实验三 方法及数组

学生姓名：黄晨箬　　学　　号：6109119066　　专业班级：计算机 193 班

实验类型：■ 验证 □ 综合 □ 设计 □ 创新　实验日期：2021.4.27　实验成绩：

一、实验名称

方法及数组

二、实验目的

1、学习和掌握方法的定义和使用

2、培养学生模块化编程的思想

3、培养学生编程过程中采用抽象和逐步求精的理念

4、开发和调用具有数组参数和数组返回值的方法

5、加深对一维数组用法的理解和使用，掌握基本的数据查找和排序算法

6、加深对多维数组用法的理解和使用。

三、实验内容

采用记事本或集成开发环境编写Java语言源程序，完成下列习题任务的运行及调试。

习题1：（密码检测）一些网站在用户注册设定密码时需遵循一些规则。编写一个方法，检测字符串是否是一个有效密码。假设密码规则如下：

（1） 密码必须至少8位字符。

（2） 密码仅能包含字母、数字和下划线。

（3） 密码必须包含至少两个数字。

编写一个程序，提示用户输入一个密码，如果符合规则，则显示Valid Password，否则显示Invalid Password。

习题 2：（搜索最大块） 给定一个元素为 0 或 1 的方阵，编写一个程序，找到一个元素都为 1 的最大的子方阵。程序提示用户输入矩阵的行数。然后显示最大的子方阵的第一个元素位置，以及该子方阵的行数。运行示例如下，其中绿色字体为用户输入。

```
Enter the number of rows in the square matrix: 5
Enter the matrix row by row:
1 0 1 0 1
1 1 1 0 1
1 0 1 1 1
1 0 1 1 1
1 0 1 1 1
The maximum square submatrix is at (2, 2) with size 3
```

习题3-5：完成课本下列章节的编程题内容。

习题3：　第七章 243页　7.19 （是否排好序了）

习题4：　第七章 243页　7.23 （游戏：储物柜难题）
习题5：　第七章 246页　7.35 （猜字词游戏 ）

四、实验仪器设备及耗材

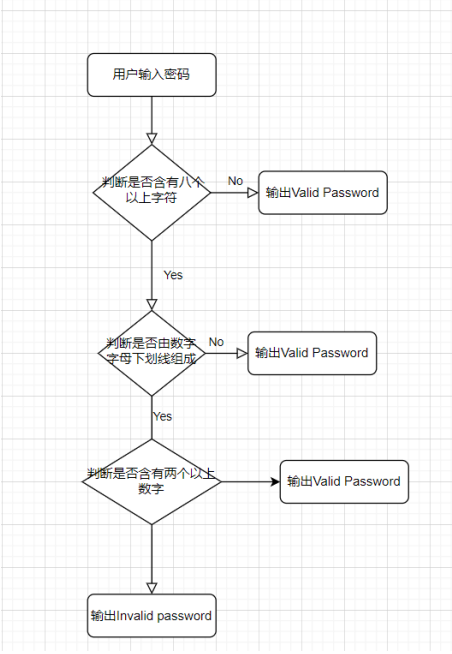
1、PC微机；
2、DOS操作系统或 Windows 操作系统；
3、Eclipse程序集成环境。

五、实验步骤

1、根据题目要求，画出程序流程图；
2、给出习题X程序的java数据结构；
3、编写习题X程序源代码；
4、调试程序：
5、给出一些测试数据，检查输出结果。

六、实验数据及处理结果

习题一：
[程序流程图]



[数据结构设计]

根据题意，用户所输入的密码需要满足三个条件；
首先将用户输入的密码每一位都存入数组里，判断数组的长度从而判断位数是否满足要求
然后再通过正则表达式对数组每一位进行判定，看是否有违法字符
如果以上都合法，则没有问题，只要有一项违法，就输出错误信息

