2021 SYSTEM PROGRAMMING
Lab2 Report – Shell Lab

자유전공학부 2012-13311 안 효 지

1. How to implement

1) void eval(char* cmdline)

(1) logic flow

해당 함수는 shell의 cmdline을 처리하는 함수이다. ① empty line이 들어온 경우 바로 return 해준다. ② builtin_cmd인지 판단, builtin이면 즉시 처리한다. ③ builtin이 아닌 경우, fork() 후 execve()를 통해 child process에서 해당 job이 돌게 한다. ④ parent process인 shell에서는 parseline()을 통해 'background' job인지 판단하여 addjob()을 한다. Foreground job이라면 해당 process가 종료될 때까지 기다리고 background job이라면 기다릴 필요 없이 바로 다음 cmdline을 eval한다.

(2) Synchronizing flows by blocking signals

eval()에서 주의할 것은 SIGCHLD를 block/unblock하는 타이밍이다. 이를 조절해주지 않으면 scheduler가 어떻게 스케줄링을 하느냐에 따라 원치 않는 결과를 야기할 수 있다. 우리는 parent process에서 addjob()을 한 후, sigchld_handler에서 deletejob()을 수행해야 한다. 하지만 parent process가 addjob()을 수행하기 전에 child process가 terminate되어 sigchld_handler까지 호출이 되고, 존재하지도 않는 job을 delete()한 후에 다시 parent process로 넘어와 이미 끝나버린 job을 addjob()하는 경우가 발생할 수 있다. 이를 방지하기 위해서는 fork()를 하기 전에 sigchld를 block해놓고, parent process는 addjob()이 끝난 후에 unblock, child process는 execve()를 하기 전에 unblock을 해주어야 한다. 그러면 parent process가 addjob()을 수행하기 전에 child가 terminate 되어도 signal receiving은 addjob()을 한 이후에 이루어지므로 언급한 버그를 피할 수 있다.

(3) setpgid(0, 0)

child process가 execve()를 하기 전에 process group id를 해당 child process의 pid로 setting해준다. 그렇지 않으면 signal을 보낼 target이 정확히 설정되지 않아 제대로 된 signaling이 이루어지지 않을 수 있다.

2) int builtin_cmd(char** argv)

(1) built-in-command는 항상 argv[0]에 위치한다. 해당 인자가 quit, jobs, fg/bg인지 판단한다.
(2) quit: 에러로 종료되는 것이 아니기 때문에 exit(0)을 해주었다. 또한, shell process 자체가 종료되는 것이기 때문에 return 값은 굳이 넣어주지 않았다.
(3) jobs: skeleton code에 있는 listjobs() 호출하고 builtin command가 맞기 때문에 return 1.

(4) fg/bg : do_bgfg()를 호출하고 return 1.
(5) 상기한 명령어에 해당하지 않으면 return 0.

3) void do_bgfg(char** argv)

(1) no argv[1]
argv[0]은 bg/fg이지만 그 뒤(argv[1])에 argument가 들어오지 않은 경우를 먼저 체크, 해당하면 tshref의 실행결과와 동일한 문구가 출력되도록 한다.
(2) valid formation
argv[1]이 jid/pid의 올바른 형식을 갖추었는지 체크한다. jid인 경우에는 argv[1]이 %로 시작하고, pid는 숫자이다. 이도 저도 아니면 잘못 들어온 arg이다.
(3) valid jid/pid
형식을 갖추었다면, 실재하는 jid/pid인지 확인해야 한다. 각각 Getjobjid()/getjobpid()로 joblist에 해당 jid/pid를 가진 job이 존재하는지 확인, 존재하지 않으면 에러메시지를 띄우고 return한다.
(4) execute fg/bg
(i) fg command는 ST/BG인 job을 FG로 바꾼다.
① 정확하게 하려면 해당 job이 현재 ST/BG인지, 즉, FG인 상태에서 fg command가 들어온 것은 아닌지 확인을 해야 한다고 생각했었다. 하지만 기존의 foreground job이 running인 상태에서 shell이 새로운 command line을 받을 수는 없다. 따라서 이 부분에서는 FG인지 여부를 판별하지 않아도 된다. 만약 현재 FG인 process가 `fg pid`의 형태로 들어온다면, 저 명령어가 eval되는 시점은 이미 해당 pid process가 종료된 후이므로 “No such process”가 뜰 것이다. 또한, 그러므로 해당 job의 state를 판별하지 않아도 된다.
② SIGCONT를 보내고 해당 job->state = FG로 바꾼 후, foreground로 왔기 때문에 waitfg()를 호출하여 작업이 끝날 때까지 shell이 기다리도록 한다.
(ii) bg command는 ST job을 BG로 바꾸어야 하므로, SIGCONT를 보내고 job->state = BG.

