

**ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH**

**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN**







**HỆ ĐIỀU HÀNH**

**LỚP: IT007.O18.1**

**BÁO CÁO BÀI TẬP THỰC HÀNH 3**

**Họ tên: Trần Đình Khánh Đăng**

**MSSV: 22520195**



Câu 1: Mối quan hệ cha-con giữa các tiến trình

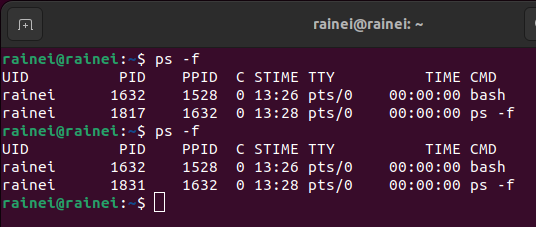
1. Vẽ cây quan hệ parent-child của các tiến trình bên dưới:

|  |  |  |  |
| --- | --- | --- | --- |
| UID | PID | PPID | COMMAND |
| 88 | 86 | 1 | WindowServer |
| 501 | 281 | 86 | iTunes |
| 501 | 282 | 86 | Terminal |
| 0 | 287 | 282 | login |
| 501 | 461 | 293 | firefox-bin |
| 501 | 531 | 86 | Safari |
| 501 | 726 | 86 | Mail |
| 501 | 751 | 293 | Aquamacs |
| 501 | 293 | 287 | -bash |

Hình 1.1: Câu a

1. Trình bày cách sử dụng lệnh ps để tìm tiến trình cha của một tiến trình dựa vào PID của nó.

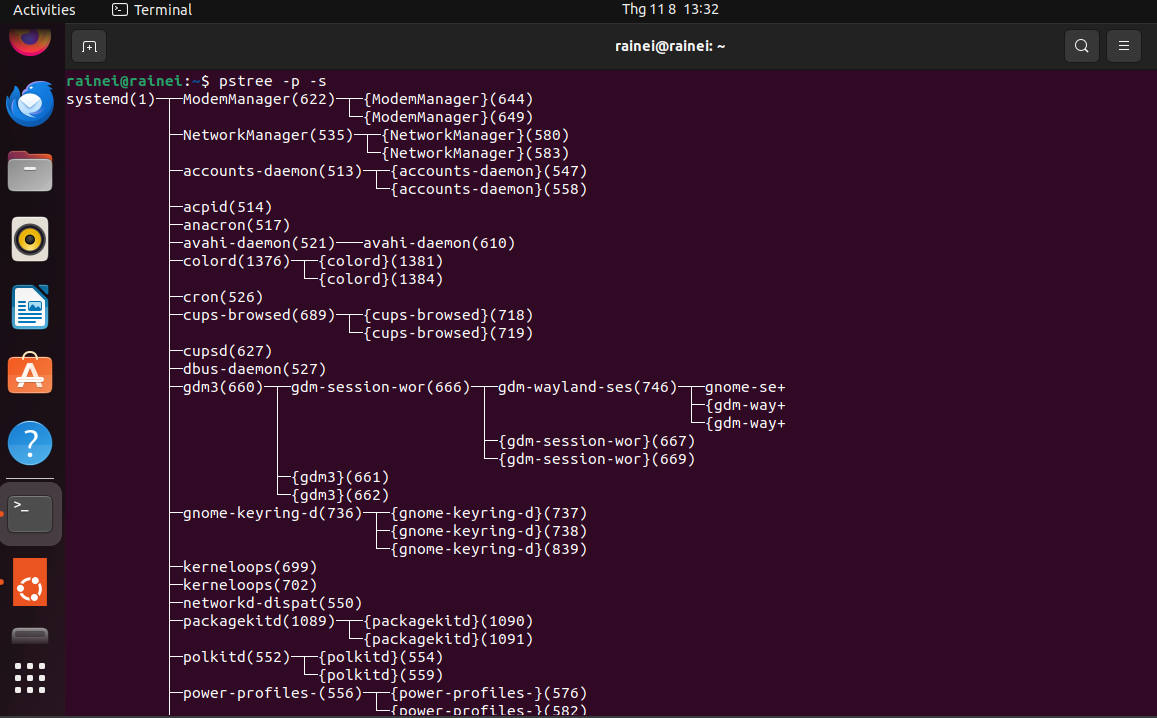
Sử dụng lệnh ps -f để hiển thị pid của tiến trình cha (PPID), sau đó sử dụng lệnh ps -f một lần nữa để xem thông tin của tiến trình cha.