[程序源代码]

```java
//判断内容是否是数字字母下划线
public static boolean rule2(String value) {
    String regex = "^\\w+$";
    return value.matches(regex);
}

//判断内容是否包含数字
public static int rule3(String value, int n) {
    String regex =  "^[0-9]*$";
    if (value.matches(regex)){
        return n+1;
    }
    else return n;
}

public static void main(String[] args) {
    System.out.print("输入你的密码: ");
    Scanner scanner = new Scanner(System.in);
    String str = scanner.nextLine();
    scanner.close();
    char[] Arr = str.toCharArray();

    //使用n来对数字的个数进行计数，使用ruler对密码是否合法进行判定
    int n = 0;
    boolean ruler = true;

    for (char c : Arr) {

        //对是否含有八个以上字符进行判断
        if (Arr.length < 8) {
            ruler = false;
            break;
        }
```

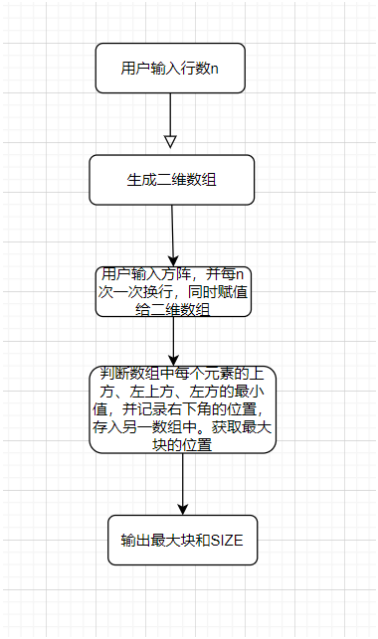
```
            //对是否包含数字字母下划线进行判断
            boolean rule = rule2(String.valueOf(c));
            if (!rule) {
                ruler = false;
                break;
            }

            //判断是否含有两个以上的数字
            n = rule3(String.valueOf(c), n);
        }
        if (n < 2) {
            ruler = false;
        }

        //判定密码是否合法，如果不合法则输出错误信息
        if (!ruler) {
            System.out.println("Invalid Password");
        }
        else {
            System.out.println("Valid Password");
        }

    }
}
```

[程序运行结果]

```
输入你的密码: 1394iu393
Valid Password
```

习题二：

[程序流程图]



[数据结构设计]

```
/*
    首先输入行数，根据用户输入的行数生成一个二维数组。
    接下来，如果用户输入n，则输入时，每n个一次换行，对数组对应位置的值赋值
    记录方阵的数值，以便于输出最大块
 */
```

[程序源代码]

```
    //用于查询最大块的函数
    public static int[] check(int[][]a, int n) {
        int max = 1, x = 0, y = 0;
        for(int i=1;i<n;i++)
        {
            for(int j=1;j<n;j++)
            {
                if(a[i][j]==1)
                {
                    //求出上方、左方以及上左的最小值
                    int min = Math.min(a[i - 1][j], a[i][j - 1]);
                    min = Math.min(a[i - 1][j - 1], min);
                    a[i][j] = min + 1;
                    //记录方阵右下角的位置
                    if(max < a[i][j])
                    {
                        max = a[i][j];
                        x=i;y=j;
                    }
                }
            }
        }
        //获取所得结果之后放入数组，以便展示
        int[] Arr = {max,x-max+1,y-max+1};
        return Arr;
    }
```

