition=random.getFilePointer();
              long mark=lastFifthEndPosition;
              int j=1,n=-1;
              while((mark>=0)&&(j<=6))
               {
                    mark--;
                    random.seek(mark);
                    n=random.readByte();
                    if(n=='\n')
                    {
                       lastFifthEndPosition=random.getFilePointer();
                       j++;
                    }
               }
           random.seek(lastFifthEndPosition);
           long startPoint=random.getFilePointer();
           while(startPoint<length)
               {
                  int m=random.readByte();
                  System.out.print((char)m);
               }
           random.close();
        }
      catch(IOException ee)
        {
        }
    }
}
```

【程序 3】拷贝目录

拷贝一个目录(文件)到指定路径

```
/**
    *拷贝一个目录或者文件到指定路径下
    *@paramsource
    *@paramtarget
    */
    publicvoid copy(File source,File target)
    {
      File tarpath = new File(target,source.getName());
      if(source.isDirectory())
      {
          tarpath.mkdir();
          File[] dir = source.listFiles();
          for (int i = 0; i < dir.length; i++) {
              copy(dir[i],tarpath);
          }
      }else
      {
          try {
              InputStream is = new FileInputStream(source);
              OutputStream os = new FileOutputStream(tarpath);
              byte[] buf = newbyte[1024];
              int len = 0;
              while((len = is.read(buf))!=-1)
              {
                  os.write(buf,0,len);
              }
              is.close();
              os.close();
          } catch (FileNotFoundException e) {
              // TODO Auto-generated catch block
              e.printStackTrace();
          } catch (IOException e) {
              // TODO Auto-generated catch block
              e.printStackTrace();
          }
      }
    }
```

八 数据库程序练习

【程序 1】怎样连接数据库

1 连接 oracle 数据库：

```
Connection connection = null;
      try {
         // Load the JDBC driver
         String                driverName                =
"oracle.jdbc.driver.OracleDriver";
         Class.forName(driverName);

         // Create a connection to the database
         String serverName = "127.0.0.1";
         String portNumber = "1521";
         String sid = "mydatabase";
         String url = "jdbc:oracle:thin:@" + serverName + ":"
+ portNumber
               + ":" + sid;
         String username = "username";
         String password = "password";
         connection   =   DriverManager.getConnection(url,
username, password);
      } catch (ClassNotFoundException e) {
         // Could not find the database driver
      } catch (SQLException e) {
         // Could not connect to the database
      }
```

2 连接 MySql 数据库

```
Connection connection = null;
      try {
         // Load the JDBC driver
         String driverName = "org.gjt.mm.mysql.Driver"; //
MySQL MM JDBC driver
         Class.forName(driverName);

         // Create a connection to the database
         String serverName = "localhost";
         String mydatabase = "mydatabase";
         String url = "jdbc:mysql://" + serverName + "/" +
mydatabase; // a JDBC url
```

```
            String username = "username";
            String password = "password";
            connection    =    DriverManager.getConnection(url,
username, password);
        } catch (ClassNotFoundException e) {
            // Could not find the database driver
        } catch (SQLException e) {
            // Could not connect to the database
        }
```

3 连接 SqlServer 数据库

```
Connection connection = null;
        try {
                                        String    driverName    =
"com.jnetdirect.jsql.JSQLDriver"; // NetDirect JDBC driver
            String serverName = "127.0.0.1";
            String portNumber = "1433";
            String mydatabase = serverName + ":" + portNumber;
            String url = "jdbc:JSQLConnect://" + mydatabase; //
a JDBC url
            String username = "username";
            String password = "password";

            // Load the JDBC driver
            Class.forName(driverName);

            // Create a connection to the database
              connection = DriverManager.getConnection(url,
username, password);
        } catch (ClassNotFoundException e) {
            // Could not find the database driver
        } catch (SQLException e) {
            // Could not connect to the database
        }
```

【程序 2】创建数据库表格

```
try {
            Statement stmt = connection.createStatement();

            // Create table called my_table
            String   sql   =   "CREATE   TABLE   my_table(col_string
```

```
VARCHAR(254))";

            stmt.executeUpdate(sql);
        } catch (SQLException e) {
            }
```

【程序 3】创建带各种数据类型的 SqlServer 数据库表格

```
try {
            Statement stmt = connection.createStatement();
             //     Column Name          SQLServer Type          Java
Type
            String sql = "CREATE TABLE sqlserver_all_table("
                + "col_boolean        BIT, "              // boolean
                + "col_byte           TINYINT, "          // byte
                + "col_short          SMALLINT, "          // short
                + "col_int            INTEGER, "          // int
                + "col_float          REAL, "             // float
                + "col_double         DOUBLE PRECISION, "   // double
                    + "col_bigdecimal       DECIMAL(13,0), "       //
BigDecimal; can also be NUMERIC(p,s)
                + "col_string         VARCHAR(254), "      // String
                + "col_date           DATETIME, "          // Date
                + "col_time           DATETIME, "          // Time
                    + "col_timestamp        TIMESTAMP, "           //
Timestamp
                    + "col_characterstream  TEXT, "                //
CharacterStream or AsciiStream (< 2 GBytes)
                    + "col_binarystream     IMAGE)";               //
BinaryStream (< 2 GBytes)

            stmt.executeUpdate(sql);
        } catch (SQLException e) {
        }
```

【程序 4】创建带各种数据类型的 Msql 数据库表格

```
try {
            Statement stmt = connection.createStatement();
```

```
          String sql = "CREATE TABLE mysql_all_table("
                + "col_boolean      BOOL, " // boolean
                + "col_byte         TINYINT, " // byte
                + "col_short        SMALLINT, " // short
                + "col_int          INTEGER, " // int
                + "col_long         BIGINT, " // long
                + "col_float        FLOAT, " // float
                + "col_double        DOUBLE PRECISION, " //
double
                +  "col_bigdecimal      DECIMAL(13,0),  " //
BigDecimal
                + "col_string       VARCHAR(254), " // String
                + "col_date         DATE, " // Date
                + "col_time         TIME, " // Time
                + "col_timestamp    TIMESTAMP, " // Timestamp
                + "col_asciistream   TEXT, " // AsciiStream (<
2^16 bytes)
                +   "col_binarystream     LONGBLOB,   "   //
BinaryStream (< 2^32
                                          // bytes)
                + "col_blob         BLOB)"; // Blob (< 2^16
bytes)

            stmt.executeUpdate(sql);
        } catch (SQLException e) {
        }
```

【程序 5】返回结果集

```
try {
            // Create a result set containing all data from
my_table
           Statement stmt = connection.createStatement();
            ResultSet rs = stmt.executeQuery("SELECT * FROM
my_table");
        } catch (SQLException e) {
        }
```

【程序 6】从结果集中返回各种类型的数据(MySql)

```
try {
```

```
        // Create a result set containing all data from
mysql_all_table
        Statement stmt = connection.createStatement();
        ResultSet rs = stmt.executeQuery("SELECT * FROM
mysql_all_table");

