Tên: Lê Anh Thư

MSSV: 20521985

MÔN HỌC: HỆ ĐIỀU HÀNH

CÂU HỎI VÀ BÀI TẬP CHƯƠNG 1, 2, 3

CHƯƠNG 1

Câu 1 : Định nghĩa hệ điều hành?

- Hệ điều hành là chương trình trung gian giữa phần cứng máy tính và người sử dụng, có chức năng điều khiển và phối hợp việc sử dụng phần cứng và cung cấp các dịch vụ cơ bản cho các ứng dụng.

Câu 2 : Cấu trúc hệ thống máy tính gồm những phần nào?

- Phần cứng (hardware): Bao gồm các tài nguyên phần cứng của máy tính như CPU, bộ nhớ, các thiết bị I/O. Ví dụ: Chuột, bàn phím, màn hình, …
- Hệ điều hành (operating system): Phân phối tài nguyên, điều khiển và phối hợp các hoạt động của các chương trình trong hệ thống. Ví dụ một số loại HĐH: MS – DOS, Linux, Window
- Chương trình ứng dụng (application programs): Sử dụng hệ thống tài nguyên để giải quyết một bài toán tính toán nào đó của người sử dụng. Ví dụ: game
- Users (people, machines, other computers)

Câu 3 : Hệ điều hành có những chức năng gì?

- Phân chia thời gian xử lý và định thời CPU.
- Phối hợp và đồng bộ hoạt động giữa các processes (coordination & synchronization).
- Quản lý tài nguyên hệ thống (thiết bị I/O, bộ nhớ, file chứa dữ liệu,…).
- Thực hiện và kiểm soát access control, protection.
- Duy trì sự nhất quán (integrity) của hệ thống, kiểm soát lỗi và phục hồi hệ thống khi có lỗi (error recovery).
- Cung cấp giao diện làm việc cho users.

Câu 4 : Dưới góc độ hình thức xử lý, hệ điều hành chia thành những loại nào? Trong mỗi loại có những yêu cầu gì với hệ điều hành?

- Hệ thống xử lý theo lô

+ Hệ thống đơn chương (uniprograming OS):

- Tác vụ được thi hành tuần tự.
- Yêu cầu:

- Bộ giám sát thường trực
- CPU và các thao tác nhập xuất:
  - Xử lý offline
  - Đồng bộ hóa các thao tác bên ngoài – Spooling (Simultaneous Peripheral Operation On Line)

+ Hệ thống đa chương (multiprogramming OS):

- Nhiều công việc được nạp đồng thời vào bộ nhớ chính, tận dụng được thời gian rảnh của các tiến trình đang trong giai đoạn chờ thực thi.
- Yêu cầu:
  - Định thời công việc (job scheduling): chọn job trong job pool trên đĩa và nạp nó vào bộ nhớ để thực thi.
  - Quản lý bộ nhớ (memory management).
  - Định thời CPU (CPU scheduling).
  - Cấp phát tài nguyên (đĩa, máy in,…).
  - Bảo vệ.

+ Hệ thống chia sẻ thời gian:

- Là hệ thống đa nhiệm, lập lịch cho các tiến trình thực thi trên CPU.
- Yêu cầu:
  - Định thời công việc (job scheduling).
  - Quản lý bộ nhớ (memory management).
    - Virtual memory
  - Quản lý các quá trình (process management)
    - Định thời CPU
    - Đồng bộ các quá trình (synchronization)
    - Giao tiếp giữa các quá trình (process communication)
    - Tránh deadlock
  - Quản lý hệ thống file, hệ thống lưu trữ.
  - Cấp phát hợp lý các tài nguyên.
  - Bảo vệ (protection).

+Hệ thống song song:

- Nhiều CPU, chia sẻ computer bus, clock
- Ưu điểm:
    - Năng suất: càng nhiều CPU thì càng xử lý công việc nhanh.
    - Multiprocessor system ít tốn kém hơn multiple single-processor system.
    - Độ tin cậy: khi một processor hỏng thì công việc của nó được chia sẻ giữa các processor còn lại.
- Phân loại: Đa xử lý đối xứng và đa xử lý bất đối xứng.

