2018 시스템 프로그래밍

- Lab 06 -

제출일자	2018.11.08
분 반	02
이 름	진승언
학 번	201404377

Trace 번호 (00)

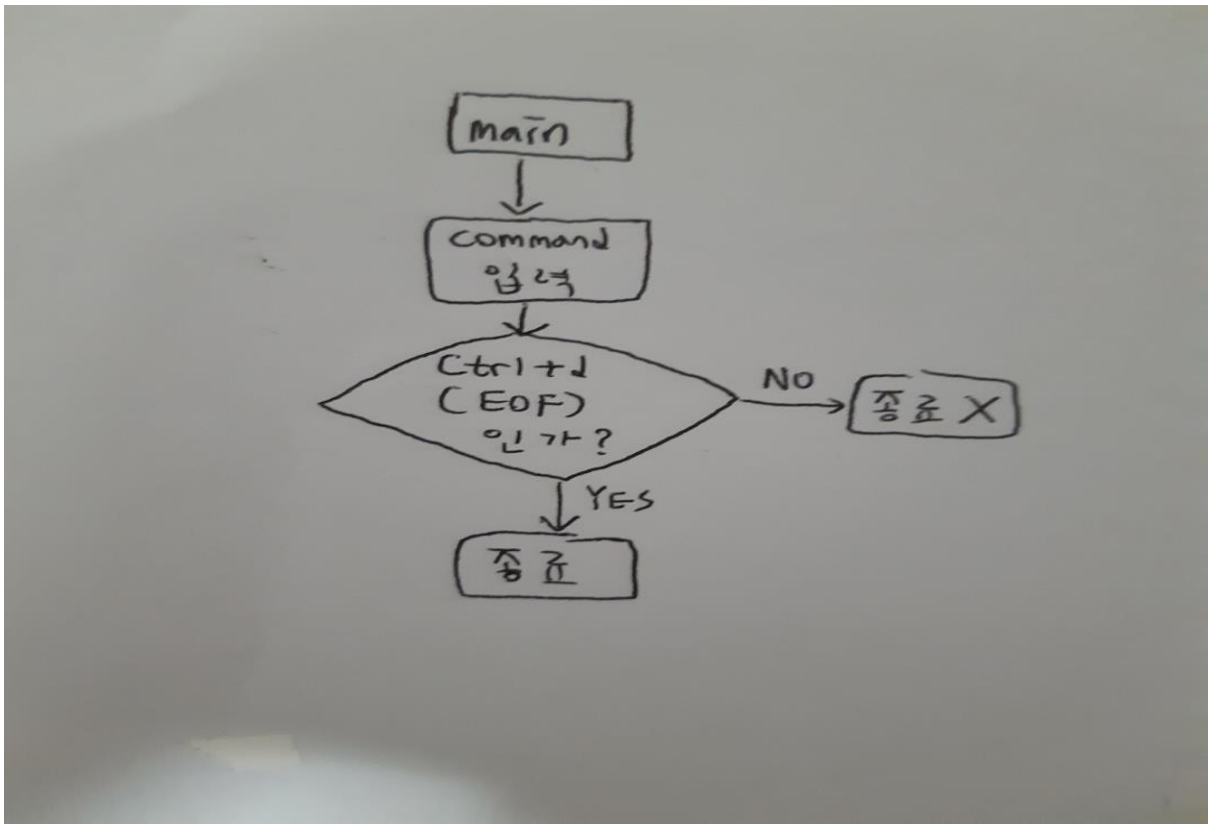
c201404377@2018-sp: ~/shlab-handout

```
c201404377@2018-sp:~/shlab-handout$ vi tsh.c
c201404377@2018-sp:~/shlab-handout$ ./sdriver -V -t 00 -s ./tsh
Running trace00.txt...
Success: The test and reference outputs for trace00.txt matched!
Test output:
#
# trace00.txt - Properly terminate on EOF.
#

Reference output:
#
# trace00.txt - Properly terminate on EOF.
#

c201404377@2018-sp:~/shlab-handout$ ./tsh
eslab_tsh> c201404377@2018-sp:~/shlab-handout$
c201404377@2018-sp:~/shlab-handout$
```