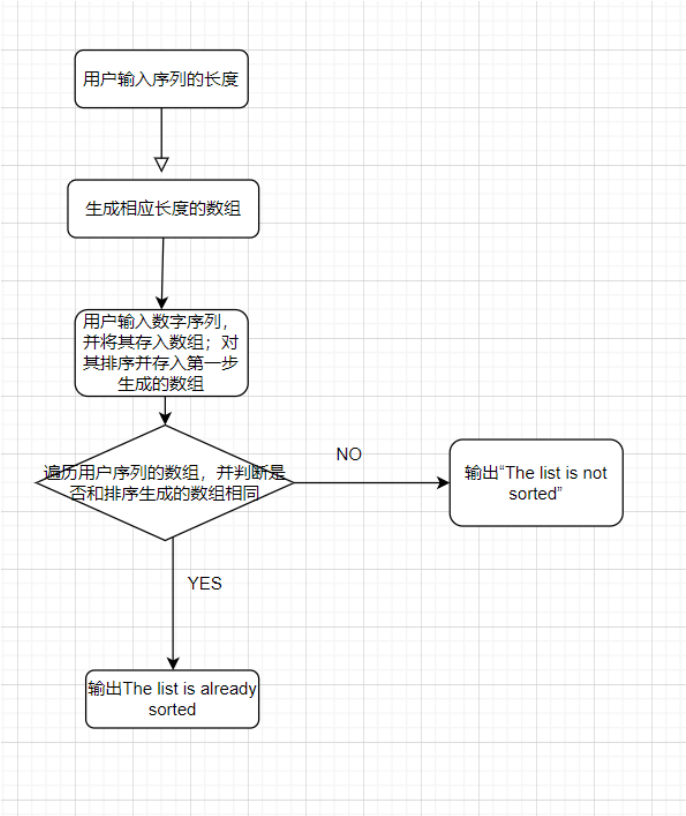
```java
public static void main(String[] args) {
    System.out.print("Enter the number of rows in the square matrix: ");
    Scanner input = new Scanner(System.in);
    int n = input.nextInt();

    int [][] arr = new int[n][n];

    //用户输入方阵
    for (int i = 0; i < n; i++){

        Scanner scanner = new Scanner(System.in);
        String str = scanner.nextLine();
        String s[]  = str.split( regex: " ");

        for (int j = 0; j < n; j++){
            arr[i][j] = Integer.parseInt(s[j]);
        }
    }

    int[] key = check(arr, n);
    System.out.print("The maximum square submatrix is at ("+key[1]+","+key[2]+")  ");
    System.out.println("with size "+key[0]);

}
}
```

[程序运行结果]

```
Enter the number of rows in the square matrix: 5
1 0 1 0 1
1 1 1 0 1
1 0 1 1 1
1 0 1 1 1
1 0 1 1 1
The maximum square submatrix is at (2,2)  with size 3
```

习题三：

[程序流程图]



[数据结构设计]

```
/*
    首先用户需要输入这个序列的长度，通过这个长度来创造一个相对应长度的数组
    然后用户输入一个数字序列，我们需要对其判断是否已经排好序了
    那么就要将用户输入的序列放进一个数组表示
    然后从这个数组把各个元素拿出来进行排序，并依次将其放到另外一个数组里
    然后判断这两个数组是否是相同的
*/
```

[程序源代码]

```java
//用户输入序列的长度
System.out.print("Enter the size of the list: ");
Scanner input = new Scanner(System.in);
int size = input.nextInt();

//用户输入其序列，将序列中的各个数依次存进数组里
System.out.print("Enter the contents of the list: ");
Scanner scanner = new Scanner(System.in);
String str = scanner.nextLine();
String[] user = str.split( regex: " ");

//对用户输入的序列进行遍历
System.out.print("The list has " + size + " integers ");
for (int i = 0; i < size; i++){
    System.out.print(user[i]);
}

System.out.println();

//将用户输入的序列放入另一个数组存放
String[] answer = new String[size];
if (size >= 0) System.arraycopy(user,  srcPos: 0, answer,  destPos: 0, size);

//对用户给出的序列进行升序排列
Arrays.sort(user);
```

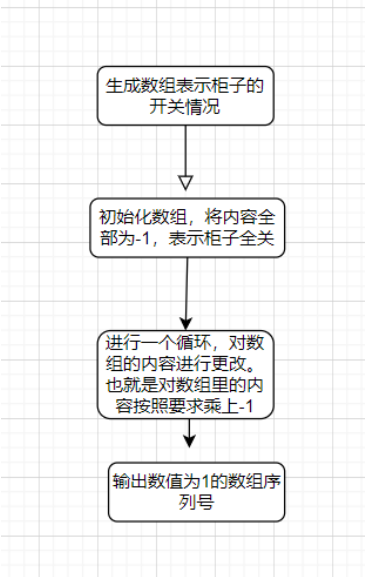
```java
if (Arrays.equals(user, answer)) {
    System.out.println("The list is already sorted");
}
else {
    System.out.println("The list is not sorted");
}
```