+ Hệ thống phân tán:

- Mỗi processor có bộ nhớ riêng, giao tiếp với nhau qua các kênh nối như mạng, bus tốc độ cao nhưng người dùng chỉ thấy một hệ thống đơn nhất.
- Ưu điểm:
    - Chia sẻ tài nguyên (resource sharing)
    - Chia sẻ sức mạnh tính toán (computational sharing)
    - Độ tin cậy cao (high reliability)
    - Độ sẵn sàng cao (high availability): các dịch vụ của hệ thống được cung cấp liên tục cho dù một thành phần hardware trở nên hỏng.
- Các mô hình hệ thống song song: client-sever và peer-to-peer

+Hệ thống xử lý thời gian thực:

- Sử dụng trong các thiết bị chuyên dụng như điều khiển các thử nghiệm khoa học, điều khiển trong y khoa, dây chuyền công nghiệp, thiết bị gia dụng, quân sự.
- Ràng buộc về thời gian: hard và soft real-time.
    - Hard real-time.
        - Hạn chế (hoặc không có) bộ nhớ phụ, tất cả dữ liệu nằm trong bộ nhớ chính (RAM hoặc ROM).
        - Yêu cầu về thời gian đáp ứng/xử lý rất nghiêm ngặt, thường sử dụng trong điều khiển công nghiệp, robotics,…
    - Soft real-time: Thường được dùng trong lĩnh vực multimedia, virtual reality với yêu cầu mềm dẻo hơn về thời gian đáp ứng.

Câu 5 : Dưới góc độ loại máy tính, hệ điều hành chia thành những loại nào?
- Hệ điều hành dành cho máy MainFrame.
- Hệ điều hành dành cho máy Server.
- Hệ điều hành dành cho máy nhiều CPU.
- Hệ điều hành dành cho máy tính cá nhân (PC).
- Hệ điều hành dành cho máy PDA (Embedded OS - hệ điều hành nhúng).
- Hệ điều hành dành cho máy chuyên biệt.
- Hệ điều hành dành cho thẻ chíp (SmartCard).

Câu 6 : Nêu lịch sử phát triển của HĐH:
- Thế hệ 1 (1945 - 1955)
  - Thiết kế, xây dựng, lập trình, thao tác: do 1 nhóm người.
  - Lưu trên phiếu đục lỗ.
- Thế hệ 2 (1955 - 1965)
  - Xuất hiện sự phân công công việc
  - Hệ thống sử lý theo lô ra đời, lưu trên băng từ.
  - Hoạt động dưới sự điều khiển đặc biệt của 1 chương trình .
  - Chưa xuất hiện hệ điều hành.
- Thế hệ 3 (1965 - 1980)
  - Ra đời hệ điều hành, khái niệm đa chương.
  - HĐH chia sẻ thời gian như CTSS của MIT.
  - MULTICS, UNIX.
- Thế hệ 4 (1980)
  - Ra đời máy tính cá nhân, IBM PC.
  - HĐH MS-DOS, MacOS (Apple Macintosh), HĐH mạng,…

Câu 7: Những yêu cầu của hệ thống chia sẻ thời gian?

Yêu cầu đối với OS trong hệ thống time-sharing
- Định thời công việc (job scheduling)
- Quản lý bộ nhớ (memory management)
  - Virtual memory
- Quản lý các quá trình (process management)
  - Định thời CPU
  - Đồng bộ các quá trình (synchronization)
  - Giao tiếp giữa các quá trình (process communication)
  - Tránh deadlock

- Quản lý hệ thống file, hệ thống lưu trữ
- Cấp phát hợp lý các tài nguyên
- Bảo vệ (protection)

Câu 8: Đặc điểm của hệ thống đa chương?

Đặc điểm của hệ thống đa chương là:
- Nhiều công việc được nạp đồng thời vào bộ nhớ chính
- Khi một tiến trình thực hiện I/O, một tiến trình khác được thực thi
- Tận dụng được thời gian rảnh, tăng hiệu suất sử dụng CPU (CPU utilization)

CHƯƠNG 2

Câu 1 : Hệ điều hành bao gồm những thành phần nào? Cụ thể từng thành phần?
- Quản lý tiến trình
- Quản lý bộ nhớ chính
- Quản lý file
- Quản lý hệ thống I/O
- Quản lý hệ thống lưu trữ thứ cấp
- Hệ thống bảo vệ
- Hệ thống thông dịch lệnh

Câu 2 : Các cơ chế trao đổi thông tin giữa các tiến trình?
Chia sẻ bộ nhớ (Shared memory)
Chuyển thông điệp (Message passing)
Dùng tín hiệu

Pipe

Câu 3 : Cấu trúc hệ thống gồm những loại nào? Cho ví dụ từng loại (theo sách tham khảo)
Cấu trúc Monolithic - Original UNIX
Ví dụ: Linux, UNIX