4) void waitfg(pid_t pid)

본 함수는 argument로 들어온 pid를 가진 process가 foreground job이 아닐 때까지 기다리는 역할을 한다. 즉, foreground job이 끝날 때까지 shell에게 기다리라고 명령하는 것이다. 교재에 나와 있는대로 sigsuspend()를 사용해보려 하였으나 에러가 나는데 디버깅에 실패하여 sleep(1)을 사용하였다.

5) void sigchld_handler(int sig)

child process가 종료되거나 정지된 경우, kernel은 parent process에게 SIGCHLD를 보낸다. 본 lab에서는 SIGCHLD가 도착하면 handler를 실행하여 reap과 deletejob()을 해준다.
① waitpid(-1, &status, WNOHAND ¦ WUNTRACED)를 사용했다. 첫 인자인 -1은 wait set을 모든 child process로 설정한다는 의미이다. 세 번째 인자 중 하나인 WNOHANG은 child process가 wait의 대상이 되지 않았다면 기다리지 말고 즉시 return하라는 의미이다. WUNTRACED는 terminated된

child process 뿐만 아니라 stopped child process도 wait의 대상으로 여기라는 의미이다. 즉, waitpid()가 호출되는 시점에 wait set에 속해있는 child process 중 stopped/terminated 상태인 것이 있으면 return하라는 의미이다. Terminated child process는 reap도 해준다.

② 그런데 waidpid()가 while문 안에 들어가 있다. Waitpid()가 호출될 시점에 wait의 대상이 되어 있는 child process를 빠짐없이 처리하기 위해서이다.

③ WIFEXITED는 signal에 의하지 않고 정상적으로 종료된 child인 경우에 true를 return한다. 별다른 부가작업 없이 deletejob()만 해주면 된다.

④ WIFSIGNALED는 signal에 의해 종료된 child의 경우에 true를 return한다. 이 때, WTERMSIG를 사용하면 어떤 signal인지도 알려준다. 메시지를 출력하고 deletejob()을 해준다.

⑤ WIFSTOPPED는 signal에 의해 정지되어 있는 child의 경우에 true를 return한다. WSTOPSIG를 사용하면 어떤 signal이 정지를 시켰는지 알려준다. 종료된 것이 아니기에 deletejob()은 해주지 않고 메시지만 출력한다.

6) void sigint_handler(int sig)

해당 signal이 들어오면 tsh는 종료되지 않고 foreground process group만 종료되어야 한다. 핸들러가 없으면 tsh까지 다 종료가 되어 버리기 때문에 signal handler를 사용해 fg group에만 SIGINT를 보내야 한다. Fgpid()를 이용해 foreground job의 pid를 구하고, kill(-pid, SIGINT)를 통해 해당 pid뿐만이 아니라 해당 pid가 속한 모든 foreground group에 signal을 보낸다.

7) void sigtstp_handler(int sig)

6)과 마찬가지로 tsh 자체를 중지시키는 것이 아니라 foreground process group만 중지시켜야 하므로 핸들러를 설치하고 fg group에 SIGTSTP를 보내면 된다. fgpid()를 이용해 foreground job의 pid를 구하고, kill(-pid, SIGTSTP)를 통해 해당 pid뿐만이 아니라 해당 pid가 속한 모든 foreground group에 signal을 보낸다.

2. What was difficult

① Eval()에서 맨 처음에 sigprocmask(BLOCK, …..)의 위치를 잘못 두어서 디버깅에 많은 시간을 쏟았다. Fork()하기 전에 block 해야 한다는 것은 알고 있었지만, builtin_cmd() 전부터 블록해버리는 바람에 제대로 돌아가지 않았다. 무엇이 잘못되었는지 발견하는 데 한참 걸렸다.