        // Fetch each row from the result set
        while (rs.next()) {
           boolean bool = rs.getBoolean("col_boolean");
           byte b = rs.getByte("col_byte");
           short s = rs.getShort("col_short");
           int i = rs.getInt("col_int");
           long l = rs.getLong("col_long");
           float f = rs.getFloat("col_float");
           double d = rs.getDouble("col_double");
           BigDecimal bd =
rs.getBigDecimal("col_bigdecimal");
           String str = rs.getString("col_string");
           Date date = rs.getDate("col_date");
           Time t = rs.getTime("col_time");
           Timestamp ts =
rs.getTimestamp("col_timestamp");
           InputStream ais =
rs.getAsciiStream("col_asciistream");
           InputStream bis =
rs.getBinaryStream("col_binarystream");
           Blob blob = rs.getBlob("col_blob");
        }
     } catch (SQLException e) {
     }
```

【程序 7】返回二进制大对象数据

```
try {
         Statement stmt = connection.createStatement();
         ResultSet rs = stmt.executeQuery("SELECT col_blob
FROM mysql_all_table");

         if (rs.next()) {
            // Get the BLOB from the result set
            Blob blob = rs.getBlob("col_blob");
```

```
            // Get the number bytes in the BLOB
            long blobLength = blob.length();

            // Get bytes from the BLOB in a byte array
            int pos = 1;   // position is 1-based
            int len = 10;
            byte[] bytes = blob.getBytes(pos, len);

            // Get bytes from the BLOB using a stream
            InputStream is = blob.getBinaryStream();
            int b = is.read();
        }
    } catch (IOException e) {
    } catch (SQLException e) {
    }
```

【程序 8】通过预编译语句插入数据

```
try {
        // Prepare a statement to insert a record
        String sql = "INSERT INTO mysql_all_table("
            + "col_boolean,"
            + "col_byte,"
            + "col_short,"
            + "col_int,"
            + "col_long,"
            + "col_float,"
            + "col_double,"
            + "col_bigdecimal,"
            + "col_string,"
            + "col_date,"
            + "col_time,"
            + "col_timestamp,"
            + "col_asciistream,"
            + "col_binarystream,"
            + "col_blob) "
            + "VALUES(?,?,?,?,?,?,?,?,?,?,?,?,?,?,?)";
                        PreparedStatement   pstmt   =
connection.prepareStatement(sql);

        // Set the values
        pstmt.setBoolean(1, true);
```

```
            pstmt.setByte(2, (byte)123);
            pstmt.setShort(3, (short)123);
            pstmt.setInt(4, 123);
            pstmt.setLong(5, 123L);
            pstmt.setFloat(6, 1.23F);
            pstmt.setDouble(7, 1.23D);
            pstmt.setBigDecimal(8, new BigDecimal(1.23));
            pstmt.setString(9, "a string");
                                    pstmt.setDate(10,    new
java.sql.Date(System.currentTimeMillis()));
                                    pstmt.setTime(11,    new
Time(System.currentTimeMillis()));
                                pstmt.setTimestamp(12,    new
Timestamp(System.currentTimeMillis()));

            // Set the ascii stream
            File file = new File("infilename1");
            FileInputStream is = new FileInputStream(file);
            pstmt.setAsciiStream(13, is, (int)file.length());

            // Set the binary stream
            file = new File("infilename2");
            is = new FileInputStream(file);
                            pstmt.setBinaryStream(14,    is,
(int)file.length());

            // Set the blob
            file = new File("infilename3");
            is = new FileInputStream(file);
                            pstmt.setBinaryStream(15,    is,
(int)file.length());

            // Insert the row
            pstmt.executeUpdate();
        } catch (SQLException e) {
        } catch (FileNotFoundException e) {
        }
```

【程序 9】向数据库插入大数据

```
try {
            // Prepare a statement to insert binary data
```

```
            String sql = "INSERT INTO mysql_all_table
(col_binarystream) VALUES(?)";
                        PreparedStatement pstmt =
connection.prepareStatement(sql);

        // Create some binary data
        byte[] buffer = "some data".getBytes();

        // Set value for the prepared statement
        pstmt.setBytes(1, buffer);

        // Insert the data
        pstmt.executeUpdate();
        pstmt.close();


        // Select records from the table
        Statement stmt = connection.createStatement();
        ResultSet resultSet = stmt.executeQuery("SELECT *
FROM mysql_all_table");
        while (resultSet.next()) {
            // Get data from the binary column
                                    byte[] bytes =
resultSet.getBytes("col_binarystream");
        }
    } catch (SQLException e) {
        }
```

【程序 10】向数据库批量插入数据

```
try {
            // Disable auto-commit
            connection.setAutoCommit(false);

            // Create a prepared statement
            String sql = "INSERT INTO my_table VALUES(?)";
                        PreparedStatement pstmt =
connection.prepareStatement(sql);
```

```
            // Insert 10 rows of data
            for (int i=0; i<10; i++) {
                pstmt.setString(1, ""+i);
                pstmt.addBatch();
            }

            // Execute the batch
            int [] updateCounts = pstmt.executeBatch();

            // All statements were successfully executed.
            // updateCounts contains one element for each
batched statement.
            // updateCounts[i] contains the number of rows
affected by that statement.
            processUpdateCounts(updateCounts);

            // Since there were no errors, commit
            connection.commit();
        } catch (BatchUpdateException e) {
            // Not all of the statements were successfully
executed
            int[] updateCounts = e.getUpdateCounts();

            // Some databases will continue to execute after
one fails.
             // If so, updateCounts.length will equal the
number of batched statements.
            // If not, updateCounts.length will equal the
number of successfully executed statements
            processUpdateCounts(updateCounts);

              // Either commit the successfully executed
statements or rollback the entire batch
            connection.rollback();
        } catch (SQLException e) {
        }

        public static void processUpdateCounts(int[]
updateCounts) {
            for (int i=0; i<updateCounts.length; i++) {
                if (updateCounts[i] >= 0) {
                    // Successfully executed; the number
represents number of affected rows
                } else if (updateCounts[i] ==
```

```
Statement.SUCCESS_NO_INFO) {
                        // Successfully executed; number of
affected rows not available
                            } else if (updateCounts[i] ==
Statement.EXECUTE_FAILED) {
                        // Failed to execute
                    }
                }
            }
```

【程序 11】通过滚动光标获得行号

```
try {
        // Create a scrollable result set
        Statement stmt = connection.createStatement(
              ResultSet.TYPE_SCROLL_INSENSITIVE,
              ResultSet.CONCUR_READ_ONLY);
        ResultSet resultSet = stmt.executeQuery("SELECT *
FROM my_table");

        // Move to the end of the result set
        resultSet.last();

        // Get the row number of the last row which is also
the row count
        int rowCount = resultSet.getRow();
      } catch (SQLException e) {

