Name: Phùng Lê Phi

ID: 24521324

Class: IT007.Q15.2

OPERATING SYSTEM
LAB 6'S REPORT

SUMMARY

Self-scrores: 100%

Note: Export file to **PDF and name the file by following format:*
Student ID_LABx.pdf
Đây là bài lab cuối của môn hệ điều hành.
Em xin chúc thầy thật nhiều sức khỏe và có nhiều thành công hơn trong công việc và cuộc sống.

Section 6.4

1. Thực thi lệnh trong tiến trình con

1. Thực thi lệnh trong tiến trình con

- Ví dụ: khi thực hiện

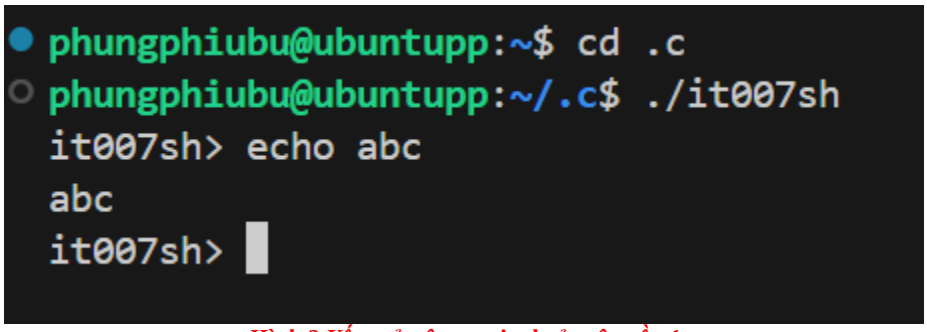
```
it007sh> echo abc
```

Kết quả sẽ in ra chuỗi abc, kết thúc dòng lệnh sẽ hiển thị dấu nhắc it007sh> để người dùng nhập lệnh tiếp theo. Lưu ý rằng trong khi lệnh `echo abc` đang thực thi, không cho người dùng nhập command mới.

- Gợi ý: Xem hình 1.

Hình 1.Yêu cầu chi tiết của phần 1

Kết quả:



Hình 2.Kết quả trên terminal của yêu cầu 1

→ Qua đó kết quả sẽ in ra chuỗi abc, kết thúc dòng lệnh sẽ hiển thị dấu nhắc it007sh> để người dùng nhập lệnh tiếp theo.

Phân tích :

```
child_pid = fork();
if (child_pid == 0) {
    signal(SIGINT, SIG_DFL);
    execvp(args[0], args);
```

```
    exit(1);
}
```

Bảng 1. Các thành phần trong code liên quan tới yêu cầu 1

- fork() tạo tiến trình con.
- child_pid == 0 → đoạn này chạy trong tiến trình con.
- execvp(args[0], args) thay thế tiến trình con bằng chương trình mà người dùng nhập (echo, ls, pwd…)
- Khi chương trình chạy xong → thoát → shell in prompt mới.

2. Tạo tính năng sử dụng lại câu lệnh gần đây

2. Tạo tính năng sử dụng lại câu lệnh gần đây

- Cho phép người dùng thực thi lệnh gần đây bằng cách sử dụng các phím lên/xuống (để chọn lệnh) và nhấn Enter.

  Ví dụ: Nếu các lệnh đã nhập vào shell (theo thứ tự)

  ```
  echo abc
  ls -l
  pwd
  ```

  thì khi sử dụng các phím lên (↑) /xuống (↓), shell sẽ lần lượt hiển thị lại các lệnh trên để người dùng lựa chọn. Nhấn phím Enter để thực thi lệnh đang hiển thị.

Hình 3.Yêu cầu chi tiết của phần 2

Kết quả:

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS
phungphiubu@ubuntupp:~$ cd .c
phungphiubu@ubuntupp:~/.c$ ./it007sh
it007sh> echo abc
abc
it007sh> ls
count.sh   it007sh.c  lab522.c  lab53.c             SJF         test_execl.c  test_shm_B
count.txt  lab5_1     lab52.c   lab6.c              SJF.c       test_fork     test_shm_B.c
ex_3       lab5_1.c   lab53     out.txt             SRTF        test_fork.c   test_system
ex_3.c     lab52      lab533    producer_consumer   SRTF.c      test_shm_A    test_system.c
it007sh    lab522     lab533.c  producer_consumer.c test_execl  test_shm_A.c  time.c
it007sh> pwd
/home/phungphiubu/.c
it007sh>
```

Hình 4.Các lệnh được nhập vào theo thứ tự

Khi nhấn nút phím lên lần 1: lệnh pwd được ghi

