【2017 Network System Programming Homework 1】

This homework focuses on making simple shell parser.

◆ Rules:

1. Please use **C language** in this homework and run your program on **Ubuntu 16.04**.
2. Please provide **Makefile** to compile your homework; otherwise, you will get **ZERO**.
3. **Do not copy others homework.**
4. If you have any question, please send email to sp_ta@net.nsysu.edu.tw or come to F5018, but TA does not help to debug.

◆ Upload:

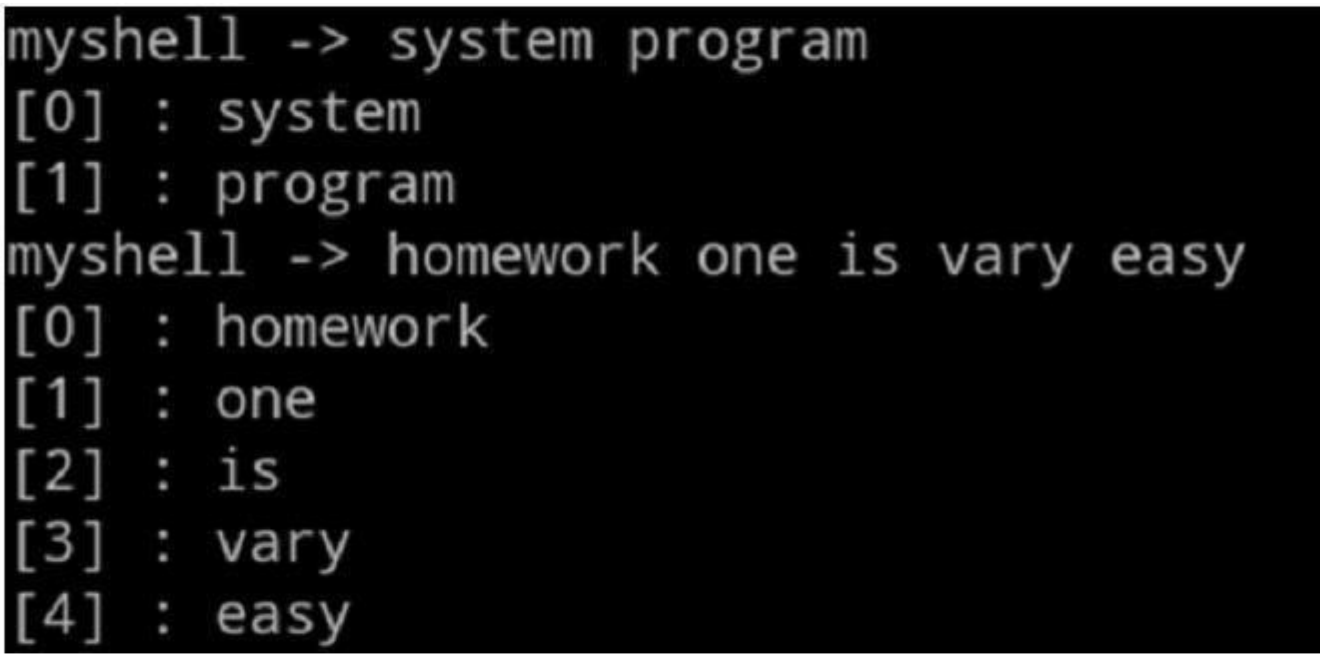
1. Please compress your homework into **zip** or **tar** archive.
2. Naming rules: "**StudentID_SP_HW1.zip**". For example: M043040001_SP_HW1.zip
3. Upload your homework to **NSYSU Cyber University**.
4. **Deadline: 2017/10/10 (Tue.) 23:59; if it is overdue, you will also get ZERO.**

◆ Part 1:

1. Edit the **parse.c** file to use **strtok()** and **realloc( )** to implement the **parse()** and **free_argv()** functions.