② SIGINT/SIGTSTP가 들어오면 각각의 handler가 kill()을 이용해 다시 해당 signal을 프로세스에게 보낸다. 이 때, 그러면 또 signal handler가 호출되어 또 kill()을 보내고, 무한으로 반복되지 않을까 의문이 있었다. 하지만 직접 코드를 짜서 실행해보니 잘 작동하였다. 생각해보니 signal handler는 1. 6), 7) 에서 언급했듯이 tsh에게는 signal의 영향이 닿지 않게 하기 위하여 shell process에만 설치가 된 것이기 때문에, shell process에서 kill(SIGINT)를 보내면 child process에서는 핸들러가 불리지 않는다는 것을 알게 되었다.

3. What was surprising

① shell이 정상적으로 동작하기 위해서는 sigint, sigtstp로 인한 동작 및 reap과 deletejob()을 sigchld_handler에서 해준다는 사실을 처음 알게 되었다.
② shell이 어떤 과정으로 fg job이 끝날 때까지 기다리는지 알게 되었다.
③ waitpid()가 그저 terminated child process를 reap만 해주는 줄 알고 있었는데, 교재를 자세히 읽어보니 여러 옵션이 있다는 것도 알게 되었다.

4. Result screen shot

r<#>.out : tshref.c output
m<#>.out : my tsh.c output

File: r1.out

```
./sdriver.pl -t trace01.txt -s ./tshref -a "-p"
#
# trace01.txt - Properly terminate on EOF.
#
```

File: m1.out

```
./sdriver.pl -t trace01.txt -s ./tsh -a "-p"
#
# trace01.txt - Properly terminate on EOF.
#
```

File: r2.out

```
./sdriver.pl -t trace02.txt -s ./tshref -a "-p"
#
# trace02.txt - Process builtin quit command.
#
```

File: m2.out

```
./sdriver.pl -t trace02.txt -s ./tsh -a "-p"
#
# trace02.txt - Process builtin quit command.
#
```

File: r3.out

```
./sdriver.pl -t trace03.txt -s ./tshref -a "-p"
#
# trace03.txt - Run a foreground job.
#
tsh> quit
```

File: m3.out

```
./sdriver.pl -t trace03.txt -s ./tsh -a "-p"
#
# trace03.txt - Run a foreground job.
#
tsh> quit
```

File: r4.out

```
./sdriver.pl -t trace04.txt -s ./tshref -a "-p"
#
# trace04.txt - Run a background job.
#
tsh> ./myspin 1 &
[1] (20866) ./myspin 1 &
```

File: m4.out

```
./sdriver.pl -t trace04.txt -s ./tsh -a "-p"
#
# trace04.txt - Run a background job.
#
tsh> ./myspin 1 &
[1] (21547) ./myspin 1 &
```

File: r5.out

```
./sdriver.pl -t trace05.txt -s ./tshref -a "-p"
#
# trace05.txt - Process jobs builtin command.
#
tsh> ./myspin 2 &
[1] (20930) ./myspin 2 &
tsh> ./myspin 3 &
[2] (20932) ./myspin 3 &
tsh> jobs
[1] (20930) Running ./myspin 2 &
[2] (20932) Running ./myspin 3 &
```

File: m5.out

```
./sdriver.pl -t trace05.txt -s ./tsh -a "-p"
#
# trace05.txt - Process jobs builtin command.
#
tsh> ./myspin 2 &
[1] (21564) ./myspin 2 &
tsh> ./myspin 3 &
[2] (21566) ./myspin 3 &
tsh> jobs
[1] (21564) Running ./myspin 2 &
[2] (21566) Running ./myspin 3 &
```

File: r6.out

```
./sdriver.pl -t trace06.txt -s ./tshref -a "-p"
#
# trace06.txt - Forward SIGINT to foreground job.
#
tsh> ./myspin 4
Job [1] (20988) terminated by signal 2
```

File: m6.out

```
./sdriver.pl -t trace06.txt -s ./tsh -a "-p"
#
# trace06.txt - Forward SIGINT to foreground job.
#
tsh> ./myspin 4
Job [1] (21618) terminated by signal 2
```

File: r7.out

```
./sdriver.pl -t trace07.txt -s ./tshref -a "-p"
#
# trace07.txt - Forward SIGINT only to foreground job.
#
tsh> ./myspin 4 &
[1] (21039) ./myspin 4 &
tsh> ./myspin 5
Job [2] (21041) terminated by signal 2
tsh> jobs
[1] (21039) Running ./myspin 4 &
```