각 trace 별 프로우 차트



```
if (feof(stdin)) { /* End of file (ctrl-d) */
    fflush(stdout);
    fflush(stderr);
    exit(0);
}
```

EOF(End of File)이 입력되면 종료된다. 여기서 EOF란 'ctrl + d' 를 입력했을 때를 의미한다. 이 기능은 처음부터 main()에 위 사진과 같이 구현되어있으므로 따로 건들게 없었다. 따라서 구현하지 않은 tsh도 tshref의 동작과 동일하다.

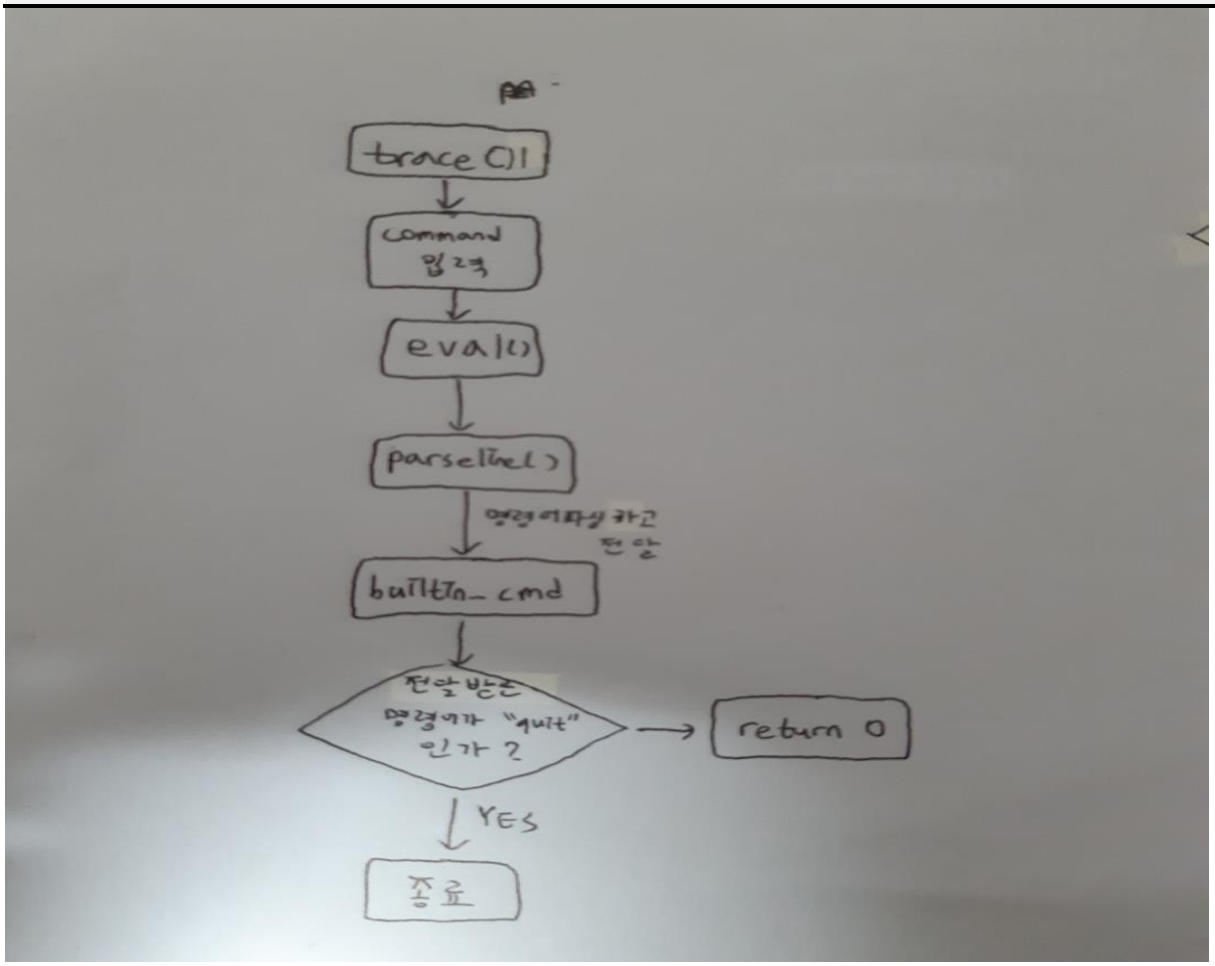
Trace 번호 (01)

```
c201404377@2018-sp: ~/shlab-handout
c201404377@2018-sp:~/shlab-handout$ ./sdriver -V -t 01 -s ./tshref
Running trace01.txt...
Success: The test and reference outputs for trace01.txt matched!
Test output:
#
# trace01.txt - Process builtin quit command.
#

Reference output:
#
# trace01.txt - Process builtin quit command.
#

c201404377@2018-sp:~/shlab-handout$ ./tsh
eslab_tsh> quit
c201404377@2018-sp:~/shlab-handout$
```