```
phungphiubu@ubuntupp:~$ cd .c
phungphiubu@ubuntupp:~/.c$ ./it007sh
it007sh> echo abc
abc
it007sh> ls
count.sh   it007sh.c  lab522.c  lab53.c             SJF         test_execl.c  test_shm_B
count.txt  lab5_1     lab52.c   lab6.c              SJF.c       test_fork     test_shm_B.c
ex_3       lab5_1.c   lab53     out.txt             SRTF        test_fork.c   test_system
ex_3.c     lab52      lab533    producer_consumer   SRTF.c      test_shm_A    test_system.c
it007sh    lab522     lab533.c  producer_consumer.c test_execl  test_shm_A.c  time.c
it007sh> pwd
/home/phungphiubu/.c
it007sh> pwd
```

Hình 5.Terminal hiện kết quả khi nhấn phím lên lần 1

Khi nhấn phím lên lần 2: lệnh ls được ghi

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS
phungphiubu@ubuntupp:~$ cd .c
phungphiubu@ubuntupp:~/.c$ ./it007sh
it007sh> echo abc
abc
it007sh> ls
count.sh   it007sh.c  lab522.c  lab53.c             SJF         test_execl.c  test_shm_B
count.txt  lab5_1     lab52.c   lab6.c              SJF.c       test_fork     test_shm_B.c
ex_3       lab5_1.c   lab53     out.txt             SRTF        test_fork.c   test_system
ex_3.c     lab52      lab533    producer_consumer   SRTF.c      test_shm_A    test_system.c
it007sh    lab522     lab533.c  producer_consumer.c test_execl  test_shm_A.c  time.c
it007sh> pwd
/home/phungphiubu/.c
it007sh> ls
```

Hình 6.Terminal hiện kết quả nhấn phím lên lần 2

Khi nhấn phím xuống: lệnh pwd được ghi

```
phungphiubu@ubuntupp:~$ cd .c
phungphiubu@ubuntupp:~/.c$ ./it007sh
it007sh> echo abc
abc
it007sh> ls
count.sh   it007sh.c  lab522.c  lab53.c             SJF         test_execl.c  test_shm_B
count.txt  lab5_1     lab52.c   lab6.c              SJF.c       test_fork     test_shm_B.c
ex_3       lab5_1.c   lab53     out.txt             SRTF        test_fork.c   test_system
ex_3.c     lab52      lab533    producer_consumer   SRTF.c      test_shm_A    test_system.c
it007sh    lab522     lab533.c  producer_consumer.c test_execl  test_shm_A.c  time.c
it007sh> pwd
/home/phungphiubu/.c
it007sh> pwd
```

Hình 7.Terminal hiện kết quả nhấn phím xuống

Nhấn enter để thực thi lệnh:

```
phungphiubu@ubuntupp:~$ cd .c
phungphiubu@ubuntupp:~/.c$ ./it007sh
it007sh> echo abc
abc
it007sh> ls
count.sh   it007sh.c  lab522.c  lab53.c             SJF         test_execl.c  test_shm_B
count.txt  lab5_1     lab52.c   lab6.c              SJF.c       test_fork     test_shm_B.c
ex_3       lab5_1.c   lab53     out.txt             SRTF        test_fork.c   test_system
ex_3.c     lab52      lab533    producer_consumer   SRTF.c      test_shm_A    test_system.c
it007sh    lab522     lab533.c  producer_consumer.c test_execl  test_shm_A.c  time.c
it007sh> pwd
/home/phungphiubu/.c
it007sh> pwd
/home/phungphiubu/.c
it007sh> ls
count.sh   it007sh.c  lab522.c  lab53.c             SJF         test_execl.c  test_shm_B
count.txt  lab5_1     lab52.c   lab6.c              SJF.c       test_fork     test_shm_B.c
ex_3       lab5_1.c   lab53     out.txt             SRTF        test_fork.c   test_system
ex_3.c     lab52      lab533    producer_consumer   SRTF.c      test_shm_A    test_system.c
it007sh    lab522     lab533.c  producer_consumer.c test_execl  test_shm_A.c  time.c
it007sh>
```

Hình 8.Terminal hiện kết quả thực thi lệnh

→Qua đó thấy được khi sử dụng các phím lên (↑) /xuống (↓), shell sẽ lần lượt hiển thị lại các lệnh trên để người dùng lựa chọn. Nhấn phím Enter để thực thi lệnh đang hiển thị.

