

**ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN
KHOA CÔNG NGHỆ THÔNG TIN**



**BÁO CÁO ĐỒ ÁN: SIMPLE SHELL
MÔN HỌC: HỆ ĐIỀU HÀNH**

LỚP : 18CNTN
SINH VIÊN THỰC HIỆN : 18120019 – Nguyễn Hoàng Dũng
18120052 – Lê Hạnh Linh

Thành phố Hồ Chí Minh, ngày 5 tháng 11 năm 2020

Mục Lục

| | | |
|-------------|--|---|
| I. | BIẾN TOÀN CỤC | 2 |
| a. | int pipes[]: | 2 |
| b. | int user_read, user_write: | 2 |
| II. | ĐỊNH NGHĨA HÀM | 2 |
| a. | void io_redirect(char sign, char *file) | 2 |
| b. | void parse_iostream(char **argv, bool *background) | 2 |
| c. | void parse_command(char *args, char **argv) | 3 |
| d. | pid_t execute_command(char **argv, int pipe_read, int pipe_write) | 3 |
| e. | void count_command(char *str, char **cmd) | 3 |
| f. | void pipe_command(int index, int *pipe_read, int *pipe_write) | 4 |
| g. | bool is_internal_command(char **argv) | 4 |
| h. | void main_process(char *line) | 4 |
| i. | bool check_last_command(char *line, char *hist) | 5 |
| j. | void open_pipes() | 5 |
| III. | PHÂN CÔNG CÀI ĐẶT | 5 |
| IV. | DEMO | 6 |

I. BIẾN TOÀN CỤC

a. `int pipes[]`:

- Mảng chứa lần lượt các giá trị read end và write end của các pipe.
- Mỗi pipe chiếm 2 vị trí liên tiếp: chẵn là read end, lẻ là write end.

```
cmd0    cmd1    cmd2    cmd3    cmd4
  pipe0    pipe1    pipe2    pipe3
  [0,1]    [2,3]    [4,5]    [6,7]
```

b. `int user_read, user_write`:

- Lưu giá trị file descriptor để read và write với user.
- Gọi hàm `dup()` để lấy giá trị mặc định từ STDIN/OUT.

II. ĐỊNH NGHĨA HÀM

a. `void io_redirect(char sign, char *file)`

Input:

- `sign`: kí tự < hoặc >
- `file`: chuỗi kí tự chứa tên file

Xử lý:

- Nếu `sign` là < thì mở file theo chế độ đọc và gán cho `user_read`.
- Nếu `sign` là > thì mở file theo chế độ ghi và gán cho `user_write`.

b. `void parse_iostream(char **argv, bool *background)`

Input:

- `argv`: biến trỏ đến danh sách các con trỏ quản lý argument)
- `background`: flag chứa tín hiệu đánh dấu parent chờ child process

Xử lý:

- Duyệt `argv` nếu có các kí tự <, > thì gọi hàm `io_redirect`.
- Nếu có & thì gán `background = true`
⇒ Báo hiệu parent chờ child process kết thúc

c. void parse_command(char *args, char **argv)

Input:

- args: chuỗi kí tự chứa 1 command user nhập vào
- argv: biến để chứa danh sách các con trỏ quản lý argument

Xử lý:

- Từ args, tách thành các argument và lưu vào argv

d. pid_t execute_command(char **argv, int pipe_read, int pipe_write)

Input:

- argv: biến để chứa danh sách các con trỏ quản lý argument
- pipe_read: read end của pipe kế trước để child process đọc input
- pipe_write: write end của pipe kế sau để child process ghi output

Xử lý:

1. Gọi hàm fork() khởi tạo child process.
2. Nếu là child process:
 - Gọi dup2() để đọc ghi từ pipe_read và pipe_write.
 - Gọi hàm execvp() và truyền argv để thực hiện command.
3. Nếu là parent process:
 - Close file descriptor (pipe_read, pipe_write)
 - Chờ child process nếu có background = true

Output:

- Child process ID

e. void count_command(char *str, char **cmd)

Input:

- str: chứa chuỗi command user nhập (có thể có pipe)
- cmd: biến để chứa danh sách các con trỏ quản lý command.

