**Question 1**

代码段A：

代码执行远远超过6次

代码段B：

代码共执行6次

原因：代码段A执行时，子进程也会创建进程。代码段B由于子进程返回的pid为0，所以每次创建子进程后，由于返回值为0，循环中断，所以只会执行6次，只有6个进程。

**Question 2**

子进程在执行exit（）函数后，几乎放弃进程退出前占用的内存，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。处于这种状态的子进程就是僵尸进程。

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的信息，wait（）函数就会返回并使进程恢复执行。

**Question 3**

信号全称软中断信号，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。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。

**Question 4**

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。

**Question 5**

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。