一、corePoolSize 线程池核心线程大小

线程池中会维护一个最小的线程数量，即使这些线程处理空闲状态，他们也不会 被销毁，除非设置了allowCoreThreadTimeOut。这里的最小线程数量即是corePoolSize。

二、maximumPoolSize 线程池最大线程数量

一个任务被提交到线程池后，首先会缓存到工作队列（后面会介绍）中，如果工作队列满了，则会创建一个新线程，然后从工作队列中的取出一个任务交由新线程来处理，而将刚提交的任务放入工作队列。线程池不会无限制的去创建新线程，它会有一个最大线程数量的限制，这个数量即由maximunPoolSize来指定。

三、keepAliveTime 空闲线程存活时间

一个线程如果处于空闲状态，并且当前的线程数量大于corePoolSize，那么在指定时间后，这个空闲线程会被销毁，这里的指定时间由keepAliveTime来设定

四、unit 空间线程存活时间单位

keepAliveTime的计量单位
五、workQueue 工作队列

新任务被提交后，会先进入到此工作队列中，任务调度时再从队列中取出任务。jdk中提供了四种工作队列：

①ArrayBlockingQueue

基于数组的有界阻塞队列，按FIFO排序。新任务进来后，会放到该队列的队尾，有界的数组可以防止资源耗尽问题。当线程池中线程数量达到corePoolSize后，再有新任务进来，则会将任务放入该队列的队尾，等待被调度。如果队列已经是满的，则创建一个新线程，如果线程数量已经达到maxPoolSize，则会执行拒绝策略。

②LinkedBlockingQuene

基于链表的无界阻塞队列（其实最大容量为Interger.MAX），按照FIFO排序。由于该队列的近似无界性，当线程池中线程数量达到corePoolSize后，再有新任务进来，会一直存入该队列，而不会去创建新线程直到maxPoolSize，因此使用该工作队列时，参数maxPoolSize其实是不起作用的。

③SynchronousQuene

一个不缓存任务的阻塞队列，生产者放入一个任务必须等到消费者取出这个任务。也就是说新任务进来时，不会缓存，而是直接被调度执行该任务，如果没有可用线程，则创建新线程，如果线程数量达到maxPoolSize，则执行拒绝策略。

④PriorityBlockingQueue

具有优先级的无界阻塞队列，优先级通过参数Comparator实现。

六、threadFactory 线程工厂

创建一个新线程时使用的工厂，可以用来设定线程名、是否为daemon线程等等

七、handler 拒绝策略

当工作队列中的任务已到达最大限制，并且线程池中的线程数量也达到最大限制，这时如果有新任务提交进来，该如何处理呢。这里的拒绝策略，就是解决这个问题的，jdk中提供了4中拒绝策略：

①CallerRunsPolicy

该策略下，在调用者线程中直接执行被拒绝任务的run方法，除非线程池已经shutdown，则直接抛弃任务。

②AbortPolicy

该策略下，直接丢弃任务，并抛出RejectedExecutionException异常。

③DiscardPolicy

该策略下，直接丢弃任务，什么都不做。

④DiscardOldestPolicy

该策略下，抛弃进入队列最早的那个任务，然后尝试把这次拒绝的任务放入队列