Cấu trúc Layered Approach
Ví dụ: THE

Cấu trúc Microkernels
Ví dụ: Mach. QNX

Cấu trúc Modules
Ví dụ: Linux, Solaris

Cấu trúc Hybrid Systems
Ví dụ: Windows NT

Câu 4 : Chương trình hệ thống gồm những phần nào?
- Quản lý hệ thống file: như create, delete, rename, list
- Thông tin trạng thái: như date, time, dung lượng bộ nhớ trống
- Soạn thảo file: như file editor
- Hỗ trợ ngôn ngữ lập trình: như compiler, assembler, interpreter
- Nạp, thực thi, giúp tìm lỗi chương trình: như loader, debugger
- Giao tiếp: như email, talk, web browser

Câu 5 : Lời gọi hệ thống là gì và dùng để làm gì?
- Lời gọi hệ thống là việc một chương trình máy tính yêu cầu một dịch vụ từ nhân của hệ điều hành mà nó được thực thi.
- Tác dụng:
  - Dùng để giao tiếp giữa tiến trình và hệ điều hành
  - Cung cấp giao diện giữa tiến trình và hệ điều hành

Câu 6 : Hệ điều hành cung cấp những dịch vụ nào?
- Thực thi chương trình
- Thực hiện các thao tác I/O theo yêu cầu của chương trình
- Các thao tác trên hệ thống file
- Trao đổi thông tin giữa các tiến trình qua hai cách:
  - Chia sẻ bộ nhớ (Shared memory)
  - Chuyển thông điệp (Message passing)

- Phát hiện lỗi
- Ngoài ra còn các dịch vụ giúp tăng hiệu suất của hệ thống:
  - Cấp phát tài nguyên (resource allocation):
    - CPU, bộ nhớ chính, ổ đĩa,…
    - OS có các routine tương ứng.

+ Kế toán (accounting): Nhằm lưu vết user để tính phí hoặc đơn giản để thống kê.
+ Bảo vệ (protection)
  - Hai tiến trình khác nhau không được ảnh hưởng nhau
  - Kiểm soát được các truy xuất tài nguyên của hệ thống

+ An ninh (security): Chỉ các user được phép sử dụng hệ thống mới truy cập được tài nguyên của hệ thống (vd: thông qua username và password)

Câu 7 : Các khái niệm liên quan đến máy ảo?

- Máy ảo là phần mềm tạo ra môi trường giữa hệ nên máy tính và người dùng, người dùng có thể thực thi phần mềm trên máy ảo
- Ví dụ: Virtual Box, Parallels

CHƯƠNG 3

Câu 1: Một tiến trình chứa những thành phần gì?
Một tiến trình bao gồm :
Trong bộ nhớ (memory) :

- Text Section (Program code) : Chứa những đoạn mã chương trình đã được biên dịch bởi compiler.
- Data Section (khu vực dữ liệu) : Chứa các biến toàn cục (global variables) và các biến tĩnh (static variables)
- Heap : Dùng để lưu trữ các bộ nhớ được cấp phát động.
- Stack : Dùng để lưu trữ các biến cục bộ (local variables).

Câu 2*:* Tiến trình có những trạng thái nào? Cách tiến trình chuyển trạng thái?

Các trạng thái của tiến trình :

- new : tiến trình vừa được tạo.
- ready : tiến trình đã có đủ tài nguyên, đang chờ được cấp CPU để chạy.
- running : các lệnh của tiến trình đang được thực thi.
- waiting: tiến trình đợi I/O hoàn tất.
- terminated: tiến trình đã kết thúc (đã thực thi xong).

Cách tiến trình chuyển trạng thái :
Đầu tiên, khi vừa khởi tạo, tiến trình sẽ ở trạng thái là new.
Thông qua bộ định thời dài hạn Long-term scheduling (hay còn gọi là bộ định thời công việc – Job Scheduler), tiến trình của chúng ta từ new sẽ được sắp xếp vị trí để "chui" vào trong hàng đợi ready.
Ở một số hệ điều hành có thêm bộ định thời Medium-term Scheduler. Thông qua bộ định thời này, các tiến trình sẽ được swap-out (chuyển tiến trình từ bộ nhớ chính sang bộ nhớ phụ) và swap-in (chuyển tiến trình từ bộ nhớ phụ vào bộ nhớ).
Thông qua bộ định thời Short-term Scheduling (hay còn được gọi là Dispatcher), từ trạng thái ready tiến trình sẽ được sắp xếp để chuyển qua trạng thái running(là trạng thái chạy – hay trạng thái sử dụng CPU – của tiến trình).