Hình 1.2: Câu b

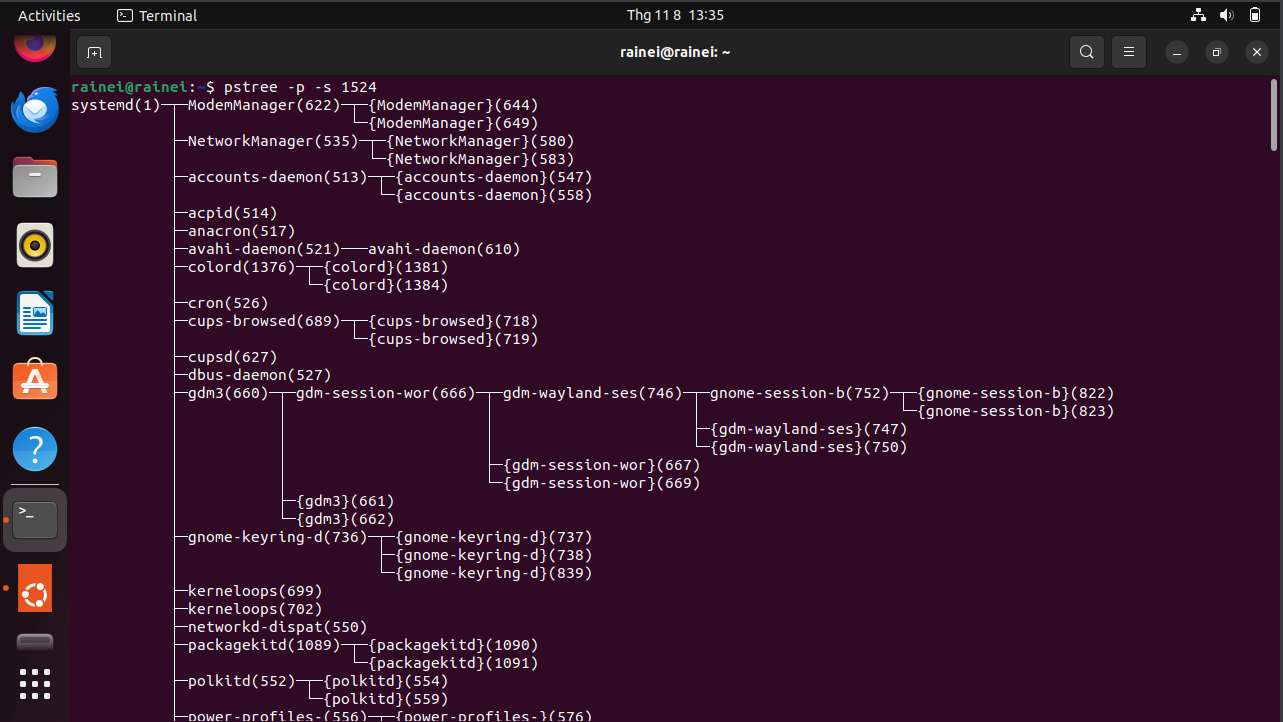
1. Tìm hiểu và cài đặt lệnh pstree (nếu chưa được cài đặt), sau đó trình bày cách sử dụng lệnh này để tìm tiến trình cha của một tiến trình dựa vào PID của nó.

Dùng lệnh pstree -p -s để hiển thị cả cây tiến trình tương ứng với tiến trình con được nhập vào.



