1.String 为什么要设计成不可变的?
答: String 设计成不可变的原因包括、⑴设计考虑、⑵效率问题、⑶安全性 三大方面， 其中
⑴设计考虑- 当创建一个String对象时,假如此字符串值已经存在于常量池中,则不会创建一个新的对象,而是引用已经存在的对象。
⑵效率问题- String的不可变的特点 字符串不变性保证了hash码的唯一性,因此可以放心地进行缓存，是一种性能优化手段,意味着不必每次都去计算新的哈希码 比如在hashMap中。
⑶安全性- String被许多的Java类(库)用来当做参数,例如 网络连接地址URL,文件路径path,还有反射机制所需要的String参数等, 假若String不是固定不变的,将会引起各种安全隐患。

2.final、finally 和 finalize 的区别?
答: final 关键字 用于修饰 类、变量、方法 具有不可变的特性 ，⑴被修饰的类不可被继承、⑵被修饰的变量值不可被修改、⑶被修饰的方法 只能被调用，不能被覆盖，但是可以被重载。
finally 子句 是try-catch异常捕捉的一部分,finally子句中的语句是一定会被执行的，比如在io、db操作时 可以在finally子句处理关闭io、db资源, 保证了资源的合理回收。
finalize 方法 来自于java.lang.Object，用于回收资源。可以为任何一个类添加finalize方法。finalize方法将在垃圾回收器清除对象之前调用，在实际应用中，不要依赖使用该方法回收任何短缺的资源，这是因为很难知道这个方法什么时候被调用。

3.static 关键字有什么作用?
答: static关键字主要作用有点，⑴为某特定数据类型或对象分配单的存储空间，而与创建对象的个数无关。⑵希望某个方法或属性与类而不是对象关联在一起，也就是说，在不创建对象的情况下就可以通过类来直接调用方法或使用类的属性。 static在Java语言中主要有四种使用情况:成员变量、成员方法、代码块及内部类。
(1)static 成员变量(类变量) Java类静态变量属于类,只要静态变量所在的类被加载,这个静态变量就会被分配空间,因此就可以被使用了 静态变量的引用有两种方式，分别为“类静态变量”和“对象静态变量”。
A.类静态变量 直接类名.操作符 引用。 B.对象静态变量 只要对象被创建后 实例变量才会被分配空间,才能被使用,它在内存中存在多个副本 只能用“对象.静态变量”的方式来引用。
(2)static 成员方法 与变量类似,static 方法是类的方法，可以通过“类.静态方法”的方式调用。static方法中不能使用this和super关键字，不能调用非static方法，只能访问所属类的静态成员变量和成员方法，因为当static方法被调用时，这个类的对象可能还没被创建，即使已经被创建了，也无法确定调用哪个对象的方法。同理，static 方法也不能访问非static类型的变量。实现了单例。
(3)Static代码块 Static代码块又叫静态代码块，在类中独立于成员变量和成员函数的代码块。它不在任何一个方法体内，JVM在加载类的时候执行静态代码块，多个存在则顺序执行，常用来初始化静态变量。而且只被执行一次，如果是在主类中，会优先

于main方法执行。
(4)Static内部类 被static修饰的类相当于外部类，但是内部类不能与外部类的类名相同，只有内部类才能被定义为static（可以理解为内部类也是类的一个成员）。

4.列举 Java 的集合以及集合之间的继承关系?
答: 集合类存放的都是对象的引用，而非对象本身，出于表达上的便利，我们称集合中的对象就是指集合中对象的引用（reference)。集合类型主要有3种：set(集）、list(列表）和map(映射)。
继承关系图====> https://upload-images.jianshu.io/upload_images/2529760-c09de6b64377d284.png?imageMogr2/auto-orient/
简书解释====> https://www.jianshu.com/p/e203b23acf45
Collection和Map最大的区别就是Collection存储的是一组对象；Map是以“键值对”的形式对对象进行的管理。
Iterator是迭代器，Iterable是接口。很多类，像List、Set、HashMap不直接实现迭代器接口Iterator，而是去实现Iterable接口。
Collection是一个集合接口。它提供了对集合对象进行进本操作的通用接口方法。
Collections是一个工具类。内有多个对集合对象进行操作的静态方法，不能实例化。