[程序运行结果]

```
Enter the size of the list: 8
Enter the contents of the list: 10 1 5 16 61 9 11 1
The list has 8 integers 10 1 5 16 61 9 11 1
The list is not sorted
```

习题四：

[程序流程图]



[数据结构设计]

```
用-1表示关，用1表示开。于是创造一个数组来存放柜子的开关情况。最后检验时只要遍历，然后把值为1的元素的索引值提出即可
```

[程序源代码]

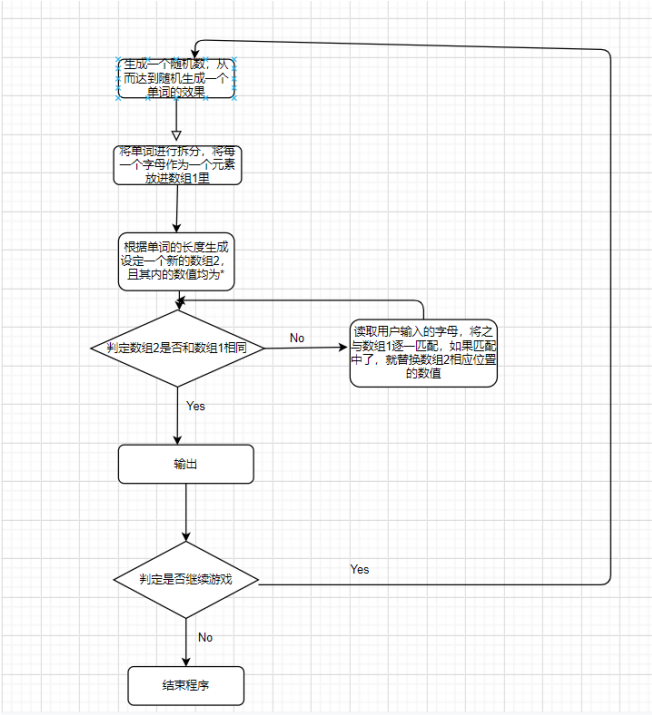
```java
public class cupboard {
    public static void main(String[] args) {

        //初始化柜子的情况（均为关）
        int[] array = new int[100];
        for (int i = 0; i < 100; i++) {
            array[i] = -1;
        }

        for (int stu = 0; stu < 100; stu++) {
            for (int i = stu; i < 100; i = i + stu + 1){
                array[i] = -1 * array[i];
            }
        }

        for (int i = 0; i < 100; i++) {
            if (array[i] == 1) {
                System.out.print(i+1 + " ");
            }
        }
```

[程序运行结果]

```
1 4 9 16 25 36 49 64 81 100
进程已结束，退出代码为 0
```

习题五：

[程序流程图]



[数据结构设计]

```
首先需要生成一个随机数，从而达到随机生成一个单词的效果
取出这个单词，对他进行拆分，每个字母作为一个元素放进另一个数组2里
设定一个新的数组3，根据单词的长度生成，每个元素都是*
读取用户输入的字母，并将之与数组2逐位匹配，如果没有相匹配的，输出is not in the word，并计数元素+1
如果有匹配的，那么就将这一位的值赋给数组3的对应元素，输出数组3

以上的行为需要先判定数组3是不是等于数组2，如果不是则一直运行，如果是，则输出结果。
结束后，判定y和n ，决定游戏是否重复。所以可以将上面的环节写成一个函数，这里就能直接调用
```

