1. String能被继承吗？为什么？

不能被继承，因为String类有final修饰符，而final修饰的类是不能被继承的。

1. ArrayList 是如何实现的,ArrayList 和 LinkedList 有什么区别？ArrayList 是如何实现库容的？

ArrayList是通过数组来实现的，读取性能很高，随机访问时间复杂度为O(1)，适用于读大于写的场景

LinkedList是是通过双向队列来实现的，更新效率更高，写只需要修改前后两个节点的相关引用，但是读取效率比较低，需要最多遍历一半长度的队列，适用与写大于读的场景

只要size > 数组的长度，就会触发grow，其中增长比例是原来的容量的一半

1. == 和 equals 的区别是什么？

一、对象类型不同

1、equals()：是超类Object中的方法。

2、==：是操作符。

二、比7a64e58685e5aeb931333366306437较的对象不同

1、equals()：用来检测两个对象是否相等，即两个对象的内容是否相等。

2、==：用于比较引用和比较基本数据类型时具有不同的功能。

三、运行速度不同

1、equals()：没有==运行速度快。

2、==：运行速度比equals()快，因为==只是比较引用。

1. 用过哪些 Map 类，都有什么区别，HashMap 是线程安全的吗？并发下使用的 Map 是什么，他们内部原理分别是什么方式？ hashcode，扩容， 默认容量？
2. 进程和线程的区别，进程间如何通讯，线程间如何通讯？

(1)、地址空间：同一进程的线程共享本进程的地址空间，而进程之间则是独立的地址空间；

(2)、资源拥有：同一进程内的线程共享本进程的资源，但是进程之间的资源是独立的；

(3)、一个进程崩溃后，在保护模式下不会对其他进程产生影响，但是一个线程崩溃整个进程都死掉，所以多进程要比多线程健壮；

(4)、进程切换时，消耗的资源大，效率高。所以涉及到频繁的切换时，使用线程要好于进程。同样如果要求同时进行并且又要共享某些变量的并发操作，只能用线程不能用进程；

(5)、执行过程：每个独立的进程有一个程序运行的入口、顺序执行序列和程序出口。但是线程不能独立执行，必须依存在应用程序中，由应用程序提供多个线程执行控制。

(6)、线程是处理器调度的基本单位，但是进程不是。

进程通讯

(1)、管道( pipe )：

管道是一种半双工的通信方式，数据只能单向流动，而且只能在具有亲缘关系的进程间使用。进程的亲缘关系通常是指父子进程关系。

(2)、有名管道 (namedpipe) ：

有名管道也是半双工的通信方式，但是它允许无亲缘关系进程间的通信。

(3)、信号量(semophore ) ：

信号量是一个计数器，可以用来控制多个进程对共享资源的访问。它常作为一种锁机制，防止某进程正在访问共享资源时，其他进程也访问该资源。因此，主要作为进程间以及同一进程内不同线程之间的同步手段。

(4)、消息队列( messagequeue ) ：

消息队列是由消息的链表，存放在内核中并由消息队列标识符标识。消息队列克服了信号传递信息少、管道只能承载无格式字节流以及缓冲区大小受限等缺点。

(5)、信号 (sinal ) ：

信号是一种比较复杂的通信方式，用于通知接收进程某个事件已经发生。

(6)、共享内存(shared memory ) ：

共享内存就是映射一段能被其他进程所访问的内存，这段共享内存由一个进程创建，但多个进程都可以访问。共享内存是最快的 IPC 方式，它是针对其他进程间通信方式运行效率低而专门设计的。它往往与其他通信机制，如信号两，配合使用，来实现进程间的同步和通信。