File: m7.out

```
./sdriver.pl -t trace07.txt -s ./tsh -a "-p"
#
# trace07.txt - Forward SIGINT only to foreground job.
#
tsh> ./myspin 4 &
[1] (21669) ./myspin 4 &
tsh> ./myspin 5
Job [2] (21671) terminated by signal 2
tsh> jobs
[1] (21669) Running ./myspin 4 &
```

File: r8.out

```
./sdriver.pl -t trace08.txt -s ./tshref -a "-p"
#
# trace08.txt - Forward SIGTSTP only to foreground job.
#
tsh> ./myspin 4 &
[1] (21093) ./myspin 4 &
tsh> ./myspin 5
Job [2] (21095) stopped by signal 20
tsh> jobs
[1] (21093) Running ./myspin 4 &
[2] (21095) Stopped ./myspin 5
```

File: m8.out

```
./sdriver.pl -t trace08.txt -s ./tsh -a "-p"
#
# trace08.txt - Forward SIGTSTP only to foreground job.
#
tsh> ./myspin 4 &
[1] (21689) ./myspin 4 &
tsh> ./myspin 5
Job [2] (21691) stopped by signal 20
tsh> jobs
[1] (21689) Running ./myspin 4 &
[2] (21691) Stopped ./myspin 5
```

File: r9.out

```
./sdriver.pl -t trace09.txt -s ./tshref -a "-p"
#
# trace09.txt - Process bg builtin command
#
tsh> ./myspin 4 &
[1] (21147) ./myspin 4 &
tsh> ./myspin 5
Job [2] (21149) stopped by signal 20
tsh> jobs
[1] (21147) Running ./myspin 4 &
[2] (21149) Stopped ./myspin 5
tsh> bg %2
[2] (21149) ./myspin 5
tsh> jobs
[1] (21147) Running ./myspin 4 &
[2] (21149) Running ./myspin 5
```

File: m9.out

```
./sdriver.pl -t trace09.txt -s ./tsh -a "-p"
#
# trace09.txt - Process bg builtin command
#
tsh> ./myspin 4 &
[1] (21743) ./myspin 4 &
tsh> ./myspin 5
Job [2] (21745) stopped by signal 20
tsh> jobs
[1] (21743) Running ./myspin 4 &
[2] (21745) Stopped ./myspin 5
tsh> bg %2
[2] (21745) ./myspin 5
tsh> jobs
[1] (21743) Running ./myspin 4 &
[2] (21745) Running ./myspin 5
```

File: r10.out

```
./sdriver.pl -t trace10.txt -s ./tshref -a "-p"
#
# trace10.txt - Process fg builtin command.
#
tsh> ./myspin 4 &
[1] (21209) ./myspin 4 &
tsh> fg %1
Job [1] (21209) stopped by signal 20
tsh> jobs
[1] (21209) Stopped ./myspin 4 &
tsh> fg %1
tsh> jobs
```