각 trace 별 프로우 차트



Trace 해결 방법 설명

```
170 void eval(char *cmdline)
171 {
172     char *argv[MAXARGS];
173
174     //명령어를 parseline을 통해 분리
175     parseline(cmdline, argv);
176     //parsing된 명령어를 전달
177     builtin_cmd(argv);
178
179     return;
180 }
```

```
189 int builtin_cmd(char **argv)
190 {
191     char *cmd = argv[0];
192
193     if(!strcmp(cmd, "quit")){ /* quit command */
194         exit(0);
195     }
196
197     return 0; /* not a builtin command */
198 }
```

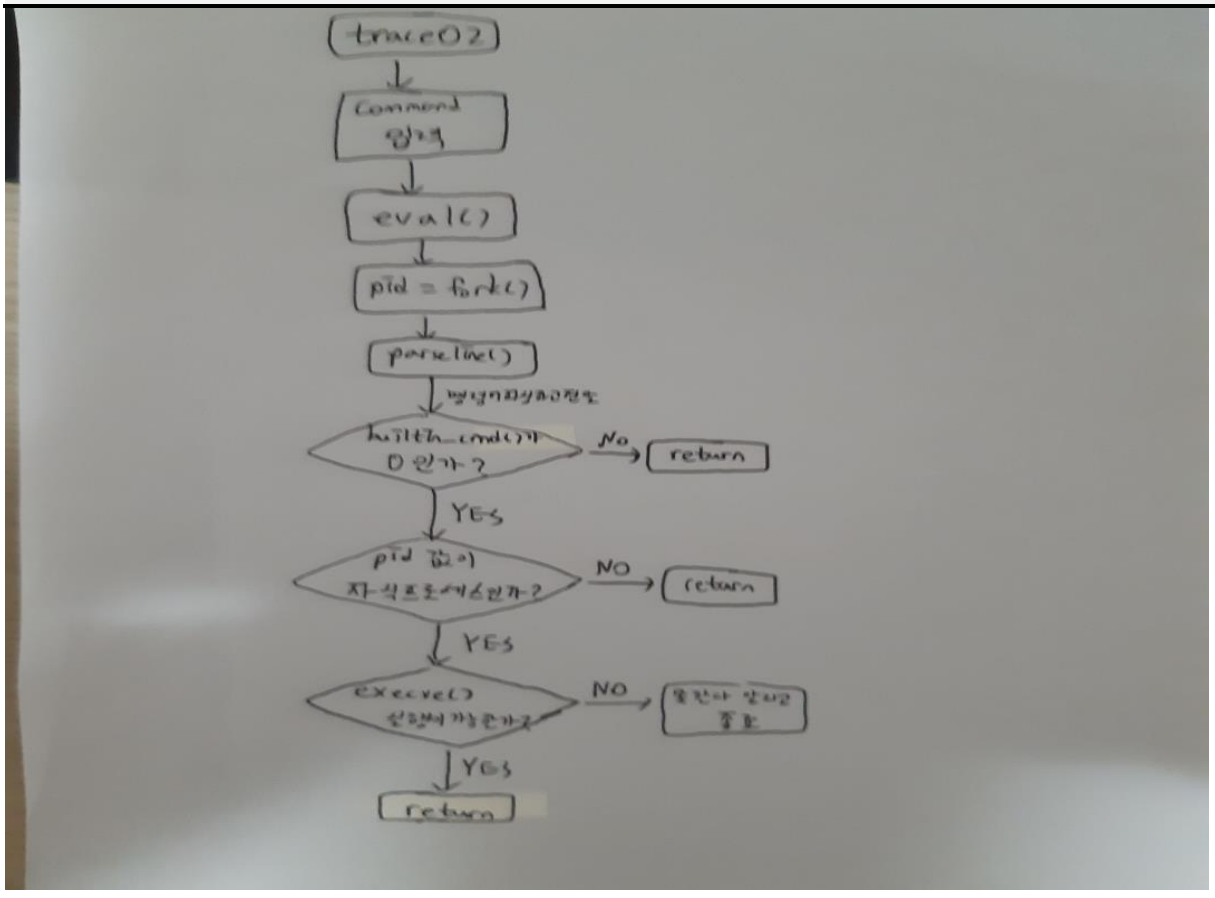
- ❖ eval() 함수에서 입력 받은 명령어를 parseline() 함수에서 파싱하고 builtin_cmd() 함수로 전달되게 하였다. builtin_cmd에서는 해당 명령어가 'quit'인 경우 쉘을 종료할 수 있도록 하였다. 해당 명령어가 quit인지 비교하는것은 strcmp함수를 이용하였다. strcmp함수는 문자열이 같으면 0을 반한한다. 그러므로 해당 명령어가 quit라면 0이반환되고 !(반전)이 되서 exit(0)가 실행되어 종료된다.