2. Files provided:
   shell.h
   shell.c
   parse.c

3. Example:



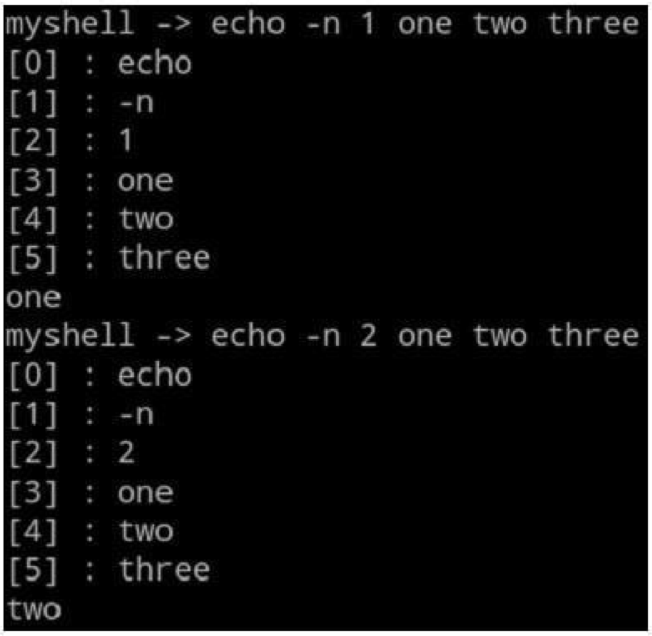
```
myshell -> system program
[0] : system
[1] : program
myshell -> homework one is vary easy
[0] : homework
[1] : one
[2] : is
[3] : vary
[4] : easy
```

◆ Part 2:

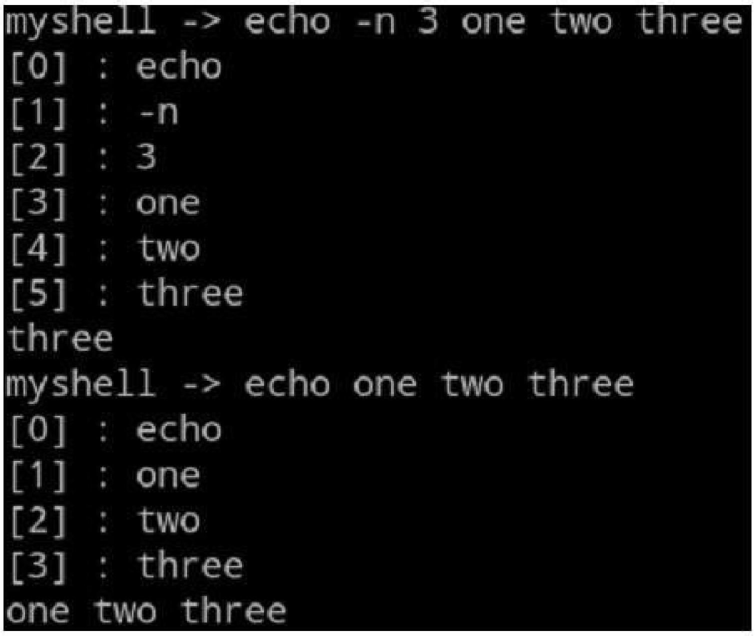
1. Add code to the **builtin.c** stub to recognize the **echo, quit, exit, logout and bye** commands. Write functions implementing these commands, and add a new line for each command to table inbuilts[] just above the line {NULL, NULL}.

