<Operating System>

Assignment #1

Implement My Own Shell

“mysh-0”

소프트웨어학과

201321005

박주원

1. Work Description

이번 과제는 Implement my own shell 로 shell command parser를 직접 만들고, 3개의 명령어(pwd, cd, alias)를 내 shell에 built-in하는 것이다. pwd 명령어는 현재 자신의 디렉토리를 표시해주는 명령어이고, cd 명령어는 자신이 원하는 위치로 디렉토리를 변경해주는 명령어이다. alias는 특정 명령을 자신이 원하는 단어로 변경해주는 명령어다. 예를 들어 alias cd_to_etc="cd /etc"의 명령 경우 "cd /etc" 명령을 "cd_to_etc"로 사용할 수 있도록 하는 것이다.

1) utils.c "mysh_parse_command()"

과제를 수행하기 위해서 첫 번째로 input string을 parse하여 Command unit에 넣어야 했다. strtok() 함수를 사용하여 스페이스 단위로 tokenizing 하였다. 처음에 pwd나 cd command 같은 경우에는 맞게 parse 되었지만, alias 같은 경우엔 "pwd" 명령어를 alias할 땐 알맞지만 "cd /etc" 같은 cd 명령어를 alias할 때는 argc값이 3으로 나오게 되므로 alias 명령어의 경우에 조건문을 사용하여 맨 처음의 스페이스만을 tokenized했다.

2) commands.c "do_pwd()", "validate_pwd_argv()"

"pwd" 명령어를 built-in하기 위해 commands.c 파일의 validate_pwd_argv() 함수와 do_pwd() 함수를 작성해야했다. do_pwd() 함수의 skeleton code를 보면 validate_pwd_argv() 함수의 return 값이 1이어야만 pwd 명령어가 실행되야 한다는 것을 볼 수 있었다. 따라서 validate_pwd_argv() 함수에서 argc 값이 1이고 argv[0]의 값이 pwd 일 경우에만 1을 return하도록 하였다. do_pwd() 함수에서는 pwd의 기능을 구현하였는데, 이미 getcwd()라는 함수가 있어서 단지 getcwd()를 통해 현재 내가 작업하고 있는 디렉토리를 받아올 수 있었다.

3) commands.c "do_cd()", "validate_cd_argv()"

"cd" 명령어를 built-in하기 위해 commands.c 파일의 validate_cd_argv() 함수와 do_cd() 함수를 작성해야 했다. do_cd()의 경우에도 validate_cd_argv() 함수를 통해 input string 값을 validate 하는 과정을 거쳐야했다. 따라서 validate_cd_argv() 함수에서 argc 값이 2이고 argv[0]의 값이 cd일 경우에 1을 return 하도록 구현하였다. cd 명령어의 경우엔 "cd ..", "cd /etc"와 같이 상위 디렉토리나 원하는 위치의 디렉토리로 설정하는 명령어이므로 cd 뒤에 목적하는 바가 있어야한다. 그래서 argc 값이 2이고 이번과제에서 가장 상위의 디렉토리로 설정하는 "cd"의 경우엔 고려하지 않아도 됐기 때문에 쉽게 구현할 수 있었다. pwd의 getcwd() 함수처럼 cd 명령어도 디렉토리를 변경해주는 chdir()이라는 이미 만들어진 함수를 통해 구현하였다.

4) main.c "do_alias()", "validate_alias_argv()"

"alias" 명령어를 built-in하기 위해 main.c 파일의 validate_alias_argv() 함수와 do_alias() 함수를 작성해야 했다. "alias"도 마찬가지로 validate 함수를 통해 input string

을 validate 해야 했다. alias의 경우 argc값이 2여야만 했기 때문에 argc가 2이고 argv[0]값이 alias인 경우에만 1을 return하도록 하였다. do_alias()에서 temp라는 변수에 parse해온 argv[1]의 값을 strtok() 함수를 통해 "="으로 tokenized했다. 이렇게 alias할 명령을 strcpy()를 통해 alias 변수에 삽입하고, "\""을 tokenized 하여 ""안에 있는 내가 만든 명령을 command 변수에 삽입하도록 하였다.

2. Lessons

우선 이번 과제를 수행하기 위해서는 git을 기본적으로 사용할 줄 알아야했다. 평소에 git을 써본 적이 없어서 많이 당황했지만, 조교님이 youtube Live를 통해 필요한 부분을 자세히 설명해주셔서 무리없이 사용할 수 있었다. 이번 기회에 fork를 하고 Ubuntu에 clone을 하고 다시 git에 commit, push하는 명령들을 배우게 되면서 git에 대한 기본적인 사용방법을 배우게 되었다. 이 과정을 통해 git의 편리함과 효율성을 느끼게 되었다.

나만의 shell을 구현하면서, 평소에 linux환경에서 자주 사용했던 "cd"와 "pwd" 명령어를 직접 구현하게 되어 매우 흥미롭고 신기하였다. "alias" 명령어 같은 경우 이번 과제를 통해 처음 알게 되었다. 실제로 사용하면 매우 편리하게 사용 가능할 것 같았다. C programming을 어느 정도 해왔기 때문에 문자열 함수들을 사용하여 편리하게 구현할 수 있었던 것 같다.

과제를 하면서 어려움을 겪은 부분은 memory allocation 부분 이었다. 처음엔 input string을 parse하여 argv[]에 값을 저장할 때 argv[] 변수의 memory 할당을 적게하여 alias 명령에서 argv[1]값이 짤려서 저장되었다. 이 부분에서 입력 받은 만큼의 memory를 할당하고 싶었으나, skeleton code를 바탕으로 구현해야 했기 때문에 어느 정도 무리가 있었다. 그래서 어느 정도 충분한 memory를 할당하고 과제를 진행했다.

과제를 진행하면서 C programming에 좀 더 전문성을 가지게 된 것 같고, system software의 기초 단계를 잘 이해할 수 있었다.

3. Feedback

과제의 난이도는 적절했던 것 같았다. Skeleton code를 바탕으로 구현하는 것이었기 때문에, 큰 어려움은 없었던 것 같다. 하지만 C programming에 익숙하지 않거나 코드 이해가 어려운 사람들에겐 매우 어려운 과제였을 것 같다. git에 대한 사용법을 Youtube Live에서 자세히 설명해 주셨듯이 Skeleton code도 주석으로만이 아닌 Youtube Live를 통해 직접 설명해 주셨으면, google group에 많은 질문이 나오진 않았을 것 같다. Youtube Live를 통해 과제에 대한 기초 지식 설명과 Group을 통해 질문을 주고 받는 방식은 정말 효율적이어서 도움이 많이 되었다. 질문에 대해 조교님이 빠르게 답변해주시고 자세히 설명해주셔서 무리없이 잘 진행할 수 있었던 것 같다.