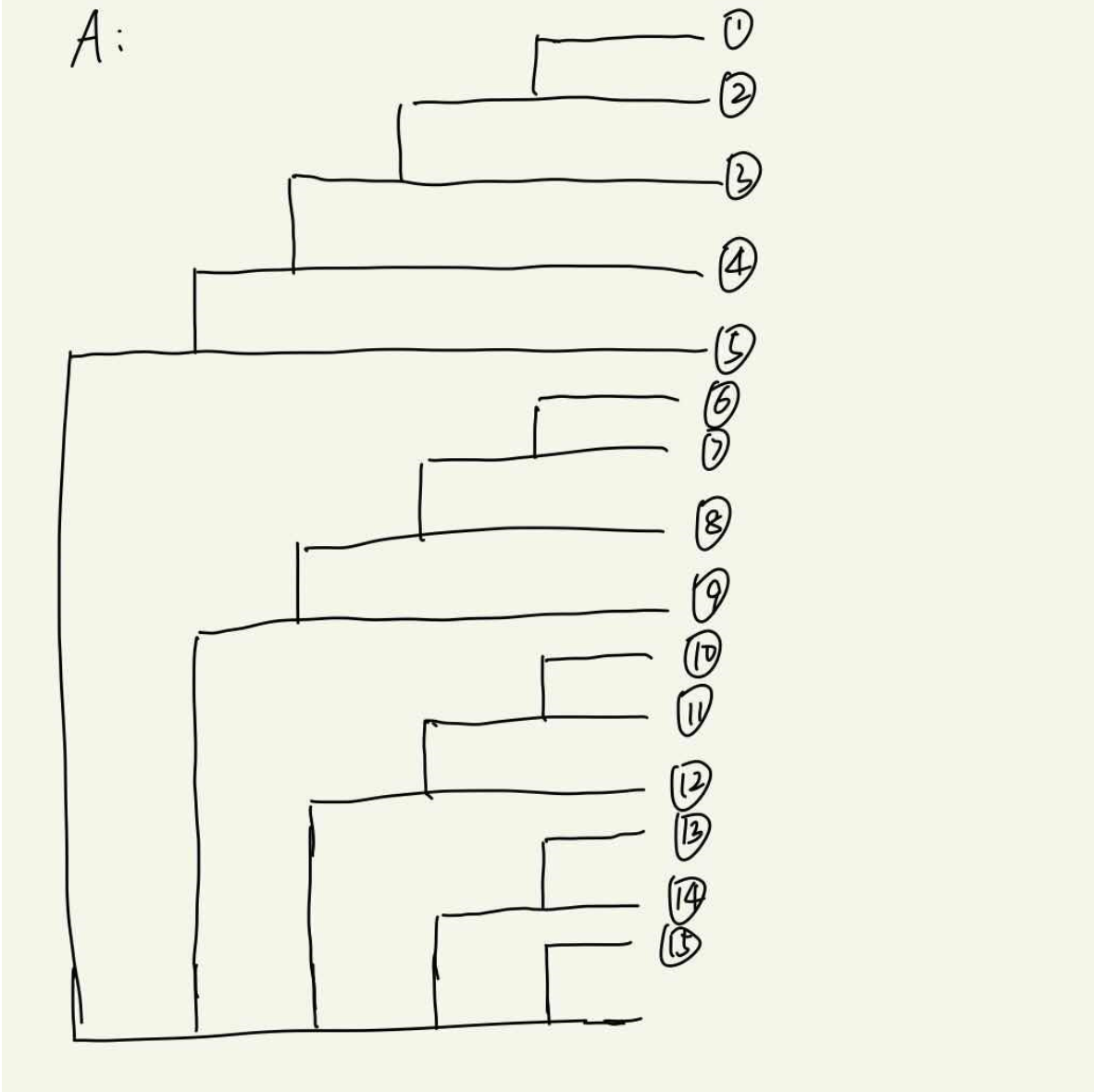
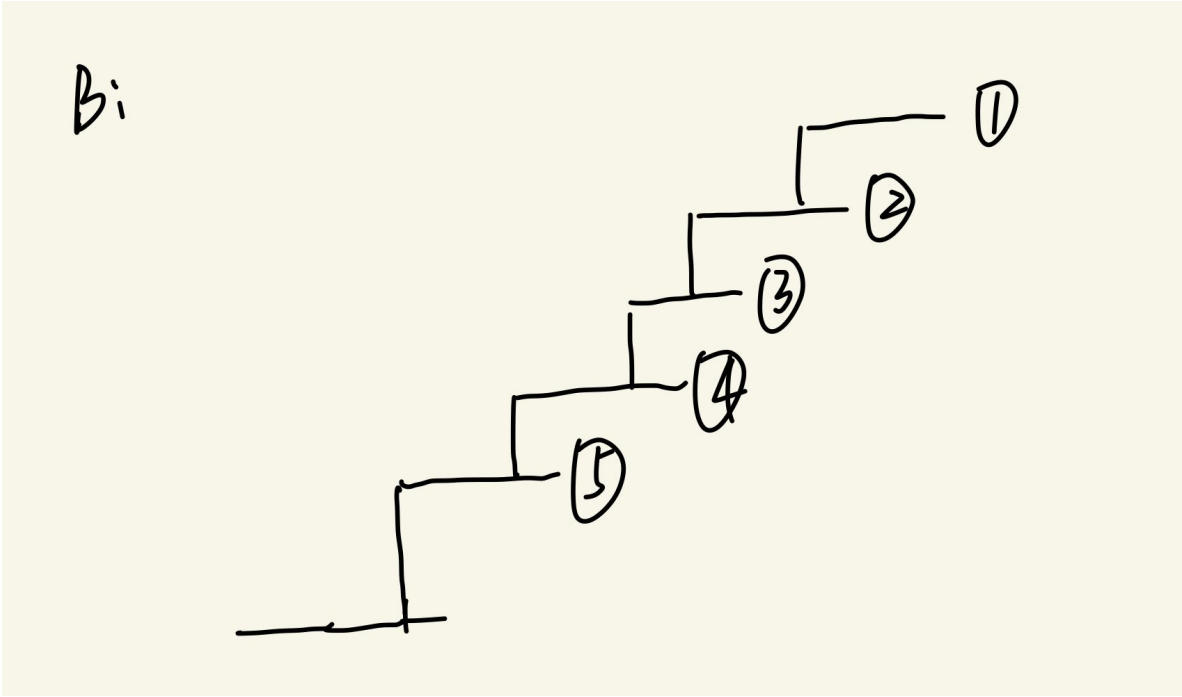
1

进程是描述程序执行过程和资源共享的基本单位，这个概念的作用是为了刻画一个进程执行时的资源分配，共享资源情况，执行过程，以及其他所有的控制和管理这个程序运行的东西需要使用到的一种可以存储、保存、维护、管理这些东西的特殊的数据结构。

2

代码段A和代码段B的执行结果区别在于最终创建的进程的数量，经过分析，代码A创建了15个进程，而代码B创建了5个进程。
有这样的区别主要在于代码B循环中有一个判断当前进程是否为父进程的操作，如果为父进程，则直接退出循环，避免了父进程在接下来的循环中还会继续创建进程，这样看来，代码B比较合理。





3

僵尸进程：这是一个针对子进程的概念，当子进程终止时，如果没有被正确地清除，即父进程没有调用wait()函数，则该子进程成为僵尸进程。

根据上面的僵尸进程概念的描述，可以知道，一个子进程要成为僵尸进程，那么该子进程便要在父进程调用wait()函数前已死亡，但是wait()不仅可以等待活着的子进程，也可以对僵尸进程起到效果，可以将其退出状态抽取出来返回，并且清除该子进程。
具体的过程如下：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进程的信息，wait()函数就会返回并使进程恢复执行。

4

信号是实现Linux系统中进程之间通信的方式之一，本质上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。

5

在进程的PCB中存在两个信号集，一个是信号掩码（signal mask），另一个是未决信号（signal pending）。若mask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pending中该信号对应的位为1，此时该信号即处于未决状态，之后除非该信号被解除屏蔽，否则内核不会再向进程发送这个信号。
信号存在未决状态的作用是为了解决下述情况：对进程来说，若信号的发送过于密集，即在处理信号的同时在此收到信号，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，不会再次发送；对于信号接收方的进程来说，又未对该信号做出应有的处理就丢弃。有了信号屏蔽机制（即信号存在未决状态），便可以将后到的信号设置为未决状态，之后决定是否向进程再次发送该信号，便不会出现常规信号不可靠这个问题了。

6

首先要创建一片共享内存空间，然后将需要用到这片共享内存的程序与其绑定，然后创建一个二值信号量（这里针对两个进程之间的读写同步），定义P操作是使得信号量-1，定义V操作是使得信号量+1。
之后在代码中会发生写入操作的地方，先执行P操作，使得该写操作的进程不会与其他进程同时使用共享内存，然后在写操作执行完成后执行V操作，这样其他读操作的进程就可以正常访问了。