2. File provided:
   Builtin.c

3. echo example:
   echo print all strings
   echo -n N: print the specified string

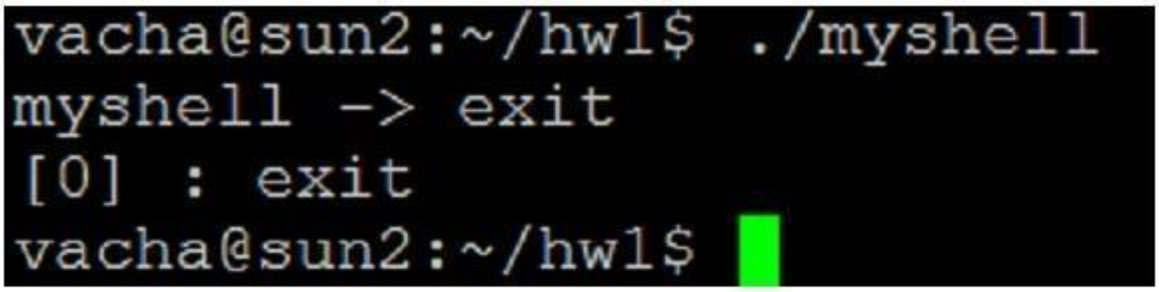


```
myshell -> echo -n 1 one two three
[0] : echo
[1] : -n
[2] : 1
[3] : one
[4] : two
[5] : three
one
myshell -> echo -n 2 one two three
[0] : echo
[1] : -n
[2] : 2
[3] : one
[4] : two
[5] : three
two
```



```
myshell -> echo -n 3 one two three
[0] : echo
[1] : -n
[2] : 3
[3] : one
[4] : two
[5] : three
three
myshell -> echo one two three
[0] : echo
[1] : one
[2] : two
[3] : three
one two three
```

4. quit example:
   exit, quit, logout and bye terminate the program.

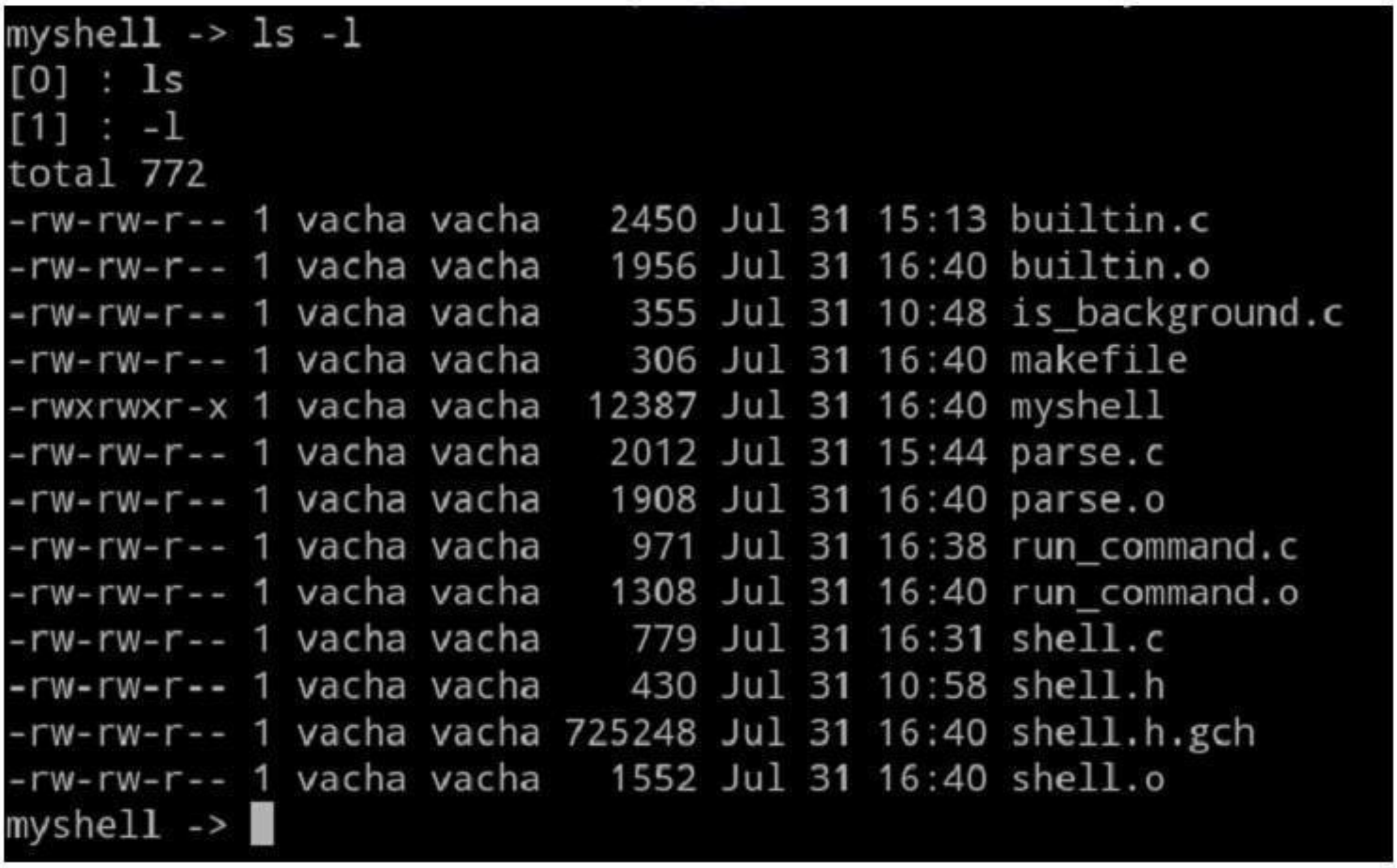


```
vacha@sun2:~/hw1$ ./myshell
myshell -> exit
[0] : exit
vacha@sun2:~/hw1$
```