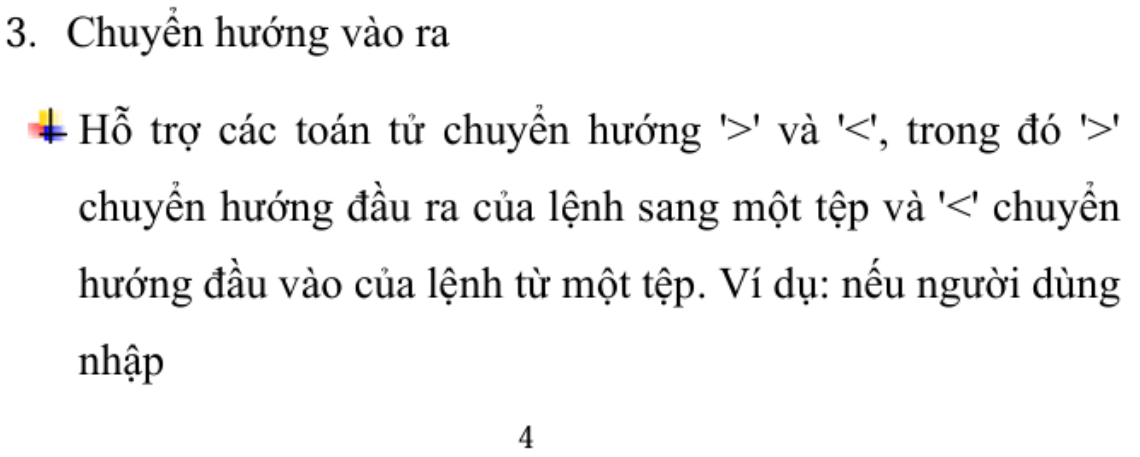
Phân tích:

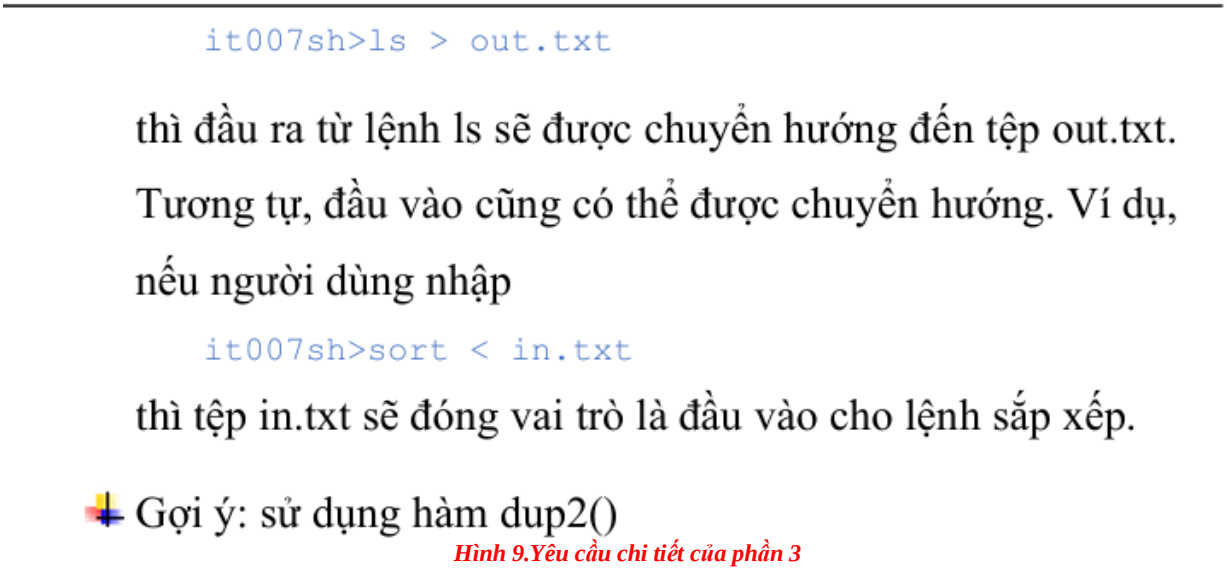
```
#include <readline/readline.h>
#include <readline/history.h>

char *line = readline("it007sh> ");
add_history(line);
```

Bảng 2. Các thành phần code liên quan tới yêu cầu 2

- readline() tự động hỗ trợ:
    - phím lên (↑): hiện lệnh trước
    - phím xuống (↓): hiện lệnh sau
- add_history(line) thêm lệnh vào lịch sử
- Người dùng nhấn Enter sẽ thực thi lệnh đang hiển thị