      }
```

【程序 12】创建一个可以更新的结果集

```
try {
         // Create a statement that will return updatable
result sets
         Statement stmt = connection.createStatement(
                          ResultSet.TYPE_SCROLL_SENSITIVE,
ResultSet.CONCUR_UPDATABLE);

         // Primary key col_string must be specified so that
```

```
the result set is updatable
            ResultSet resultSet = stmt.executeQuery("SELECT
col_string FROM my_table");
        } catch (SQLException e) {

        }
```

【程序 13】调用存储过程

```
CallableStatement cs;
          try {
            // Call a procedure with no parameters
              cs = connection.prepareCall("{call myproc}");
              cs.execute();

            // Call a procedure with one IN parameter
              cs = connection.prepareCall("{call myprocin(?)}");

              // Set the value for the IN parameter
              cs.setString(1, "a string");

              // Execute the stored procedure
              cs.execute();

            // Call a procedure with one OUT parameter
              cs = connection.prepareCall("{call myprocout(?)}");

              // Register the type of the OUT parameter
              cs.registerOutParameter(1, Types.VARCHAR);

               // Execute the stored procedure and retrieve the OUT
value
              cs.execute();
              String outParam = cs.getString(1);     // OUT parameter

            // Call a procedure with one IN/OUT parameter
              cs = connection.prepareCall("{call myprocinout(?)}");

              // Register the type of the IN/OUT parameter
              cs.registerOutParameter(1, Types.VARCHAR);

              // Set the value for the IN/OUT parameter
```

```
            cs.setString(1, "a string");

            // Execute the stored procedure and retrieve the IN/OUT
value
            cs.execute();
            outParam = cs.getString(1);          // OUT parameter
        } catch (SQLException e) {

         }
```

【程序 14】导出一个 mysql 表数据到文件

```
try {
          // Create the statement
          Statement stmt = connection.createStatement();

          // Export the data
          String filename = "c:\\\\temp\\\\outfile.txt";
          String tablename = "mysql_2_table";
          stmt.executeUpdate("SELECT * INTO OUTFILE \"" + filename
                  + "\" FROM " + tablename);
      } catch (SQLException e) {

      }
```

【程序 15】从文本文件导入数据到 mysql 表

```
 try {
            // Create the statement
            Statement stmt = connection.createStatement();

            // Load the data
            String filename = "c:\\\\temp\\\\infile.txt";
            String tablename = "mysql_2_table";
              stmt.executeUpdate("LOAD DATA INFILE \"" +
filename + "\" INTO TABLE " + tablename);

             // If the file is comma-separated, use this
statement
```

```
                stmt.executeUpdate("LOAD DATA INFILE \"" +
filename + "\" INTO TABLE "
                                        + tablename + " FIELDS
TERMINATED BY ','");

             // If the file is terminated by \r\n, use this
statement
                stmt.executeUpdate("LOAD DATA INFILE \"" +
filename + "\" INTO TABLE "
                                    + tablename + " LINES TERMINATED
BY '\\r\\n'");
          } catch (SQLException e) {

          }
```

九 网络程序练习

【程序 1】读取 URL 对象处的资源

利用 URL 对象读取网络资源。URL 类中有一个方法 openStream(),一个 URL 对象可以使用这个方法获得一个输入流，然后用这个输入流读取 URL 对象处的资源。

```
import java.net.*;
import java.io.*;
public class E2
{
  public static void main(String args[])
    {
       try{
          URL url=new URL("http://www.tsinghua.edu.cn/welcome.html");
          InputStream in=url.openStream();
          int b,j=1000;byte tom[]=new byte[2500];
          while((b=in.read(tom,0,j))!=-1)
              {
                 String s=new String (tom,0,j);
                 System.out.println(s);
              }
           in.close();
         }
       catch(Exception e){}
     }
}
```

【程序 2】网络传输对象

编程实现序列化的 Student（sno,sname）对象在网络上的传输

```
package com.softeem.demo;

import java.io.IOException;
import java.io.ObjectInputStream;
import java.io.ObjectOutputStream;
import java.io.Serializable;
import java.net.ServerSocket;
import java.net.Socket;

class Student implements Serializable {
	private int sno;
	private String sname;

	public Student(int sno, String sname) {
		this.sno = sno;
		this.sname = sname;
	}

	public int getSno() {
		return sno;
	}

	public void setSno(int sno) {
		this.sno = sno;
	}

	public String getSname() {
		return sname;
	}

	public void setSname(String sname) {
		this.sname = sname;
	}

	@Override
	public String toString() {
		return "学号:" + sno + ";姓名:" + sname;
	}
```

```
}

class MyClient extends Thread {
	@Override
	public void run() {
		try {
			Socket s = new Socket("localhost", 9999);
				ObjectInputStream ois = new
ObjectInputStream(s.getInputStream());
			Student stu = (Student) ois.readObject();
			 System.out.println("客户端程序收到服务器端程序传输
过来的学生对象>> " + stu);
			ois.close();
			s.close();
		} catch (IOException e) {
			// TODO Auto-generated catch block
			e.printStackTrace();
		} catch (ClassNotFoundException e) {
			// TODO Auto-generated catch block
			e.printStackTrace();
		}
	}
}

class MyServer extends Thread {

	@Override
	public void run() {
		try {
			ServerSocket ss = new ServerSocket(9999);
			Socket s = ss.accept();
				ObjectOutputStream ops = new
ObjectOutputStream(s.getOutputStream());
			Student stu = new Student(1, "赵本山");
			ops.writeObject(stu);
			ops.close();
			s.close();
			ss.close();
		} catch (IOException e) {
			// TODO Auto-generated catch block
			e.printStackTrace();
		}
	}
```

```
}

public class TestTransfer {
      public static void main(String[] args) {
            new MyServer().start();
            new MyClient().start();
      }
}
```

【程序 3】广播练习

制作一个广播，要求能通过窗口中的菜单选择要广播的文件或停止广播。广播文件时，每次广播文件的一行，并且可以重复广播一个文件。

```
import java.awt.* import java.io.*;
import java.net.*;
import java.awt.*;
import java.awt.event.*;
import javax.swing.Timer;

public class BroadCastWord extends Frame implements ActionListener
{
    int port;                                          //组播的端口.
    InetAddress group=null;                            //组播组的地址.
    MulticastSocket socket=null;                       //多点广播套接字.
    Timer time=null;                                   //计时器,负责每隔
多长时间播放一个内容.
    FileDialog open=null;                              //选择文件对话框.
    Button select,开始广播,停止广播;
    File file=null;                                    //要播放的文件.
    String FileDir=null,fileName=null;
    FileReader in=null;                                //负责读取文件的
流.
    BufferedReader bufferIn=null;
    int     token=0;                                   //文件的长度.
    TextArea  显示正在播放内容,显示已播放的内容;
   public BroadCastWord()
    {
    super("单词广播系统");
    select=new Button("选择要广播的文件");
    开始广播=new Button("开始广播");
    停止广播=new Button("停止广播");
    select.addActionListener(this);
    开始广播.addActionListener(this);
    停止广播.addActionListener(this);
```