◆ Part 3:

1. Edit the **run_command.c** file so a child process is created to run the command, and the parent waits for the child process to terminate. Check for builtin commands first, create a new process only for commands which are not built in. Use the parser from pervious labs to create from the command line the argv array passed to the child.

   Hint: You can use **exec()/execvp()** function, **waitpid()** function and the **fork()** system call.
2. File provided:
   run_command.c
3. Example:



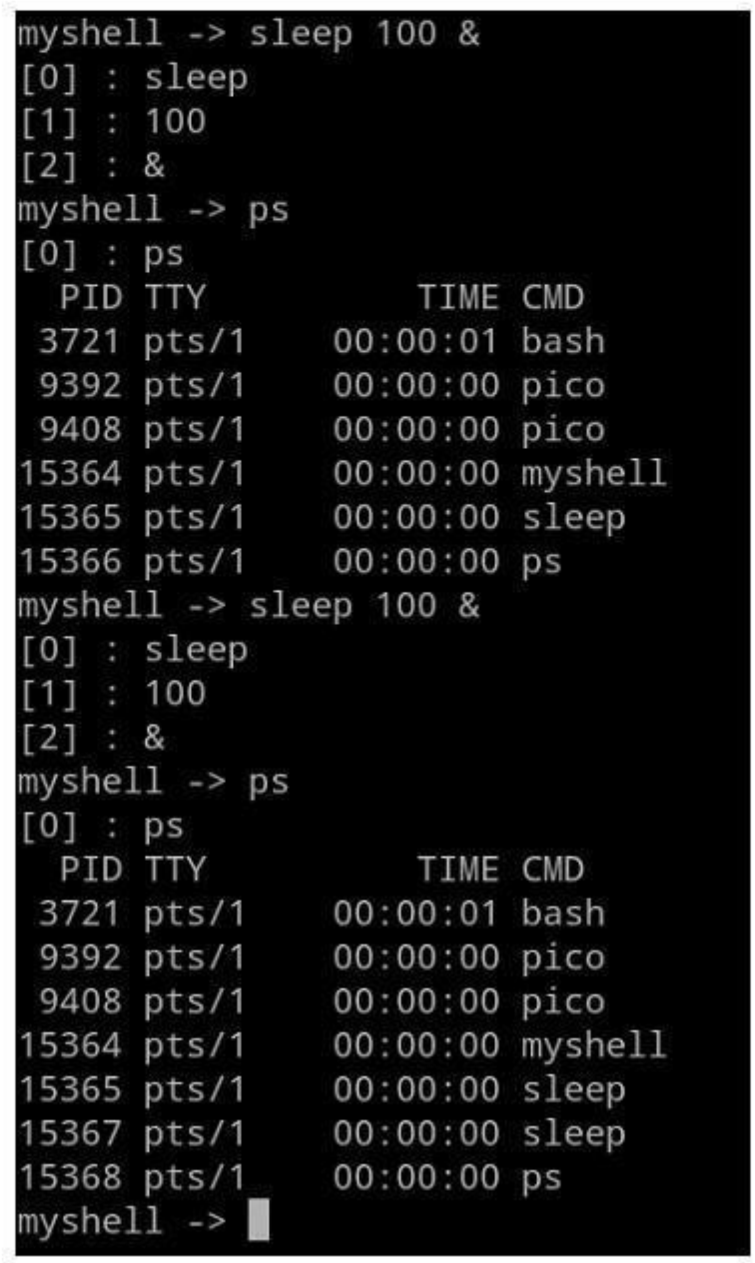
```
myshell -> ls -l
[0] : ls
[1] : -l
total 772
-rw-rw-r-- 1 vacha vacha   2450 Jul 31 15:13 builtin.c
-rw-rw-r-- 1 vacha vacha   1956 Jul 31 16:40 builtin.o
-rw-rw-r-- 1 vacha vacha    355 Jul 31 10:48 is_background.c
-rw-rw-r-- 1 vacha vacha    306 Jul 31 16:40 makefile
-rwxrwxr-x 1 vacha vacha  12387 Jul 31 16:40 myshell
-rw-rw-r-- 1 vacha vacha   2012 Jul 31 15:44 parse.c
-rw-rw-r-- 1 vacha vacha   1908 Jul 31 16:40 parse.o
-rw-rw-r-- 1 vacha vacha    971 Jul 31 16:38 run_command.c
-rw-rw-r-- 1 vacha vacha   1308 Jul 31 16:40 run_command.o
-rw-rw-r-- 1 vacha vacha    779 Jul 31 16:31 shell.c
