进程是一个二进制程序的执行过程，在Linux操作系统中，向命令行输入一条命令，按下回车，便会有一个程序被启动。 当程序执行结束后，进程随之消失，进程所用的资源被系统回收。

A代码段会创建32个子进程，B代码段会创建5个子进程。因为代码段A没有进行判断，父进程在第一次循环创建的子进程也会共同参加第二次循环。代码段B进行判断，如果PID==0，也就是如果是子进程，那么就不再循环，所以只会创建5个子进程。



僵尸进程放弃了进程退出前所占用的所有内存，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收集。当进程调用了exit（）函数后，就会变成僵尸进程。如果父进程中没有回收子进程的代码，子进程会一直处于僵尸态。也就是没有父进程回收的退出的子进程是僵尸进程。

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；若没有变为将事态的子进程，wait（）函数会让进程一直阻塞。只要捕获到一个变为僵尸态的子进程的信息，wait（）函数就会返回并使进程恢复执行。

信号全程软中断信号，其本质是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。



发送的信号被阻塞，无法到达进程时，内核就会将该信号的状态设置为未决状态。

未决状态可以解决常规信号不可靠的问题，如果信号掩码中某个信号对应的位被设置为1，信号会被屏蔽，进入阻塞状态，此时内核会修改未决信号集中该信号对应的位为1，使该信号处于未决状态，之后除非该信号被解除屏蔽，否则内存不会再向进程发送这个信号。



先创建一个信号量，再把信号量初始化，在共享内存写入数据时，如果内存没满，就执行信号量的P操作，使信号量减一；在共享内存读取数据时，如果内存不为空，就执行信号量的V操作，使信号量加一。