Hình 1.3.1: Câu c

Có thể dùng lệnh **pstree -p -s <pid của tiến trình con cần tìm tiến trình cha>**

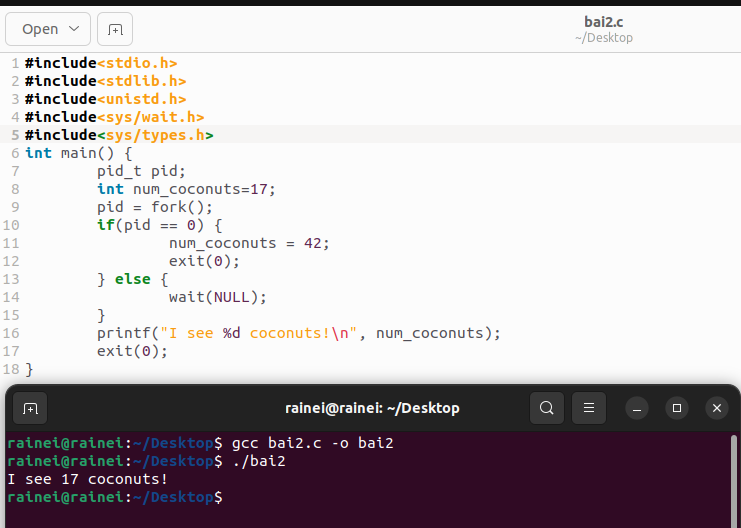
****

Hình 1.3.2: Câu c

Câu 2: Chương trình sẽ in ra kết quả gì? Giải thích tại sao.

Chương trình bị lỗi biên dịch do thiếu thư viện unistd.h và stdlib.h.

Sửa lại:



Hình 2: Chương trình

Bây giờ, chương trình in ra **“I see 17 coconuts!”**

Câu 3:

**Các hàm được sử dụng để làm thay đổi thuộc tính của pthread:**

• **pthread\_attr\_init (pthread\_attr\_t \*attr)**: để khởi tạo đối tượng thuộc tính luồng với các giá trị mặc định.

**• pthread\_attr\_destroy (pthread\_attr\_t \*attr)**: để xóa bộ nhớ cấp phát cho đối tượng thuộc tính luồng và làm cho nó không hợp lệ.

• **pthread\_attr\_setdetachstate (pthread\_attr\_t \*attr, int detachstate)**: để đặt trạng thái detach của luồng (joinable hoặc detached) cho đối tượng thuộc tính luồng.

**• pthread\_attr\_getdetachstate (const pthread\_attr\_t \*attr, int \*detachstate)**: để lấy trạng thái detach của luồng từ đối tượng thuộc tính luồng.

**• pthread\_attr\_setscope (pthread\_attr\_t \*attr, int scope)**: để đặt phạm vi tranh chấp của luồng (process hoặc system) cho đối tượng thuộc tính luồng.

**• pthread\_attr\_getscope (const pthread\_attr\_t \*attr, int \*scope)**: để lấy phạm vi tranh chấp của luồng từ đối tượng thuộc tính luồng.

**• pthread\_attr\_setstackaddr (pthread\_attr\_t \*attr, void \*stackaddr)**: để đặt địa chỉ ngăn xếp của luồng cho đối tượng thuộc tính luồng.

**• pthread\_attr\_getstackaddr (const pthread\_attr\_t \*attr, void \*\*stackaddr)**: để lấy địa chỉ ngăn xếp của luồng từ đối tượng thuộc tính luồng.

**• pthread\_attr\_setstacksize (pthread\_attr\_t \*attr, size\_t stacksize)**: để đặt kích thước ngăn xếp của luồng cho đối tượng thuộc tính luồng.

**• pthread\_attr\_getstacksize (const pthread\_attr\_t \*attr, size\_t \*stacksize)**: để lấy kích thước ngăn xếp của luồng từ đối tượng thuộc tính luồng.

• **pthread\_attr\_setschedpolicy (pthread\_attr\_t \*attr, int policy)**: để đặt chính sách lập lịch của luồng cho đối tượng thuộc tính luồng.

• **pthread\_attr\_getschedpolicy (const pthread\_attr\_t \*restrict attr, int \*restrict policy)**: để lấy chính sách lập lịch của luồng từ đối tượng thuộc tính luồng.

• **pthread\_attr\_setguardsize (pthread\_attr\_t \*attr, size\_t guardsize)**: để đặt kích thước bảo vệ ngăn xếp của luồng cho đối tượng thuộc tính luồng.