-rw-rw-r-- 1 vacha vacha    430 Jul 31 10:58 shell.h
-rw-rw-r-- 1 vacha vacha 725248 Jul 31 16:40 shell.h.gch
-rw-rw-r-- 1 vacha vacha   1552 Jul 31 16:40 shell.o
myshell -> 
```

◆ Part 4:

1. Edit the **is_background.c** file to detect an **"&"**. Alter the **run_command.c** file so that if a task is running in the background, the parent does not wait. Do not worry about background processes becoming zombies at this point; this will be addressed later.

2. File provided:
   is_background.c

3. Example:



```
myshell -> sleep 100 &
[0] : sleep
[1] : 100
[2] : &
myshell -> ps
[0] : ps
  PID TTY          TIME CMD
 3721 pts/1    00:00:01 bash
 9392 pts/1    00:00:00 pico
 9408 pts/1    00:00:00 pico
15364 pts/1    00:00:00 myshell
15365 pts/1    00:00:00 sleep
15366 pts/1    00:00:00 ps
myshell -> sleep 100 &
[0] : sleep
[1] : 100
[2] : &
myshell -> ps
[0] : ps
  PID TTY          TIME CMD
 3721 pts/1    00:00:01 bash
 9392 pts/1    00:00:00 pico
 9408 pts/1    00:00:00 pico
15364 pts/1    00:00:00 myshell
15365 pts/1    00:00:00 sleep
15367 pts/1    00:00:00 sleep
15368 pts/1    00:00:00 ps
myshell -> 
```