3. Chuyển hướng vào ra



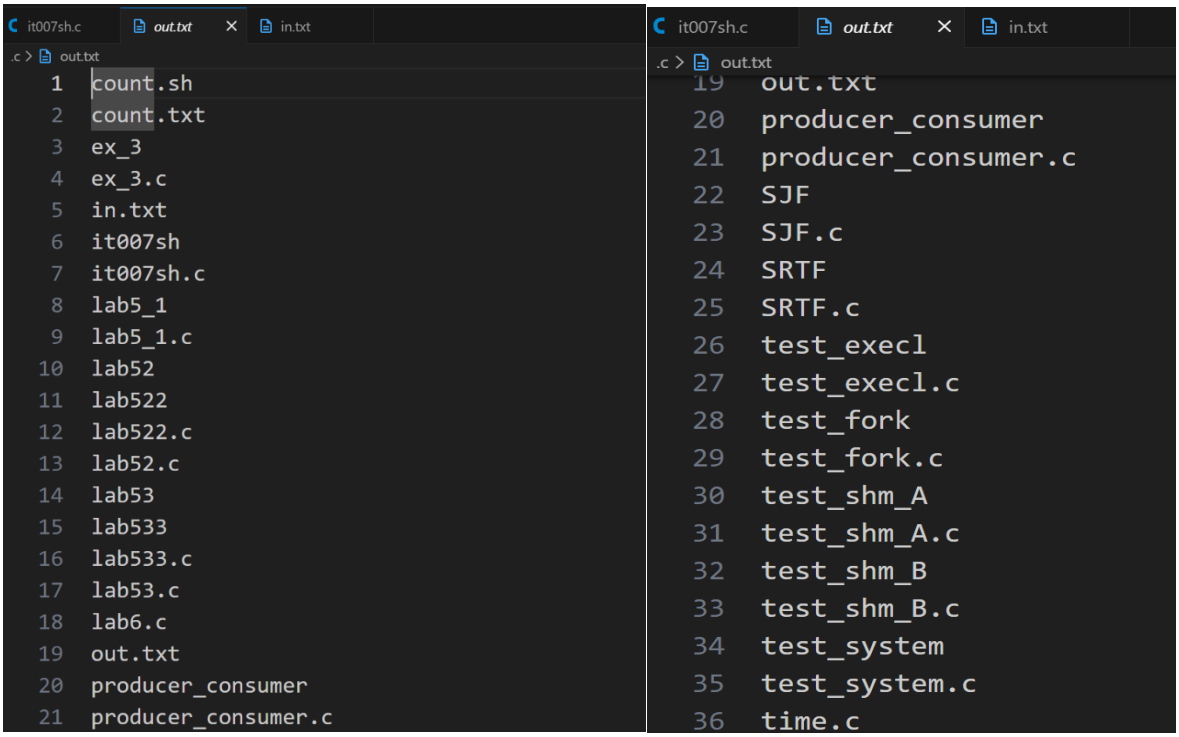


Hình 9.Yêu cầu chi tiết của phần 3

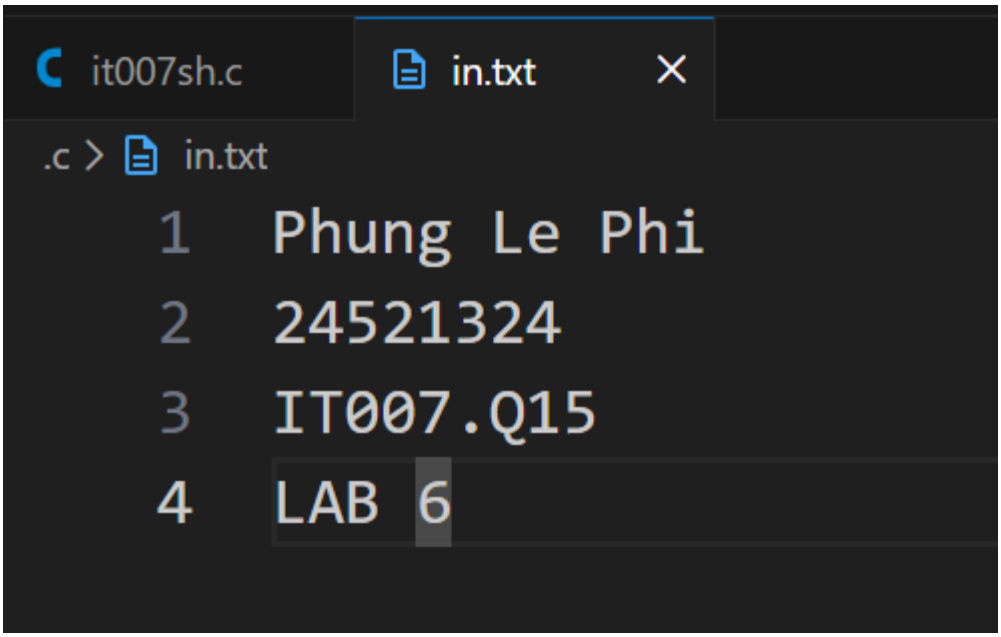
Kết quả:

```
it007sh> ls > out.txt
it007sh> sort < in.txt
24521324
IT007.Q15
LAB 6
Phung Le Phi
it007sh>
```

Hình 10.Kết quả trên terminal của yêu cầu 3



Hình 11. File out.txt sau khi thực thi lệnh ls > out.txt



Hình 12.File in.txt khi chưa thực thi lệnh sort < in.txt

→Qua đó hỗ trợ các toán tử chuyển hướng '>' và '<', trong đó '>' chuyển hướng đầu ra của lệnh sang một tệp và '<' chuyển hướng đầu vào của lệnh từ một tệp.