**Một vài chương trình minh họa:**

**pthread\_attr\_init**

Hàm pthread\_attr\_init có prototype như sau:

**#include <pthread.h>**

**int pthread\_attr\_init(pthread\_attr\_t \*attr);**

Trong đó, attr là con trỏ trỏ đến đối tượng thuộc tính luồng cần khởi tạo. Hàm này trả về 0 nếu thành công, hoặc một số dướng là mã lỗi nếu có lỗi.

**Code và kết quả:**

Ảnh có chứa văn bản, đồ điện tử, ảnh chụp màn hình, phần mềm

Mô tả được tạo tự động

**Công dụng trong chương trình**: Tác dụng của hàm **pthread\_attr\_init** trong ví dụ trên là để khởi tạo một đối tượng thuộc tính luồng với các giá trị mặc định cho các thuộc tính như trạng thái detach, kích thước ngăn xếp, chính sách lập lịch, v.v. Điều này có nghĩa là khi tạo luồng mới với đối tượng thuộc tính luồng này, luồng sẽ có các thuộc tính theo giá trị mặc định của hệ thống. Đây là một cách để thiết lập các thuộc tính cho các luồng mới một cách nhanh chóng và tiện lợi.

**pthread\_attr\_destroy**

Hàm **pthread\_attr\_destroy** có prototype như sau:

**#include <pthread.h>   
int pthread\_attr\_destroy(pthread\_attr\_t \*attr);**

Trong đó, **attr** là con trỏ trỏ đến đối tượng thuộc tính luồng cần xóa. Hàm này trả về 0 nếu thành công, hoặc một số dướng là mã lỗi nếu có lỗi.

**Code và kết quả:**



**Công dụng trong chương trình:** Tác dụng của hàm **pthread\_attr\_destroy** trong chướng trình trên là để xóa bộ nhớ cấp phát cho đối tượng thuộc tính luồng và làm cho nó không hợp lệ. Điều này có nghĩa là đối tượng thuộc tính luồng này không thể được sử dụng để tạo các luồng mới. Đây là một cách để tránh rò rỉ bộ nhớ và giảm thiểu lãng phí tài nguyên của tiến trình.

Câu 4: Viết chương trình làm các công việc sau theo thứ tự:

Ảnh có chứa văn bản, ảnh chụp màn hình, phần mềm, Trang web

Mô tả được tạo tự động

Hình 4: Chương trình

1. In ra dòng chữ “Welcome to IT007, I am 22520195!”.

Ảnh có chứa văn bản, Phông chữ, ảnh chụp màn hình, Đồ họa

Mô tả được tạo tự động

Hình 4.1: Câu a

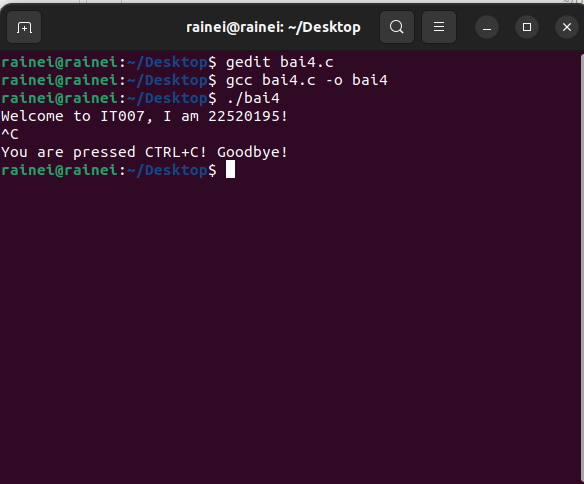
1. Mở tệp abcd.txt bằng vim editor

Ảnh có chứa văn bản, ảnh chụp màn hình, phần mềm, Phần mềm đa phương tiện

Mô tả được tạo tự động

Hình 4.2: Câu b

1. Tắt vim editor khi người dùng nhất CRTL+C.



Hình 4.3: Câu c

1. Khi người dùng nhấn CTRL+C thì in ra dòng chữ: “You are pressed CTRL+C! Goodbye!”.

Ảnh có chứa văn bản, đồ điện tử, ảnh chụp màn hình, màn hình

Mô tả được tạo tự động

Hình 4.4: Câu d