# ShellLab-Report

21302010042

侯斌洋

本次lab要求实现一个简易的shell，且shell的框架已经给出，故只需要实现其中的一些关键函数即可。这些关键函数其实在CSAPP课本中也给出了大致的思路，不过需要做一些修改来进行完善。

此外，对于系统调用返回值的错误检查，用到了很多包装函数，这些函数也是CSAPP书中提到的csapp.c文件中的函数，相关的说明均已给出了注释。

*下面介绍各个函数实现的大致思路*

**eval：**

整体框架参考CSAPP中图8-24和图8-40的程序，首先通过调用已经给出的parseline函数来将命令行转化为参数列表，之后调用builtin\_cmd函数来区分第一个参数是否为内置命令，若为内置命令则在builtin\_cmd函数内部执行相应的操作，若不为内置命令则把第一个参数当作一个可执行程序进行执行，在这个过程中要创建新的子进程并在子进程中运行该程序。关于函数中的对于信号的阻塞以及前台后台程序的区分等详细内容不在此赘述，函数中都有详细的注释。

**builtin\_cmd：**

该函数主要是通过if语句来判断命令的类型，若为内置命令则执行相应的操作，这些操作集成在不同的函数中，之后返回1或者直接退出（quit）；若不为内置命令则返回0。函数整体思路较为简单明确。

**do\_bgfg：**

该函数主要实现对于前台以及后台进程的一些操作。由于参数既可以为JID也可以为PID，故需要进行区分并根据参数的类型进行不同的操作来获取整个作业的信息，包括PID、JID和job的地址。之后对于bg命令，要在后台重启进程，则利用kill函数发送SIGCONT信号并设置作业的状态；对于fg命令，要在前台重启进程，则利用kill函数发送SIGCONT信号并设置作业的状态，之后还需要一直等待该前台的结束。函数中的各种错误检测不在此赘述，请看详细代码以及注释。

**waitfg：**

该函数较为简单，用一个循环来等待前台作业的结束，用fgpid()作为循环检测条件。当前台作业运行结束，被回收并从jobs中删除后，fgpid()返回0结束等待。

**sigchld\_handler：**

框架参考CSAPP中图8-40的程序，通过循环来回收所有的僵死子进程。由于需要对停止的或因为SIFINT终止的子进程进行提示，故在回收时需要利用WIFEXITED等函数来检查子进程的退出状态。详细内容见代码。

**sigtstp\_handler：**

该函数较为简单由于SIGTSTP信号是针对前台进程组的，故只需要获取前台作业PID并利用kill函数将该信号发送到该进程所在进程组的每一个进程即可。

**sigint\_handler：**

该函数与sigtstp\_handler思路基本相同。