Phân tích :

```
if (strcmp(t, "<") == 0) *infile = token;
if (strcmp(t, ">") == 0) *outfile = token;
if (infile) {
    int fd = open(infile, O_RDONLY);
    dup2(fd, STDIN_FILENO);
}
if (outfile) {
    int fd = open(outfile, O_WRONLY | O_CREAT | O_TRUNC, 0644);
    dup2(fd, STDOUT_FILENO);
}
```

Bảng 3. Các thành code liên quan tới yêu cầu 3

- Parser tách tên file sau < hoặc >
- dup2(fd, STDIN_FILENO) → thay stdin bằng file (input redirect)
- dup2(fd, STDOUT_FILENO) → thay stdout bằng file (output redirect)

4. Giao tiếp sử dụng cơ chế đường ống

4. Giao tiếp sử dụng cơ chế đường ống

- Cho phép đầu ra của một lệnh đóng vai trò là đầu vào cho lệnh khác bằng cách sử dụng một đường ống. Ví dụ: Khi người dùng nhập

  ```
  it007sh>ls -l | less
  ```

  thì đầu ra của lệnh `ls -l` đóng vai trò là đầu vào cho lệnh `less`.

- Gợi ý: sử dụng hàm pipe() và dup2().

Hình 13.Yêu cầu chi tiết của phần 4

Kết quả:

```
phungphiubu@ubuntupp:~$ cd .c
phungphiubu@ubuntupp:~/.c$ ./it007sh
it007sh> ls -l | less
```

```
total 404
-rwxrwxr-x 1 phungphiubu phungphiubu   177 ต.ค.  24 13:50 count.sh
-rw-rw-r-- 1 phungphiubu phungphiubu    15 ต.ค.  25 21:16 count.txt
-rwxrwxr-x 1 phungphiubu phungphiubu 16992 ต.ค.  25 20:59 ex_3
-rw-rw-r-- 1 phungphiubu phungphiubu  1093 ต.ค.  25 20:59 ex_3.c
-rw-rw-r-- 1 phungphiubu phungphiubu    37 ธ.ค.  12 14:46 in.txt
-rwxrwxr-x 1 phungphiubu phungphiubu 17776 ธ.ค.  12 14:09 it007sh
-rwxrwxr-x 1 phungphiubu phungphiubu  3075 ธ.ค.  12 14:08 it007sh.c
-rwxrwxr-x 1 phungphiubu phungphiubu 17320 พ.ย.  28 13:58 lab5_1
-rwxrwxr-x 1 phungphiubu phungphiubu  1193 พ.ย.  28 13:58 lab5_1.c
-rwxrwxr-x 1 phungphiubu phungphiubu 17176 พ.ย.  28 14:19 lab52
-rwxrwxr-x 1 phungphiubu phungphiubu 17392 พ.ย.  28 14:42 lab522
-rwxrwxr-x 1 phungphiubu phungphiubu  1151 พ.ย.  28 14:41 lab522.c
-rwxrwxr-x 1 phungphiubu phungphiubu   924 พ.ย.  28 14:18 lab52.c
-rwxrwxr-x 1 phungphiubu phungphiubu 16984 พ.ย.  28 15:08 lab53
-rwxrwxr-x 1 phungphiubu phungphiubu 17288 พ.ย.  28 15:40 lab533
-rwxrwxr-x 1 phungphiubu phungphiubu  1044 พ.ย.  28 15:39 lab533.c
-rwxrwxr-x 1 phungphiubu phungphiubu   662 พ.ย.  28 15:07 lab53.c
-rw-rw-r-- 1 phungphiubu phungphiubu     0 ธ.ค.  12 13:23 lab6.c
-rw-r--r-- 1 phungphiubu phungphiubu   334 ธ.ค.  12 14:47 out.txt
-rwxrwxr-x 1 phungphiubu phungphiubu 17504 ต.ค.  25 21:56 producer_consumer
-rw-rw-r-- 1 phungphiubu phungphiubu  2197 ต.ค.  25 21:56 producer_consumer.c
-rwxrwxr-x 1 phungphiubu phungphiubu 21496 พ.ย.  16 21:12 SJF
-rwxrwxr-x 1 phungphiubu phungphiubu 11007 พ.ย.  16 21:12 SJF.c
-rwxrwxr-x 1 phungphiubu phungphiubu 21504 พ.ย.  16 22:21 SRTF
-rwxrwxr-x 1 phungphiubu phungphiubu 10809 พ.ย.  16 22:21 SRTF.c
-rwxrwxr-x 1 phungphiubu phungphiubu 17000 ต.ค.  25 16:45 test_execl
-rw-rw-r-- 1 phungphiubu phungphiubu  1064 ต.ค.  24 13:54 test_execl.c
-rwxrwxr-x 1 phungphiubu phungphiubu 16960 ต.ค.  25 16:08 test_fork
-rwxrwxr-x 1 phungphiubu phungphiubu   960 ต.ค.  24 13:43 test_fork.c
-rwxrwxr-x 1 phungphiubu phungphiubu 17056 ต.ค.  25 18:15 test_shm_A
-rw-rw-r-- 1 phungphiubu phungphiubu  1523 ต.ค.  25 18:19 test_shm_A.c
-rwxrwxr-x 1 phungphiubu phungphiubu 17008 ต.ค.  25 18:15 test_shm_B
-rw-rw-r-- 1 phungphiubu phungphiubu  1381 ต.ค.  25 19:02 test_shm_B.c
-rwxrwxr-x 1 phungphiubu phungphiubu 16920 ต.ค.  25 17:39 test_system
-rw-rw-r-- 1 phungphiubu phungphiubu   796 ต.ค.  24 14:06 test_system.c
drwxrwxr-x 2 phungphiubu phungphiubu  4096 ต.ค.  25 20:28 time.c
(END)
```

