2018 시스템 프로그래밍
- Lab 06 -

제출일자	2018.11.08
분 반	02
이 름	박상현
학 번	201702012

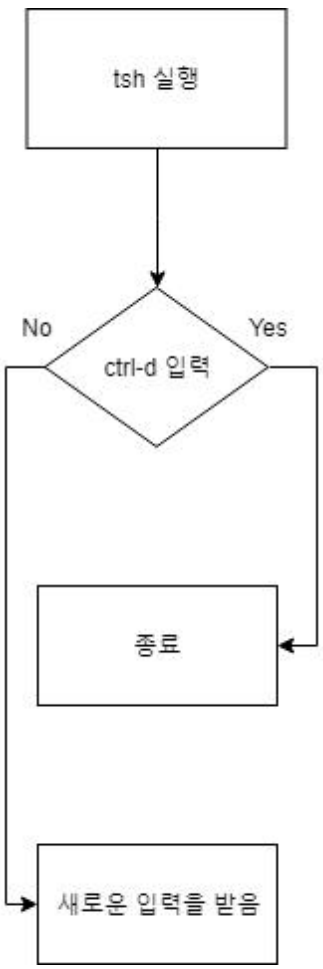
Trace 00

```
c201702012@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.
#

c201702012@2018-sp:~/shlab-handout$
```

trace 00 플로우 차트



trace 00 해결 방법 설명

```
170 void eval(char *cmdline) //     ////////////////////////////////////////////////////////eval/////////
171 {
172
173     char *argv[MAXARGS];    //  command 저장
174     pid_t pid;
175
176     //  명령어를 parseline을 통해 분리
177     parseline(cmdline, argv);
178     //  parsing된 명령어를 전달
179     builtin_cmd(argv);
180
181     if(feof(stdin)){ /* trace00    End of file (ctrl - d) */
182         fflush(stdout);
183         exit(0);
184     }
185
186     if(!builtin_cmd(argv)){  /* trace02  */
```

void eval(char *cmdline)함수에 End-Of-File인 경우, 즉 ctrl-d를 누른 경우 tsh 가 종료되도록 코드를 작성하였다.

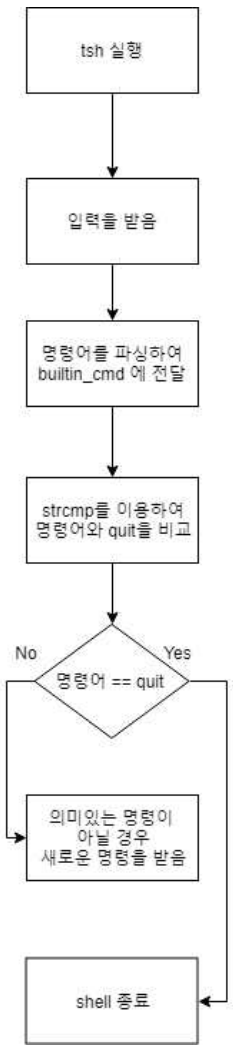
Trace 01

```
c201702012@2018-sp:~/shlab-handout$ ./sdriver -V -t 01 -s ./tsh
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.
#

c201702012@2018-sp:~/shlab-handout$
```

trace 01 플로우 차트



trace 01 해결 방법 설명

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

```
200 int builtin_cmd(char **argv)    //      ////////////////////////////////////////////////builtin//////
201 {
202     char *cmd = argv[0];
203
204     if(!strcmp(cmd, "quit")){ /* trace01  quit command */
205         exit(0);
206     }
207
208
209     return 0;   /* not a builtin command */
210 }
211
```

parseline(cmdline, argv)를 통해 eval()에서 input으로 받는 cmdline을 파싱하고, 파싱된 명령어를 builtim_층(argv) 함수에 전달한다. 그후, 파싱한 명령어와 "quit"을 strcmp로 비교하고, strcmp가 같을 때 0을 반환하므로, !를 앞에 붙여 조건문을 작성한다.