Xử lý:

- Đếm số lượng command.
- Lưu con trỏ đến các command này vào 1 mảng con trỏ do cmd quản lý.

f. void pipe_command(int index, int *pipe_read, int *pipe_write)

Input:

- index: thứ tự của command hiện tại
- pipe_read: read end của pipe kế trước để child process đọc input
- pipe_write: pipe_write: write end của pipe tương ứng để child process ghi output

Xử lý:

- Nếu là command đầu: pipe_read = user_read
- Nếu là command cuối: pipe_write = user_write
- Command thứ i sẽ:

Đọc input (pipe_read) từ pipes[i * 2 - 2]

Ghi output (pipe_write) ra pipes[i * 2 + 1]

```
cmd0    cmd1    cmd2    cmd3    cmd4
pipe0    pipe1    pipe2    pipe3
[0,1]    [2,3]    [4,5]    [6,7]
```

g. bool is_internal_command(char **argv)

Input:

- argv: biến để chứa danh sách các con trỏ quản lý argument

Xử lý:

- Xét xem argv lưu command của shell (cd, exit, pwd) hay không.

Output:

- True: command đó là internal
- False: không phải là internal

h. void main_process(char *line)

Input:

- line: chuỗi kí tự user nhập

Xử lý:

- Gọi các hàm trên để xử lý lần lượt các command.

i. bool check_last_command(char *line, char *hist)

Input:

- line: chuỗi kí tự user nhập
- hist: chuỗi kí tự trước đó user đã nhập

Xử lý:

- Xét input của user là command hay là yêu cầu thực hiện command gần nhất trong history (!!).
- Nếu là command thì copy vào buffer hist. Ngược lại thực hiện command trong buffer hist.

Output:

- True: Nếu command là !! và trước đó có command, hoặc command khác !!
- False: Nếu command là !! và trước đó không có command

j. void open_pipes()

Global:

- pipes[]: mảng chứa lần lượt các giá trị read end và write end của các pipe

Xử lý:

- Gọi hàm pipe() tại các vị trí pipes + i * 2

III. PHÂN CÔNG CÀI ĐẶT

| Hàm | Người phụ trách | Mức độ hoàn thành |
|---------------------|-------------------|-------------------|
| io_redirect | Lê Hạnh Linh | 100% |
| parse_iostream | Nguyễn Hoàng Dũng | 100% |
| parse_command | Nguyễn Hoàng Dũng | 100% |
| execute_command | Nguyễn Hoàng Dũng | 100% |
| count_command | Lê Hạnh Linh | 100% |
| pipe_command | Lê Hạnh Linh | 100% |
| main_process | Lê Hạnh Linh | 100% |
| is_internal_command | Nguyễn Hoàng Dũng | 100% |

IV. DEMO

1. Change dir to project directory

```
File  Actions  Edit  View  Help
nhdvn@kali:~$ cd Desktop/sshell/
nhdvn@kali:~/Desktop/sshell$
```

2. Compile shell.c into executable binary file

```
File  Actions  Edit  View  Help
nhdvn@kali:~/Desktop/sshell$ make shell
cc      shell.c      -o shell
nhdvn@kali:~/Desktop/sshell$
```

3. Run the binary

```
File  Actions  Edit  View  Help
nhdvn@kali:~/Desktop/sshell$ make shell
cc      shell.c      -o shell
nhdvn@kali:~/Desktop/sshell$ ./shell
ssh →
```

4. Simple command with child process

Cách kiểm tra:

- In ra pid của child process sau khi gọi hàm fork()
- Gọi command với lệnh ps

Giải thích:

Từ parent, ta fork child process để thực hiện lệnh ps và in ra pid của child process. Mà bản thân lệnh ps in ra các process status đang chạy.

```
ssh → cat in
Child PID: 2176

optimus
elgamal
omicron
unicron
epsilon
plutoni
entropy

ssh → ps
Child PID: 2177

  PID TTY          TIME CMD
 1684 pts/1        00:00:00 bash
 2167 pts/1        00:00:00 shell
 2177 pts/1        00:00:00 ps

ssh →
```

