2017029743 배병재
assignment3

1. builtin_cmd(char** argv) :

argv[0]값이 built_in 명령어인지 파악하는 함수이다.
quit일 경우 exit를 통해 나가게 하고,
jobs일 경우 listjobs를 통해 현재 jobs를 출력하게 해준다.

bg혹은 fg일 경우 do_bgfg함수를 호출 시켜 fg bg에 대한 명령어를 실행시키게 한다.
built-in 명령어에 경우는 1을 return 시키고, 아닌 경우 0을 return 시킨다.

2. void eval(char* cmdline) :

인자로 들어온 cmdline을 parseline함수로 분리해준다.
bg를 parseline의 리턴값으로 받아준다.
cmdline에서 &가 붙어있을 경우 bg 실행으로 판단한다.
이후 분리된 문자열을 builtin_cmd를 통해 built_in 명령어인지 체크한다.
built_in 명령어일 경우 그 명령어를 실행하고,

입력받은 명령어가 builtin 명령어가 아니라면 자식 프로세스를 하나 생성하고,
자식 프로세스에서 입력받은 명령어로 새로운 프로그램을 실행한다.
fork로 새로운 프로세스를 만들고,
fork의 리턴값이 0이면 자식프로세스의 실행으로 분기가 된다. 이때 프로세스 그룹 id가 호출하는 프로세스 pid로 하는 새 프로세스 그룹을 만든다, 프로세스 pid를 새 그룹에 추가한다. INT나 TSTP가 명령어로 주어질 때, kill함수가 프로세서 그룹에 전송하기 때문에 유니크한 pgid를 가지고 있는 자식들은 pgid를 설정하지 않을 경우 child 프로세서가 시그널을 받지 못하게 된다. setpgid(0,0)을 통해 프로세서 그룹 id를 설정해주고,
새로운 프로그램을 실행하는 방법으로 execve()를 사용한다.
execve()함수는 현재 프로그램의 컨텍스트 내에서 새로운 프로그램을 로드하고 실행한다.

fork는 2가지 값을 리턴하게되는데, 0을 리턴하는 경우는 child processor가 실행하는 구문이 되고, 0이 아닌 pid를 리턴할 경우, 그 pid는 자식 프로세서의 pid가 된다. 만약 0이 아닌 음수일 경우 foreground인지 background인지에 따라 나뉘게 된다. background는 명령어에 &가 붙을 경우 back ground로 처리해준다. background일 경우 job목록에 추가해주고,
job 내용을 출력해준다.
foreground인 경우, job목록에 추가해주고 waitfg를 통해 자식 프로세서의 종료까지 기달려준다. 즉, job이 없어지거나 stop상태가 될 때까지 대기해준다.

3. void waitfg(pid_t pid) :

wait_fg는 child프로세서가 foreground job으로 실행하게 될 경우 부모 프로세서가 자식 프로세서를 기다려주기 위해 실행하는 함수이다.
pid에 해당하는 job이 없거나, job에 상태가 stopped가 될 때까지 대기하게 만들어준다.