

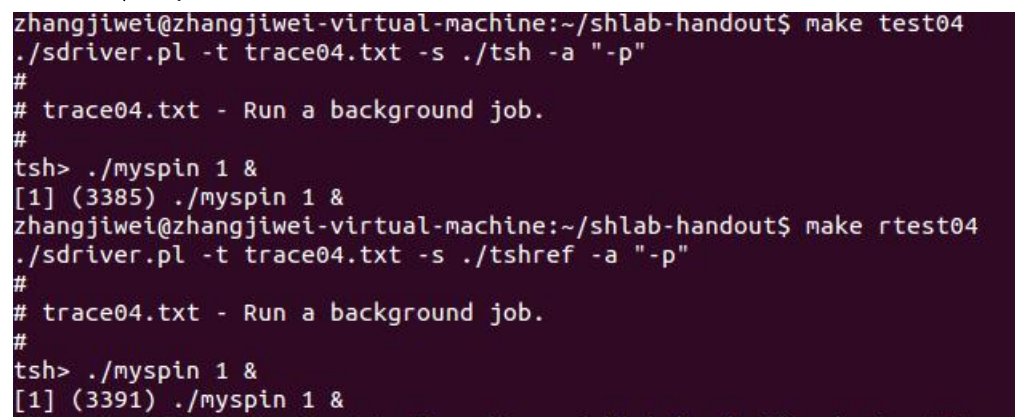
实验日志 2.9

【40%】编程实现 eval() 的后台作业管理功能并使用 trace04 验证；

eval() :

```
void eval(char *cmdline)
{
    char buf[MAXLINE];
    char **argv;
    int bg;
    pid_t pid;
    strcpy(buf,cmdline);
    parseline(buf,argv);
    if(argv[0]==NULL)
        return;
    if(!builtin_cmd(argv))
    {
        if((pid=fork())==0)
        {
            if(execve(argv[0],argv,enviro) < 0)
            {
                printf("%s:Command not found.\n",argv[0]);
                exit(0);
            }
        }
        else
        {
            if(bg)
                addjob(jobs,pid,BG,cmdline);
            if(bg)
            {
                printf("[%d] (%d) %s",pid2jid(pid),pid,cmdline);
            }
        }
    }
    return;
}
```

Trace04 验证:



```
zhangjiwei@zhangjiwei-virtual-machine:~/shlab-handout$ make test04
./sdriver.pl -t trace04.txt -s ./tsh -a "-p"
#
# trace04.txt - Run a background job.
#
tsh> ./myspin 1 &
[1] (3385) ./myspin 1 &
zhangjiwei@zhangjiwei-virtual-machine:~/shlab-handout$ make rtest04
./sdriver.pl -t trace04.txt -s ./tshref -a "-p"
#
# trace04.txt - Run a background job.
#
tsh> ./myspin 1 &
[1] (3391) ./myspin 1 &
```

【20%】学习 trace 测试文件符号（空格，&，#等）、命令、用户程

序 myspin 含义；

文件符号：

空格：用来分隔 ASCII 文本单词序列，分隔命令中的参数。

&：如果一个命令以&结尾，说明它在后台运行

#：以#开头的行是注释行，被解释器忽略，只输出不执行

%：命令行的前缀%表示 jid。%5 即为 jid 5 其中 5 表示 pid 5

命令：

包括内建命令和外部命令

内建命令：quit jobs bg fg 等等 支持作业控制，这些命令由 shell 程序识别并在 shell 程序内部完成运行，通常 在 linux 系统加载运行时 shell 就被加载并驻留在系统内存中，其执行速度比外部命令快。 可以使用 type 来确定一个命令是否是内置命令。

外部命令：在系统加载时并不随系统一起被加载到内存中，而是在需要时才将其调用内存。通常外部命令的实体并不包含在 shell 中，但是其命令执行过程是由 shell 程序控制的。shell 程序管理外部命令执行的路径查找、加载存放，并控制命令的执行。外部命令是在 bash 之外额外安装的，通常放在/bin, /sbin, /usr/bin, /usr/sbin, /usr/local/sbin 等等

用户程序 myspin:

实现的是睡眠函数的功能，命令 myspin<n>表示让程序睡眠 n 秒，与 myint 的区别在于这个函数睡眠好后就自动退出了，没有检测系统错误。

```
/*
 * myspin.c - A handy program for testing your tiny shell
 *
 * usage: myspin <n>
 * Sleeps for <n> seconds in 1-second chunks.
 */
#include <stdio.h>
#include <unistd.h>
#include <stdlib.h>

int main(int argc, char **argv)
{
    int i, secs;

    if (argc != 2) {
        fprintf(stderr, "Usage: %s <n>\n", argv[0]);
        exit(0);
    }
    secs = atoi(argv[1]);
    for (i=0; i < secs; i++)
        sleep(1);
    exit(0);
}
```

【40%】编程实现 jobs 内建命令，使用 trace05 验证。

Clearjob 函数：清楚该 job

Initjobs 函数：初始化 job

Maxjid 函数：返回分配的最大 job ID

Addjob 函数：将一个 job 加入 job 列表

Deletejob 函数：将 PID 为 pid 的 job 列表中删除

Fgpid 函数：返回当前前台工作的 job 的 PID 如果没有则返回 0

Getjobpid 函数：通过 PID 在 job 列表中查找一个 job

Getjobjid 函数：通过 job 的 PID 得到 job 的 PID

Listjobs 函数：打印 job 列表

在 builtin_cmd 函数中添加函数命令为 jobs 的函数

```

/*
int builtin_cmd(char **argv)
{
    if(!strcmp(argv[0], "quit"))//
        exit(0);
    if(!strcmp(argv[0], "&"))
        return 1;
    if(!strcmp(argv[0], "jobs"))
    {
        listjobs(jobs);
        return 1;
    }
    return 0;    /* not a builtin command */
}

```

如果命令为 jobs 则执行 listjobs 函数 并将所有作业打印

```

zhangjiwei@zhangjiwei-virtual-machine:~/shlab-handout$ make test05
./sdriver.pl -t trace05.txt -s ./tsh -a "-p"
#
# trace05.txt - Process jobs builtin command.
#
tsh> ./myspin 2 &
[1] (3398) ./myspin 2 &
tsh> ./myspin 3 &
[2] (3400) ./myspin 3 &
tsh> jobs
[1] (3398) Running ./myspin 2 &
[2] (3400) Running ./myspin 3 &
zhangjiwei@zhangjiwei-virtual-machine:~/shlab-handout$ make rtest05
./sdriver.pl -t trace05.txt -s ./tshref -a "-p"
#
# trace05.txt - Process jobs builtin command.
#
tsh> ./myspin 2 &
[1] (3407) ./myspin 2 &
tsh> ./myspin 3 &
[2] (3409) ./myspin 3 &
tsh> jobs
[1] (3407) Running ./myspin 2 &
[2] (3409) Running ./myspin 3 &
zhangjiwei@zhangjiwei-virtual-machine:~/shlab-handout$

```

验证成功!