5. Simple command with &

Cách kiểm tra:

- Gọi lệnh ping để và thêm & để cho child process chạy background.
- Gõ tiếp lệnh ps sẽ thấy được thực hiện ngay, trong khi đó lệnh ping vẫn tiếp tục in ra màn hình.
- Lệnh ps in đồng thời cả pid của child process thực hiện lệnh ping và child process thực hiện lệnh ps.

```
ssh → ping -c 5 google.com &
Child PID: 2205

ssh → PING google.com (216.58.200.14) 56(84) bytes of data.
64 bytes from hkg12s11-in-f14.1e100.net (216.58.200.14): icmp_seq=1 ttl=56 time=29.1 ms
ps
Child PID: 2206

  PID TTY          TIME CMD
 1684 pts/1        00:00:00 bash
 2167 pts/1        00:00:00 shell
 2205 pts/1        00:00:00 ping
 2206 pts/1        00:00:00 ps

ssh → 64 bytes from hkg12s11-in-f14.1e100.net (216.58.200.14): icmp_seq=2 ttl=56 time=29.3 ms
64 bytes from hkg12s11-in-f14.1e100.net (216.58.200.14): icmp_seq=3 ttl=56 time=29.4 ms
64 bytes from hkg12s11-in-f14.1e100.net (216.58.200.14): icmp_seq=4 ttl=56 time=28.5 ms
64 bytes from hkg12s11-in-f14.1e100.net (216.58.200.14): icmp_seq=5 ttl=56 time=28.6 ms

--- google.com ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4004ms
rtt min/avg/max/mdev = 28.472/28.963/29.351/0.366 ms
```

Giải thích:

- Nếu không có & thì lệnh ps phải đợi lệnh ping thực hiện hết trước.
- Lúc này, lệnh ps được thực hiện thì không còn thấy lệnh ping trong process status nữa.

```
ssh → ping -c 5 google.com
Child PID: 2203

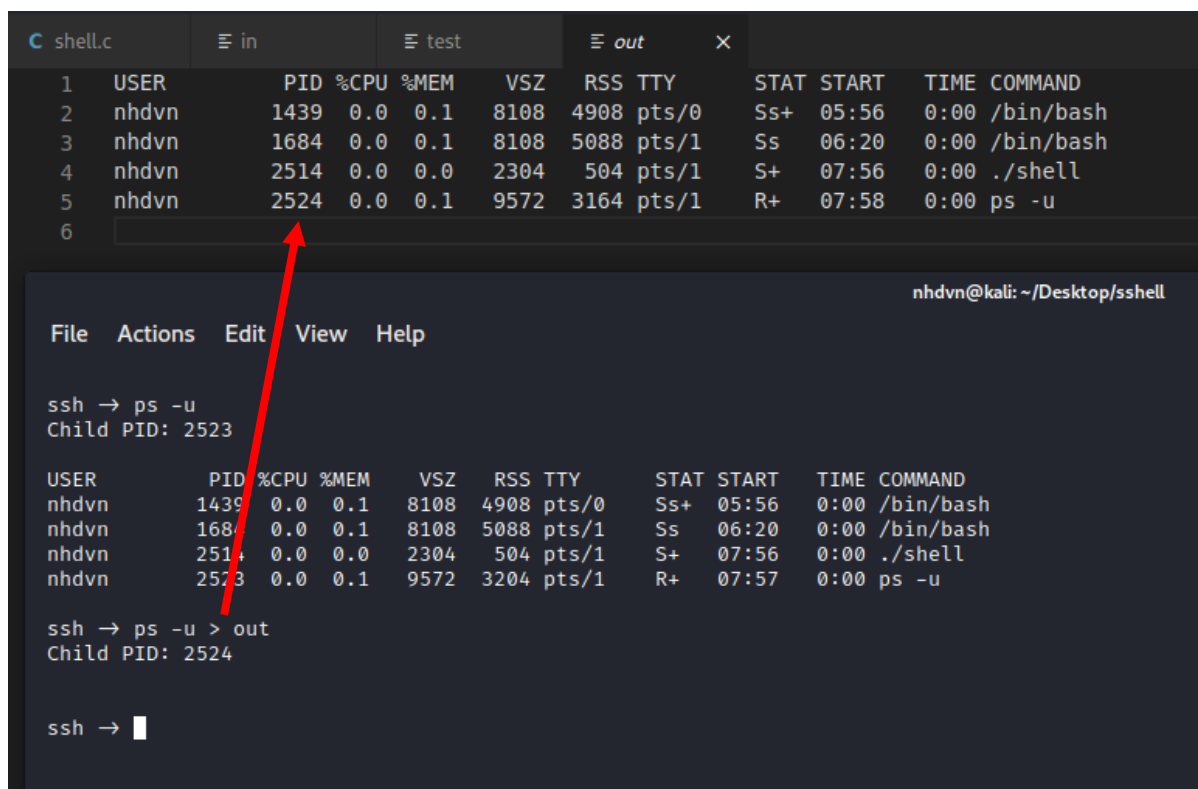
PING google.com (216.58.220.206) 56(84) bytes of data.
64 bytes from del01s08-in-f206.1e100.net (216.58.220.206): icmp_seq=1 ttl=56 time=31.8 ms
64 bytes from del01s08-in-f206.1e100.net (216.58.220.206): icmp_seq=2 ttl=56 time=28.4 ms
64 bytes from del01s08-in-f206.1e100.net (216.58.220.206): icmp_seq=3 ttl=56 time=28.9 ms
64 bytes from del01s08-in-f206.1e100.net (216.58.220.206): icmp_seq=4 ttl=56 time=29.3 ms
64 bytes from del01s08-in-f206.1e100.net (216.58.220.206): icmp_seq=5 ttl=56 time=29.2 ms

--- google.com ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4011ms
rtt min/avg/max/mdev = 28.446/29.537/31.827/1.185 ms

ssh → Child PID: 2204

  PID TTY          TIME CMD
 1684 pts/1        00:00:00 bash
 2167 pts/1        00:00:00 shell
 2204 pts/1        00:00:00 ps
```


6. Ouput redirection



The screenshot shows a terminal window with a menu bar (File, Actions, Edit, View, Help) and a title bar (shell.c, in, test, out, X). The terminal displays the following commands and outputs:

```
ssh → ps -u
Child PID: 2523
```

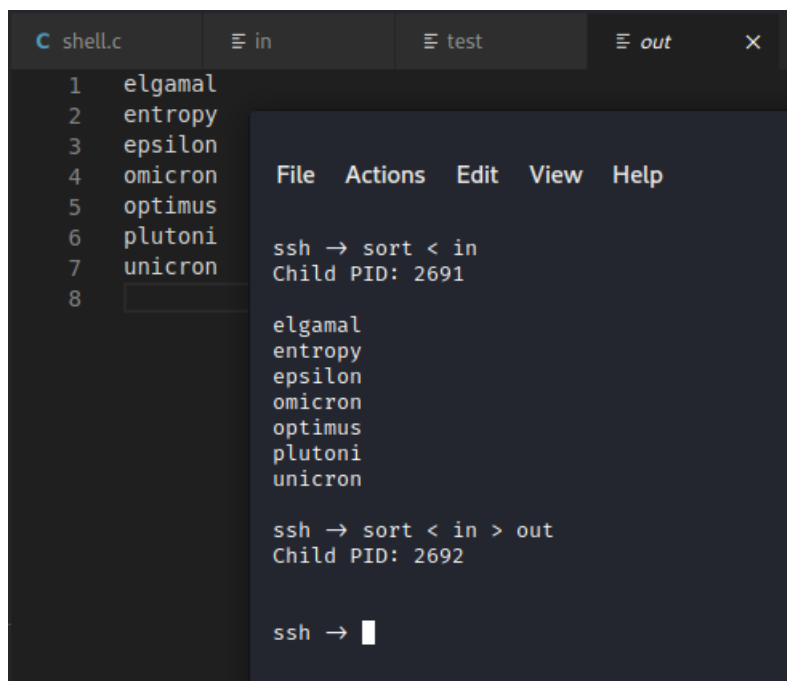
| USER | PID | %CPU | %MEM | VSZ | RSS | TTY | STAT | START | TIME | COMMAND |
|-------|------|------|------|------|------|-------|------|-------|------|-----------|
| nhdvn | 1439 | 0.0 | 0.1 | 8108 | 4908 | pts/0 | Ss+ | 05:56 | 0:00 | /bin/bash |
| nhdvn | 1684 | 0.0 | 0.1 | 8108 | 5088 | pts/1 | Ss | 06:20 | 0:00 | /bin/bash |
| nhdvn | 2514 | 0.0 | 0.0 | 2304 | 504 | pts/1 | S+ | 07:56 | 0:00 | ./shell |
| nhdvn | 2524 | 0.0 | 0.1 | 9572 | 3164 | pts/1 | R+ | 07:58 | 0:00 | ps -u |

```
ssh → ps -u > out
Child PID: 2524
```

```
ssh →
```