(7)、套接字(socket ) ：

套接口也是一种进程间通信机制，与其他通信机制不同的是，它可用于不同设备及其间的进程通信。

线程通信

(1)、锁机制：包括互斥锁、条件变量、读写锁

a. 互斥锁提供了以排他方式防止数据结构被并发修改的方法。

b. 读写锁允许多个线程同时读共享数据，而对写操作是互斥的。

c. 条件变量可以以原子的方式阻塞进程，直到某个特定条件为真为止。对条件的测试是在互斥锁的保护下进行的。条件变量始终与互斥锁一起使用。

(2)、信号量机制(Semaphore)：包括无名线程信号量和命名线程信号量

(3)、信号机制(Signal)：类似进程间的信号处理线程间的通信目的主要是用于线程同步，所以线程没有像进程通信中的用于数据交换的通信机制。

1. Cookie和Session的区别？

1、存储位置不同

cookie的数据信息存放在客户端浏览器上。

session的数据信息存放在服务器上。

2、存储容量不同

单个cookie保存的数据<=4KB，一个站点最多保存20个Cookie。

对于session来说并没有上限，但出于对服务器端的性能考虑，session内不要存放过多的东西，并且设置session删除机制。

3、存储方式不同

cookie中只能保管ASCII字符串，并需要通过编码方式存储为Unicode字符或者二进制数据。

session中能够存储任何类型的数据，包括且不限于string，integer，list，map等。

4、隐私策略不同

cookie对客户端是可见的，别有用心的人可以分析存放在本地的cookie并进行cookie欺骗，所以它是不安全的。

session存储在服务器上，对客户端是透明对，不存在敏感信息泄漏的风险。

5、有效期上不同

开发可以通过设置cookie的属性，达到使cookie长期有效的效果。

session依赖于名为JSESSIONID的cookie，而cookie JSESSIONID的过期时间默认为-1，只需关闭窗口该session就会失效，因而session不能达到长期有效的效果。

6、服务器压力不同

cookie保管在客户端，不占用服务器资源。对于并发用户十分多的网站，cookie是很好的选择。

session是保管在服务器端的，每个用户都会产生一个session。假如并发访问的用户十分多，会产生十分多的session，耗费大量的内存。

7、浏览器支持不同

假如客户端浏览器不支持cookie：

cookie是需要客户端浏览器支持的，假如客户端禁用了cookie，或者不支持cookie，则会话跟踪会失效。关于WAP上的应用，常规的cookie就派不上用场了。

运用session需要使用URL地址重写的方式。一切用到session程序的URL都要进行URL地址重写，否则session会话跟踪还会失效。

假如客户端支持cookie：

cookie既能够设为本浏览器窗口以及子窗口内有效，也能够设为一切窗口内有效。

session只能在本窗口以及子窗口内有效。

8、跨域支持上不同

cookie支持跨域名访问。

session不支持跨域名访问。

1. equals方法实现

equals()方法需要根据业务而来，取对象属性中标识对象业务唯一标识来进行比较

实现了equals方法，同时需要实现hashcode方法，为了维护统一性

public boolean equals(Object anObject) {

if (this == anObject) {

return true;

} if (anObject instanceof String) {

String anotherString = (String)anObject;

int n = value.length;

if (n == anotherString.value.length) {

char v1[] = value;

char v2[] = anotherString.value;

int i = 0;

while (n-- != 0) {

if (v1[i] != v2[i])

return false;

i++;

}

return true;

}

}

return false;

1. 线程状态BLOCKED和WAITING有什么区别？

线程处于BLOCKED状态的场景。

当前线程在等待一个monitor lock，比如等待执行synchronized代码块或者使用synchronized标记的方法。

在synchronized块中循环调用Object类型的wait方法

线程处于WAITING状态的场景。

调用Object对象的wait方法，但没有指定超时值。

调用Thread对象的join方法，但没有指定超时值。

调用LockSupport对象的park方法。

1. 事务的实现原理

原子性：使用 undo log ，从而达到回滚

持久性：使用 redo log，从而达到故障后恢复

隔离性：使用锁以及MVCC,运用的优化思想有读写分离，读读并行，读写并行

一致性：通过回滚，以及恢复，和在并发环境下的隔离做到一致性。

1. 反射的原理，反射创建类实例的三种方式是什么？

反射就是指程序在运行时能够动态的获取到一个类的类型信息的一种操作

通过一个类的全限定名来获取定义此类的二进制字节流。

将这个字节流所代表的的静态存储结构转化为方法区的运行时数据结构。

在内存中生成一个代表这个类的java.lang.Class对象，作为方法区这个类的各种数据的访问入口。

获取一个类的Class对象的三种方法：

// 1

Class class = Class.forName("com.zkw.TestClass");

// 2

Class class = TestClass.class;

// 3

TestClass tc = new TestClass();

Class class = tc.getClass();

