第一章：

导读从“hallo world”说起，简单的代码下是富含深度的提问。诚然，我在学习代码的路途中确实没有对这些深思熟虑过，提到的问题我基本都无法答出来。在过去学习的路上，基本都是有需求就完成而不是深挖里面的奥义。这本书让我很好的开阔了视野，了解到计算机的奥秘。

计算机是什么？这本书把计算机深化成最重要的三个东西：cpu、内存、i/o控制芯片。而在计算机进化的路上，cpu计算频率增加导致内存跟不上等说明三者有着很强的协调关系。即使cpu更新很快但还是无法满足人们日益增长的需求，于是诞生了对称多处理器。这里面多核的讲法很有意思，理论上来说多增加一个cpu，系统的运行速度就会越快，但是作者举了个非常好的例子解释道：就比如一个女人可以花10个月生出一个孩子，但是10个女人并不能在一个月就生出一个孩子一样。

接着讲到了层的概念，这里我看的并不是很明了。作者讲到每个层次都需要相互通信，通信需要协议，即接口，接口由下层定义，由上层完成需求。那我们使用的c，c++是否也属于接口，我认为应该是的。我对于上下层的概念并不是很清晰，在和智锐讨论求证后我了解了应用层，传输层等，开始了有了基本的认识。

操作系统我在看完后的一个浅显的理解就是：把硬件的处理变成编程人员看的懂的形式，设计接口让人们能直接在计算机上操作。

关于隔离，虚拟地址真是个天才的想法，它是对层的深度理解。通过增加中间层解决物理地址之间容易混乱崩溃。这也正好验证了作者开头引用的名句“计算机科学领域的任何问题都可以通过增加一个间接的中间层来解决”

然后是线程的概念，阐述了多线程的优势，以及它为什么有优势，它是通过什么样的处理方式来达到快速响应任务。但是这作为一个概念的词语，我没有看到实例来具象化这个线程，所以理解起来会有点吃力。

第六章：

程序类似与菜谱，进程类似于做菜，简明生动。

由于程序执行的指令和数据都必须要在内存之中，但是物理内存又不足以支撑，故而引用了覆盖装入和页映射，这利用了程序的局部性原理，将常用的的模块装在内存中，将不用的模块存在磁盘中。

覆盖装入应该是程序员排了个树状模块，每个依赖关系都组织成一个树状结构，等到要使用的时候，再调用其中之一。

页映射就是把所有指令和数据变成若干个页，等到使用时再装载。

从操作系统的角度看可执行文件的装载进程的建立，创建一个进程，然后装载相应的可执行文件并且执行，在有虚拟存储的情况下，主要做的事情如下：创建一个独立的虚拟地址空间，读取可执行文件头，并且建立虚拟空间与可执行文件的映射关系将CPU的指令寄存器设置成可执行文件的入口地址，启动执行。

创建虚拟空间即是做了个页目录，可执行文件头即是将这个页和物理内存联系在一起操作系统通过设置CPU的指令寄存器将控制权交给进程。由此进程开始执行。

由于此时数据并没有真正存入到内存之中，所以cpu会检测到空页，这就是页错误。操作系统会自动找到空页所在的虚拟空间定位出他对应的数据再讲这数据存到物理内存中，由于数据内存可能会大于物理内存，所以需要操作系统在这之中起到调配的作用。

很多情况下，一个称重的栈和堆都有一个对应的VMA。堆和栈我的理解是，堆是一个无序的什么都能往里存的空间，栈是有序的先进后出的地址，由栈的地址去找堆的内容。