File: m10.out

```
./sdriver.pl -t trace10.txt -s ./tsh -a "-p"
#
# trace10.txt - Process fg builtin command.
#
tsh> ./myspin 4 &
[1] (21801) ./myspin 4 &
tsh> fg %1
Job [1] (21801) stopped by signal 20
tsh> jobs
[1] (21801) Stopped ./myspin 4 &
tsh> fg %1
tsh> jobs
```

```
stu3@ubuntu:~/workspace$ make rtest11
./sdriver.pl -t trace11.txt -s ./tshref -a "-p"
#
# trace11.txt - Forward SIGINT to every process in foreground process group
#
tsh> ./mysplit 4
Job [1] (512733) terminated by signal 2
tsh> /bin/ps a
    PID TTY      STAT   TIME COMMAND
    662 hvc0     Ss+    0:00 /sbin/agetty -o -p -- \u --keep-baud 115200,38400,9600 hvc0 vt220
    669 tty1     Ss+    0:00 /sbin/agetty -o -p -- \u --noclear tty1 linux
 512452 pts/0    Ss     0:00 -bash
 512705 pts/1    Ss+    0:00 -bash
 512728 pts/0    S+     0:00 make rtest11
 512729 pts/0    S+     0:00 /bin/sh -c ./sdriver.pl -t trace11.txt -s ./tshref -a "-p"
 512730 pts/0    S+     0:00 /usr/bin/perl ./sdriver.pl -t trace11.txt -s ./tshref -a -p
 512731 pts/0    S+     0:00 ./tshref -p
 512736 pts/0    R      0:00 /bin/ps a
```

```
stu3@ubuntu:~/workspace$ make test11
./sdriver.pl -t trace11.txt -s ./tsh -a "-p"
#
# trace11.txt - Forward SIGINT to every process in foreground process group
#
tsh> ./mysplit 4
Job [1] (512724) terminated by signal 2
tsh> /bin/ps a
    PID TTY      STAT   TIME COMMAND
    662 hvc0     Ss+    0:00 /sbin/agetty -o -p -- \u --keep-baud 115200,38400,9600 hvc0 vt220
    669 tty1     Ss+    0:00 /sbin/agetty -o -p -- \u --noclear tty1 linux
 512452 pts/0    Ss+    0:00 -bash
 512705 pts/1    Ss     0:00 -bash
 512719 pts/1    S+     0:00 make test11
 512720 pts/1    S+     0:00 /bin/sh -c ./sdriver.pl -t trace11.txt -s ./tsh -a "-p"
 512721 pts/1    S+     0:00 /usr/bin/perl ./sdriver.pl -t trace11.txt -s ./tsh -a -p
 512722 pts/1    S+     0:00 ./tsh -p
 512727 pts/1    R      0:00 /bin/ps a
```

```
stu3@ubuntu:~/workspace$ make rtest12
./sdriver.pl -t trace12.txt -s ./tshref -a "-p"
#
# trace12.txt - Forward SIGTSTP to every process in foreground process group
#
tsh> ./mysplit 4
Job [1] (512744) stopped by signal 20
tsh> jobs
[1] (512744) Stopped ./mysplit 4
tsh> /bin/ps a
    PID TTY      STAT   TIME COMMAND
    662 hvc0     Ss+    0:00 /sbin/agetty -o -p -- \u --keep-baud 115200,38400,9600 hvc0 vt220
    669 tty1     Ss+    0:00 /sbin/agetty -o -p -- \u --noclear tty1 linux
 512452 pts/0    Ss     0:00 -bash
 512705 pts/1    Ss+    0:00 -bash
 512739 pts/0    S+     0:00 make rtest12
 512740 pts/0    S+     0:00 /bin/sh -c ./sdriver.pl -t trace12.txt -s ./tshref -a "-p"
 512741 pts/0    S+     0:00 /usr/bin/perl ./sdriver.pl -t trace12.txt -s ./tshref -a -p
 512742 pts/0    S+     0:00 ./tshref -p
 512744 pts/0    T      0:00 ./mysplit 4
 512745 pts/0    T      0:00 ./mysplit 4
 512748 pts/0    R      0:00 /bin/ps a
```

```
stu3@ubuntu:~/workspace$ make test12
./sdriver.pl -t trace12.txt -s ./tsh -a "-p"
#
# trace12.txt - Forward SIGTSTP to every process in foreground process group
#
tsh> ./mysplit 4
Job [1] (512754) stopped by signal 20
tsh> jobs
[1] (512754) Stopped ./mysplit 4
tsh> /bin/ps a
    PID TTY      STAT   TIME COMMAND
    662 hvc0     Ss+    0:00 /sbin/agetty -o -p -- \u --keep-baud 115200,38400,9600 hvc0 vt220
    669 tty1     Ss+    0:00 /sbin/agetty -o -p -- \u --noclear tty1 linux
 512452 pts/0    Ss+    0:00 -bash
 512705 pts/1    Ss     0:00 -bash