```
    time=new Timer(2000,this);                          //每隔2秒"振铃"一次,
导致ActionEvent事件.
    open=new FileDialog(this,"选择要广播的文件",FileDialog.LOAD);
    显示正在播放内容=new TextArea(10,10);
    显示正在播放内容.setForeground(Color.blue);
    显示已播放的内容=new TextArea(10,10);
    Panel north=new Panel();
    north.add(select);
    north.add(开始广播);
    north.add(停止广播);
    add(north,BorderLayout.NORTH);
    Panel center=new Panel();
    center.setLayout(new GridLayout(1,2));
    center.add(显示正在播放内容);
    center.add(显示已播放的内容);
    add(center,BorderLayout.CENTER);
    validate();
    try
        {
         port=5858;                                     //设置组播组的监
听端口为5000.
         group=InetAddress.getByName("239.255.8.0");   //设置组播组的地址
为239.255.0.0.
         socket=new MulticastSocket(port);              //多点广播套接字将在
port端口广播.
         socket.setTimeToLive(1);                       //多点广播套接字发送数
据报范围为本地网络.
         socket.joinGroup(group);                       //加入组播组,加入group
后,socket发送的数据报.
                                                        //可以被加入到group中
的成员接收到.
        }
  catch(Exception e)
         {
           System.out.println("Error: "+ e);
         }
  setBounds(100,50,360,380);
  setVisible(true);
  addWindowListener(new WindowAdapter()
                          { public void windowClosing(WindowEvent e)
                            { System.exit(0);
                            }
                          });
 }
```

```
public void actionPerformed(ActionEvent e)
{
   if(e.getSource()==select)
      {
       显示已播放的内容.setText(null);
       open.setVisible(true);
       fileName=open.getFile();
       FileDir=open.getDirectory();
       if(fileName!=null)
        {
          time.stop();                                          //停止振铃,
打断播放.
           file=new File(FileDir,fileName);
           try
               {
                   file=new File(FileDir,fileName);
                   in=new FileReader(file);
                   bufferIn=new BufferedReader(in);
               }
           catch(IOException ee)
               {
               }
        }
      }
   else if(e.getSource()==开始广播)
      {
         time.start();
      }
   else if(e.getSource()==time)
      {
        String s=null;
         try
               {
                 if(token==-1)
                     {
                       file=new File(FileDir,fileName);
                       in=new FileReader(file);
                       bufferIn=new BufferedReader(in);
                     }
                 s=bufferIn.readLine();
                 if(s!=null)
                    {
                      token=0;
                      显示正在播放内容.setText("正在广播的内容:\n"+s);
```

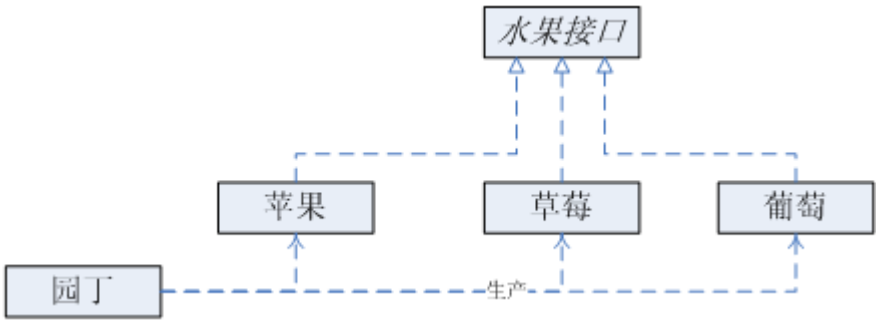
```
                显示已播放的内容.append(s+"\n");
                DatagramPacket                          packet=null;
//待发送的数据包.
                byte data[]=s.getBytes();
                packet=new DatagramPacket(data,data.length,group,port);
                socket.send(packet);
//发送数据包.
                }
              else
                {
                  token=-1;
                }
            }
          catch(IOException ee)
            {
            }
      }
    else if(e.getSource()==停止广播)
      {
        time.stop();
      }
  }

  public static void main(String[] args)
    {
      BroadCastWord broad=new BroadCastWord();
    }
}
```

十 模式练习

【程序 1】简单工厂模式（一般模式）

问题：有一个农场，生产各种水果，有苹果（Apple）、草莓（Strawberry）、葡萄（Grape）；农场的园丁（FruitGardener）要根据客户的需求，提供相应的水果。



1 产品接口－水果接口：Fruit.java

```
package com.lavasoft.patterns.simplefactory.ybgc;
/**
 * Created by IntelliJ IDEA.
 * FileName:Fruit.java
 * User:    LavaSoft
 * Date:    2006-12-1
 * Time:    0:26:51
 * 《Java 与模式》（--阎宏博士著）读书笔记
 * 工厂模式模式--简单工厂模式--一般模式
 * ReadMe:  抽象产品角色:工厂的水果产品接口--水果
 */
public interface Fruit {
    /**
     * 种植
     */
    void plant();
    /**
     * 生长
     */
    void grow();
    /**
     * 收获
     */
    void harvest();
}
```

2 产品－平果类：Apple.java

```
package com.lavasoft.patterns.simplefactory.ybgc;
/**
 * Created by IntelliJ IDEA.
 * FileName:Apple.java
 * User:    LavaSoft
 * Date:    2006-12-1
 * Time:    0:47:25
```

```
 * 《Java 与模式》（--阎宏博士著）读书笔记
 * 工厂模式模式--简单工厂模式--一般模式
 * ReadMe:  水果工厂的产品:苹果
 */
public class Apple implements Fruit {
    private int treeAge;
    /**
     * 种植
     */
    public void plant() {
        System.out.println("Apple has been planted.");
    }
    /**
     * 生长
     */
    public void grow() {
        System.out.println("Apple is growing...");
    }
    /**
     * 收获
     */
    public void harvest() {
        System.out.println("Apple has been harvested.");
    }
    /**
     * @return 返回树龄
     */
    public int getTreeAge() {
        return treeAge;
    }
    /**
     * 设置树龄
     */
    public void setTreeAge(int treeAge) {
        this.treeAge = treeAge;
    }
}
```

3 产品－草莓类：Strawberry.java

```
package com.lavasoft.patterns.simplefactory.ybgc;
/**
 * Created by IntelliJ IDEA.
 * FileName:Strawberry.java
 * User:    LavaSoft
```

