**Shell Lab Report**

20200927 장유진

1. eval 함수 알고리즘

eval(char \*cmdline)은 cmdline을 받아서 그에 맞는 함수를 실행한다. 먼저 cmdline을 buf에 임시로 받고 buf를 띄어쓰기를 기준으로 parse하여 argv에 저장한다. 이때 parseline 함수를 사용하고 이 함수는 해당 작업이 bg(1)인지 fg(0)인지 리턴한다. cmdline이 비었으면 오류 메시지를 출력한다.

1) command line에서 입력된 작업이 builtin command인 경우

Builtin\_cmd 함수 내에서 즉시 실행한다.

2) command line에서 입력된 작업이 어떤 파일의 주소인 경우

Parent process에서 signal을 block하고 parent process의 동작을 수행한다. Parent에서는 fg, bg인 경우를 각각 나누어 fg이면 해당하는 job을 addjob하고 signal unblock 후 child가 끝날 때까지 기다린다. bg이면 해당하는 job을 addjob하고 signal unblock 후 해당 cmdline를 출력하고 종료한다. 한편, child에서는 execve를 실행하기 전에 signal unblock을 하고 입력된 프로그램을 수행한다.

2. builtin\_cmd 함수

파라미터로 받은 argv에서 argv[0]가 빌트인 함수인 quit, fg, bg, jobs 중 하나인지 확인하고 해당하는 함수 exit(0), do\_bgfg(argv), do\_bgfg(argv), listjobs(jobs)를 실행한다. 이 함수는 입력된 명령이 빌트인 함수이면 1을 리턴하고 아니면 0을 리턴한다.

3. do\_bgfg 함수

Command line에서 job id를 입력하는 경우와 process id를 입력하는 경우를 나눠서 그에 해당하는 job entry를 불러온다. 명령어가 bg라면 job entry의 state를 BG로 설정하고 해당 process가 속한 process group의 모든 process에 SIGCONT 신호를 보낸다. 명령어가 fg이면 job entry의 state를 FG로 설정하고, state를 바꾼 해당 process를 wait한다.

1) job id를 받은 경우

%뒤에 온 숫자를 JID로 저장하고 입력된 JID가 오류가 없는지 체크한다. 이후 JID를 이용해서 해당하는 job entry를 job table에서 불러온다.

2) process id를 받은 경우

받은 pid를 변수 PID에 저장하고 그 값에 오류가 없는지 체크한다. 이후 PID를 이용해서 해당하는 job entry를 job table에서 불러온다.

4. waitfg 함수

파라미터로 받은 process id가 올바른 값인지 체크하고 해당 process가 끝날 때까지 while문을 반복한다.

5. sigchld\_handler 함수

SIGCHLD signal을 받은 경우 동작하는 함수로, parent 내의 모든 child가 종료할 때까지 child가 정상적으로 종료된 경우, 정지된 경우, 어떤 signal을 받아 종료된 경우에 따라 job을 delete해주거나 해당 job entry의 state를 바꿔준다.

6. sigint\_handler 함수

입력 받은 signal을 kill 함수로 pid가 속한 process group 내의 모든 process로 그 신호를 보낸다.

7. sigint\_handler 함수

입력 받은 signal을 kill 함수로 pid가 속한 process group 내의 모든 process로 그 신호를 보낸다.

8. 알게 된 점

Signal의 동작 원리를 이해하고 활용할 수 있다.

Fork 함수, parent와 child process 사이 관계를 알 수 있다.