A red arrow points from the 'ps -u' command output to the 'ps -u > out' command.

7. Input redirection



The screenshot shows a terminal window with a menu bar (File, Actions, Edit, View, Help) and a title bar (shell.c, in, test, out, X). The terminal displays the following commands and outputs:

```
ssh → sort < in
Child PID: 2691
```

```
elgamal
entropy
epsilon
omicron
optimus
plutoni
unicron
```

```
ssh → sort < in > out
Child PID: 2692
```

```
ssh →
```

8. Pipe multiple command

```
File  Actions  Edit  View  Help

ssh → cat in | sort
Child PID: 2697

Child PID: 2698

elgamal
entropy
epsilon
omicron
optimus
plutoni
unicron

ssh →
```

```
ssh → ping -c 5 google.com | grep icmp | sort -k 8
Child PID: 2729

Child PID: 2730

Child PID: 2731

64 bytes from hkg07s28-in-f14.1e100.net (172.217.31.238): icmp_seq=5 ttl=115 time=30.1 ms
64 bytes from hkg07s28-in-f14.1e100.net (172.217.31.238): icmp_seq=4 ttl=115 time=30.4 ms
64 bytes from hkg07s28-in-f14.1e100.net (172.217.31.238): icmp_seq=3 ttl=115 time=30.8 ms
64 bytes from hkg07s28-in-f14.1e100.net (172.217.31.238): icmp_seq=1 ttl=115 time=31.8 ms
64 bytes from hkg07s28-in-f14.1e100.net (172.217.31.238): icmp_seq=2 ttl=115 time=32.4 ms

ssh →
```

```
C shellc  in  test  out  X

1  64 bytes from hkg07s28-in-f14.1e100.net (172.217.31.238): icmp_seq=2 ttl=115 time=30.1 ms
2  64 bytes from hkg07s28-in-f14.1e100.net (172.217.31.238): icmp_seq=5 ttl=115 time=30.1 ms
3  64 bytes from hkg07s28-in-f14.1e100.net (172.217.31.238): icmp_seq=3 ttl=115 time=30.6 ms
4  64 bytes from hkg07s28-in-f14.1e100.net (172.217.31.238): icmp_seq=4 ttl=115 time=30.7 ms
5  64 bytes from hkg07s28-in-f14.1e100.net (172.217.31.238): icmp_seq=1 ttl=115 time=32.4 ms
6

nhdvn@kali: ~/Desktop/sshell

File  Actions  Edit  View  Help

nhdvn@kali:~/Desktop/sshell$ ./shell
ssh → ping -c 5 google.com | grep icmp | sort -k 8 > out
Child PID: 2751

Child PID: 2752

Child PID: 2753

ssh →
```