1. 程序计数器：

程序计数器是一块较小的内存空间，可以看作线程所执行的字节码行号指示器。字节码的解释工作就是通过改变程序计数器的值进行选取下一条需要执行的指令，分支、循环、跳转、异常处理、线程恢复等基础功能都是依赖这个计数器完成的。并且，java多线程是采用时间片轮转实现的，任意时间，一个cpu都只会执行一个线程，因此切换线程后能否恢复到正确的执行位置，每个线程都需要一个独立的程序计数器，各个线程都有独立的程序计数器，相互独立，互不干扰

2. java虚拟机栈

线程私有、与线程生命周期相同，虚拟机栈描述的是java方法执行的内存模型。每个方法被执行的同时就会被创建一个栈帧、用于存放局部变量表、操作数栈、动态连接、方法出口信息等，每个方法的执行就对应着一个栈帧在虚拟机中出战入站的过程。局部变量表存放了编译器可知道各种基本类型，对象引用（不同于对象本身们可能是指向一个对象起始地址的引用指针、可能是执行一个代表对象的句柄或者其他与此对象的位置）和returnAddrss类型

3. 本地方法栈

与java虚拟机栈的功能相似，区别就是执行java native方法。有些虚拟机直接将这两个合二为一（hotspot）

4. java堆

java虚拟机管理内存中最大的一块、java堆是被所有线程共享的一块内存区域。此区域唯一的目的就是存放对象实例，机会所有的对象实例都分配在此。因为是gc管理的主要区域，因此又成为gc堆， java堆又分为新生代和老年代，再细致就是eden空间、from survior空间、to survior空间（-Xmx - Xms控制）

5. 方法区

与java堆一样，是线程共享的内存区域、用于存放已经被虚拟机加载的类信息、常量、静态变量、即时编译后的代码等数据

6. 运行常量池

是方法区的一部分、class文件中除了类信息，方法、字段、接口信息外，还有一项信息就是常量池。用于存放编译器生成的各种字面量与符号引用，这部分内容将在类加载后进入方法区的运行时常量池中存放。常见string.intern()

7. 直接内存

直接内存不是虚拟机运行时数据区的一部分，也不是java虚拟机规范中定于的内存区域，nlo中，引用了基于channel和buffer的i/o方法，它可以使用Native函数库直接分配堆外内存，然后通过一个存储在java堆中的DirectByteBuffer对象作

为这块内存的引用进行操作，直接内存的分配不受java堆大小的限制，会受本机总内存限制