demo1~8

```
import java.util.*;

public class Solution {

    Scanner scanner = new Scanner(System.in);

    public static void main(String[] args) {
        Solution solution = new Solution();
        System.out.println("请输入三位数");
        solution.demo1();
        System.out.println("输入正方形的边长");
        solution.Demo12();
        System.out.println("请输入三个数");
        solution.Demo21();
        System.out.println("输入正确题目");
        solution.Demo22();
        System.out.println("请入如距离");
        solution.Demo23();
        System.out.println("请输入人数和年龄");
        solution.Demo31();
        System.out.println("请输入n与a");
        solution.Demo32();
        System.out.println("请输入月份");
        solution.Demo33();
        System.out.println("高空抛物的高度和次数");
        solution.Demo34();
        System.out.println("我要打印乘法表啦");
        solution.Demo41();
        System.out.println("输入一个正数");
        solution.Demo42();
        System.out.println("猜数字");
        solution.Demo43();
        System.out.println("输入菱形边长");
        solution.Demo44();
        System.out.println("输入数量和数字");
        solution.Demo51();
        System.out.println("数字环，数量，数字，偏移位置");
        solution.Demo52();
        System.out.println("找一个可以活着的位置");
        solution.Demo53();
        System.out.println("输入正方矩阵边长和数字");
        solution.Demo61();
        System.out.println("输入正方矩阵边长和数字");
        solution.Demo62();
        System.out.println("杨辉三角的高度");
        solution.Demo63();
        System.out.println("输入正方矩阵边长");
        solution.Demo64();
        System.out.println("输入含英文的字符串");
```

```
        solution.Demo71();
        System.out.println("输入字符串");
        solution.Demo72();
        System.out.println("输入密码");
        solution.Demo73();
        System.out.println("密码加密");
        solution.Demo74();
        System.out.println("要输入id和密码辣");
        solution.showLoginPage();
    }

//    设计一个程序, 输入三位数a, 分别输出个,十,百位.
    public void demo1(){
        //--变量声明--
        // 输入值
        int a;
        // 个十百位
        int r1, r2, r3;
        //--接收输入--
        a = scanner.nextInt();
        //--数据处理--
    /*
        a%10表示取个位数
        a/10将数字缩小10倍    也就是十位数变成个位数
    */
        r1 = a % 10;
        r2 = a / 10 % 10;
        r3 = a / 100;
        //--输出--
        System.out.println(r3);
        System.out.println(r2);
        System.out.println(r1);
    }

    // 求正方形和圆形的面积差
    void Demo12() {
        //--变量声明--
        // PI
        final double PI = 3.14;
        // 输入值
        int l;
        // 正方形面积, 圆形面积.
        double squ_s, cir_s;
        // 结果
        double result;
        //--接收输入--
        l = scanner.nextInt();
        //--数据处理--
    /*
        正方形 = l^2
        圆形 = PI*r^2
    */
        squ_s = l * l;
        cir_s = l * l * PI / 4;
        result = squ_s - cir_s;
```

```
        //--输出--
        System.out.printf("%.2f", result);
        System.out.println();
    }

    //最大数
    void Demo21() {
        //--变量声明--
        //输入值
        int a, b, c;
        //最大值
        int max;
        //--接收输入--
        a = scanner.nextInt();
        b = scanner.nextInt();
        c = scanner.nextInt();
        //--数据处理--
        max = a;
        if (b > max) {
            max = b;
        }
        if (c > max) {
            max = c;
        }
        //--输出--
        System.out.println(max);
    }

    // 打分系统
    void Demo22() {
        //--变量声明--
        // 输入
        int n;
        // 输出
        int result;
        //--接收输入--
        n = scanner.nextInt();

        //--数据处理--
        if (n <= 10) result = n * 6;
        else if (n <= 20) result = 10 * 6 + (n - 10) * 2;
        else result = 10 * 6 + 10 * 2 + (n - 20);
        //--输出--
        System.out.println(result);
    }

    // 上班
    void Demo23() {
        //--变量声明--
        // 输入
        int n;
        // 负数表示骑车快，正数表示走路快，0表示一样快
        double result;
```

```
        //--接收输入--
        n = scanner.nextInt();

        //--数据处理--
        //用骑车速度减去走路速度  如果是正数表示骑车耗时多
        result = (27 + 23 + n / 3.0) - (n / 1.2);

        //--输出--
        // 注意double判断会有精度问题  不能直接判断==0
        if (result < 0.0001) System.out.println("骑车");
        else if (result > 0.0001) System.out.println("走路");
        else System.out.println("一样快");
    }

    // 求平均年龄
    void Demo31() {
        //--变量声明--
        // 人数, 临时记录变量, 学长的年龄总和
        int n, temp;
        double age = 0;

        //--接收输入--
        // 注意这里age+=temp   也就是存储了所有学长年龄的和
        n = scanner.nextInt();
        for (int i = 0; i < n; i++) {
            temp = scanner.nextInt();
            age += temp;
        }

        //--数据处理--
        age /= n;

        //--输出--
        System.out.printf("%.2f\n", age);
    }

    // 求2222222
    void Demo32() {
        //--变量声明--
        // 输入n, 输入a, 累加数存储变量, 答案存储变量
        int n, a;
        int num, result = 0;

        //--接收输入--
        n = scanner.nextInt();
        a = scanner.nextInt();
        num = a;

        //--数据处理--
    /*
        num用于存储a,aa,aaa这样的数字

        num = aaa
        num*10 = aaa0
        num*10+a = aaaa
```

```
    */
        for (int i = 0; i < n; i++) {
            result += num;
            num = num * 10 + a;
        }

        //--输出--
        System.out.println(result);
    }

    // 数兔子
    void Demo33() {
        //--变量声明--
        // 用户输入值  月数
        int n;
        // 用于存储当月 以及前1,2个月的兔子数
        int m1 = 1, m2 = 1, m3 = 0;

        //--接收输入--
        n = scanner.nextInt();

        //--数据处理--
        //前两个月不生兔子
        // 每个月的兔子数量   =  上一个月兔子数 + 上上一个月兔子数
        n -= 2;
        while (n-- != 0) {
            m3 = m1 + m2;
            m1 = m2;
            m2 = m3;
        }

        //--输出--
        System.out.println(m3);
    }

    // 弹球
    void Demo34() {
        //--变量声明--
        // 初始条件
        int N, M;
        // 高度和距离
        double h, l = 0;

        //--接收输入--
        M = scanner.nextInt();
        N = scanner.nextInt();
        h = M;
        //--数据处理--
        while(N-- != 0){
            h /= 2;
            l += h * 3;
        }

        //--输出--
        System.out.printf("%.2f, %.2f", h, l);
```

```
        System.out.println();
    }

    // 乘法表
    void Demo41() {
        //--变量声明--

        //--接收输入--

        //--数据处理--

        //--输出--
        for (int i = 1; i <= 9; i++) {
            for (int j = 1; j <= i; j++) {
                System.out.printf("%d * %d = %d\t", j, i, i * j);
            }
            System.out.println();
        }
    }


    // 求素数
    void Demo42() {
        //--变量声明--
        // 输入值
        int N;
        // 标识符  值为1表示素数  值为0表示非素数
        boolean flag;
        //--接收输入--
        N = scanner.nextInt();
        //--数据处理--
        //--输出--
        for (int i = 2; i <= N; i++) {
            flag = false;
            for (int j = 2; j < i/*sqrt(i)*/; j++) {
                // 如果i%j为0  则该数字是非素数   0表示假  !取反为真
                if (i % j == 0){
                    flag = true;
                    break;
                }
            }
            //如果flag是0  表示是素数 打印
            if (!flag) System.out.println(i);
        }
    }

    // 猜数字
    void Demo43() {
        //--变量声明--
        // 用户输入 让程序猜的数字
        int n;
        // 程序猜测的次数, 二分猜测法的上下限
        int count = 0, max = 100, min = 0;
        //猜测值
        int mid;
```

```
        //--接收输入--
        n = scanner.nextInt();