 512749 pts/1    S+     0:00 make test12
 512750 pts/1    S+     0:00 /bin/sh -c ./sdriver.pl -t trace12.txt -s ./tsh -a "-p"
 512751 pts/1    S+     0:00 /usr/bin/perl ./sdriver.pl -t trace12.txt -s ./tsh -a -p
 512752 pts/1    S+     0:00 ./tsh -p
 512754 pts/1    T      0:00 ./mysplit 4
 512755 pts/1    T      0:00 ./mysplit 4
 512758 pts/1    R      0:00 /bin/ps a
```

```
stu3@ubuntu:~/workspace$ make rtest13
./sdriver.pl -t trace13.txt -s ./tshref -a "-p"
#
# trace13.txt - Restart every stopped process in process group
#
tsh> ./mysplit 4
Job [1] (512768) stopped by signal 20
tsh> jobs
[1] (512768) Stopped ./mysplit 4
tsh> /bin/ps a
    PID TTY      STAT   TIME COMMAND
    662 hvc0     Ss+    0:00 /sbin/agetty -o -p -- \u --keep-baud 115200,38400,9600 hvc0 vt220
    669 tty1     Ss+    0:00 /sbin/agetty -o -p -- \u --noclear tty1 linux
 512452 pts/0    Ss     0:00 -bash
 512705 pts/1    Ss+    0:00 -bash
 512763 pts/0    S+     0:00 make rtest13
 512764 pts/0    S+     0:00 /bin/sh -c ./sdriver.pl -t trace13.txt -s ./tshref -a "-p"
 512765 pts/0    S+     0:00 /usr/bin/perl ./sdriver.pl -t trace13.txt -s ./tshref -a -p
 512766 pts/0    S+     0:00 ./tshref -p
 512768 pts/0    T      0:00 ./mysplit 4
 512769 pts/0    T      0:00 ./mysplit 4
 512772 pts/0    R      0:00 /bin/ps a
tsh> fg %1
tsh> /bin/ps a
    PID TTY      STAT   TIME COMMAND
    662 hvc0     Ss+    0:00 /sbin/agetty -o -p -- \u --keep-baud 115200,38400,9600 hvc0 vt220
    669 tty1     Ss+    0:00 /sbin/agetty -o -p -- \u --noclear tty1 linux
 512452 pts/0    Ss     0:00 -bash
 512705 pts/1    Ss+    0:00 -bash
 512763 pts/0    S+     0:00 make rtest13
 512764 pts/0    S+     0:00 /bin/sh -c ./sdriver.pl -t trace13.txt -s ./tshref -a "-p"
 512765 pts/0    S+     0:00 /usr/bin/perl ./sdriver.pl -t trace13.txt -s ./tshref -a -p
 512766 pts/0    S+     0:00 ./tshref -p
 512775 pts/0    R      0:00 /bin/ps a
```

```
stu3@ubuntu:~/workspace$ make test13
./sdriver.pl -t trace13.txt -s ./tsh -a "-p"
#
# trace13.txt - Restart every stopped process in process group
#
tsh> ./mysplit 4
Job [1] (512781) stopped by signal 20
tsh> jobs
[1] (512781) Stopped ./mysplit 4
tsh> /bin/ps a
    PID TTY      STAT   TIME COMMAND
    662 hvc0     Ss+    0:00 /sbin/agetty -o -p -- \u --keep-baud 115200,38400,9600 hvc0 vt220
    669 tty1     Ss+    0:00 /sbin/agetty -o -p -- \u --noclear tty1 linux
 512452 pts/0    Ss+    0:00 -bash
 512705 pts/1    Ss     0:00 -bash
 512776 pts/1    S+     0:00 make test13
 512777 pts/1    S+     0:00 /bin/sh -c ./sdriver.pl -t trace13.txt -s ./tsh -a "-p"
 512778 pts/1    S+     0:00 /usr/bin/perl ./sdriver.pl -t trace13.txt -s ./tsh -a -p
 512779 pts/1    S+     0:00 ./tsh -p
 512781 pts/1    T      0:00 ./mysplit 4
 512782 pts/1    T      0:00 ./mysplit 4
 512785 pts/1    R      0:00 /bin/ps a
tsh> fg %1
tsh> /bin/ps a
    PID TTY      STAT   TIME COMMAND
    662 hvc0     Ss+    0:00 /sbin/agetty -o -p -- \u --keep-baud 115200,38400,9600 hvc0 vt220
    669 tty1     Ss+    0:00 /sbin/agetty -o -p -- \u --noclear tty1 linux
 512452 pts/0    Ss+    0:00 -bash
 512705 pts/1    Ss     0:00 -bash
 512776 pts/1    S+     0:00 make test13
 512777 pts/1    S+     0:00 /bin/sh -c ./sdriver.pl -t trace13.txt -s ./tsh -a "-p"
 512778 pts/1    S+     0:00 /usr/bin/perl ./sdriver.pl -t trace13.txt -s ./tsh -a -p
 512779 pts/1    S+     0:00 ./tsh -p
 512788 pts/1    R      0:00 /bin/ps a
```