Hình 14.Kết quả trên terminal của yêu cầu 4

→ Qua đó cho phép đầu ra của một lệnh đóng vai trò là đầu vào cho lệnh khác bằng cách sử dụng một đường ống.

Phân tích :

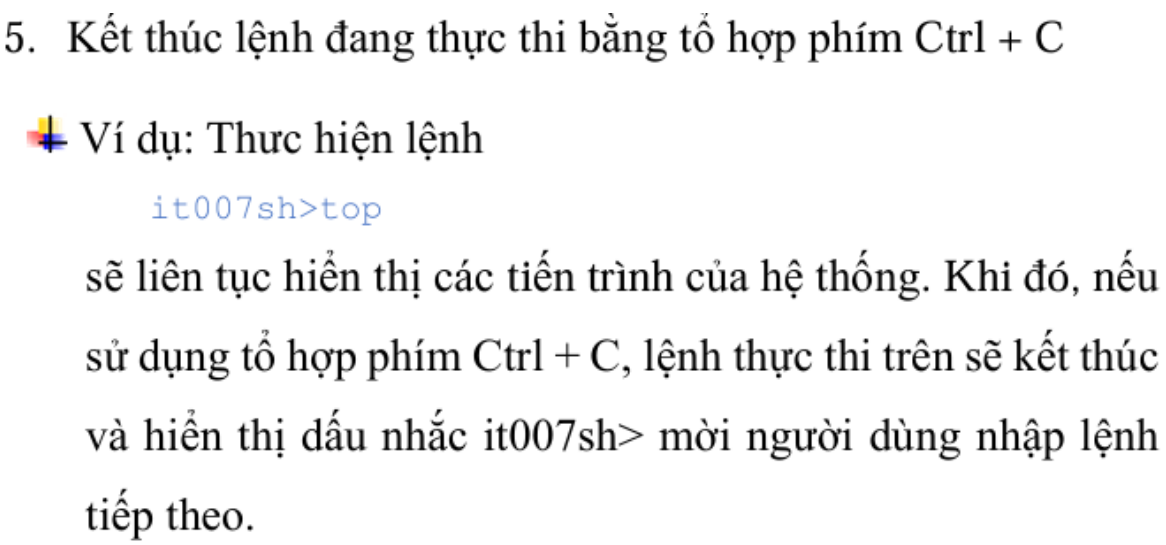
```
int n = split_pipe(line, cmds);
pipe(pipefd + 2*i);
```

```
if (i > 0)
   dup2(pipefd[(i-1)*2], STDIN_FILENO);
if (i < n-1)
   dup2(pipefd[i*2+1], STDOUT_FILENO);
```

Bảng 4.Các thành phần code liên qua tới yêu cầu 4

- split_pipe() chia chuỗi lệnh thành từng phần trước/sau dấu |
- Mỗi cặp lệnh có 1 pipe
- Lệnh đầu → ghi vào pipe
- Lệnh giữa → đọc bên trái, ghi bên phải
- Lệnh cuối → đọc từ pipe

