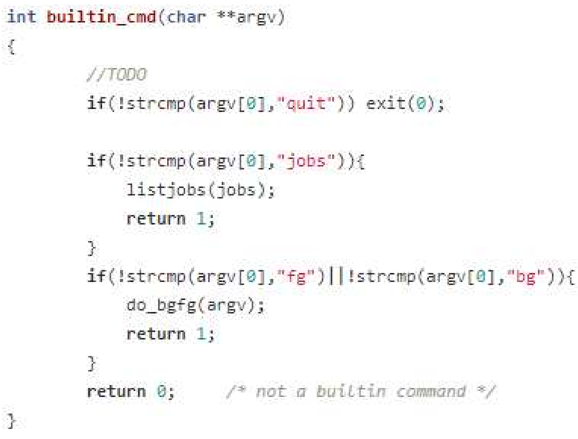
Report for Assignment 4   20131218 박기훈

1. builtin_cmd funciton

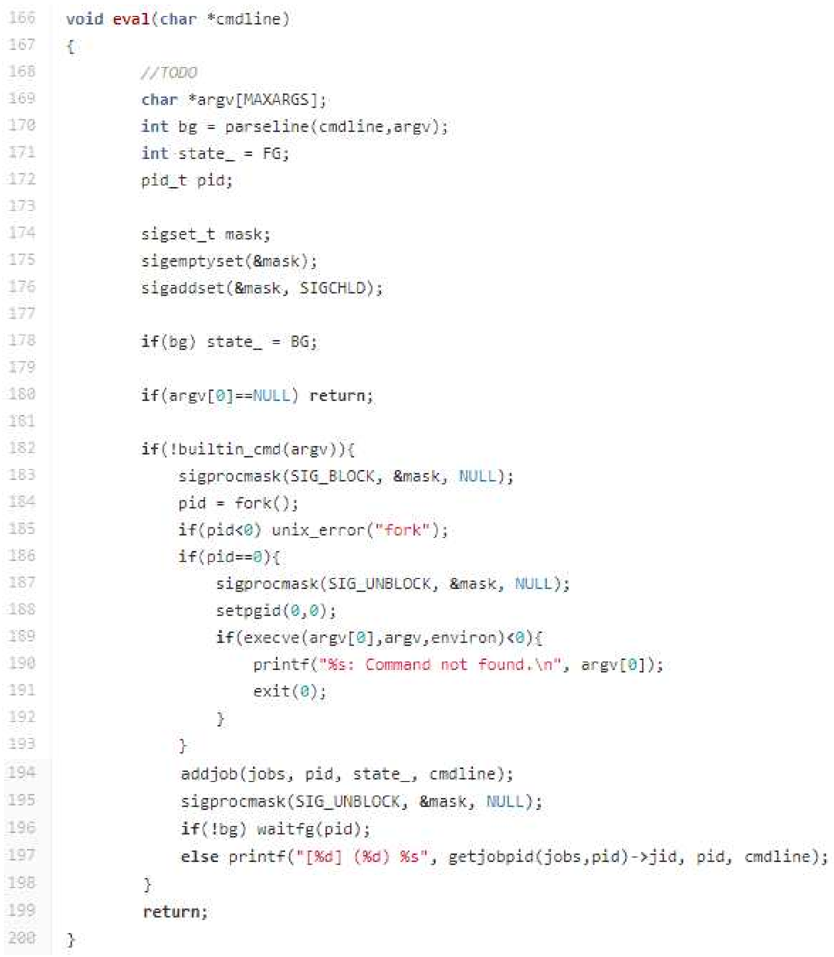


```c
int builtin_cmd(char **argv)
{
        //TODO
        if(!strcmp(argv[0],"quit")) exit(0);

        if(!strcmp(argv[0],"jobs")){
            listjobs(jobs);
            return 1;
        }
        if(!strcmp(argv[0],"fg")||!strcmp(argv[0],"bg")){
            do_bgfg(argv);
            return 1;
        }
        return 0;     /* not a builtin command */
}
```

First of all, I made the builtin_cmd function. In this function, it takes argv array which has command line. (Command line is divided and saved into argv array through parseline function.) Then by using strcmp, check whether argv[0] is built in command (quit, jobs, fg, bg). In case of 'jobs', by using listjobs function, I can enumerate the list of job. In case of 'fg' or 'bg', by using do_bgfg function, save the FG or BG into state of each job structure. And in case of these built in command, by returning 1 values, I can make the code to deal with case when command is not built in command in eval function.

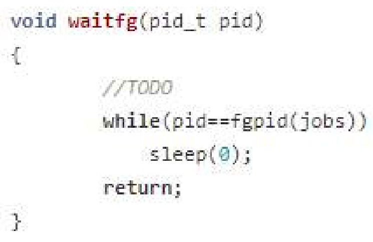
2. eval function, waitfg function



```c
166  void eval(char *cmdline)
167  {
168          //TODO
169          char *argv[MAXARGS];
170          int bg = parseline(cmdline,argv);
171          int state_ = FG;
172          pid_t pid;
173
174          sigset_t mask;
175          sigemptyset(&mask);
176          sigaddset(&mask, SIGCHLD);
177
178          if(bg) state_ = BG;
179
180          if(argv[0]==NULL) return;
181
182          if(!builtin_cmd(argv)){
183              sigprocmask(SIG_BLOCK, &mask, NULL);
184              pid = fork();
185              if(pid<0) unix_error("fork");
186              if(pid==0){
187                  sigprocmask(SIG_UNBLOCK, &mask, NULL);
188                  setpgid(0,0);
189                  if(execve(argv[0],argv,environ)<0){
190                      printf("%s: Command not found.\n", argv[0]);
191                      exit(0);
192                  }
193              }
194              addjob(jobs, pid, state_, cmdline);
195              sigprocmask(SIG_UNBLOCK, &mask, NULL);
196              if(!bg) waitfg(pid);
197              else printf("[%d] (%d) %s", getjobpid(jobs,pid)->jid, pid, cmdline);
198          }
199          return;
200  }
```

In main function, it takes command line and store at cmdline. Then, eval function takes *cmdline as parameter. First, declare *argv[MAXARGS] and by using parseline function, we divided cmdline and stored each command at the argv arrays. To make child process, declare pid_t pid; which is for running jobs in context of the child.
At line 174-176, 'sigset_t mask' is making the signal set, mask. 'sigemptyset(&mask) vacates the set(&mask), and sigaddset add SIGCHLD into set(&mask).
At line 182, if cmdline is built in command, execute it immediately. But if not, we execute the command by using child process.
At line 183, sigprocmask set the signal to waiting state. In first parameter, by using SIG_BLOCK, set signal to waiting state. And at line 187, with unblock, signal can be processed. At line 188, setpgid(0,0) is very important because each child process should have unique process group ID so that our background

children don't receive SIGINT when ctrl-c. At line 189, by using execve function, we can execute command which is not built in command. In execve parameter, argv[0] is path of command, environ is environment variable.



```
void waitfg(pid_t pid)
{
        //TODO
        while(pid==fgpid(jobs))
            sleep(0);
        return;
}
```

At line 194, I used addjob function, of which role is just adding the new jobs into job list. In this function parameter, state_ means that if command is background then 2(BG), if not 1(FG). At line 195, Unblock to process the signal. And at line 196-197, divides the case whether job is executed in foreground or background. If job is executed in foreground, only that job can be executed in foreground. So by using waitfg funciton, block until process pid is no longer the foreground process. If job is executed in background, print the messege like line 197.