

## 实验十一 字符串类、System 类、Runtime 类的应用

### 一、实验目的

- (1) 掌握 String 类中常用方法的使用。
- (2) 掌握 System 类中常用方法的使用。
- (4) 掌握 Runtime 类中常用方法的使用。

### 二、知识回顾

#### (1) 字符串

字符串就是指一连串的字符，它是由许多单个字符连接而成的，如多个英文字母所组成的一个英文单词。

String 类的初始化方式：

方式 1：使用字符串常量直接初始化一个 String 对象，例如：

```
String str1 = "abc";
```

方式 2：使用 String 类的构造方法初始化字符串对象。例如：

```
String str1 = new String("abc");
```

#### (2) String 类的常见操作方法

```
int indexOf(int ch)
```

```
int lastIndexOf(int ch)
```

```
int indexOf(String str)
```

```
int lastIndexOf(String str)
```

```
char charAt(int index)
```

```
Boolean endsWith(String suffix)
```

```
int length()
```

```
boolean equals(Object anObject)
```

```
boolean isEmpty()
```

```
boolean startsWith(String prefix)
```

```
boolean contains(CharSequence cs)
```

```
String toLowerCase()
```

```
String toUpperCase()
```

```
static String valueOf(int i)
```

```
char[] toCharArray()
```

```
String replace(CharSequence oldstr, CharSequence newstr)
```

```
String[] split(String regex)
```

```
String substring(int beginIndex)
```

`String substring(int beginIndex, int endIndex)`

`String trim()`

### (3) 字符串的转换操作

程序开发中，经常需要对字符串进行转换操作。例如，将字符串转换成数组的形式，将字符串中的字符进行大小写转换等。

### (4) 字符串的替换和去除空格操作

程序开发中，用户输入数据时经常会有一些错误和空格，这时可以使用 `String` 类的 `replace()` 和 `trim()` 方法，进行字符串的替换和去除空格操作。

### (5) 字符串的判断操作

操作字符串时，经常需要对字符串进行一些判断，如判断字符串是否以指定的字符串开始、结束，是否包含指定的字符串，字符串是否为空等。

### (6) “==” 和 `equals()` 两种方式对字符串进行比较的区别

`equals()` 方法用于比较两个字符串中的字符是否相等，`==` 方法用于比较两个字符串对象的地址是否相同。

### (7) 字符串的截取和分割

在 `String` 类中，`substring()` 方法用于截取字符串的一部分，`split()` 方法用于将字符串按照某个字符进行分割。

### (8) `StringBuffer` 类

Java 提供了一个 `StringBuffer` 类(也称字符串缓冲区)。`StringBuffer` 类和 `String` 类最大的区别在于它的内容和长度都是可以改变的。`StringBuffer` 类似一个字符容器，当在其中添加或删除字符时，并不会产生新的 `StringBuffer` 对象。

### (9) `String`、`StringBuffer`、`StringBuilder` 的区别

`String` 类表示的字符串是常量，一旦创建后，内容和长度都是无法改变的。而 `StringBuilder` 和 `StringBuffer` 表示字符容器，其内容和长度可以随时修改。在操作字符串时，如果该字符串仅用于表示数据类型，则使用 `String` 类即可，但是如果需要对字符串中的字符进行增删操作，则使用 `StringBuffer` 与 `StringBuilder` 类。如果有大量字符串拼接操作，不要求线程安全的情况下，采用 `StringBuilder` 更高效。相反如果需要线程安全则需要使用 `StringBuffer`。`String` 类对象可以用操作符“+”进行连接，而 `StringBuffer` 类对象之间不能。

(10) `System` 类定义了一些与系统相关的属性和方法，它所提供的属性和方法都是静态的，因此，想要引用这些属性和方法，直接使用 `System` 类调用即可，如：

`arraycopy()` 方法

`currentTimeMillis()` 方法

`getProperties()` 和 `getProperty()` 方法

`gc()` 方法

(11) `Runtime` 类用于表示虚拟机运行时的状态，它用于封装 JVM 虚拟机进程。每次使用 `java` 命令启动虚拟机都对应一个 `Runtime` 实例，并且只有一个实例，因此在