        do{
            //--数据处理--
        /*
            猜测值 = (上限 + 下限) /2
            如上限100   下限0    猜测值=50
            上限100   下限50   猜测值=75

            如果猜测值比n小,  则下限 = 猜测值+1
            如果猜测值比n大,  则上限 = 猜测值-1
        */
            count++;
            mid = (max + min) / 2;
            if (mid > n) {
                max = mid - 1;
            }
            else if (mid < n) {
                min = mid + 1;
            }

            //--输出--
            System.out.println(mid);

        } while (mid != n);
        System.out.println("最终猜测了" + count + "次" );
    }

    // 打印菱形
    void Demo44() {
        //--变量声明--
        //菱形边长
        int n;
        //--接收输入--
        n = scanner.nextInt();

        //--数据处理--
    /*
        用坐标系的思路
        设坐标点(x,y)
        当x的绝对值 + y的绝对值   小于n时,   这个坐标点在菱形内
        当x的绝对值 + y的绝对值   大于等于n时, 这个坐标点在菱形外

        如: (2,2)在菱形内, (3,2)在菱形外, (-4,0)在菱形内, (4,1)在菱形外
    */
        //--输出--
        // 当n为5时, 坐标系的上下限分别是-4~4   最大高/宽度为9
        for (int i = -n+1; i < n; i++) {
            for (int j = -n+1; j < n; j++) {

                // i的绝对值+j的绝对值 小于n   则在菱形范围内
                if (Math.abs(i) + Math.abs(j) < n) {
                    System.out.print("*");
                }
                else {
```

```
                System.out.print(" ");
            }
        }
        System.out.println();
    }
}

// 去重
void Demo51() {
    //--变量声明--
    //数据总数，去重后的数量
    int N, len;
    //待去重的数据
    int[] arr = new int[100];

    //--接收输入--
    N = scanner.nextInt();
    len = N;
    for (int i = 0; i < N; i++) {
        arr[i] = scanner.nextInt();
    }
    //--数据处理--
/*
    将每个元素都和后面的元素进行判断
        如果[i]和[j]值重复，将[j]的值设为-1.  并且len--
    {1,3,2,6,-1,-1,4,8,-1,-1}

    都判断一遍后，将值为-1的元素  用后一个元素值进行覆盖
    {1,3,2,6,4,8}
*/
    for (int i = 0; i < N; i++) {
        for (int j = i + 1; j < N; j++) {
            if (arr[i] == -1) break;

            if (arr[i] == arr[j]) {
                arr[j] = -1;
                len--;
            }
        }
    }
/*
    快慢指针思路

    i表示慢下标，j表示快下标
    快下标：每次循环往后移动一位
    慢下标：条件符合时往后移动一位

    每次循环时，arr[i] = arr[j]
    慢下标条件：覆盖后，如果当前[i]的值不为-1  则i++(往后移动一位)

    下面样例的截取点是在arr[i] = arr[j]后  i++前

    ij
    1       2       -1      3       -1      4       -1      5       6
```

```
            ij
    1       2       -1      3       -1      4       -1      5       6

                    ij
    1       2       -1      3       -1      4       -1      5       6

                    i       j
    1       2       3       3       -1      4       -1      5       6

                            i       j
    1       2       3       -1      -1      4       -1      5       6

                            i               j
    1       2       3       4       -1      4       -1      5       6

                                    i               j
    1       2       3       4       -1      4       -1      5       6

                                    i                       j
    1       2       3       4       5       4       -1      5       6

                                            i                       j
    1       2       3       4       5       6       -1      5       6
*/
    for (int i = 0, j = 0;
         j < N;
         j++)
    {

        arr[i] = arr[j];
        if (arr[i] != -1) {
            i++;
        }

    }


    //--输出--
    for (int i = 0; i < len; i++) {
        System.out.printf("%d ", arr[i]);
    }
    System.out.println();
}

// 数字环
void Demo52() {
    //--变量声明--
    // 数字环存储数组, 长度, 移动位数
    int[] arr, result;
    int len, m;
    // 转换后数组

    //--接收输入--
    len = scanner.nextInt();
    arr = new int[len];
```

```
        result = new int[len];
        for (int i = 0; i < len;i++) {
            arr[i] = scanner.nextInt();
        }
        m = scanner.nextInt();

        //--数据处理--
        for (int i = 0; i < len; i++) {
            result[(i + m) % len] = arr[i];
        }

        //--输出--
        for (int i = 0; i < len; i++) {
            System.out.printf("%d ", result[i]);
        }
        System.out.println();
    }

    // 约瑟夫环
    void Demo53() {
        //--变量声明--
        // 玩家数量, 被枪毙数, 玩家当前状况
        // 值为1表示存活  值为0表示被枪毙了
        int n, m;
        int[] arr = new int[100];
        // 目前活着的人数, flag用于标记当前数到几
        int num, flag = 0;

        //--接收输入--
        n = scanner.nextInt();
        m = scanner.nextInt();
        num = n;
        for (int i = 0; i < n; i++) {
            arr[i] = 1;
        }

        //--数据处理--
        // 当num值为1时, 游戏结束
        for (int i = 0; num - 1 != 0; i++) {
            if (i == n) i = 0;

            if (arr[i] != 0) {
                flag++;
                if (flag == m) {
                    arr[i] = 0;
                    flag = 0;
                    num --;
                }
            }
        }
        //--输出--
        for (int i = 0; i < n; i++) {
            if (arr[i] != 0) System.out.printf("%d", i + 1);
        }
        System.out.println();
```

```
}

// 矩阵转置
void Demo61() {
    //--变量声明--
    // 矩阵边长，矩阵本身，转置后数组
    int[][] arr, result;
    int size;
    //--接收输入--
    size = scanner.nextInt();
    arr = new int[size][size];
    result = new int[size][size];
    for (int i = 0; i < size; i++) {
        for (int j = 0; j < size; j++) {
            arr[i][j] = scanner.nextInt();
        }
    }
    //--数据处理--
    for (int i = 0; i < size; i++) {
        for (int j = 0; j < size; j++) {
            result[j][i] = arr[i][j];
        }
    }


    //--输出--
    for (int i = 0; i < size; i++) {
        for (int j = 0; j < size; j++) {
            System.out.printf("%d ", result[i][j]);
        }
        System.out.println();
    }
}

// 颈椎病
void Demo62() {
    //--变量声明--
    // 矩阵边长，矩阵本身，转置后数组
    int[][] arr, result;
    int size;
    //--接收输入--
    size = scanner.nextInt();
    arr = new int[size][size];
    result = new int[size][size];
    for (int i = 0; i < size; i++) {
        for (int j = 0; j < size; j++) {
            arr[i][j] = scanner.nextInt();
        }
    }

    //--数据处理--
/*
    假设size为3
    [0][0]->[0][2]
    [0][1]->[1][2]
```

```
        [1][2]->[2][1]
        [2][1]->[1][0]
        规律 x转y    y转x
        x转y时 size-x=y
        y转x时 y=x
    */
        for (int i = 0; i < size; i++) {
            for (int j = 0; j < size; j++) {
                result[j][size - i - 1] = arr[i][j];
            }
        }
        //--输出--
        for (int i = 0; i < size; i++) {
            for (int j = 0; j < size; j++) {
                System.out.printf("%d ", result[i][j]);
            }
            System.out.println();
        }
    }

    // 杨辉三角
    void Demo63() {
        //--变量声明--
        // 杨辉三角的高度
        int n;
        // 存储用数组,  注意杨辉三角只会占用一半的空间, 所以这里可以用malloc动态分配来避免空
间浪费
        int[][] arr = new int[10][10];

        //--接收输入--
        n = scanner.nextInt();
        arr[0][0] = 1;

        //--数据处理--
        // 左右两侧数据直接填充即可
        for (int i = 1; i < n; i++) {
            arr[i][0] = arr[i][i] = 1;
            for (int j = 1; j < i; j++) {
                arr[i][j] = arr[i - 1][j - 1] + arr[i - 1][j];
            }
        }
        //--输出--
        for (int i = 0; i < n; i++) {
            for (int j = 0; j <= i; j++) {
                System.out.printf("%d ", arr[i][j]);
            }