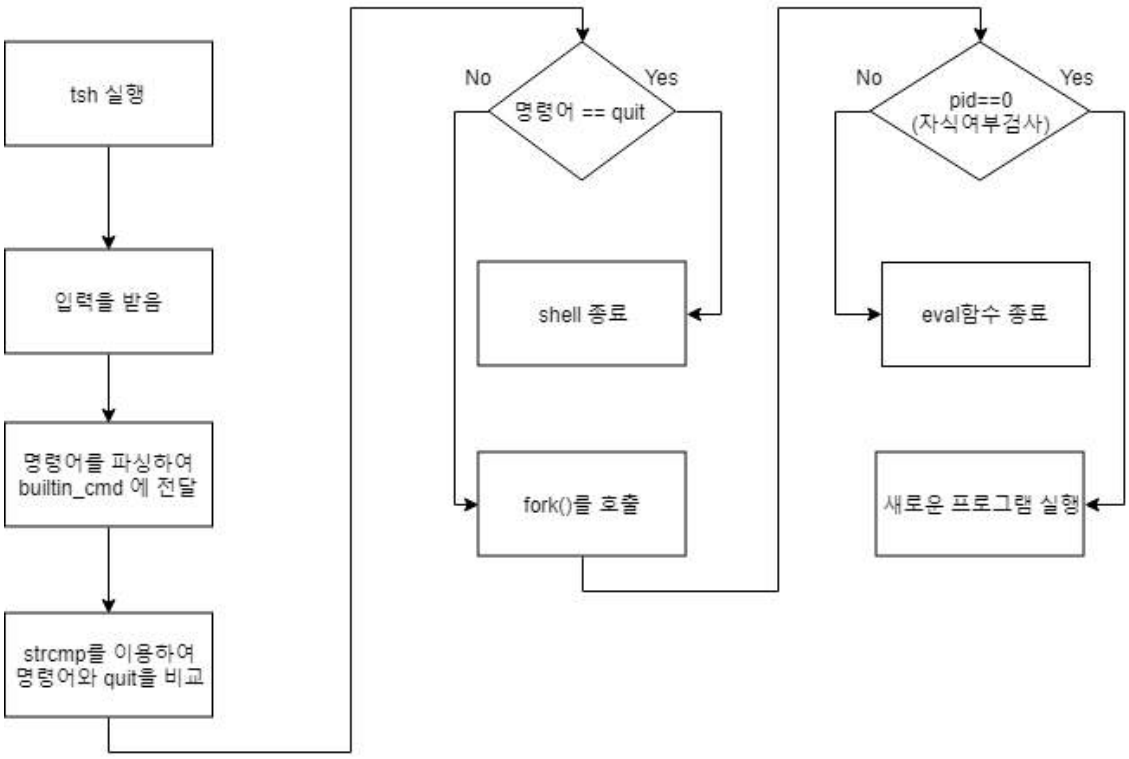
Trace 02

```
c201702012@2018-sp:~/shlab-handout$ ./sdriver -V -t 02 -s ./tsh
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/c201702012/bin:/home/sys02/c201702012/.local/bin:/usr/local/sbin:/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/c201702012/bin:/home/sys02/c201702012/.local/bin:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin
:/sbin:/bin:/usr/games:/usr/local/games

c201702012@2018-sp:~/shlab-handout$
```

trace 02 플로우 차트



trace 02 해결 방법 설명

```
170 void eval(char *cmdline) //     ////////////////////////////////////////////////////////eval/////////
171 {
172
173     char *argv[MAXARGS];    //  command 저장
174     pid_t pid;
175
176     // 명령어를 parseline을 통해 분리
177     parseline(cmdline, argv);
178     // parsing된 명령어를 전달
179     builtin_cmd(argv);
180
181     if(feof(stdin)){ /* trace00    End of file (ctrl - d) */
182         fflush(stdout);
183         exit(0);
184     }
185
186     if(!builtin_cmd(argv)){  /* trace02  */
187         pid = fork();
188         if( pid  == 0/* Child Process 체크  */ ){    //  //  Child Process 인 경우 , execve() 수행
189             if((execve(argv[0], argv, environ) < 0 )){
190                     printf("%s : command not found\n", argv);
191                     exit(0);
192             }
193         }
194     }
195
196
197     return;
198 }
```

Foreground 작업 형태로 프로그램을 실행하기 위한 작업을 한다.
테스트에서 ./myenv 를 실행하는 것을 알 수 있다.
주어진 코드를 참조하여 작성하였는데, Child Process인 경우는 fork()의 리턴이 0인 경우이다. 주어진 <174> pid_t pid; 를 활용하기 위하여 <187>에 주어진 pid에 fork()를 실행함과 동시에 리턴값을 저장하는 것을 볼 수 있다. 이때 fork()가 실행되며, 자식 프로세스를 생성한다. <188>의 if조건문을 통하여 pid==0인 경우 execve()를 수행하도록 코드를 작성하였다.