# 1.1、String面试题

## 字符型常量和字符串常量的区别

形式上: 字符常量是单引号引起的一个字符 字符串常量是双引号引起的若干个字符

含义上: 字符常量相当于一个整形值(ASCII值),可以参加表达式运算 字符串常量代表一个地址值(该字符串在内存中存放位置)

占内存大小 字符常量只占一个字节 字符串常量占若干个字节(至少一个字符结束标志)

## 什么是字符串常量池？

字符串常量池位于堆内存中，专门用来存储字符串常量，可以提高内存的使用率，避免开辟多块空间存储相同的字符串，在创建字符串时 JVM 会首先检查字符串常量池，如果该字符串已经存在池中，则返回它的引用，如果不存在，则实例化一个字符串放到池中，并返回其引用。

## String底层

String 底层就是一个 char 类型的数组，如果要描述一段文本，就需要用多个 char 类型的变量，也就是一个 char 类型数组，比如“你好” 就是长度为2的数组 char[] chars = {‘你’,‘好’};

## 字符串特性

* 不变性：String 是只读字符串，是一个典型的 immutable 对象，对它进行任何操作，其实都是创建一个新的对象，再把引用指向该对象。不变模式的主要作用在于当一个对象需要被多线程共享并频繁访问时，可以保证数据的一致性。
* 常量池优化：String 对象创建之后，会在字符串常量池中进行缓存，如果下次创建同样的对象时，会直接返回缓存的引用。
* final：使用 final 来定义 String 类，表示 String 类不能被继承，提高了系统的安全性。

**private** **final** **char** value[];

## String str="i"与 String str=new String(“i”)一样吗？

不一样，因为内存的分配方式不一样。String str="i"的方式，java 虚拟机会将其分配到常量池中；而 String str=new String(“i”) 则会被分到堆内存中。

## String 类的常用方法都有那些？

charAt()：返回指定索引处的字符。

endsWith()：判断是否以什么字符串结尾。

equals()：字符串比较。

equalsIgnoreCase()：忽略大小写比较。

getBytes()：将String编码为byte序列，并存储到新的byte数组

indexOf(String str)：返回指定第一次字符的索引。

indexOf(String str,int fromIndex)：返回指定第一次字符的索引，从指定开始

lastIndexOf(int ch)：返回指定字符在此字符串最右边出现的索引

lastIndexOf(int ch，int fromIndex)：返回指定字符在此字符串最后一次出现的索引，从指定的索引开始反向搜索

length()：返回字符串长度。

matches(String regs)：告知字符串是否匹配给指定的正则表达式。

replace()：字符串替换, 支持bai字符的替换du,也支持字符串的替换。

replaceAll(String regex，String replacement)：基于规则表达式的替换替换, 比如,可以通过replaceAll("\\d", "\*")把一个字符串所有的数字字符都换成星号。

split(String regx)：分割字符串，返回一个分割后的字符串数组。

startWith()：判断是否以指定前缀开始。

substring(int beginIndex,int endIndex)：截取字符串，返回一个新的字符串，它是此字符串的子字符串，包前不包后。

toCharArray()：将此字符串转换为一个新的字符数组。

toUpperCase()：将字符串转成大写。

toLowerCase()：将字符串转成小写。

trim()：去除字符串两端空白。

String.ValueOf()：底层是toString()，区别是value对空指针进行了处理。

## 在使用 HashMap 的时候，用 String 做 key 有什么好处？

* HashMap 内部实现是通过 key 的 hashcode 来确定 value 的存储位置，因为字符串是不可变的，所以当创建字符串时，它的 hashcode 被缓存下来，不需要再次计算，所以相比于其他对象更快

**String和StringBuffer、StringBuilder的区别是什么？String为什么是不可变的**

**可变性**

String类中使用字符数组保存字符串，private　final　char　value[]，所以string对象是不可变的。StringBuilder与StringBuffer都继承自AbstractStringBuilder类，在AbstractStringBuilder中也是使用字符数组保存字符串，char[] value，这两种对象都是可变的。

**线程安全性**

String中的对象是不可变的，也就可以理解为常量，线程安全。AbstractStringBuilder是StringBuilder与StringBuffer的公共父类，定义了一些字符串的基本操作，如expandCapacity、append、insert、indexOf等公共方法。StringBuffer对方法加了同步锁或者对调用的方法加了同步锁，所以是线程安全的。StringBuilder并没有对方法进行加同步锁，所以是非线程安全的。

**性能**

每次对String 类型进行改变的时候，都会生成一个新的String对象，然后将指针指向新的String 对象。StringBuffer每次都会对StringBuffer对象本身进行操作，而不是生成新的对象并改变对象引用。相同情况下使用StirngBuilder 相比使用StringBuffer 仅能获得10%~15% 左右的性能提升，但却要冒多线程不安全的风险。

**对于三者使用的总结**

如果要操作少量的数据用 = String

单线程操作字符串缓冲区 下操作大量数据 = StringBuilder

多线程操作字符串缓冲区 下操作大量数据 = StringBuffer