|  |
| --- |
| Name: Nguyễn Hữu Tứ  ID:19522453  Class:IT007.L21.1 |

OPERATING SYSTEM  
LAB X’S REPORT

**SUMMARY**

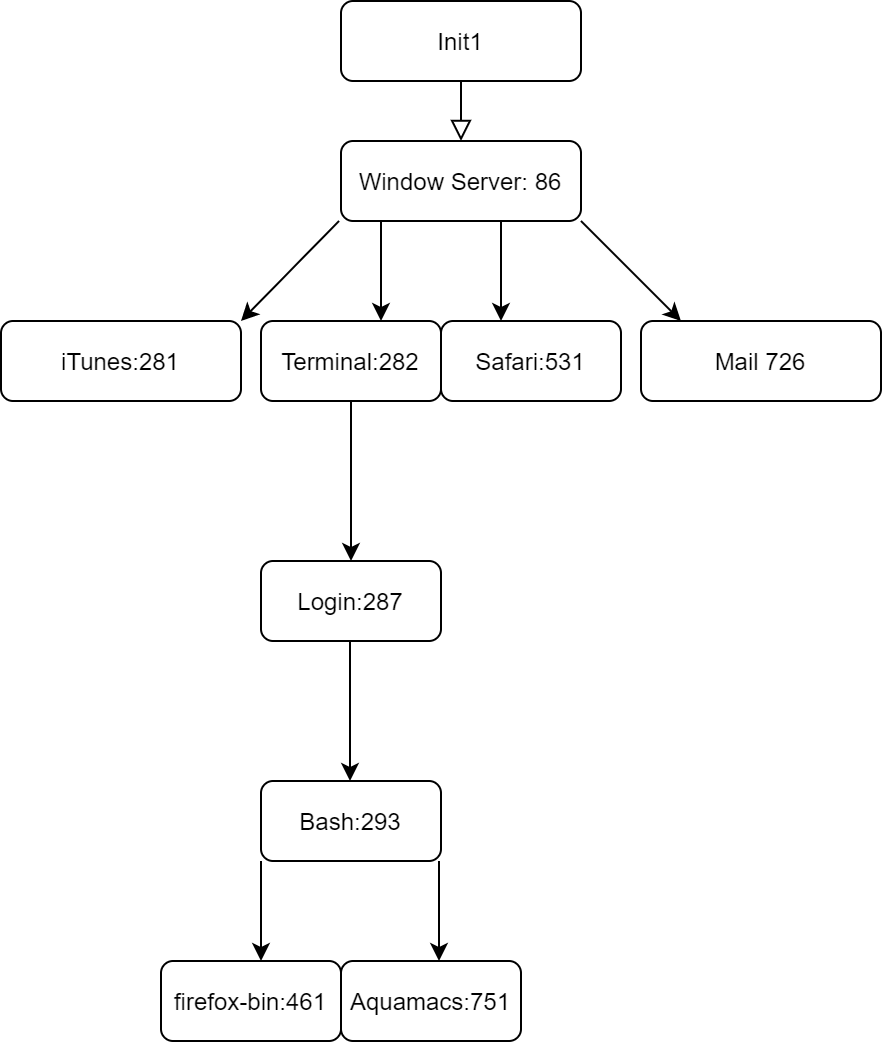
|  |  |  |  |
| --- | --- | --- | --- |
| **Task** | | **Status** | **Page** |
|  | Task name 4 | Done | 1 |
| Task name 5 | Done | 2 |
| Task name 6 | Done | 4 |
| Task name 7 | Done | 7 |

**Self-scrores:**

# Section 1.5

## Task name 1: Mối quan hệ cha-con giữa các tiến trình

Vẽ cây quan hệ parent-child của các tiến trình:

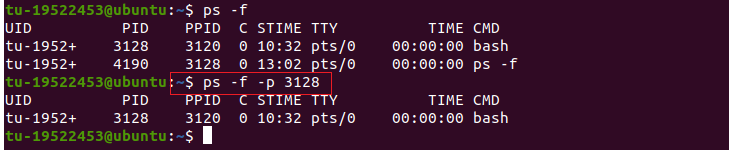


Cách sử dụng lệnh ps để tìm tiến trình cha của một một tiến trình dựa vào PID của nó :

- Sử dụng ps -f -p để hiển thị đầy đủ thông tin của một tiến trình dựa vào PID của nó:

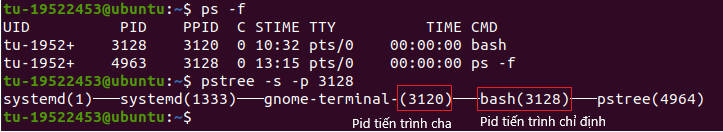
• PID: ID của tiến trình

• PPID: ID của tiến trình cha của nó

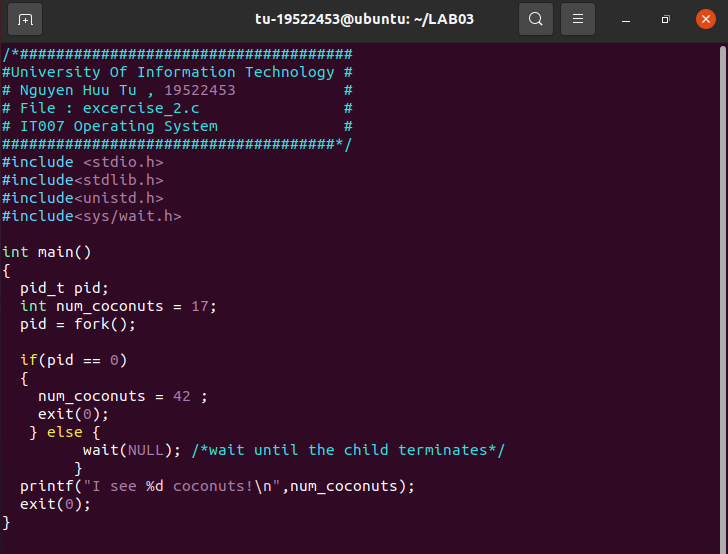


Tìm hiểu và cài đặt lệnh pstree (nếu chưa được cài đặt), sau đó trình bày cách sử dụng lệnh này để tìm tiến trình cha của một tiến trình dựa vào PID của nó

- Sử dụng pstree -s -p để hiển thị một phần của cây tiến trình, cho phép xem PID tiến trình cha của một tiến trình chỉ định



## Task name 2:





* Chương trình sẽ in ra dòng chữ “I see 17 coconuts !”
* Giải thích :

Đầu tiên num\_coconuts = 17

Khi đến dòng lệnh pid = fork() thì sinh ra 1 tiến trình con. Gán pid = giá trị trả về của hàm fork() => pid của tiến trình con = 0, và pid của tiến trình cha > 0.

Sau đó, cả 2 tiến trình cùng chạy, tiến trình cha đến dòng lệnh wait(NULL) sẽ đợi cho đến khi tiến trình con terminated.

Tiến trình con gán num\_coconuts = 42 nhưng khi chạy đến dòng lệnh exit(0) đã bị thoát ra. Lúc này giá trị num\_coconuts ở tiến trình cha vẫn là 17.

Dòng lệnh printf("I see %d coconuts!\n", num\_coconuts); chỉ có tiến trình cha thực hiện vì tiến trình con sau khi thực hiện lệnh exit(0) đã bị thoát ra.

## Task name 3: Các hàm được sử dụng để làm thay đổi thuộc tính thread

**pthread\_attr\_init**: dùng để khởi tạo đối tượng thuộc tính với giá trị mặc định của chúng. Các thuộc tính có thể là: scope, detachstate, stackaddr,stacksize, priority, inheritshed, shedpolicy.

**pthread\_attr\_destroy**: dùng giải phóng tài nguyên của đối tượng thuộc tính đã được tạo.

**pthread\_attr\_setdetachstate**: thiết lập thuộc tính tách được

**pthread\_attr\_getdetachstate**: lấy thuộc tính tách được của đối tượng

**pthread\_attr\_setguardsize**: thiết lập guard size (??) của đối tượng thuộc tính

**pthread\_attr\_setscope**: để tạo thread có giới hạn (bound) hoặc thread không giới hạn (unbound).

**pthread\_attr\_getscope**: lấy về giá trị thread scope, từ đó biết được là thread là bound hay unbound.

**pthread\_attr\_setshedpolicy**: để thiết lập thuật toán lập lịch tiến trình. Có thể là SHED\_FIFO (first-in-first-out), SCHED\_RR (round-robin) or SCHED\_OTHER.

**pthread\_attr\_getschedpolicy**: lấy về thuật toán lập lịch tiến trình hiện tại của hệ thống.

**pthread\_attr\_setschedparams**: thiết lập các tham số lập lịch.

**pthread\_attr\_getschedparams**: lấy về các tham số lập lịch đã được định nghĩa bởi hàm pthread\_attr\_setschedparam().

**pthread\_attr\_setstacksize**: thiết lập kích thước stack của thread.

**pthread\_attr\_getstacksize**: lấy về kích thước stack của thread được thiết lập bởi thread\_attr\_setstacksize().

**pthread\_attr\_setstackaddr**: thiết lập địa chỉ stack của thread.

**pthread\_attr\_getstackaddr**: trả về địa chỉ stack của thread đã được thiết lập bởi hàm pthread\_attr\_setstackaddr.

## Task name 4:

1. In ra dòng chữ: “Welcome to IT007, I am !”