Runtime 类定义的时候，它的构造方法已经被私有化了(单例设计模式的应用)，对象不可以直接实例化。其常用的方法如下：

```
getRuntime()  
exec(String command)  
freeMemory()  
maxMemory()  
availableProcessors()  
totalMemory()
```

### 三、实验内容

#### 第 1 题：

在使用一些 APP 时，通常都需要填写用户名和密码。用户名和密码输入都正确才会登录成功，否则会提示用户名或密码错误。

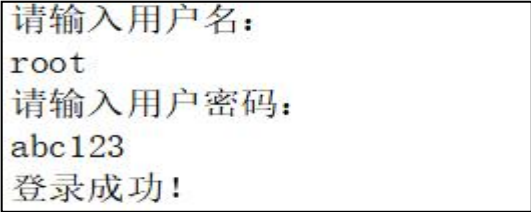
本例要求编写一个程序，模拟用户登录。程序要求如下：

- (1) 用户名和密码正确，提示登录成功。
- (2) 用户名或密码不正确，提示“用户名或密码错误”。

(3) 总共有 3 次登录机会，在 3 次内(包含三次)输入正确的用户名和密码后给出登录成功的相应提示。超过 3 次用户名或密码输入有误，则提示登录失败，无法再继续登录。

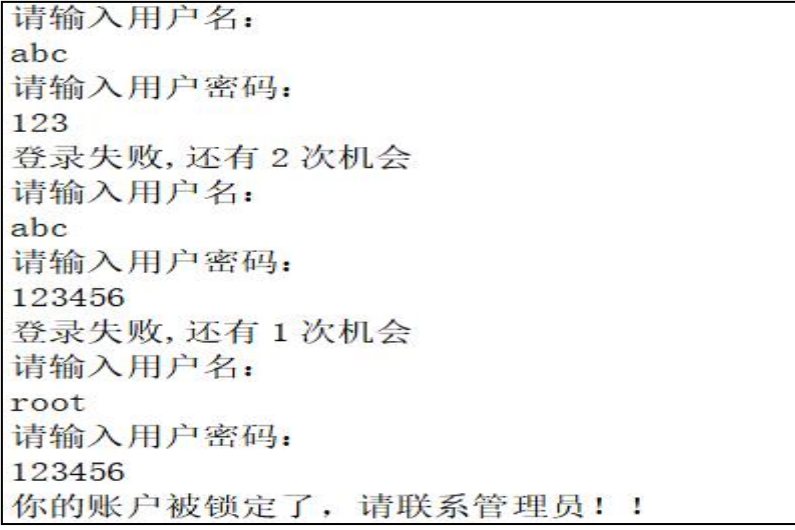
在登录时，需要比较用户输入的用户名密码与已知的用户名密码是否相同，本案例要求使用 Scanner 类以及 String 类的相关方法实现比较操作。

参考运行界面如图 11-1、11-2 所示。



```
请输入用户名:  
root  
请输入用户密码:  
abc123  
登录成功!
```

图 11-1 运行界面 (1)



```
请输入用户名:  
abc  
请输入用户密码:  
123  
登录失败, 还有 2 次机会  
请输入用户名:  
abc  
请输入用户密码:  
123456  
登录失败, 还有 1 次机会  
请输入用户名:  
root  
请输入用户密码:  
123456  
你的账户被锁定了, 请联系管理员!!
```

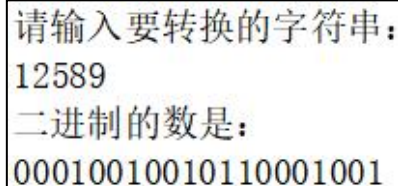
图 11-2 运行界面 (2)

## 第 2 题:

本例要求编写一个程序，从键盘录入一个字符串，将字符串转换为二进制数。在转换时，将字符串中的**每个字符单独转换为一个二进制数**，将所有二进制数连接起来进行输出。