            System.out.println();
        }
    }

    // 包围圈
    void Demo64() {
        //--变量声明--
        int[][] tree = new int[10][10];
        int size;
```

```
        // 当前坐标xy, 移动方向toward 0左 1下 2右 3上, 填充数
        // 注意 由于第一次探查前会先移动 x++  所以x需要修正为-1
        int x = -1, y = 0, toward = 0, num = 0;
        // 一个方向的移动步数
        int s;

        //--接收输入--
        size = scanner.nextInt();
        s = size;

        //--数据处理--
        // 移动规律  如果是4  则是4+3+3+2+2+1+1 = 16  同理 如果是5  5+4+4+3+3+2+2+1+1
        // 外部循环: 每奇数次循环,  最大步数-1.  一个方向移动完后 对方向进行修改
        // 内部循环: 循环(步数)次, 每次移动根据方向对xy进行修改

        for (int i = 0; num<size*size; i++) {
            // 奇数次循环 最大步数-1
            if (i % 2 == 1) s--;
            for (int j = 0; j < s; j++) {
                // 获取方向并移动
                switch (toward) {
                    case 0:
                        x++;
                        break;
                    case 1:
                        y++;
                        break;
                    case 2:
                        x--;
                        break;
                    case 3:
                        y--;
                        break;
                }

                tree[y][x] = ++num;
            }

            //  改变方向  注意 当toward为3  也就是向上时, 需要取模操作 也就是变为0
            // (1+1) %4 = 2   (3+1) %4=0
            toward = (toward + 1) % 4;
        }


        //--输出--
        for (int i = 0; i < size; i++) {
            for (int j = 0; j < size; j++) {
                System.out.printf("%d ", tree[i][j]);
            }
            System.out.println();
        }
    }

    // 大小写转换
    void Demo71() {
```

```
        //--变量声明--
        // 存放字符串的数组
        String str = scanner.nextLine();
        //--接收输入--

        //--数据处理--
        // 判断ascii码是否在大小写范围内  如果在  则手动转换
        char[] chars = str.toCharArray();
        for (int i = 0; i < chars.length; i++) {
//              要不要好奇 'a' = ?
//              'A' = ?
//              if (chars[i]  >= 'a' && chars[i] <= 'z') chars[i] += 32; 行不行（狗
头）
            if (chars[i]  >= 65 && chars[i] <= 90) chars[i] += 32;
            else if (chars[i] >= 97 && chars[i] <= 122) chars[i] -= 32;
        }
        System.out.println(new String(chars));
    }

    // 字符串反转
    void Demo72() {
        //--变量声明--
        // 存放字符串的数组, 存放答案的数组
        String str;
        char[]  result;
        // 字符串的长度
        int len;
        //--接收输入--
        str = scanner.nextLine();
        //--数据处理--
        // 获取长度
        len = str.length();
        result = new char[len];
        // 倒序填入result字符串
        for (int i = 0; i < len; i++) {
            result[i] = str.charAt(len - i - 1);
        }
        //--输出--
        System.out.println(new String(result));
    }

    // 安全密码
    void Demo73() {
        //--变量声明--
        // 用户输入
        String password;
        // 用于验证是否合法, 0:长度, 1:开头大写, 2:包含小写, 3:包含数字, 4:包含特殊符号
        boolean[] flag = new boolean[5];

        //--接收输入--
        password = scanner.nextLine();

        //--数据处理--
        //开头大写判断
```

```
        if (password.charAt(0) >= 65 && password.charAt(0) <= 90) flag[0] =
true;

        int length = password.length();
        for (int i = 1; i < length; i++) {
            //小写字母
            char c = password.charAt(i);
            if (c >= 97 && c <= 122) flag[2] = true;
                //数字
            else if (c >= 48 && c <= 57) flag[3] = true;
                //特殊符号
            else if (c == '~' || c == '!' || c == '@' || c == '#' || c == '$' ||
c == '%' || c == '*')
                flag[4] = true;

            // 长度
            flag[1] = (i >= 8 && i <= 16);
        }

        //--输出--
        if (flag[0] && flag[1] && flag[2] && flag[3] && flag[4]) {
            System.out.println("true");
        }
        else {
            System.out.println("false");
        }
    }

    // 加密
    void Demo74() {
        //--变量声明--
        String password;

        //--接收输入--
        password = scanner.nextLine();
        //--数据处理--
    /*
        思路: 先减到0~25  然后+5取模
        如果超过25了(字母xyz), 则会因为取模成bcd

        比如z
        90 - 65=25
        25 + 5 = 30  %26 = 4
        4+65 = 69  = e;

        数字同理
    */
        char[] chars = password.toCharArray();
        for (int i = 0; i < chars.length; i++) {
            //大写字母
            if (chars[i] >= 65 && chars[i] <= 90) {
                chars[i] = (char) ((chars[i] - 65 + 5) % 26 + 65);
            }
            //小写字母
            else if (chars[i] >= 97 && chars[i] <= 122) {
```

```
                chars[i] = (char) ((chars[i] - 97 + 5) % 26 + 97);
            }
            //数字
            else if (chars[i] >= 48 && chars[i] <= 57) {
                chars[i] = (char) ((chars[i] - 48 + 5) % 10 + 48);

            }
        }

        //--输出--
        System.out.println(new String(chars));
    }


    int[] ids  = { 10001,10002,10003,10004, 0};
    String[] names = {"张三", "李四", "王五", "赵六", ""};
    String[] passwords = { "aaaaa", "bbbbb", "ccccc", "ddddd", ""};
    int uNum = 4;
    /*
    功能: 根据id 查询用户是否存在, 如果存在返回用户名,  如果不存在返回空
    参数:
        uid: 用户id
    返回值:
        如果用户存在, 返回用户名.
        如果用户不存在, 返回NULL
*/

    //函数
    String selectUserById(int uid) {
        for (int i = 0; i < uNum; i++) {
            if (ids[i] == uid) {
                return names[i];
            }
        }
        return null;
    }


    /*
        功能: 根据id 查询用户密码, 如果存在返回用户密码,  如果不存在返回空
        参数:
            uid: 用户id
        返回值:
            如果密码存在, 返回密码.
            如果密码不存在, 返回NULL
    */
    String selectPassById(int uid) {
        for (int i = 0; i < uNum; i++) {
            if (ids[i] == uid) {
                return passwords[i];
            }
        }
        return null;
    }
```

```
    /*
        功能: 传入用户id和密码, 根据上面两个函数(selectUserById, selectPassById)来获取相应用户数据, 并判断是否登录成功
            传入用户id 查询用户名是否存在, 并获取用户密码
            如果用户存在  则判断密码是否正确
        参数:
            uid: 用户账户
            password: 用户密码
        返回值:
            如果账号不存在, 返回1
            如果密码错误, 返回2
            如果登录成功, 返回0
    */
    int login(int uid, String password) {
        if (selectUserById(uid) == null) {
            return 1;
        }
        String pass = selectPassById(uid);
        return pass.equals(password) ? 0 : 2;

    }

    /*
        功能: 提示用户输入账号密码, 根据login函数判断是否登录成功,
        如果登录成功提示正在进入加载界面
        如果登录失败
            密码错误: 提示密码错误, 并让用户重新登录
            账号不存在: 提示账号不存在, 并提示正在进入注册界面
        参数: 无
        返回值: 无
    */
    void showLoginPage() {
        //--变量声明--
        //用户id
        int uid;
        //用户密码
        String password;
        //结果选项
    /*  如果账号不存在, 返回1
        如果密码错误, 返回2
        如果登录成功, 返回0
        */
        int result;

        //--接收输入--
        System.out.println("请输入账号");
        uid = scanner.nextInt();
        System.out.println("请输入密码");
        scanner.nextLine();
        password = scanner.nextLine();
        //--数据处理--
        result = login(uid, password);

        //--输出--
```

```
        switch (result) {
            case 1:
                System.out.print("账号不存在，正在进入注册界面");
                break;
            case 2:
                System.out.print("密码错误，请重新登录");
                break;
            case 0:
                System.out.print("登录成功，正在进入首页");
                break;
        }

    }


}
```

