String 类型不能对其本省修改，修改之后的就得另赋值给其他字符串

1. String str="Hello Java";
2. String str; str="Hello Java";
3. String()
4. String (String original) String str1=new String("Hello Java");

String str2=new String(str1);

1. String (char[ ]value)
2. char a[]={'H','e','l','l','0'}; String sChar=new String(a); a[1]='s'
3. String(char[] value ,int offset ,int count) char a[]={'H','e','l','l','o'};String sChar=new String(a,1,4); a[1]='s';
4. Contact ; 字符串 1.concat(字符串 2); 返回值是string
5. Length 字符串名.length(); 返回int型
6. toLowerCase() 字符串名.toLowerCase() //将字符串中的字母全部转换为小写，非字母不受影响 返回的是String类型
7. toUpperCase() 字符串名.toUpperCase() //将字符串中的字母全部转换为大写，非字母不受影响 返回的是String类型
8. trim() 字符串名.trim() 去除掉前后的空格，而中间的去不掉，返回的是String类型
9. 提取子字符串 ； substring(int beginIndex) 形式 此方式用于提取从索引位置开始至结尾处的字符串部分 ； substring(int beginIndex，int endIndex) 形式 此方法中的 beginIndex 表示截取的起始索引，截取的字符串中包括起始索引对应的字符；endIndex 表示结束索引，截取的字符串中不包括结束索引对应的字符 ； 返回的是String类型
10. 分割字符串 ；返回字符串数组的类型 ； str.split(String sign)
11. str.split(String sign,int limit) ；str 为需要分割的目标字符串；sign 为指定的分割符，可以是任意字符串；limit 表示分割后生成的字符串的限制个数，如果不指定，则表示不限制，直到将整个目标字符串完全分割为止。String Colors="Red,Black,White,Yellow,Blue";

String[] arr1=Colors.split(","); //不限制元素个数

String[] arr2=Colors.split(",",3); //限制元素个数为3

1. 替换字符串 ；replace() 方法用于将目标字符串中的指定字符（串）替换成新的字符（串）

字符串.replace(String oldChar, String newChar)

其中，oldChar 表示被替换的字符串；newChar 表示用于替换的字符串。replace() 方法会将字符串中所有 oldChar 替换成 newChar

字符串.replaceFirst(String regex, String replacement)

replaceFirst() 方法用于将目标字符串中匹配某正则表达式的第一个子字符串替换成新的字符串，

字符串.replaceAll(String regex, String replacement)

replaceAll() 方法用于将目标字符串中匹配某正则表达式的所有子字符串替换成新的字符串

1. 将字符串转换成字节数组 ；

byte[] bt=value.getBytes(); //将字符串转换为字节数组；

for(int i=0;i<bt.length;i++)

{

bt[i]=(byte)(bt[i]^(int)secret); //通过异或运算进行加密

}

String newresult=new String(bt,0,bt.length); //将加密后的字符串保存到 newresult 变量中

1. 字符串的查找

indexOf() 方法 indexOf() 方法用于返回字符（串）在指 定字符串中首次出现的索引位置，如果能找到，则返回索引值，否则返回 -1。

str.indexOf(value)

str.indexOf(value,int fromIndex)

注意：lastIndexOf() 方法的查找策略是从右往左查找，如果不指定起始索引，则默认从字符串的末尾开始查找

lastlndexOf() 方法

lastIndexOf() 方法用于返回字符（串）在指定字符串中最后一次出现的索引位置，如果能找到则返回索引值，否则返回 -1。该方法也有两种重载形式：str.lastIndexOf(value)；str.lastlndexOf(value, int fromIndex)

1. 根据索引查找 字符串名.charAt(索引值) ；返回的是字符类型char类型
2. 字符串比较 equels(s1)
3. 字符串比较忽略大小写 equelsIgnoreCase(s1)
4. 比较 compareTo(s1) compareToIgnoreCase(s1)忽略大小写
5. 判断s1是否是该字符串的子串 contains(s1)
6. startsWith(prefix) 如果字符串以特定的前缀开始，返回true ;endsWith 如果字符串以特定的后缀开始，返回true;

Stringbuffer

**字符串反转**

StringBuffer 类中的 reverse() 方法用于将字符串序列用其反转的形式取代。

StringBuffer 对象.reverse();

StringBuffer sb=new StringBuffer("java");

sb.reverse();

System.out.println(sb); //输出：avaj

**构造方法**

StringBuffer 类提供了 3 个构造方法来创建一个字符串，如下所示：

StringBuffer() 构造一个空的字符串缓冲区，并且初始化为 16 个字符的容量。

StringBuffer(int length) 创建一个空的字符串缓冲区，并且初始化为指定长度 length 的容量。

StringBuffer(String str) 创建一个字符串缓冲区，并将其内容初始化为指定的字符串内容 str，字符串缓冲区的初始容量为 16 加上字符串 str 的长度也是字符串构建器的容量，也就是capacity()的大小。

//定义一个空的字符串缓冲区，含有16个字符的容量

StringBuffer str1=new StringBuffer();

//定义一个含有10个字符容量的字符串缓冲区

StringBuffer str2=new StringBuffer(10);

//定义一个含有(16+4)的字符串缓冲区，"青春无悔"为4个字符

StringBuffer str3=new StringBuffer("青春无悔");

/\*

\*输出字符串的容量大小

\*capacity()方法返回字符串的容量大小

\*/

System.out.println(str1.capacity()); //输出 16

System.out.println(str2.capacity()); //输出 10

System.out.println(str3.capacity()); //输出 20