```
 * Date:    2006-12-1
 * Time:    0:45:09
 * 《Java 与模式》（--阎宏博士著）读书笔记
 * 工厂模式模式--简单工厂模式--一般模式
 * ReadMe:  水果工厂的产品:草莓
 */
public class Strawberry implements Fruit {
    /**
     * 生长
     */
    public void grow() {
        System.out.println("Strawberry is growing...");
    }
    /**
     * 收获
     */
    public void harvest() {
        System.out.println("Strawberry has been harvested.");
    }
    /**
     * 种植
     */
    public void plant() {
        System.out.println("Strawberry has been planted.");
    }
    /**
     * 辅助方法
     */
    public static void log(String msg) {
        System.out.println(msg);
    }
}
```

4 产品－葡萄类：Grape.java

```
package com.lavasoft.patterns.simplefactory.ybgc;
/**
 * Created by IntelliJ IDEA.
 * FileName:Grape.java
 * User:    LavaSoft
 * Date:    2006-12-1
 * Time:    0:36:56
 * 《Java 与模式》（--阎宏博士著）读书笔记
 * 工厂模式模式--简单工厂模式--一般模式
 * ReadMe:  水果工厂的产品:葡萄
```

```
 */
public class Grape implements Fruit {
    private boolean seedless;   //是否有籽
    /**
     * 种植
     */
    public void plant() {
        System.out.println("Grape has been planted.");
    }
    /**
     * 生长
     */
    public void grow() {
        System.out.println("Grape is growing...");
    }
    /**
     * 收获
     */
    public void harvest() {
        System.out.println("Grape has been harvested.");
    }
    /**
     * @return 是否有籽
     */
    public boolean getSeedless() {
        return seedless;
    }
    /**
     * 有无籽的赋值方法
     */
    public void setSeedless(boolean seedless) {
        this.seedless = seedless;
    }
    /**
     * 辅助方法
     */
    public static void log(String msg) {
        System.out.println(msg);
    }
}
```

5 工厂一园丁类：FruitGardener.java

```
package com.lavasoft.patterns.simplefactory.ybgc;
```

```
/**
 * Created by IntelliJ IDEA.
 * FileName:FruitGardener.java
 * User:    LavaSoft
 * Date:    2006-12-1
 * Time:    1:03:27
 * 《Java 与模式》（--阎宏博士著）读书笔记
 * 工厂模式模式--简单工厂模式--一般模式
 * ReadMe:  工厂类角色：水果园丁,生产水果产品
 */
public class FruitGardener {
    /**
     * 静态工厂方法
     * @param which :具体的产品名称
     * @return 一个水果对象
     * @throws BadFruitException
     */
    public static Fruit factory(String which) throws BadFruitException
{
        if (which.equalsIgnoreCase("apple")) {
            return new Apple();
        } else if (which.equalsIgnoreCase("strawberry")) {
            return new Strawberry();
        } else if (which.equalsIgnoreCase("grape")) {
            return new Grape();
        } else {
            throw new BadFruitException("Bad fruit request");
        }
    }
}
```

6 工厂异常定义类：BadFruitException.java

```
package com.lavasoft.patterns.simplefactory.ybgc;
/**
 * Created by IntelliJ IDEA.
 * FileName:BadFruitException.java
 * User:    LavaSoft
 * Date:    2006-12-1
 * Time:    1:04:56
 * 《Java 与模式》（--阎宏博士著）读书笔记
 * 工厂模式模式--简单工厂模式--一般模式
 * ReadMe:  工厂的异常处理类
 */
public class BadFruitException extends Exception {
```

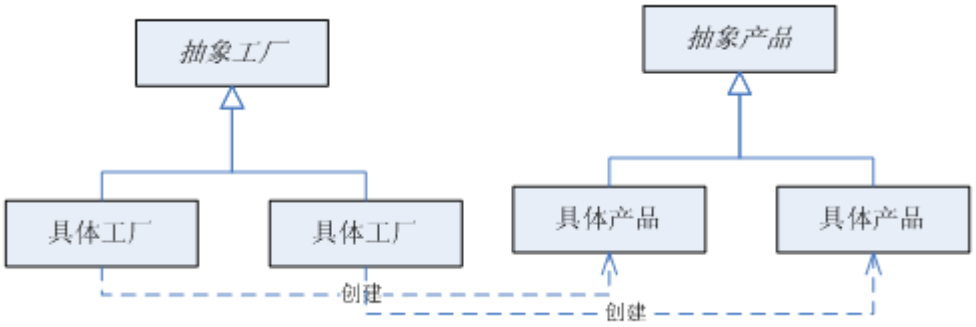
```
    public BadFruitException(String msg) {
        super(msg);     //调用父类的构造方法
    }
}
```

7 一般工厂模式的测试类

```
package com.lavasoft.patterns.simplefactory.ybgc;
/**
 * Created by IntelliJ IDEA.
 * FileName:TestApp.java
 * User:    LavaSoft
 * Date:    2006-12-1
 * Time:    1:12:08
 * 《Java 与模式》（--阎宏博士著）读书笔记
 * 工厂模式模式--简单工厂模式--一般模式
 * ReadMe:  一般工厂模式的测试类
 */
public class TestApp {
    /**
     * 测试方法
     */
    private void test(String fruitName) {
        try {
            Fruit f = FruitGardener.factory(fruitName);
            System.out.println("恭喜!生产了一个水果对象:" + fruitName);
        } catch (BadFruitException e) {
            System.out.println("对不起!工厂目前不能生产你所要的产品:" +
fruitName);
            System.out.println(e.getMessage());     //输出异常信息
            e.printStackTrace();                    //输出异常堆栈信息
        }
    }
    /**
     * 应用入口方法
     */
    public static void main(String args[]) {
        TestApp t = new TestApp();
        t.test("apple");
        t.test("grape");
        t.test("strawberry");
        t.test("car");  //此处会抛异常,水果工厂能生产 car(轿车)吗!哈哈哈
哈...
    }
}
```

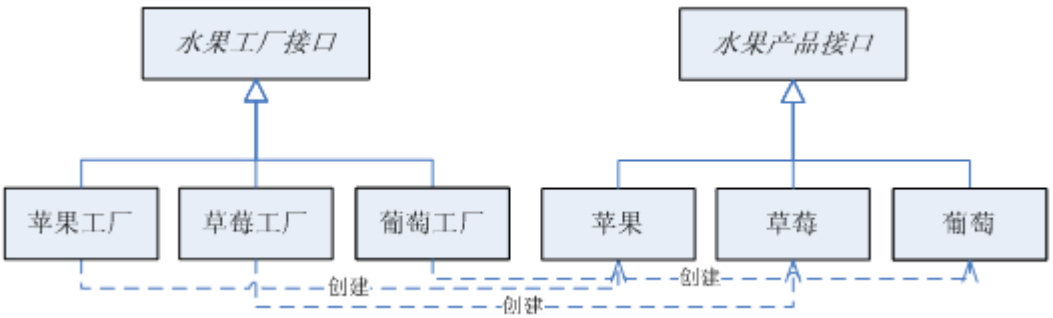
【程序 2】工厂方法模式在农场系统中的实现

在工厂方法模式中，一般都有一个平行的等级结构，也就是说工厂和产品是对应的的。抽象工厂对应抽象产品，具体工厂对应具体产品。简单的示意图如下：



背景

在简单工厂模式中，有个全能的园丁，控制所有作物的种植、生长和收获。现在农场规模变大了，管理更加专业化了。过去全能的园丁没有了，每一种作物都有专门的园丁管理，形成了规模化和专业化生产。



实现源码：

1 水果产品接口 Fruit.java