案例在实现时，要求使用 Math 类、String 类以及 Scanner 等常见 Java API 的常用方法实现。

参考运行界面如图 11-3 所示。



```
请输入要转换的字符串:
12589
二进制的数是:
00010010010110001001
```

图 11-3 运行界面

## 四、实验步骤

### 第 1 题:

Step 1: 建立项目;

Step 2: 定义 UserLogin 类，建立 main 方法;

Step 3: 设定初始用户名和密码；在循环中输入用户名和密码，将输入的用户名和密码与设定的用户名密码匹配，并根据情况给出提示。参考代码如下：

```
public class UserLogin {
    public static void main(String[] args) {
        //已知用户名密码，定义两个字符串表示
        String username = "root";
        String password = "abc123";
        for (int i = 0; i < 3; i++) {
            // 键盘录入要登录的用户名密码。用 Scanner 实现
            Scanner sc = new Scanner(System.in);
            System.out.println("请输入用户名: ");
            String uname = sc.nextLine();
            System.out.println("请输入用户密码: ");
            String pwd = sc.nextLine();
            //将输入的用户名密码和已知的用户名密码进行比较，给出相应的提示，
            // 字符串内容比较用 equals 方法实现。
            if (uname.equals(username) && pwd.equals(password)) {
                System.out.println("登录成功!");
                break;
            } else {
                if (2 - i == 0) {
                    System.out.println("你的账户被锁定了，请联系管理员!!");
                } else {
                    System.out.println("登录失败, 还有" + (2 - i) + "次机会");
                }
            }
        }
    }
}
```

## 第 2 题:

Step 1: 建立项目;

Step 2: 定义 DecToBinary 类, 建立 main 方法;

Step 3: 依次取出用户输入的串, 将每位数字字符变成对应的整数, 然后求出它对应的二进制数字并存入二维数组中。最后将二维数组输出, 即可得到结果。参考代码如下:

```
import java.util.Scanner;
public class DecToBinary {
    public static void main(String[] args) {
        //键盘录入要转化的字符串。用 Scanner 实现。
        Scanner sc = new Scanner(System.in);
        System.out.println("请输入要转换的字符串: ");
        String ss = sc.nextLine();
        //因为一个十进制数字 0-9, 可用 4 位二进制数字表示
        //因此定义一个二维数组来存储转换后的二进制数字。其中 4 代表每一行长度。
        //根据输入的字符串的长度 ss.length() 确定二维数组的行数, 而每行的长度固定为 4。
        int [][] arr = new int[ss.length()][4];
        //利用 for 循环遍历字符串, 遍历后用 String 的 charAt() 方法获取每个字符, 将字符
        //转化成 int。char 与 int 进行运算, char 的数值要减去 48, 因为 ASCII 码中
        // '0' 的值是 48, '1' 就是 49。
        for (int i = 0; i < ss.length(); i++) {
            int charss = (int) ss.charAt(i) - 48;
            for (int j = 0; j < 4; j++) {
                /*将 int 型数变成二进制数可以采用将该数依次除以 2 的 N(N=3, 2, 1, 0) 次方的结果%2
                * 例如: 整数 9 对应的二进制数字为: 9/8%2=1, 9/4%2=0, 9/2%2=0, 9/1%2=1,
                * 因此, 结果为 1001.
                * 2 的 N 次方, 可以用 Math.pow(2, N) 方法来求。
                * 将每位二进制数字赋值给数组。
                */
                arr[i][j] = (int)((charss/Math.pow(2, 3-j))%2);
            }
        }
        // 最后用双重 for 循环遍历二维数组。将结果输出到控制台。
        System.out.println("二进制的数是: ");
        // 最后用双重 for 循环遍历二维数组。将结果输出到控制台。
        for (int i = 0; i < ss.length(); i++) {
            for (int j = 0; j < 4; j++) {
                System.out.print(arr[i][j]);
            }
        }
    }
}
```

## 五、实验总结及思考(实验中遇到的问题及相应的解决方法)

总结:

思考: