

1. 见PPT10-18页。
2. **缺陷：.**

这个外壳程序最大的缺陷是，当程序在后台运行时，父进程在创建子进程之后，并不会回收子进程，而是简单地输出子进程的processID和命令行文本之后就直接退出了eval函数，转而读取下一条命令行指令。由于父进程（shell）永远在内存中运行，不会结束。这样，子进程在终止之后并不会被init进程回收，而将会成为一个僵死进程，造成了内存泄漏。

**修正：**

修正这个缺陷的办法是引入信号在父进程和子进程之间通讯。在子进程结束的时候，内核会发送一个SIGCHLD的信号给父进程，通常来说父进程对于这个信号的默认行为是忽略。我们可以在父进程中使用signal函数修改父进程对于这个信号的默认行为。具体做法是，在父进程中定义一个signal handler的函数，通过这个函数来回收子进程。在main函数的signal函数调用中传入这个handler从而改变该进程对SIGCHLD的默认行为。

由于上下文切换的特点，父进程对子进程信号的接受不是当子进程一发出就进行的，因此当父进程创建的子进程有多个的时候，很可能父进程来未来得及处理之前的pending signal，另一个子进程也结束了。由于pending signal是不会排队等待的（can’t be queued），因此如果之前的pending signal未被接收，之后来的同类型的信号会被简单地丢弃。因此在父进程中，一旦接收到一个SIGCHLD信号，意味着至少一个子进程结束了，因此在我们写的handler函数中，需要通过一个while循环，不断地对子进程进行回收，直到wait函数返回-1，说明无法再回收子进程了，这时候我们要检查一下是发生了错误退出while循环还是无法回收子进程而退出循环，如果发生错误，需要打印错误信息。（更详细的介绍见PPT，我只是简单讲一下思路，代码在PPT上）