1. 进程是正在执行的一个程序或命令，每个进程都是一个运行的实体，都有自己的地址空间，并占用一定的系统资源。进程存在于内存当中，是抽象的存在，进程的存在方便系统管理系统中所有运行中的程序。
2. 代码A会生成32个进程，而代码B会生成6个进程。代码A中，每次循环中，调用fork()函数会复制原程序，此后程序状态完全复制，父进程与子进程继续循环调用fork()函数，则每次循环程序变为两倍，最终生成个进程即32个进程。 在代码B中，由于子进程的fork()函数返回值为0，父进程fork()返回值非零，则子进程的循环将会终止，最终只有第一个进程在创建子进程，最终创建了5个子进程，共6个进程。
3. 当进程调用了exit()函数后，该进程并不是马上消失，而是留下一个称为僵尸进程的数据结构。僵尸进程是Linux系统中的另一种特殊进程，它几乎放弃进程退出前占有的所有内存，既没有可执行代码也不能被调度，只在进程列表中保留一个位置，记载进程的退出状态等信息供父进程手机。若父进程中没有回收子进程的代码，子进程将会一直处于僵尸态。

调用wait()函数的进程会被挂起，进入阻塞状态，直到子进程变为僵尸态，wait()函数捕获到该子进程的退出信息时才会转为运行态，回收子进程资源并返回；若没有变为僵尸态的子进程，wait()函数会让进程一直阻塞。若当前进程有多个子进程，只要捕获到一个变为僵尸态的子进程信息，wait()函数就会返回并使进程恢复执行。

1. 信号全程软中断信号，其本质时软件层次上对中断机制的一种模拟，用于提醒进程某件事情已经发生。信号的编号从1开始，其中1~31号信号为常规信号，也就是早期信号模型中的不可靠信号；34~64号信号为实时信号，是用于计算机底层开发中的信号。信号被应用于进程间通信，但它实际并不由进程发送，在遇到某种情况时，内核会发送某个信号到某个进程。
2. 未决状态：发送的信号被阻塞，无法到达进程，内核就会将该信号的状态设置为未决。

作用：对进程来说，若信号的发送过于密集，即在处理信号的同时再次收到信号，那么进程会将后到的信号丢弃。对于信号的发送方来说，应该发送的信号已经发送，自然不会再次发送；但对于作为信号接收方的进程来说，未对信号做出应有处理，这显然是不符合预期的。所以需要信号处于未决状态，防止要处理的信号丢失。

1. 两个进程，对同一个共享内存读写。一个进程写，另一个进程读。通过信号量可以控制读写进程执行顺序，保证在写进程操作未完成时，需要防止其他进程从共享内存中读取数据。进程先从外部文件读取数据，将数据放入共享内存中，再通过fork函数创造子进程，用exec函数族转换子进程为写进程，通过写进程将共享内存的数据写入外部文件。