```
package com.lavasoft.patterns.factorymethod.ybms;
/**
 * Created by IntelliJ IDEA.
 * FileName:Fruit.java
 * User:    LavaSoft
 * Date:    2006-12-3
 * Time:    17:25:48
```

```
 * 《Java 与模式》（--阎宏博士著）读书笔记
 * 工厂模式--工厂方法模式--一般性模式（农场应用）
 * ReadMe:  水果接口
 */
public interface Fruit {
    /**
     * 种植
     */
    void plant();
    /**
     * 生长
     */
    void grow();
    /**
     * 收获
     */
    void harvest();
}
```

2 具体产品苹果 Apple.java

```
package com.lavasoft.patterns.factorymethod.ybms;
/**
 * Created by IntelliJ IDEA.
 * FileName:Apple.java
 * User:    LavaSoft
 * Date:    2006-12-1
 * Time:    0:47:25
 * 《Java 与模式》（--阎宏博士著）读书笔记
 * 工厂模式--工厂方法模式--一般性模式（农场应用）
 * ReadMe:  水果工厂的产品:苹果
 */
public class Apple implements Fruit {
    private int treeAge;
    /**
     * 种植
     */
    public void plant() {
        System.out.println("Apple has been planted.");
    }
    /**
     * 生长
     */
    public void grow() {
```

```
        System.out.println("Apple is growing...");
    }
    /**
     * 收获
     */
    public void harvest() {
        System.out.println("Apple has been harvested.");
    }
    /**
     * @return 返回树龄
     */
    public int getTreeAge() {
        return treeAge;
    }
    /**
     * 设置树龄
     */
    public void setTreeAge(int treeAge) {
        this.treeAge = treeAge;
    }
}
```

3 具体产品葡萄：Grape.java

```
package com.lavasoft.patterns.factorymethod.ybms;
/**
 * Created by IntelliJ IDEA.
 * FileName:Grape.java
 * User:    LavaSoft
 * Date:    2006-12-1
 * Time:    0:36:56
 * 《Java 与模式》（--阎宏博士著）读书笔记
 * 工厂模式--工厂方法模式--一般性模式（农场应用）
 * ReadMe:  水果工厂的产品:葡萄
 */
public class Grape implements Fruit {
    private boolean seedless;   //是否有籽
    /**
     * 种植
     */
    public void plant() {
        System.out.println("Grape has been planted.");
    }
    /**
     * 生长
```

```
     */
    public void grow() {
        System.out.println("Grape is growing...");
    }
    /**
     * 收获
     */
    public void harvest() {
        System.out.println("Grape has been harvested.");
    }
    /**
     * @return 是否有籽
     */
    public boolean getSeedless() {
        return seedless;
    }
    /**
     * 有无籽的赋值方法
     */
    public void setSeedless(boolean seedless) {
        this.seedless = seedless;
    }
    /**
     * 辅助方法
     */
    public static void log(String msg) {
        System.out.println(msg);
    }
}
```

4 具体产品草莓：Strawberry.java

```
package com.lavasoft.patterns.factorymethod.ybms;

/**
 * Created by IntelliJ IDEA.
 * FileName:Strawberry.java
 * User:    LavaSoft
 * Date:    2006-12-1
 * Time:    0:45:09
 * 《Java 与模式》（--阎宏博士著）读书笔记
 * 工厂模式--工厂方法模式--一般性模式（农场应用）
 * ReadMe:  水果工厂的产品:草莓
 */
public class Strawberry implements Fruit {
```

```
    /**
     * 生长
     */
    public void grow() {
        System.out.println("Strawberry is growing...");
    }
    /**
     * 收获
     */
    public void harvest() {
        System.out.println("Strawberry has been harvested.");
    }
    /**
     * 种植
     */
    public void plant() {
        System.out.println("Strawberry has been planted.");
    }
    /**
     * 辅助方法
     */
    public static void log(String msg) {
        System.out.println(msg);
    }
}
```

5 水果工厂接口：FruitGardener.java

```
package com.lavasoft.patterns.factorymethod.ybms;
/**
 * Created by IntelliJ IDEA.
 * FileName:FruitGardener.java
 * User:    LavaSoft
 * Date:    2006-12-3
 * Time:    17:22:52
 * 《Java 与模式》（--阎宏博士著）读书笔记
 * 工厂模式--工厂方法模式--一般性模式（农场应用）
 * ReadMe:  水果工厂接口
 */
public interface FruitGardener {
    /**
     * 工厂方法
     *
     * @return 水果
     */
```

```
    public Fruit factory();
}
```

6 苹果工厂：AppleGardener.java

```
package com.lavasoft.patterns.factorymethod.ybms;
/**
 * Created by IntelliJ IDEA.
 * FileName:AppleGardener.java
 * User:    LavaSoft
 * Date:    2006-12-3
 * Time:    17:45:29
 * 《Java 与模式》（--阎宏博士著）读书笔记
 * 工厂模式--工厂方法模式--一般性模式（农场应用）
 * ReadMe:  苹果工厂方法
 */
public class AppleGardener implements FruitGardener {
    /**
     * 工厂方法
     *
     * @return 苹果
     */
    public Fruit factory() {
        Fruit f = new Apple();
        System.out.println("水果工厂（AppletGardener）成功创建一个水
果：苹果！");
        return f;
    }
}
```

7 葡萄工厂：GrapeGardener.java

```
package com.lavasoft.patterns.factorymethod.ybms;
/**
 * Created by IntelliJ IDEA.
 * FileName:GrapeGardener.java
 * User:    LavaSoft
 * Date:    2006-12-3
 * Time:    17:51:41
 * 《Java 与模式》（--阎宏博士著）读书笔记
 * 工厂模式--工厂方法模式--一般性模式（农场应用）
 * ReadMe:  添加说明
 */
public class GrapeGardener implements FruitGardener {
    /**
     * 工厂方法
```

```
     *
     * @return 葡萄
     */
    public Fruit factory() {
        Fruit f = new Grape();
        System.out.println("水果工厂（GrapeGardener）成功创建一个水果:
葡萄！");
        return f;
    }
}
```

8 草莓工厂：StrawberryGardener.java

```
package com.lavasoft.patterns.factorymethod.ybms;
/**
 * Created by IntelliJ IDEA.
 * FileName:StrawberryGardener.java
 * User:    LavaSoft
 * Date:    2006-12-3
 * Time:    17:53:30
 * 《Java 与模式》（--阎宏博士著）读书笔记
 * 工厂模式--工厂方法模式--一般性模式（农场应用）
 * ReadMe:  添加说明
 */
public class StrawberryGardener implements FruitGardener {
    /**
     * 工厂方法
     *
     * @return 草莓
     */
    public Fruit factory() {
        Fruit f = new Strawberry();
        System.out.println("水果工厂（StrawberryGardener）成功创建一
个水果：草莓！");
        return f;
    }
}
```

9 测试类（客户端）：TestApp.java

```
package com.lavasoft.patterns.factorymethod.ybms;
/**
 * Created by IntelliJ IDEA.
 * FileName:TestApp.java
 * User:    LavaSoft
 * Date:    2006-12-3
```

```
 * Time:    17:54:48
 * 《Java 与模式》（--阎宏博士著）读书笔记
 * 工厂模式--工厂方法模式--一般性模式（农场应用）
 * ReadMe:  测试类（客户端）
 */
public class TestApp {
    private FruitGardener f1, f2, f3;
    private Fruit p1, p2, p3;
    private void test() {
        //实力化水果工厂
        f1 = new AppleGardener();
        f2 = new GrapeGardener();
        f3 = new StrawberryGardener();
        //从水果工厂生产水果
        p1 = f1.factory();
        p2 = f2.factory();
        p3 = f3.factory();
    }
        public static void main(String args[]) {
        TestApp test = new TestApp();
        test.test();
    }
}
```

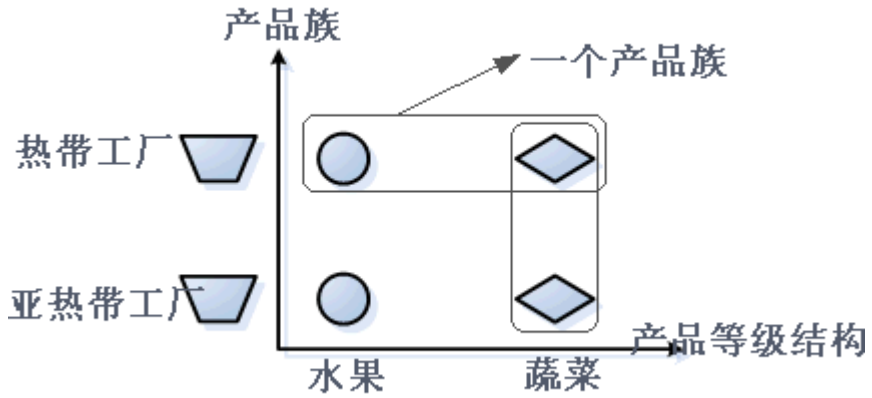
【程序 3】抽象工厂模式在农场中的实现

背景

聪明的农场主总是让自己的庄园越来越有价值,“农场”在经历了简单工厂模式和工厂模式后，不断的扩大生产。如今，再次面临新的大发展，一项重要的工作就是引进塑料大棚技术，在大棚里种植热带（Tropical）和亚热带（Northern）的水果和蔬菜，用以满足市场需求，获取更大的利益。

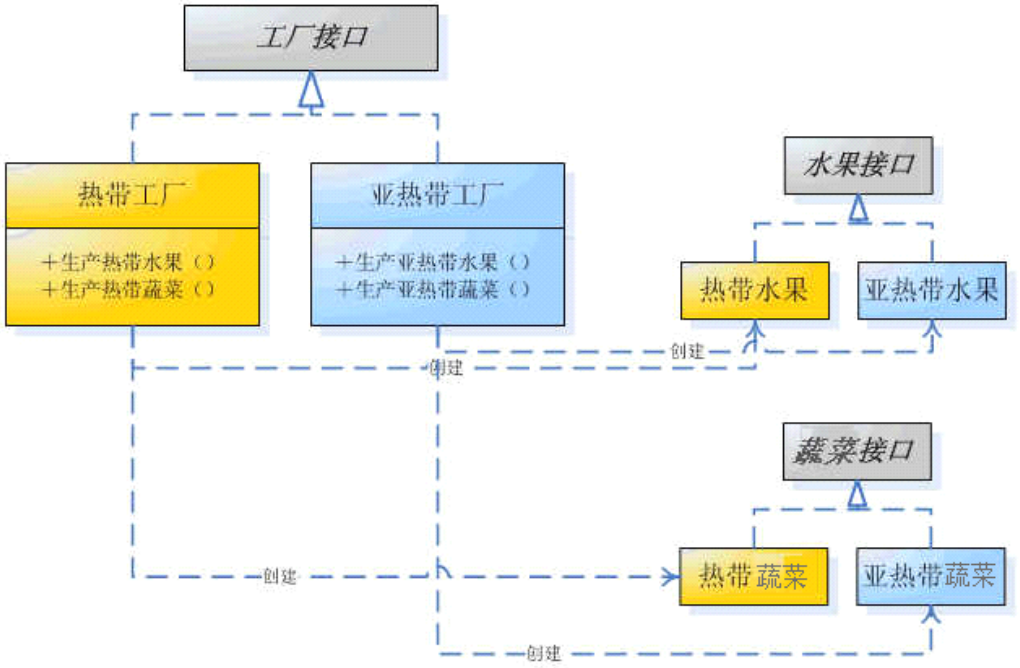
产品角色图

经过分析，对产品角色进行分析得出下图



系统设计

经过分析，所谓的各个园丁其实就是工厂角色，而蔬菜和水果则是产品角色。将抽象工厂模式用于农场中，系统设计图如下：



实现源码

1 抽象工厂：Gardener.java

```
package com.lavasoft.patterns.abstractfactory.ybms;
/**
 * Created by IntelliJ IDEA.
 * FileName:Gardener.java
 * User:     LavaSoft
 * Date:     2006-12-5
 * Time:     22:55:23
 *  《Java 与模式》（--阎宏博士著）读书笔记
```

```
 * 工厂模式--抽象工厂模式--一般性模式（农场应用）
 * ReadMe:  抽象工厂角色：工厂接口
 */
public interface Gardener {
    public Fruit createFruit(String name);
    public Veggie createVeggie(String name);
}
```

2 抽象水果产品：Fruit.java

```
package com.lavasoft.patterns.abstractfactory.ybms;
/**
 * Created by IntelliJ IDEA.
 * FileName:Fruit.java
 * User:    LavaSoft
 * Date:    2006-12-5
 * Time:    22:54:15
 * 《Java 与模式》（--阎宏博士著）读书笔记
 * 工厂模式--抽象工厂模式--一般性模式（农场应用）
 * ReadMe:  抽象产品角色：水果接口
 */
public interface Fruit {
}
```

3 抽象蔬菜产品：Veggie.java

```
package com.lavasoft.patterns.abstractfactory.ybms;
/**
 * Created by IntelliJ IDEA.
 * FileName:Veggie.java
 * User:    LavaSoft
 * Date:    2006-12-5
 * Time:    22:56:22
 * 《Java 与模式》（--阎宏博士著）读书笔记
 * 工厂模式--抽象工厂模式--一般性模式（农场应用）
 * ReadMe:  抽象产品角色：蔬菜接口
 */
public interface Veggie {
}
```

4 热带水果：TropicalFruit.java

```
package com.lavasoft.patterns.abstractfactory.ybms;
/**
 * Created by IntelliJ IDEA.
 * FileName:TropicalFruit.java
 * User:    LavaSoft
```

```
 * Date:    2006-12-5
 * Time:    22:57:08
 * 《Java 与模式》（--阎宏博士著）读书笔记
 * 工厂模式--抽象工厂模式--一般性模式（农场应用）
 * ReadMe:  具体产品角色：热带水果
 */
public class TropicalFruit implements Fruit {
    private String name;
    public TropicalFruit(String name) {
        System.out.println("热带工厂为您创建了：热带水果－"+name);
    }
    public String getName() {
        return name;
    }
    public void setName(String name) {
        this.name = name;
    }
}
```

5 热带蔬菜：TropicalVeggie.java

```
package com.lavasoft.patterns.abstractfactory.ybms;
/**
 * Created by IntelliJ IDEA.
 * FileName:TropicalVeggie.java
 * User:    LavaSoft
 * Date:    2006-12-5
 * Time:    22:58:03
 * 《Java 与模式》（--阎宏博士著）读书笔记
 * 工厂模式--抽象工厂模式--一般性模式（农场应用）
 * ReadMe:  具体产品角色：热带蔬菜
 */
public class TropicalVeggie implements Veggie {
    private String name;
    public TropicalVeggie(String name) {
        System.out.println("热带工厂为您创建了：热带蔬菜－"+name);
    }
    public String getName() {
        return name;
    }
    public void setName(String name) {
        this.name = name;
    }
}
```

6 亚热带水果：NorthernFruit.java

```
package com.lavasoft.patterns.abstractfactory.ybms;
/**
 * Created by IntelliJ IDEA.
 * FileName:NorthernFruit.java
 * User:    LavaSoft
 * Date:    2006-12-5
 * Time:    22:58:55
 * 《Java 与模式》（--阎宏博士著）读书笔记
 * 工厂模式--抽象工厂模式--一般性模式（农场应用）
 * ReadMe:  具体产品角色：亚热带水果
 */
public class NorthernFruit implements Fruit {
    private String name;
    public NorthernFruit(String name) {
        System.out.println("亚热带工厂为您创建了：亚热带水果－"+name);
    }
    public String getName() {
        return name;
    }
    public void setName(String name) {
        this.name = name;
    }
}
```

7 亚热带蔬菜：NorthernVeggie.java

```
package com.lavasoft.patterns.abstractfactory.ybms;
/**
 * Created by IntelliJ IDEA.
 * FileName:NorthernVeggie.java
 * User:    LavaSoft
 * Date:    2006-12-5
 * Time:    22:59:36
 * 《Java 与模式》（--阎宏博士著）读书笔记
 * 工厂模式--抽象工厂模式--一般性模式（农场应用）
 * ReadMe:  具体产品角色：亚热带蔬菜
 */
public class NorthernVeggie implements Veggie {
    private String name;
    public NorthernVeggie(String name) {
        System.out.println("亚热带工厂为您创建了：亚热带蔬菜－"+name);
    }
```

```
    public String getName() {
        return name;
    }
    public void setName(String name) {
        this.name = name;
    }
}
```

8 热带工厂：TropicalGardener.java

```
/**
 * Created by IntelliJ IDEA.
 * FileName:TropicalGardener.java
 * User:    LavaSoft
 * Date:    2006-12-5
 * Time:    23:01:49
 * 《Java 与模式》（--阎宏博士著）读书笔记
 * 工厂模式--抽象工厂模式--一般性模式（农场应用）
 * ReadMe:  具体工厂角色：热带工厂
 */
public class TropicalGardener implements Gardener {
    public Fruit createFruit(String name) {
        return new TropicalFruit(name);
    }
    public Veggie createVeggie(String name) {
        return new TropicalVeggie(name);
    }
}
```

9 亚热带工厂：NorthernGardener.java

```
package com.lavasoft.patterns.abstractfactory.ybms;
/**
 * Created by IntelliJ IDEA.
 * FileName:NorthernGardener.java
 * User:    LavaSoft
 * Date:    2006-12-5
 * Time:    23:00:31
 * 《Java 与模式》（--阎宏博士著）读书笔记
 * 工厂模式--抽象工厂模式--一般性模式（农场应用）
 * ReadMe:  具体工厂角色：亚热带工厂
 */
public class NorthernGardener implements Gardener {
    public Fruit createFruit(String name) {
        return new NorthernFruit(name);
    }
```

```
    public Veggie createVeggie(String name) {
        return new NorthernVeggie(name);
    }
}
```

10 测试类（客户端）：TestApp.java

```
package com.lavasoft.patterns.abstractfactory.ybms;
/**
 * Created by IntelliJ IDEA.
 * FileName:TestApp.java
 * User:    LavaSoft
 * Date:    2006-12-5
 * Time:    23:03:22
 * 《Java 与模式》（--阎宏博士著）读书笔记
 * 工厂模式--抽象工厂模式--一般性模式（农场应用）
 * ReadMe:  测试类（客户端）
 */
public class TestApp {
    private void test(){
        Veggie tv,nv;
        Fruit tf,nf;
        TropicalGardener tg=new TropicalGardener();
        NorthernGardener ng=new NorthernGardener();
            tv=tg.createVeggie("热带菜叶");
        nv=ng.createVeggie("东北甜菜");
        tf=tg.createFruit("海南椰子");
        nf=ng.createFruit("雪梨");
    }
    public static void main(String args[]){
        TestApp test=new TestApp();
        test.test();
    }
}
```

【程序 4】 单列模式

做一个单列模式的类，只加载一次属性文件

```
package com.softeem.demo;
import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.io.IOException;
import java.io.InputStream;
```

```
import java.util.Properties;

/**
*@authorleno
*单子模式，保证在整个应用期间只加载一次配置属性文件
*/
publicclass Singleton {

    privatestatic Singleton instance;
    privatestaticfinal String CONFIG_FILE_PATH =
"E:\\config.properties";
    private Properties config;
    private Singleton()
    {
      config = new Properties();
      InputStream is;
      try {
          is = new FileInputStream(CONFIG_FILE_PATH);
          config.load(is);
          is.close();
      } catch (FileNotFoundException e) {
          // TODO Auto-generated catch block
          e.printStackTrace();
      } catch (IOException e) {
          // TODO Auto-generated catch block
          e.printStackTrace();
      }
    }
    publicstatic Singleton getInstance()
    {
      if(instance==null)
      {
          instance = new Singleton();
      }
      returninstance;
    }
    public Properties getConfig() {
      returnconfig;
    }
    publicvoid setConfig(Properties config) {
      this.config = config;
    }
```

```
}
```