Trace 번호 (02)

```
c201404377@2018-sp: ~/shlab-handout
c201404377@2018-sp:~/shlab-handout$ ./sdriver -V -t 02 -s ./tshref
Running trace02.txt...
Success: The test and reference outputs for trace02.txt matched!
Test output:
#
# trace02.txt - Run a foreground job that prints an environment variable
#
# IMPORTANT: You must pass this trace before attempting any later
# traces.  In order to synchronize with your child jobs, the driver
# relies on your shell properly setting the environment.
OSTYPE=/home/sys02/c201404377/bin:/home/sys02/c201404377/.local/bin:/usr/local/s
bin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games:/usr/local/games

Reference output:
#
# trace02.txt - Run a foreground job that prints an environment variable
#
# IMPORTANT: You must pass this trace before attempting any later
# traces.  In order to synchronize with your child jobs, the driver
# relies on your shell properly setting the environment.
OSTYPE=/home/sys02/c201404377/bin:/home/sys02/c201404377/.local/bin:/usr/local/s
bin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games:/usr/local/games
```

각 trace 별 프로우 차트



```
170 void eval(char *cmdline)
171 {
172     char *argv[MAXARGS]; //comand 저 장
173     pid_t pid = fork();          //process ID
174     //명 령 어 를  parseline을  통 해  분 리
175     parseline(cmdline, argv);
176
177
178     if(!builtin_cmd(argv)){
179         if(pid == 0){//Child Process 인  경 우 , execve()수 행
180             if((execve(argv[0], argv, environ) < 0 )){
181                 printf("%s : Command not found\n", argv);
182                 exit(0);
183             }
184         }
185     }
186     return;
187 }
```

trace02는 Foreground 작업 형태로 프로그램을 실행시키는 것이다. pid가 0이면(child process인 경우) execve()함수를 실행시키고 종료된다. execve() 함수는 현재 프로세스를 execve 함수로 호출한 프로그램으로 교체하는 것이다.(호출한 프로그램의 텍스트, 데이터, bss, 스택이 호출된 프로그램의 것으로 교체된다)

위의 결과를 보면 해당 텍스트를 echo를 foreground 형태로 실행하는 것을 볼 수 있다.