demo9 1

```
public class Management {
    private HashMap<Integer, Person> container;

    public Management() {
        this.container = new HashMap<Integer, Person>();
    }

    public boolean add(Person person) {
        if (container.containsKey(Objects.hashCode(person))) {
            return false;
        }
        container.put(person.hashCode(), person);
        person.addToManagement();
        return true;
    }

    /**
     * 因为删除学生和删除工人大差不差,所以封装了一下.
     * removeFromManagement()以及addToManagement():更新对应类型的次数
     * @param sid
     * @param type
     * @return
     */
    public boolean del(int sid, Class type) {
        if (!container.containsKey(sid) || container.get(sid).getClass() !=
type) {
            return false;
        }

        container.get(sid).removeFromManagement();
        container.remove(sid);
        return true;
    }
```

```
    public boolean delWorker(int sid) {
        return del(sid,Worker.class);
    }

    public boolean delStudent(int sid) {
        return del(sid,Student.class);
    }

    public void showStudent() {
        System.out.println("系统学生数据:");
        container.values().forEach((o1)->{
            if(o1 instanceof Student){
                o1.show();
            }
        });
    }

    public void showWorker() {
        System.out.println("系统工人数据:");
        container.values().forEach((o1)->{
            if(o1 instanceof Worker){
                o1.show();
            }
        });
    }
    public void showStat() {
        System.out.printf("系统统计数据:\n学生人数: %d\n工人人数:
%d\n",Student.count,Worker.count);
    }
}



public abstract class Person {
    private int age;
    private String name;
    private boolean sex;
    public static int count;

    public int getAge() {
        return age;
    }

    public void setAge(int age) {
        this.age = age;
    }

    public String getName() {
        return name;
    }

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