5. Kết thúc lệnh đang thực thi bằng tổ hợp phím Ctrl + C



Hình 15.Yêu cầu chi tiết của phần 5

Kết quả : (ở trang sau)

```
it007sh> top
top - 15:13:59 up  1:51,  1 user,  load average: 0.04, 0.08, 0.08
Tasks: 214 total,   1 running, 213 sleeping,   0 stopped,   0 zombie
%Cpu(s):  0.5 us,  0.7 sy,  0.0 ni, 98.5 id,  0.1 wa,  0.0 hi,  0.2 si,  0.0 st
MiB Mem :   4428.6 total,    683.1 free,   1516.4 used,   2229.0 buff/cache
MiB Swap:   1162.4 total,   1162.4 free,      0.0 used.   2640.2 avail Mem

    PID USER      PR  NI    VIRT    RES    SHR S  %CPU  %MEM     TIME+ COMMAND
   8354 phungph+  20   0   32.0g 308276  57804 S   1.0   6.8   3:00.28 node
   8079 phungph+  20   0   11.3g 183896  52656 S   0.6   4.1   3:09.16 node
   8303 phungph+  20   0 1164116  81620  47704 S   0.6   1.8   0:39.07 node
    286 root      20   0       0      0      0 I   0.3   0.0   0:04.09 kworker/1:3-mm_percpu_wq
   4165 phungph+  20   0   29128  17016   7948 S   0.3   0.4   0:04.76 systemd
   7930 phungph+  20   0   40240  19356  13704 S   0.3   0.4   0:52.32 code-bf9252a2fb
  29516 root      20   0       0      0      0 I   0.3   0.0   0:00.15 kworker/u8:2-events_unbound
  29930 phungph+  20   0   12332   4064   3276 R   0.3   0.1   0:00.06 top
      1 root      20   0  168292  11608   8456 S   0.0   0.3   0:02.84 systemd
      2 root      20   0       0      0      0 S   0.0   0.0   0:00.05 kthreadd
      3 root       0 -20       0      0      0 I   0.0   0.0   0:00.01 rcu_gp
      4 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 rcu_par_gp
      5 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 slub_flushwq
      6 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 netns
      8 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 kworker/0:0H-events_highpri
     10 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 mm_percpu_wq
     11 root      20   0       0      0      0 S   0.0   0.0   0:00.00 rcu_tasks_rude_
     12 root      20   0       0      0      0 S   0.0   0.0   0:00.00 rcu_tasks_trace
     13 root      20   0       0      0      0 S   0.0   0.0   0:00.71 ksoftirqd/0
     14 root      20   0       0      0      0 I   0.0   0.0   0:18.01 rcu_sched
     15 root      rt   0       0      0      0 S   0.0   0.0   0:00.67 migration/0
     16 root     -51   0       0      0      0 S   0.0   0.0   0:00.00 idle_inject/0
     18 root      20   0       0      0      0 S   0.0   0.0   0:00.00 cpuhp/0
     19 root      20   0       0      0      0 S   0.0   0.0   0:00.00 cpuhp/1
     20 root     -51   0       0      0      0 S   0.0   0.0   0:00.00 idle_inject/1
     21 root      rt   0       0      0      0 S   0.0   0.0   0:01.18 migration/1
     22 root      20   0       0      0      0 S   0.0   0.0   0:00.66 ksoftirqd/1
```

Hình 16.Terminal hiện kết quả sau lệnh top

```
phungphiubu@ubuntupp:~/.c$ ./it007sh
MiB Mem :   4428.6 total,    681.9 free,   1517.6 used,   2229.1 buff/cache
MiB Swap:   1162.4 total,   1162.4 free,      0.0 used.   2639.0 avail Mem

    PID USER      PR  NI    VIRT    RES    SHR S  %CPU  %MEM     TIME+ COMMAND
   8354 phungph+  20   0   32.0g 308284  57804 S   1.0   6.8   3:00.40 node
   7930 phungph+  20   0   40240  19356  13704 S   0.7   0.4   0:52.37 code-bf9252a2fb
  29930 phungph+  20   0   12332   4064   3276 R   0.7   0.1   0:00.11 top
   7895 phungph+  20   0   14136   6096   4444 S   0.3   0.1   0:49.45 sshd
   8079 phungph+  20   0   11.3g 183896  52656 S   0.3   4.1   3:09.22 node
   8214 root      20   0       0      0      0 I   0.3   0.0   0:16.35 kworker/0:8-events
   8339 phungph+  20   0 1265456  72708  47584 S   0.3   1.6   0:49.68 node
      1 root      20   0  168292  11608   8456 S   0.0   0.3   0:02.84 systemd
      2 root      20   0       0      0      0 S   0.0   0.0   0:00.05 kthreadd
      3 root       0 -20       0      0      0 I   0.0   0.0   0:00.01 rcu_gp
      4 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 rcu_par_gp
      5 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 slub_flushwq
      6 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 netns
      8 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 kworker/0:0H-events_highpri
     10 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 mm_percpu_wq
     11 root      20   0       0      0      0 S   0.0   0.0   0:00.00 rcu_tasks_rude_
     12 root      20   0       0      0      0 S   0.0   0.0   0:00.00 rcu_tasks_trace
     13 root      20   0       0      0      0 S   0.0   0.0   0:00.71 ksoftirqd/0
     14 root      20   0       0      0      0 R   0.0   0.0   0:18.04 rcu_sched
     15 root      rt   0       0      0      0 S   0.0   0.0   0:00.67 migration/0
     16 root     -51   0       0      0      0 S   0.0   0.0   0:00.00 idle_inject/0
     18 root      20   0       0      0      0 S   0.0   0.0   0:00.00 cpuhp/0
     19 root      20   0       0      0      0 S   0.0   0.0   0:00.00 cpuhp/1
     20 root     -51   0       0      0      0 S   0.0   0.0   0:00.00 idle_inject/1
     21 root      rt   0       0      0      0 S   0.0   0.0   0:01.18 migration/1
     22 root      20   0       0      0      0 S   0.0   0.0   0:00.66 ksoftirqd/1
     24 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 kworker/1:0H-events_highpri
     25 root      20   0       0      0      0 S   0.0   0.0   0:00.00 cpuhp/2
     26 root     -51   0       0      0      0 S   0.0   0.0   0:00.00 idle_inject/2
     27 root      rt   0       0      0      0 S   0.0   0.0   0:00.79 migration/2
     28 root      20   0       0      0      0 S   0.0   0.0   0:04.00 ksoftirqd/2

it007sh>
```

Hình 17.Terminal sau khi nhấn Ctrl + C

→ Qua đó sử dụng tổ hợp phím Ctrl + C, lệnh thực thi trên sẽ kết thúc và hiển thị dấu nhắc it007sh> mời người dùng nhập lệnh tiếp theo.

