# 1.String-final Class

**1.1重要方法**

#### compareTo(String str);

比较两个字符串的大小。根据返回值的正负来确定大小

**底层过程：**

按照升序规则比较两个字符串

1. 将两个字符串转化为两个字符数组。

2. 对象字符串对应的字符数组的对应位置的字符 -/减 参数字符串对应的数组对应位置的字符。

3. 如果差不为0，那么直接返回。 如果差为0，则下一位对应的字符继续相减。

4. 如果对应位置的字符全部减完，依然没有返回的话，那么就会返回对象字符串-参数字符串的长度之差。如果返回的是一个正数，说明对象字符串要大一些，如果返回的是一个负数，说明参数字符串要大一些。

System.out.println(str.compareTo("dhnalgola"));

System.out.println(str.compareToIgnoreCase("DHNALGOLA"));

#### compareToIgnoreCase(String str);

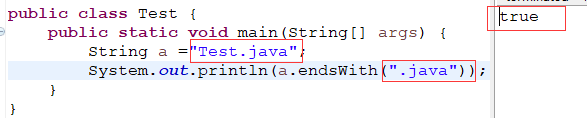
不区分大小写比较两个字符串的大小。

#### contains(String str)

判断是否包含指定的子串

#### endsWith(String s);

判断字符串是否以指定字符结尾。经常用于筛选。



#### equals(Object o)

String中已经对equals做了重写。判断两个字符串是否相等，判断的是实际值。

String str = new String("abcdefghijkmn");

// 判断是否包含指定的子串

// System.out.println(str.contains("dee"));

// 判断是否是指定的结尾---筛选

// .docx

// System.out.println(str.endsWith("mn"));

// 判断是否为指定的开头

// System.out.println(str.startsWith("a"));

// 对equals方法做过了重写，判断的是两个字符串的实际值是否一样

// 将两个字符串转化为字符数组，然后按位比较

System.out.println(str.equals(new String("ABCDEFGHIJKMN")));

// 适用于验证码，不区分大小写

System.out.println(str.equalsIgnoreCase(new String("ABCDEFGHIJKMN")));

#### getBytes();

将字符串转化为字节数组，如果不指定编码，会按照默认的系统平台码转化。

#### getBytes(Charset ch);

将字符串按照指定编码转化为字节数组。

编码：按照某种规则将字符映射成字节---记录这种规则的形式---编码表

ASCII--- 0-127---不完全的码表

ISO-8859-1---西欧码表--- 一个字符一个字节

gb2312--- 一个字符2个字节 --- 包含了常见的基本简体汉字以及部分的繁体汉字 --- gbk

Unicode编码体系--- utf-8---一个字符3个字节 --- 常见语言的常见字符

后续的所有码表默认兼容西欧码表---只要是英文，永远是一个字符对应一个字节

String str = "中文";

// 表示将字符串转化为字节数组

// 如果没有指定编码，会默认采用系统平台码

byte[] bs = str.getBytes("utf-8");

System.out.println(bs.length);

System.out.println(new String(bs, 0, 3, "UTF-8"));

// for (byte b : bs) {

// System.out.println(b);

// }

// byte[] bs = {12,55,120,127,15};

// 将字节数组转化为字符串

// 表示从字节数组的下标为1的位置开始转化，转化2个字节

// System.out.println(new String(bs,1,2));

###### 练习:截取字符串指定字节

输入一个字符串和一个数字，数字表示字节个数，然后按照指定的字节个数来截取字符串

中文english国 5 -> 中文e 6 -> 中文en 2 -> 中 3 -> 中 12-> 中文english？

public static void printStr2(String str,int number){

//判断字符串是否为空

if(str==null){

return;

}

//判断字节个数是否是非负数

if(number<0){

return;

}

//先将所有的字符看成中文

int index = number/2;

//按照这个新的长度从原字符串身上截取

String newstr = str.substring(0,index);

while(newstr.getBytes().length!=number){

//如果少来就多添加一个字符

if(newstr.getBytes().length<number){

index++;

newstr = str.substring(0,index);

}else{

index--;

newstr = str.substring(0,index);

break;

}

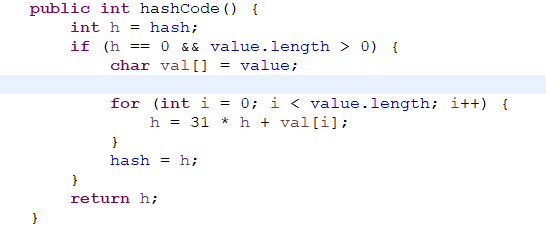
}

System.out.println(str.substring(0,index));

}

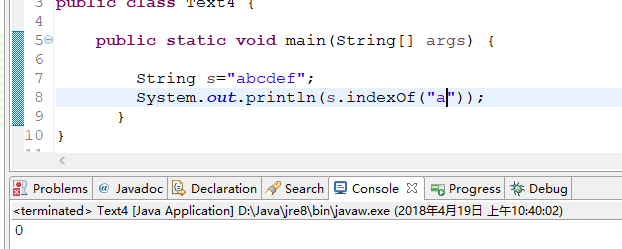
###### hashCode();

String中的hashCode方法做过了重写，同一个字符串在任何情况下的哈希码都是一样的。



#### indexOf(char c/String s);

获取指定字符c在字符串中的第一次出现的位置，如果找不到该字符会返回一个-1。



##### 练习

###### 输出字符在字符串中的出现的位置

输入一个字符串和一个字符，输出这个字符出现的所有位置。

public static void all(String str, String sub) {

// 判断str是否为null

if (str == null) {

return;

}

// 记录位置

int index = 0;

while (index < str.length()) {

// 获取指定字符第一次出现的位置

index = str.indexOf(sub, index);

if (index != -1) {

System.out.println(index);

index++;

} else {

return;

}

}

#### isEmpty();

判断字符串长度是否为0.

#### lastIndexOf(char ch/String s);

表示最后一次出现的位置。

#### lastIndexOf(char ch,int indexfrom);

从指定下标开始向前寻找最后一次出现的位置。

#### replace(char oldch,char newch);

返回一个新字符串，替换。

Java中的范围包前不包后，包左不包右，包小不包大。

#### startsWith(char ch);

判断是否指定参数作为开头。作为筛选使用。

#### subString(int index);

从指定下标开始截取。

#### subString(int beginIndex,int endIndex);

从指定位置开始截取，截取到指定位置的前一位。

#### toCharArray();

将字符串转化为字符数组。

#### toLowerCase() ;

转化为小写。

**底层**

1. 现将字符串化为字符数组。
2. 便利数组，判断每一位是否是一个大写字母。
3. 如果是一个大写字母，这以为加上32然后在强转回char类型，再赋值给原位置。
4. 将字符数组转化为字符串。

#### toUpperCase();

转为大写。

#### toString();

返回本身。

#### trim();

去掉字符串前后的空白符。

#### valueOf(int i/char c/boolean b/long l/Object o);

返回参数的对应字符串形式。

// valueOf是一个静态方法---表示将参数转化为一个字符串

// 如果传入的是一个对象，则会调用这个对象的toString方法

// System.out.println(String.valueOf(new Object()));

// System.out.println(String.valueOf(new int[] { 2, 5, 1, 7, 0 }));

// 如果传入的是字符数组，则会转化为字符串显示

// System.out.println(String.valueOf(new char[] { 'a', 'b', 'c' }));

System.out.println(new char[] { 'a', 'd', 'g' });