```
    public boolean getSex() {
        return sex;
    }

    public void setSex(boolean sex) {
        this.sex = sex;
    }

    public Person(int age, String name, boolean sex) {
        this.age = age;
        this.name = name;
        this.sex = sex;
    }

    /**
     * 多态实现计数
     */
    public void addToManagement() {
        count++;
    }

    public void removeFromManagement() {
        count--;
    }

    public Person() {
    }

    public abstract void show();

    @Override
    public String toString() {
        return "姓名: " + name + " " +
                "年龄: " + age + " " +
                "性别: " + (sex?"女":"男");
    }

    @Override
    public boolean equals(Object o) {
        if (this == o) return true;
        if (o == null || getClass() != o.getClass()) return false;
        Person person = (Person) o;
        return age == person.age && sex == person.sex &&
name.equals(person.name);
    }

    @Override
    public int hashCode() {
        return Objects.hash(age, name, sex);
    }
}


public class Student extends Person {
```

```
    private int sid;
    public static int count;


    @Override
    public void show() {
        System.out.println(toString());
    }

    @Override
    public String toString() {
        return "学号: " + sid + " " + super.toString() + " ";
    }

    public int getSid() {
        return sid;
    }


    @Override
    public boolean equals(Object o) {
        if (this == o) return true;
        if (o == null || getClass() != o.getClass()) return false;
        if (!super.equals(o)) return false;
        Student student = (Student) o;
        return sid == student.sid;
    }

    @Override
    public int hashCode() {
        return super.hashCode() + 1;
    }

    public Student(int age, String name, boolean sex) {
        super(age, name, sex);
        this.sid = hashCode();
    }

    public Student() {
    }

    @Override
    public void addToManagement() {
        super.addToManagement();
        count++;
    }

    @Override
    public void removeFromManagement() {
        super.removeFromManagement();
        count--;
    }
}

public class Worker extends Person {
```

```
    private int wid;
    public static int count;

    @Override
    public void show() {
        System.out.println(toString());
    }

    @Override
    public String toString() {
        return "工号: " + wid + " " + super.toString() ;
    }


    public int getWid() {
        return wid;
    }

    @Override
    public boolean equals(Object o) {
        if (this == o) return true;
        if (o == null || getClass() != o.getClass()) return false;
        if (!super.equals(o)) return false;
        Worker worker = (Worker) o;
        return wid == worker.wid;
    }

    @Override
    public int hashCode() {
        return super.hashCode() + 2;
    }

    public Worker(int age, String name, boolean sex) {
        super(age, name, sex);
        this.wid = hashCode();
    }

    public Worker() {
    }

    @Override
    public void addToManagement() {
        super.addToManagement();
        count++;
    }

    @Override
    public void removeFromManagement() {
        super.removeFromManagement();
        count--;
    }
}
```

demo10 1

```
public class MyNode {
    MyNode next;
    int data;
    void add(MyNode head, int val){
        MyNode aNode = new MyNode();
        aNode.data = val;
        MyNode cur = head.next;
        if(cur == null || cur.data > val){
            aNode.next = cur;
            head.next = aNode;
            return;
        }
        while(cur.next != null && cur.next.data <= val){
            cur = cur.next;
        }
        aNode.next = cur.next;
        cur.next = aNode;
    }
    public static void main(String[] args) {
        MyNode head = new MyNode();
        head.add(head, 5);
        head.add(head,1);
        head.add(head,3);
        head.add(head,2);
        head.add(head,4);
        MyNode temp = head.next;
        while(temp != null){
            System.out.print(temp.data);
            temp = temp.next;
        }
    }
}
```

demo10 2

```
public class MyArrayList<E> implements MyList<E> {
    private E[] arr;
    private int size;

    public MyArrayList() {
        arr = (E[]) new Object[10];
        this.size = 0;
    }

    @Override
    public int size() {
        return size;
    }

    @Override
```

```
    public E set(int index, E e) {
        if (index >= size) return null;
        E old = arr[index];
        arr[index] = e;
        return old;
    }

    @Override
    public boolean isEmpty() {
        return this.size == 0;
    }


    @Override
    public E get(int i) {
        if (i >= this.size) {
            //越界了
            return null;
        }
        return (E) arr[i];
    }



    @Override
    public E remove(int i) {
        if (i >= this.size) {
            //照例判定
            return null;
        }
        E old = arr[i];
        //向前覆盖
        if (size - i > 0) System.arraycopy(arr, i + 1, arr, i, size - i);
        size--;
        return old;
    }


    public boolean add(E e) {
        insert(size, e);
        return true;
    }
    public void insert(int i, E e) {
        if (i > this.size) {
            //没有连续插入
            return;
        }
        if (this.size == this.arr.length - 1) {
            //满了,扩容
            //新数组
            E[] newArr = (E[]) new Object[this.arr.length << 1];
            //拷贝原来的数据
            System.arraycopy(this.arr, 0, newArr, 0, this.size);
            this.arr = newArr;
        }
```

```
        //向后移
        if (this.size - i >= 0) System.arraycopy(arr, i, arr, i + 1, this.size -
i);
        //正常插入
        this.arr[i] = e;
        this.size++;
    }

    public static void main(String[] args) {
        MyList<String> list = new MyArrayList();
        for(int i = 0; i < 11; i++){
            list.add(i+"");
        }
        System.out.println(list.remove(5));
        System.out.println(list.set(5,"6"));
        System.out.println(list.add("11"));
        for(int i = 0; i < 15; i++){
            System.out.print(list.get(i) + "\t");
        }

    }
}
```

demo11 1

```
public class Demo111 extends JPanel{
    class MyPoint{
        //    具体点的位置
        private final Point point;
        //    线条颜色
        private final int color;
        //    是否与下一个点相连
        private final boolean hasNext;
        //    线的粗细
        private final float width;
        public MyPoint(Point point, boolean hasNext, int color, float width) {
            this.point = point;
            this.hasNext = hasNext;
            this.color = color;
            this.width = width;
        }

        public float getWidth() {
            return width;
        }

        public Point getPoint() {
            return point;
        }

        public boolean isHasNext() {
            return hasNext;
        }
```

```
    public int getColor() {
        return color;
    }
}
public static void main(String[] arg) {
    //新建框 设置名称
    JFrame frame = new JFrame("Draw lines");
    frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    //新建面板组件
    Demo111 pane = new Demo111();
    //向框架中注册面板组件
    frame.getContentPane().add(pane);
    //根据窗口里面的布局及组件的preferedSize来确定frame的最佳大小
    frame.pack();
    //设置框架可视
    frame.setVisible(true);
}
//存储4种颜色
Color[] colors = {Color.red, Color.pink, Color.black, Color.white};
//初始颜色红
int color = 0;
//初始笔宽1.0
float width = 1.0f;
//记录笔的宽度
private final ArrayList<MyPoint> points;
public Demo111() {
    //设置可以聚焦，方便接受键盘信号
    setFocusable(true);
    //记录一些列点
    points = new ArrayList<>();
    //创建并注册画线事件监听器
    LineListener listener=new LineListener();
    addMouseListener(listener);
    //创建并注册键盘事件监听器
    addKeyListener(new ColorAndSizeListener());
    addMouseMotionListener(listener);
    //设置背景板颜色为白
    setBackground(Color.white);
    //设置大小
    setPreferredSize(new Dimension(800,600));

    int width = Toolkit.getDefaultToolkit().getScreenSize().width;
    int height = Toolkit.getDefaultToolkit().getScreenSize().height;

}


@Override
public void paintComponent(Graphics page) {
    //将构造函数中设置的颜色之类的执行
    super.paintComponent(page);
    //两点一线，必须有两个点以上
    if(points.size()>=2){
        for(int i = 1; i < points.size(); i++){
```

```
                //获取之前的点
                MyPoint myPoint = points.get(i - 1);
                //判断两点是否相连
                if (myPoint.isHasNext()) {
                    //设置线的宽度
                    BasicStroke stokeLine = new BasicStroke(myPoint.getWidth());
                    //将线宽注册进页面中
                    Graphics2D g2 = (Graphics2D) page;
                    g2.setStroke(stokeLine);
                    //设置线
                    page.setColor(colors[myPoint.getColor()]);
                    //获取点的位置并连线
                    Point pointOne = myPoint.getPoint();
                    Point pointTwo = points.get(i).getPoint();
                    page.drawLine(pointOne.x, pointOne.y, pointTwo.x,
pointTwo.y);
                }
            }
        }

    }

    //键盘监听器
    private class ColorAndSizeListener implements KeyListener{

        //当点击相关键盘时的动作
        //a, d替换颜色
        //q, e替换笔的大小
        @Override
        public void keyTyped(KeyEvent e) {
            char key = e.getKeyChar();
            if ('a' == key) --color;
            else if ('d' == key) ++color;
            else if ('q' == key) --width;
            else if ('e' == key) ++width;
            if (width > 25) width = 25;
            else if (width < 0) width = 1;
            if (color < 0) color = 3;
            else if (color > 3) color = 0;
        }

        @Override
        public void keyPressed(KeyEvent e) {

        }

        @Override
        public void keyReleased(KeyEvent e) {

        }
    }

    private class LineListener implements MouseListener,MouseMotionListener {
        public void mousePressed(MouseEvent event) {}
        //当鼠标拖动时记录点的位置，是否连成线，线的颜色，线的宽度
```

```
        public void mouseDragged(MouseEvent event){
            Point point = event.getPoint();
            points.add(new MyPoint(point, true, color, width));
            repaint();
        }
        public void mouseClicked(MouseEvent event){}

        //当鼠标放松时，记录最后一个点的相关信息
        public void mouseReleased(MouseEvent event){
            Point point = event.getPoint();
            points.add(new MyPoint(point, false, color, width));
            repaint();
        }
        public void mouseEntered(MouseEvent event){}
        public void mouseExited(MouseEvent event){}
        public void mouseMoved(MouseEvent event){}
    }
}
```

demo11 2

```
/**
 * 实现一个简易的计算器
 * 二数四则运算
 */
public class Demo112 extends JFrame implements ActionListener {

    // 定义界面上的按钮控件
    final String JButtonName[] = { "7", "8", "9", "+", "4", "5", "6", "-", "1",
"2", "3", "×", ".", "0", "=", "÷" };
    // 定义字体内容
    Font font = new Font("宋体", Font.PLAIN, 17);
    // 定义面板，有二个面板 ， 第一个就是 输入框和ce ,第二个就是下面的操作的按钮
    JPanel p1, p2;
    // 创建按钮数组
    JButton Jb[] = new JButton[JButtonName.length];
    // 创建CE
    JButton ce;
    // 创建输入框
    JTextField tf;
    // 创建构造方法，使用继承了JFrame后，界面的初始化是需要在无参构造方法中实现
    public Demo112() {
        // 为每一个按钮进行点击事件
        for (int i = 0; i < JButtonName.length; i++) {
            // 实例化并声明按钮名称
            Jb[i] = new JButton("" + JButtonName[i]);
            // 设置字体
            Jb[i].setFont(font);
            // 设置按钮透明
            Jb[i].setContentAreaFilled(false);
            Jb[i].addActionListener(this);
```

```
            //添加监听器
            Jb[i].addKeyListener(new KeyListener() {
                @Override
                public void keyTyped(KeyEvent e) {

                }

                @Override
                public void keyPressed(KeyEvent e) {
                    char label = e.getKeyChar();
                    System.out.println((int)label);
                    System.out.println(label);
                    if (label == 8)
                        Reset();
                    else if ("0123456789.".contains(label+""))
                        RunNumber(label+"");
                    else
                        RunOperator(label+"");
                }

                @Override
                public void keyReleased(KeyEvent e) {

                }
            });
        }
        // 设置成4X4的布 局 网格布局
        p1 = new JPanel(new GridLayout(4, 4));
        // 把按钮添加到网格布局中
        for (int i = 0; i < JButtonName.length; i++) {
            p1.add(Jb[i]);
        }
        // 设置第2个面板
        p2 = new JPanel(new BorderLayout());
        // 为ce按钮赋值
        ce = new JButton("CE");
        ce.setContentAreaFilled(false);
        // 为CE按钮设置监听器
        ce.addActionListener(this);

        ce.addKeyListener(new KeyListener() {
            @Override
            public void keyTyped(KeyEvent e) {

            }

            @Override
            public void keyPressed(KeyEvent e) {
                char label = e.getKeyChar();
                System.out.println((int)label);
                System.out.println(label);
                if (label == 8)
                    Reset();
                else if ("0123456789.".contains(label+""))
                    RunNumber(label+"");
```

```
                else
                    RunOperator(label+"");
            }

            @Override
            public void keyReleased(KeyEvent e) {

            }
        });


        // 输入框
        tf = new JTextField("0");
        tf.setEnabled(false);
        tf.setDisabledTextColor(Color.black);
        tf.setFont(font);



        p2.add(ce, "East");
        p2.add(tf, "Center");

        //添加文本框和ce按钮放置到上方
        this.add(p2, BorderLayout.NORTH);
        //添加按钮到下方
        this.add(p1, BorderLayout.CENTER);
        //设置可见
        this.setVisible(true);

        //设置标题
        this.setTitle("简易计算器");

        //使窗口居中
        int windowsWedth = 280;
        int windowsHeight = 300;
        int width = Toolkit.getDefaultToolkit().getScreenSize().width;
        int height = Toolkit.getDefaultToolkit().getScreenSize().height;
        this.setBounds((width - windowsWedth) / 2,
                (height - windowsHeight) / 2, windowsWedth, windowsHeight);

    }

    public void actionPerformed(ActionEvent e) {
        // 返回的是按的按钮名
        String label = e.getActionCommand();
        System.out.println(label);
        if (e.getSource() == ce)
            Reset();
        else if ("0123456789.".contains(label))
            RunNumber(label);
        else
            RunOperator(label);

    }
    // 是否是第一次按数字
```

```
    boolean isFirstDigit = true;

    public void RunNumber(String key) {
        if (isFirstDigit)
            tf.setText(key);// 如果是第一次按数字，覆盖文本框的内容
        else if ((key.equals(".")) && (!tf.getText().contains(".")))// 按了点并且文
本框中之前没有点
            tf.setText(tf.getText() + ".");//如果按了多个".",只显示一个
        else if (!key.equals("."))
            tf.setText(tf.getText() + key);// 叠加显示在文本框中
        isFirstDigit = false;
        hasDeal = false;

    }

    double number = 0.0;
    String operator = "=";
    boolean hasDeal;


    public void Reset() {// 重置
        tf.setText("0");
        isFirstDigit = true;
        operator = "=";
    }

    public void RunOperator(String key) {
//          进行计算
        hasDeal = false;
        if (operator.equals("+"))
            // 将文本框中的字符串强制转化为Double类型
            number += Double.parseDouble(tf.getText());
        else if (operator.equals("-"))
            number -= Double.parseDouble(tf.getText());
        else if (operator.equals("×") || operator.equals("*"))
            number *= Double.parseDouble(tf.getText());
        else if (operator.equals("÷")) {
            number /= Double.parseDouble(tf.getText());
        } else if (operator.equals("="))
            number = Double.parseDouble(tf.getText());
        tf.setText(String.valueOf(number));
        operator = key;
        isFirstDigit = true;
    }

    public static void main(String[] args) {
        Demo112 calculator = new Demo112();
        calculator.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    }
}
```

demo12 1

```
package com.star.demo121;

import java.io.*;
import java.util.Scanner;
import java.util.TreeSet;

/**
 * 描述:封装排行榜类.
 */
class RankingList implements Serializable {
    private IRecord[] rkList;
    private int size;

    public RankingList() {
        this.rkList = new IRecord[10];
        size = 0;
    }

    /**
     * 添加新纪录,并且保持数量不超过10
     *
     * @param name
     * @param score
     */
    public void add(String name, int score) {
        int targetIdx = 0;
        while(targetIdx <= 9 && rkList[targetIdx] != null &&
rkList[targetIdx].score >score){
            targetIdx++;
        }
        if(targetIdx > 9) return; // 分太低了,榜上的都比你高,还是别上榜了
        for (int i = 9; i > targetIdx ; i--) {
            rkList[i] = rkList[i-1];
        }
        if(size<10)size++;
        IRecord newRecord = new IRecord(name, score);
        rkList[targetIdx] = newRecord;
    }

    public void show() {
        int rank = 1;
        System.out.println("名次| 昵称| 积分");
        for (int i=0;i<size;i++) {
            System.out.printf("%3d%5s%5d\n", rank, rkList[i].name,
rkList[i].score);
            rank++;
        }
    }
}

/**
 * 封装记录类.
 */
```

```
class IRecord implements Comparable<IRecord>, Serializable {
    String name;
    int score;

    public IRecord(String name, int score) {
        this.name = name;
        this.score = score;
    }

    /**
     * 配合红黑树排序,并且解决冲突
     *
     * @param o the object to be compared.
     * @return
     */
    @Override
    public int compareTo(IRecord o) {
        return score == o.score ? 1 : o.score - score;
    }
}
public class Demo121Test{
    public static void main(String[] args) throws Exception {
        test01();
    }

    static void test01(){
        try {
            File records = new File("D:/records.dat");
            if (!records.exists()) {
                System.out.println("生成排行榜文件...");
                Thread.sleep(666);
                records.createNewFile();
            }
            RankingList rankingList = null;
            try {
                FileInputStream fIS = new FileInputStream(records);
                ObjectInputStream oIS = new ObjectInputStream(fIS);
                rankingList = (RankingList) oIS.readObject();
                System.out.println("排行榜加载中...");
                Thread.sleep(666);
                fIS.close();
                oIS.close();
            } catch (Exception e) {
                System.out.println("正在初始化排行榜...");
                Thread.sleep(666);
            }
            if (rankingList == null) {
                rankingList = new RankingList();
            }
            Scanner scanner = new Scanner(System.in);
            rankingList.show();
            System.out.println("是否向排行榜输入?(y/n)");
            while ("y".equals(scanner.next())) {
                System.out.println("向排行榜输入玩家昵称和积分:");
                String name = scanner.next();
```

```
                int score = scanner.nextInt();
                rankingList.add(name, score);
                rankingList.show();
                ObjectOutputStream oOS = new ObjectOutputStream(new
FileOutputStream(records));
                oOS.writeObject(rankingList);
                oOS.flush();
                System.out.println("是否向排行榜输入?(y/n)");
            }


        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```