1. synchronized 和 volatile 的区别是什么？

作用：

synchronized 表示只有一个线程可以获取作用对象的锁，执行代码，阻塞其他线程。

volatile 表示变量在 CPU 的寄存器中是不确定的，必须从主存中读取。保证多线程环境下变量的可见性；禁止指令重排序。

区别：

synchronized 可以作用于变量、方法、对象；volatile 只能作用于变量。

synchronized 可以保证线程间的有序性（猜测是无法保证线程内的有序性，即线程内的代码可能被 CPU 指令重排序）、原子性和可见性；volatile 只保证了可见性和有序性，无法保证原子性。

synchronized 线程阻塞，volatile 线程不阻塞。

1. 什么情况下会发生栈内存溢出？

如果线程请求的栈深度大于虚拟机所允许的深度，将抛出StackOverflowError异常。 如果虚拟机在动态扩展栈时无法申请到足够的内存空间，则抛出OutOfMemoryError异常

1. 说说你对 Spring 的理解，非单例注入的原理？它的生命周期？循环注入的原理， aop 的实现原理，说说 aop 中的几个术语，它们是怎么相互工作的？
2. Java设计模式熟悉哪一种，请简单写出来

public static Singleton getInstance() {

if(singleton == null) {

synchronized (Singleton.class) {

if(singleton==null) {

singleton = new Singleton();

}

}

}

return singleton;

}

1. 冒泡排序算法？

　　public static void main(String[] args) {  
　　int[] arr = new int[]{-12,3,2,34,5,8,1};  
　　//冒泡排序  
　　for(int i = 0;i < arr.length-1;i++){  
　　　　for(int j = 0;j <arr.length-1-i;j++){  
　　　　　　if(arr[j] >arr[j+1]){  
　　　　　　int temp = arr[j];  
　　　　　　arr[j] = arr[j+1];  
　　　　　　arr[j+1] = temp;  
　　　　　　}  
　　　　}  
　　}  
　　//遍历  
　　for (int i = 0; i < arr.length; i++) {  
　　　　System.out.println(arr[i]+"\t");  
　　}  
　　}

1. 存储引擎 MyISAM 和 InnoDB区别？

1、MyISAM不支持事务，InnoDB是事务类型的存储引擎，当我们的表需要用到事务支持的时候，那肯定是不能选择MyISAM了。

2、MyISAM只支持表级锁，BDB支持页级锁和表级锁默认为页级锁，而InnoDB支持行级锁和表级锁默认为行级锁

3、MyISAM引擎不支持外键，InnoDB支持外键

4、MyISAM引擎的表在大量高并发的读写下会经常出现表损坏的情况

5、对于count()查询来说MyISAM更有优势

6、 InnoDB是为处理巨大数据量时的最大性能设计，它的CPU效率可能是任何其它基于磁盘的关系数据库引擎所不能匹敌的。

7、MyISAM支持全文索引（FULLTEXT），InnoDB不支持

8、MyISAM引擎的表的查询、更新、插入的效率要比InnoDB高

1. delete、drop、truncate区别

1、在速度上，一般来说，drop> truncate > delete。

2、在使用drop和truncate时一定要注意，虽然可以恢复，但为了减少麻烦，还是要慎重。

3、如果想删除部分数据用delete，注意带上where子句，回滚段要足够大；

   如果想删除表，当然用drop；

   如果想保留表而将所有数据删除，如果和事务无关，用truncate即可；

   如果和事务有关，或者想触发trigger，还是用delete；

   如果是整理表内部的碎片，可以用truncate跟上reuse stroage，再重新导入/插入数据

student学生表和score分数表

CREATE  TABLE  student (

id  INT(10)  NOT NULL  UNIQUE  PRIMARY KEY  , --- 学号

name  VARCHAR(20)  NOT NULL , --- 姓名

sex  VARCHAR(4)  , --- 性别

birth  YEAR, --- 出生年份

department  VARCHAR(20) , --- 院系

address  VARCHAR(50)  ---- 家庭住址

);

创建score表。SQL代码如下：

CREATE  TABLE  score (

id  INT(10)  NOT NULL  UNIQUE  PRIMARY KEY  AUTO\_INCREMENT ---编号 ,

stu\_id  INT(10)  NOT NULL , --- 学号

c\_name  VARCHAR(20) , -- 课程名称

grade  INT(10) ---分数

);

1. 计算每个学生的总成绩

 SELECT student.id,name,SUM(grade) FROM student,score

     WHERE student.id=score.stu\_id

 GROUP BY id;

1. 计算每个考试科目的平均成绩

 SELECT c\_name,AVG(grade) FROM score GROUP BY c\_name;

1. 查询同时参加计算机和英语考试的学生的信息

 SELECT \*  FROM student WHERE id =

 ( SELECT stu\_id FROM score

WHERE stu\_id IN ( SELECT stu\_id FROM score WHERE c\_name=  '计算机')  AND c\_name= '英语' );