Phân tích :

```
signal(SIGINT, sigint_handler);
void sigint_handler(int sig) {
    if (child_pid != 0) {
        kill(child_pid, SIGINT);
        printf("\n");
    } else {
        printf("\nit007sh> ");
    }
}
signal(SIGINT, SIG_DFL);
```

Bảng 5.Các thành phần code liên quan tới yêu cầu 5

- Shell cha giữ SIGINT → không bị kill
- Khi user nhấn Ctrl + C:
  - nếu tiến trình con đang chạy → bị gửi SIGINT → dừng.
  - shell cha vẫn chạy và in prompt mới.

6. Toàn bộ code của chương trình it007sh.c đáp ứng 5 yêu cầu của bài thực hành

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <string.h>
#include <signal.h>
#include <fcntl.h>
#include <sys/wait.h>
#include <readline/readline.h>
#include <readline/history.h>
```

```
#define MAX_ARGS 128
#define MAX_CMDS 64
#define MAX_LEN 1024

volatile pid_t child_pid = 0;

void sigint_handler(int sig) {
    if (child_pid != 0) {
        kill(child_pid, SIGINT);
        printf("\n");
    } else {
        printf("\nit007sh> ");
        fflush(stdout);
    }
}

void trim(char *s) {
    while (*s == ' ' || *s == '\t') s++;
    int len = strlen(s);
    while (len > 0 && (s[len-1]==' ' || s[len-1]=='\t' || s[len-1]=='\n'))
        s[--len] = '\0';
}

int split_pipe(char *line, char *cmds[]) {
    int n = 0;
    cmds[n++] = strtok(line, "|");
    while ((cmds[n] = strtok(NULL, "|")) != NULL) n++;
    return n;
}

void parse_command(char *cmd, char *args[], char **infile, char **outfile) {
    *infile = NULL;
```

```
    *outfile = NULL;
    int argc = 0;
    char *t = strtok(cmd, " ");
    while (t) {
        if (strcmp(t, "<") == 0) {
            t = strtok(NULL, " ");
            *infile = t;
        } else if (strcmp(t, ">") == 0) {
            t = strtok(NULL, " ");
            *outfile = t;
        } else {
            args[argc++] = t;
        }
        t = strtok(NULL, " ");
    }
    args[argc] = NULL;
}

void execute_pipeline(int n, char *cmds[]) {
    int pipefd[2*(n-1)];
    for (int i=0; i<n-1; i++) pipe(pipefd + 2*i);

    pid_t pids[n];

    for (int i=0; i<n; i++) {
        char *args[MAX_ARGS];
        char *infile = NULL, *outfile = NULL;

        trim(cmds[i]);
        parse_command(cmds[i], args, &infile, &outfile);

        child_pid = fork();
```

```
    pids[i] = child_pid;

    if (child_pid == 0) {
        signal(SIGINT, SIG_DFL);

        if (i > 0)
            dup2(pipefd[(i-1)*2], STDIN_FILENO);

        if (i < n-1)
            dup2(pipefd[i*2+1], STDOUT_FILENO);

        if (infile) {
            int fd = open(infile, O_RDONLY);
            dup2(fd, STDIN_FILENO);
            close(fd);
        }
        if (outfile) {
            int fd = open(outfile, O_WRONLY|O_CREAT|O_TRUNC, 0644);
            dup2(fd, STDOUT_FILENO);
            close(fd);
        }

        for (int j=0; j<2*(n-1); j++) close(pipefd[j]);

        execvp(args[0], args);
        exit(1);
    }
}

for (int i=0; i<2*(n-1); i++) close(pipefd[i]);

for (int i=0; i<n; i++) waitpid(pids[i], NULL, 0);
```

```
    child_pid = 0;
}

int main() {
    signal(SIGINT, sigint_handler);

    while (1) {
        char *line = readline("it007sh> ");
        if (!line) break;

        trim(line);
        if (strlen(line) == 0) {
            free(line);
            continue;
        }

        add_history(line);

        if (strcmp(line, "exit") == 0) {
            free(line);
            break;
        }

        char *cmds[MAX_CMDS];
        int n = split_pipe(line, cmds);

        execute_pipeline(n, cmds);

        free(line);
    }
```

```
    return 0;
}
```

Table 1. Nội dung của chương trình

HẾT