**Họ tên: Nguyễn Thị Minh Châu – MSSV: 21520645**

**BÁO CÁO THỰC HÀNH LAB 3**

**1. Mối quan hệ cha-con giữa các tiến trình**

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

Diagram

Description automatically generated

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

Định nghĩa: Lệnh ps trong Linux được sử dụng để liệt kê các tiến trình đang chạy.

Sau khi sử dụng lệnh ps sẽ hiển thị một số thông tin như:

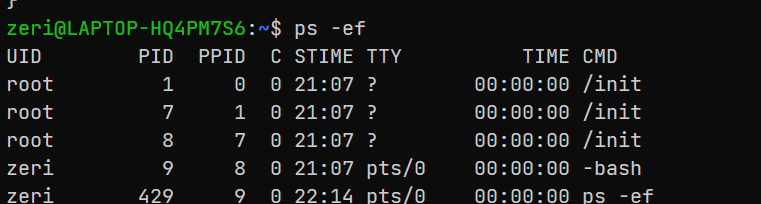
+PID: ID của tiến trình.

+TTY: Thông tin terminal mà người dùng đăng nhập.

+TIME: Lượng CPU tính bằng phút giây mà tiến trình đó chạy.

+CMD: Câu lệnh để thực hiện process đó.

Ta dùng lệnh ps -ef hoặc ps – eF



Ngoài thông tin được đề cập được thể hiện ở lệnh ps thì lệnh ps -ef cung cấp thêm một số thông

tin như :

+UID: ID của người dùng

+PPID: ID của tiến trình cha

+STIME: Thời gian mà câu lệnh đó khởi động

Đối với lệnh ps – eF, tùy chọn -F có thể xem thêm định dạng đầy đủ hơn tùy chọn -f, cụ thể như:

+SZ: Lượng bộ nhớ ảo sử dụng

+RSS: Lượng bộ nhớ sử dụng thực

c. Tìm hiểu và cài đặt lệnh pstree (Nếu chưa được cài đặt), sau đó trình bày cách sử dụng

lệnh này để tìm kiếm tiến trình cha của một tiến trình dựa vào PID của nó.

**Định nghĩa:** Tương tự như ps , nhưng thay vì liệt kê các tiến trình đang chạy, nó hiển thị chúng trong một cây. Định dạng giống như cây là một cách thuận tiện hơn để hiển thị phân cấp quy trình và làm cho đầu ra trực quan hấp dẫn hơn.

Cấu trúc đơn giản của pstree:

parent————child(1)————subchild (1)

| |--subchild (2)

|

|-child(2)

Một số trường hợp thực hiện lệnh pstree như:

- Xem thông tin tiến trình: pstree

- Xem tiến trình cùng với PID của nó: pstree -p

- Xem tiến trình cùng với câu lệnh của nó: pstree -a

- Xem tiến trình cụ thể cùng với highlight: pstree -H [PID]

- Xem cây tiến trình theo thứ tự PID: pstree -np

- Xem các tiến trình theo cây thuộc người dùng: pstree [Tên người dùng]

- Hiện thị ID của nhóm tiến trình: pstree -g

**2. Chương trình bên dưới in ra kết quả gì? Giải thích tại sao?**

**Ảnh có chứa văn bản

Mô tả được tạo tự động**

* Chương trình báo lỗi unknown type name, implicit declaration of function, incompatible

implicit declaration of built-in function, …

Đây là những lỗi về khai báo nhưng trình biên dịch không hiểu. Để sửa những lỗi trên, ta cần thêm một số thư viện:

stdlib.h: Thư viện định nghĩa một số hàm như exit(), free(), malloc(), …

unistd.h: Thư viện định nghĩa một số hàm như fork(), pipe, …

sys/types.h: Thư viện định nghĩa một số kiểu biến như pid\_t, pthread\_t, …

sys/wait.h: Thư viện định nghĩa một hàm như wait(), …

Ảnh có chứa văn bản

Mô tả được tạo tự động

Thêm các thư viện ở trên vào chương trình và chạy lại

Ảnh có chứa văn bản

Mô tả được tạo tự động

Giải thích:

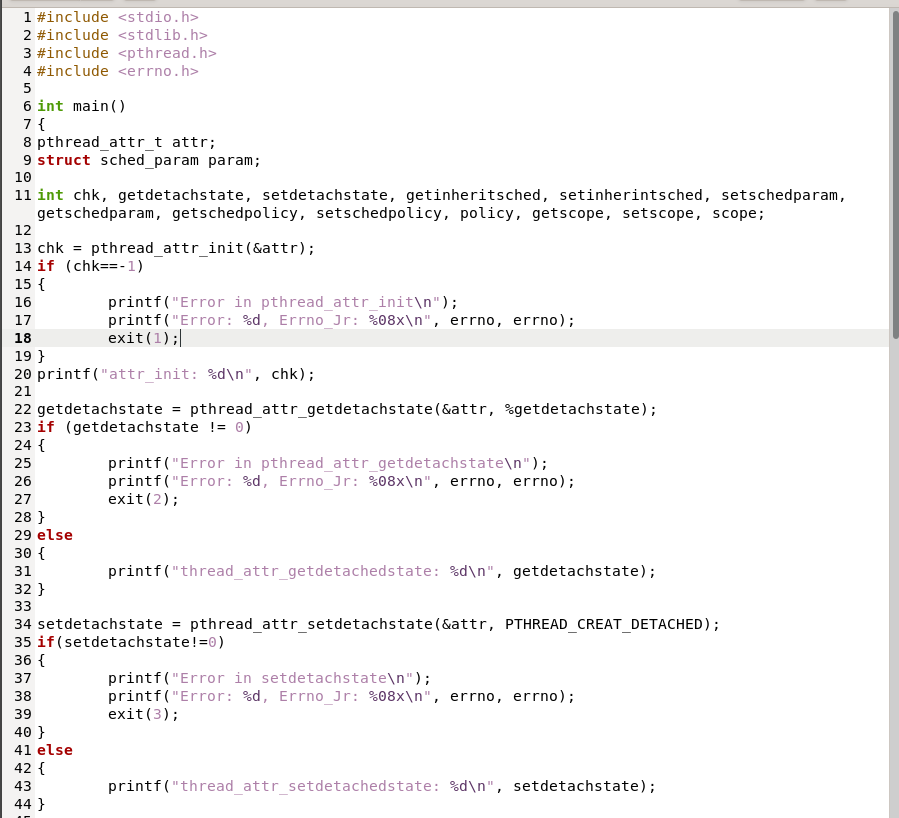
+pid = fork() tạo ra một tiến trình con với code hệt như tiến trình cha

+if(pid == 0) sẽ tiến hành thực thi các đoạn code ở tiến trình con, cụ thể là lệnh num\_coconuts = 42.

+Khi thực hiện lệnh exit(0), tiến trình con kết thúc, chương trình thực hiện tiến trình cha và biến num\_coconuts vẫn được mang giá trị 17 và thực hiện printf.

**3. Trong phần thực hành, các ví dụ chỉ sử dụng thuộc tính mặc định của pthread, hãy tìm hiểu POSIX thread và trình bày các hàm được sử dụng để làm thay đổi thuộc tính của pthread.**

|  |  |
| --- | --- |
| **Hàm** | **Ý nghĩa** |
| pthread\_attr\_init | Khởi tạo giá trị mặc định cho đối tượng |
| pthread\_attr\_destroy | Xóa bộ nhớ cấp phát khi khởi tạo |
| pthread\_attr\_getdetachstate | Truy xuất trạng thái khởi tạo của thread |
| pthread\_attr\_setdetachstate | Cập nhật lại ID và tài nguyên của thread |
| pthread\_attr\_getinheritsched | Truy xuất scheduling policy của thread |
| pthread\_attr\_setinheritsched | Cập nhật scheduling policy của thread |
| pthread\_attr\_getschedparam | Truy xuất scheduling parameter của thread |
| pthread\_attr\_setschedparam | Cập nhật lại scheduling parameter của thread |
| pthread\_attr\_getschedpoilicy | Truy xuất scheduling policy của thread |
| pthread\_attr\_setschedpoilicy | Cập nhật lại scheduling policy của thread |
| pthread\_attr\_getscope | Truy xuất phạm vi của scope |
| pthread\_attr\_setscope | Cập nhật lại phạm vi của scope |

****

**Ảnh có chứa bàn

Mô tả được tạo tự động**

**Ảnh có chứa văn bản

Mô tả được tạo tự động**

**Ảnh có chứa văn bản

Mô tả được tạo tự động**

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

Ảnh có chứa văn bản

Mô tả được tạo tự động

Ảnh có chứa văn bản

Mô tả được tạo tự động

Ảnh có chứa văn bản

Mô tả được tạo tự động