<多线程面试 44 题和答案：线程锁+线程池+线程同步等>

1、并发编程三要素?

1）原子性

原子性指的是一个或者多个操作，要么全部执行并且在执行的过程中不被其他操作打断，要么就全部都不执行。

2）可见性

可见性指多个线程操作一个共享变量时，其中一个线程对变量进行修改后，其他线程可以立即看到修改的结果。

实现可见性的方法：

synchronized 或者 Lock：保证同一个时刻只有一个线程获取锁执行代码，锁释放之前把最新的值刷新到主内存，实现可见性。

3）有序性

有序性，即程序的执行顺序按照代码的先后顺序来执行。

2、多线程的价值?

1）发挥多核 CPU 的优势

多线程，可以真正发挥出多核 CPU 的优势来，达到充分利用 CPU 的目的，采用多线程的方式去同时完成几件事情而不互相干扰。

2）防止阻塞

从程序运行效率的角度来看，单核 CPU 不但不会发挥出多线程的优势，反而会因为在单核 CPU 上运行多线程导致线程上下文的切换，而降低程序整体的效率。但是单核 CPU 我们还是要应用多线程，就是为了防止阻塞。试想，如果单核 CPU 使用单线程，那么只要这个线程阻塞了，比方说远程读取某个数据吧，对端迟迟未返回又没有设置超时时间，那么你的整个程序在数据返回回来之前就停止运行了。多线程可以防止这个问题，多条线程同时运行，哪怕一条线程的代码执行读取数据阻塞，也不会影响其它任务的执行。

3）便于建模

这是另外一个没有这么明显的优点了。假设有一个大的任务 A，单线程编程，那么就要考虑很多，建立整个程序模型比较麻烦。但是如果把这个大的任务 A 分解成几个小任务，任务 B、任务 C、任务 D，分别建立程序模型，并通过多线程分别运行这几个任务，那就简单很多了。

3、创建线程的有哪些方式?

1）继承 Thread 类创建线程类

2）通过 Runnable 接口创建线程类

3）通过 Callable 和 Future 创建线程

4.创建线程的三种方式的对比?

1）采用实现 Runnable、Callable 接口的方式创建多线程。

优势是:

线程类只是实现了 Runnable 接口或 Callable 接口，还可以继承其他类。

在这种方式下，多个线程可以共享同一个 target 对象，所以非常适合多个相同线程来处理同一份资源的情况，从而可以将 CPU、代码和数据分开，形成清晰的模型，较好地体现了面向对象的思想。

劣势是:

编程稍微复杂，如果要访问当前线程，则必须使用 Thread.currentThread()方法。

2）使用继承 Thread 类的方式创建多线程

优势是:

编写简单，如果需要访问当前线程，则无需使用 Thread.currentThread()方法，直接使用 this 即可获得当前线程。

劣势是:

线程类已经继承了 Thread 类，所以不能再继承其他父类。

3）Runnable 和 Callable 的区别

- Callable 规定（重写）的方法是 call()，Runnable 规定（重写）的方法是 run()。
- Callable 的任务执行后可返回值，而 Runnable 的任务是不能返回值的。
- Call 方法可以抛出异常，run 方法不可以。
- 运行 Callable 任务可以拿到一个 Future 对象，表示异步计算的结果。它提供了检查计算是否完成的方法，以等待计算的完成，并检索计算的结果。通过 Future 对象可以了解任务执行情况，可取消任务的执行，还可获取执行结果。

5、线程的状态流转图

线程的生命周期及五种基本状态：

同步块，这意味着同步块之外的代码是异步执行的，这比同步整个方法更提升代码的效率。请知道一条原则：**同步的范围越小越好。**

44.Java 线程数过多会造成什么异常?

1)线程的生命周期开销非常高

2)消耗过多的 CPU 资源

如果可运行的线程数量多于可用处理器的数量，那么有线程将会被闲置。大量空闲的线程会占用许多内存，给垃圾回收器带来压力，而且大量的线程在竞争 CPU 资源时还将产生其他性能的开销。

3)降低稳定性

JVM 在可创建线程的数量上存在一个限制，这个限制值将随着平台的不同而不同，并且承受着多个因素制约，包括 JVM 的启动参数、Thread 构造函数中请求栈的大小，以及底层操作系统对线程的限制等。如果破坏了这些限制，那么可能抛出 OutOfMemoryError 异常。