5.List、Set、Map 的区别?
答: List和Set是实现了Collection接口，Map是个接口,Map 接口最流行的几个实现类是 HashMap、LinkedHashMap、Hashtable 和 TreeMap。（HashMap、TreeMap最常用）。
List 特点: a可以允许重复的对象, b可以插入多个null元素, c是一个有序容器, 保持了每个元素的插入顺序，输出的顺序就是插入的顺序, d常用的实现类有 ArrayList、LinkedList 和 Vector。ArrayList 最为流行,它提供了使用索引的随意访问，而 LinkedList 则对于经常需要从 List 中添加或删除元素的场合更为合适。
Set特点: a不允许重复对象, b无序容器，你无法保证每个元素的存储顺序,TreeSet通过 Comparator 或者 Comparable 维护了一个排序顺序, c只允许一个 null 元素, d Set 接口最流行的几个实现类是 HashSet、LinkedHashSet 以及 TreeSet。最流行的是基于 HashMap 实现的 HashSet；TreeSet 还实现了 SortedSet 接口，因此 TreeSet 是一个根据其 compare() 和 compareTo() 的定义进行排序的有序容器。
Map特点: a Map不是collection的子接口或者实现类。Map是一个接口, b Map 的 每个 Entry 都持有两个对象,也就是一个键一个值，Map 可能会持有相同的值对象但键对象必须是唯一的, c TreeMap 也通过 Comparator 或者 Comparable 维护了一个排序顺序, d Map 里你可以拥有随意个 null 值但最多只能有一个 null 键, e Map 接口最流行的几个实现类是 HashMap、LinkedHashMap、Hashtable 和 TreeMap。（HashMap、TreeMap最常用）。

6.ArrayList、LinkedList 的区别?
答: 对于访问元素,ArrayList觉得优于LinkedList,因为LinkedList要移动指针, 对于新增和删除操作add和remove,LinedList比较占优势,因为ArrayList要移动数据。

ArrayList特点: a ArrayList 是线性表(数组), b get() 直接读取第几个下标,复杂度O(1), c add(E)添加元素,直接在后面插入,复杂度O(1), d add(index,E)添加元素,在第index位置上插入元素,后面的元素向后移动,复杂度O(n), c remove() 删除元素, 后面的元素需要逐个移动,复杂度O(n)。

LinkedList特点: a LinkedList是链表的操作, b get()获取第几个元素,依次遍历复杂度O(n), c add(E)添加到末尾,复杂度O(1), d add(index,E)添加到第index元素后,需要先查找到位置 然后移动指针指向操作,复杂度O(n), e remove() 删除元素,直接指针指向操作,复杂度O(1)。

7.HashMap,HashTable,ConcurrentHashMap 实现原理以及区别?

答: HashMap,HashTable,ConcurrentHashMap 都是Map的实现类。

HashMap特点:a底层数组+链表实现，可以存储null键和null值，线程不安全,b初始size为16,扩容:newsize=oldsize*2，size一定为2的n次幂c扩容针对整个Map,每次扩容时,原来数组中的元素依次重新计算存放位置,并重新插入, c 插入元素后才判断该不该扩容，有可能无效扩容（插入后如果扩容，如果没有再次插入,就会产生无效扩容）, d 当Map中元素总数超过Entry数组的75%,触发扩容操作,为了减少链表长度,元素分配更均匀, e 计算index方法: index = hash & (tab.length – 1)。

HashTable特点: a 底层数组+链表实现,无论key还是value都不能为null,线程安全,实现线程安全的方式是在修改数据时锁住整个HashTable,效率低,ConcurrentHashMap做了相关优化, b 初始size为11，扩容：newsize = olesize*2+1, c 计算index的方法：index = (hash & 0x7FFFFFFF) % tab.length。

ConcurrentHashMap特点: a 底层采用分段的数组+链表实现,线程安全。b 通过把整个Map分为N个Segment,可以提供相同的线程安全,但是效率提升N倍,默认提升16倍。(读操作不加锁,由于HashEntry的value变量是 volatile的，也能保证读取到最新的值。), c Hashtable的synchronized是针对整张Hash表的,即每次锁住整张表让线程独占,ConcurrentHashMap允许多个修改操作并发进行,其关键在于使用了锁分离技术, d 有些方法需要跨段,比如size()和containsValue(),它们可能需要锁定整个表而而不仅仅是某个段,这需要按顺序锁定所有段,操作完毕后,又按顺序释放所有段的锁, e 扩容: 段内扩容(段内元素超过该段对应Entry数组长度的75%触发扩容,不会对整个Map进行扩容), 插入前检测不需要扩容,有效避免无效扩容。

8.String、StringBuffer、StringBuilder 之间的区别?

答: 三者在执行速度方面的比较：StringBuilder > StringBuffer > String 这个优先级是相对的,

⑴如果要操作少量的数据 使用String 。

⑵单线程操作字符串缓冲区 下操作大量数据 使用StringBuilder。

⑶多线程操作字符串缓冲区 下操作大量数据 使用StringBuffer。

String 特点: a 设计考虑, 当创建一个String对象时,假如此字符串值已经存在于常量池中,则不会创建一个新的对象,而是引用已经存在的对象。 b 效率问题 String的不可变的特点 确保hash码的唯一性 来保证String 在一些容器中可以放心的缓存，c 其次是保证了 String在java类 /库 当做参数时的 安全性

StringBuffer特点: 线程安全
StringBuilder特点: 非线程安全