[程序源代码]

```java
    //用于比较正确单词和用户猜测的单词是否一致
    public static boolean match(char[]a, String[]b) {
        String[] A = new String[a.length];
        for (int i = 0; i < a.length; i++){
            A[i] = String.valueOf(a[i]);
        }

        return Arrays.equals(A,b);
    }

    public static void loop(char[]a, String[]e, int time){
        //判定用户输入的字母是否与原单词匹配，如果是，则在已猜测的数组上显示

        //用于判断是否猜错的中间数
        int j = 0;

        //判定end与answer是否相等，如果相等，则进行猜测环节
        if (!match(a,e)){
//            System.out.println("尚未猜出答案 ");

            //读取用户输入
            System.out.print("请输入你认为单词含有的字母: ");
            Scanner scanner = new Scanner(System.in);
            String str = scanner.nextLine();

            //对输入正确的位置进行赋值
            for (int i = 0; i < a.length; i++) {
                String answer = String.valueOf(a[i]);
                if (str.equals(answer)) {
```

```java
                String answer = String.valueOf(a[i]);
                if (str.equals(answer)) {
                    e[i] = answer;
                    j++;
                }
            }

            if (j == 0) {
                time++;
            }

            System.out.print("现在单词的状态是: ");
            for (int i = 0; i < e.length; i++) {
                System.out.print(e[i]);
            }
            System.out.println();
            loop(a,e,time);
        }
        else {
            System.out.print("The word is ");
            for (int i = 0; i < e.length; i++) {
                System.out.print(e[i]);
            }
            System.out.println(" 输错的次数为: " + time);
        }

    }
```

```java
    public static void game() {
        //生成单词库
        String[] words = {"white", "that", "program"};

        //用于记录猜错的次数
        int time = 0;

        //生成包含正确单词的数组2
        Random r = new Random();
        int key = r.nextInt(words.length);
        char[] answer = words[key].toCharArray();


        //生成显示的数组3
        String[] end = new String[answer.length];
        for (int i = 0; i < answer.length; i++) {
            end[i] = "*";
        }

        //执行猜测环节
        loop(answer, end, time);

        //判断是否再来一次
        System.out.print("想要再来一次吗? (y/n) :");
        Scanner scanner = new Scanner(System.in);
        String choose = scanner.nextLine();

        if (choose.equals("y")) {
            game();
```

```java
        if (choose.equals("y")) {
            game();
        }
        else {
            System.out.println("感谢你的使用");
        }
    }

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

[程序运行结果]

```
请输入你认为单词含有的字母: d
现在单词的状态是: *******
请输入你认为单词含有的字母: p
现在单词的状态是: p******
请输入你认为单词含有的字母: r
现在单词的状态是: pr**r**
请输入你认为单词含有的字母: p
现在单词的状态是: pr**r**
请输入你认为单词含有的字母: o
现在单词的状态是: pro*r**
请输入你认为单词含有的字母: m
现在单词的状态是: pro*r*m
请输入你认为单词含有的字母: g
现在单词的状态是: progr*m
请输入你认为单词含有的字母: a
现在单词的状态是: program
The word is program 输错的次数为: 1
想要再来一次吗? (y/n) :n
感谢你的使用
```

七、选做题（二选一）

1、在对话框中任意输入一段文本（不超过 100 个单词），用户可对输入的文字中的指定单词用特定字符串替换，并将替换后的文本显示出来。

2、如果一个整数其顺序和逆序数值相同，如121，则称为回文数。找出99999以内的所有正整数，使得其满足自身，自身的平方，自身的三次方均是回文数。

在该程序中要求使用以下方法：

```
//return the reversal of an integer, i.e. reverse(456) returns 654
public static long reverse(long number)
 //return true if number is Palindrome
public static boolean isPalindrome(long  number)
```

八、实验总结及体会

通过这次实验，我加深了对数组的理解，能够利用数组解决一些看似毫无头绪的问题。对模块化编程也有了新的理解，不会再把所有的内容全部写在主函数里，而是对每个功能模块化。