File: r14.out

```
./sdriver.pl -t trace14.txt -s ./tshref -a "-p"
#
# trace14.txt - Simple error handling
#
tsh> ./bogus
./bogus: Command not found
tsh> ./myspin 4 &
[1] (21270) ./myspin 4 &
tsh> fg
fg command requires PID or %jobid argument
tsh> bg
bg command requires PID or %jobid argument
tsh> fg a
fg: argument must be a PID or %jobid
tsh> bg a
bg: argument must be a PID or %jobid
tsh> fg 9999999
(9999999): No such process
tsh> bg 9999999
(9999999): No such process
tsh> fg %2
%2: No such job
tsh> fg %1
Job [1] (21270) stopped by signal 20
tsh> bg %2
%2: No such job
tsh> bg %1
[1] (21270) ./myspin 4 &
tsh> jobs
[1] (21270) Running ./myspin 4 &
```

File: m14.out

```
./sdriver.pl -t trace14.txt -s ./tsh -a "-p"
#
# trace14.txt - Simple error handling
#
tsh> ./bogus
./bogus: Command not found
tsh> ./myspin 4 &
[1] (21858) ./myspin 4 &
tsh> fg
fg command requires PID or %jobid argument
tsh> bg
bg command requires PID or %jobid argument
tsh> fg a
fg: argument must be a PID or %jobid
tsh> bg a
bg: argument must be a PID or %jobid
tsh> fg 9999999
(9999999): No such process
tsh> bg 9999999
(9999999): No such process
tsh> fg %2
%2: No such job
tsh> fg %1
Job [1] (21858) stopped by signal 20
tsh> bg %2
%2: No such job
tsh> bg %1
[1] (21858) ./myspin 4 &
tsh> jobs
[1] (21858) Running ./myspin 4 &
```

File: r15.out

```
./sdriver.pl -t trace15.txt -s ./tshref -a "-p"
#
# trace15.txt - Putting it all together
#
tsh> ./bogus
./bogus: Command not found
tsh> ./myspin 10
Job [1] (21376) terminated by signal 2
tsh> ./myspin 3 &
[1] (21378) ./myspin 3 &
tsh> ./myspin 4 &
[2] (21380) ./myspin 4 &
tsh> jobs
[1] (21378) Running ./myspin 3 &
[2] (21380) Running ./myspin 4 &
tsh> fg %1
Job [1] (21378) stopped by signal 20
tsh> jobs
[1] (21378) Stopped ./myspin 3 &
[2] (21380) Running ./myspin 4 &
tsh> bg %3
%3: No such job
tsh> bg %1
[1] (21378) ./myspin 3 &
tsh> jobs
[1] (21378) Running ./myspin 3 &
[2] (21380) Running ./myspin 4 &
tsh> fg %1
tsh> quit
```

File: m15.out

```
./sdriver.pl -t trace15.txt -s ./tsh -a "-p"
#
# trace15.txt - Putting it all together
#
tsh> ./bogus
./bogus: Command not found
tsh> ./myspin 10
Job [1] (21922) terminated by signal 2
tsh> ./myspin 3 &
[1] (21924) ./myspin 3 &
tsh> ./myspin 4 &
[2] (21926) ./myspin 4 &
tsh> jobs
[1] (21924) Running ./myspin 3 &
[2] (21926) Running ./myspin 4 &
tsh> fg %1
Job [1] (21924) stopped by signal 20
tsh> jobs
[1] (21924) Stopped ./myspin 3 &
[2] (21926) Running ./myspin 4 &
tsh> bg %3
%3: No such job
tsh> bg %1
[1] (21924) ./myspin 3 &
tsh> jobs
[1] (21924) Running ./myspin 3 &
[2] (21926) Running ./myspin 4 &
tsh> fg %1
tsh> quit
```

File: r16.out

```
./sdriver.pl -t trace16.txt -s ./tshref -a "-p"
#
# trace16.txt - Tests whether the shell can handle SIGTSTP and SIGINT
#     signals that come from other processes instead of the terminal.
#
tsh> ./mystop 2
Job [1] (21439) stopped by signal 20
tsh> jobs
[1] (21439) Stopped ./mystop 2
tsh> ./myint 2
Job [2] (21442) terminated by signal 2
```

File: m16.out

```
./sdriver.pl -t trace16.txt -s ./tsh -a "-p"
#
# trace16.txt - Tests whether the shell can handle SIGTSTP and SIGINT
#     signals that come from other processes instead of the terminal.
#
tsh> ./mystop 2
Job [1] (21985) stopped by signal 20
tsh> jobs
[1] (21985) Stopped ./mystop 2
tsh> ./myint 2
Job [2] (21988) terminated by signal 2
```