Trong khi đang ở trạng thái running, có 3 trạng thái tiếp theo mà tiến trình có thể đạt được tiếp theo :

- waiting : Khi tiến trình đang chờ I/O (VD : Khi gọi hàm print(), scanf() trong C).
- ready : Khi tiến trình bị interrupt (bị ngắt, không cho chạy nữa) bởi Short-term Scheduler. Các lý do ngắt có thể là : Ngắt thời gian (Clock Interrupt), Ngắt ngoại vi (I/O Interrupt), Lời gọi hệ thống (Operating System Call), Signal.
- terminated : Khi ứng dụng thực thi xong : Khi gặp lệnh exit, khi thực thi lệnh cuối.

Trong khi ở trạng thái waiting, tiến trình sẽ chuyển sang trạng thái ready(vào hàng đợi ready) sau khi đã thực thi xong I/O.

Câu 3*:* Tại sao phải cộng tác giữa các tiến trình?

Cộng tác giữa các tiến trình để:

- Chia sẻ dữ liệu (information sharing).
- Tăng tốc độ tính toán (computational speedup).
    - Các mạng lưới máy tính sẽ hợp với nhau để tạo thành các cluster.
    - Nếu hệ thống có nhiều CPU, chia công việc tính toán thành nhiều công việc tính toán nhỏ chạy song song.
- Thực hiện một công việc chung.
    - Xây dựng một phần mềm phức tạp bằng cách chia thành các module/process hợp tác nhau.

Câu 4: PCB là gì? Dùng để làm gì?

Mỗi tiến trình trong hệ thống đều được cấp phát một Process Control Block (PCB). Là một trong các cấu trúc dữ liệu quan trọng nhất của hệ điều hành.
PCB chứa các thông tin liên quan đến process như :

- Trạng thái tiến trình (Process State) : new, ready, running,…
- Bộ đếm chương trình (Program Counter) : Chỉ đến địa chỉ của lệnh tiếp theo sẽ được thực thi cho tiến trình này.
- Các thanh ghi CPU (CPU Registers) : Phụ thuộc vào kiến trúc máy tính. Có thể kể đến vài loại như accumulators, index registers, stack pointers, general-purpose registers, condition-code information.
- Thông tin lập thời biểu CPU (CPU Scheduling Information) : Độ ưu tiên, con trỏ đến các hàng đợi, và các tham số của việc lập thời biểu.

- Thông tin quản lý bộ nhớ (Memory-Management Information) : Chứa page tables, segment tables, memory limits (giới hạn bộ nhớ).
- Thông tin trạng thái I/O (I/O status information) : Chứa danh sách các thiết bị I/O đã được cấp phát cho tiến trình, danh sách các file tiến trình đang mở,…
- Các thông tin quan trọng khác như : Lượng CPU, thời gian sử dụng,PID,…

Câu 5: Tiểu trình là gì?

Tiểu trình : Là một đơn vị cơ bản sử dụng CPU, gồm :

- Thread ID.
- PC (Program Counter).
- Registers.
- Stack.
- Chia sẻ chung code, data, resources (file).

Câu 6: Trình tự thực thi của tiến trình cha và tiến trình con?

Khi một tiến trình con được tạo ra từ tiến trình cha (từ câu lệnh fork()) thì tiến trình con sẽ được thực thi đến hết sau đó tiến trình cha mới thực thi câu lệnh tiếp theo.

Ngoài ra tiến trình cha còn có thể thực thi song song với tiến trình con.

Câu 7: (Bài tập mẫu) Cho đoạn chương trình sau:

```
int main (int argc, char** argv)
{
        int i = 2;
        while (i < =5)
        {
                i++;
                if (i % 2 == 0)
                {
                        printf (“Hello”);
                        printf (“Hi”);
                }
                else
                {
                        printf (“Bye”);
                }
        }
        exit (0);
}
```

Hỏi trong quá trình thực thi thì tiến trình khi chạy từ chương trình trên đã trải qua những trạng thái nào? Vẽ sơ đồ chuyển trạng thái trong quá trình thực thi?

Trả lời:

Trong quá trình thực thi thì tiến trình khi chạy từ chương trình trên đã trải qua những trạng thái như sau: new – ready – running – waiting – ready – running – waiting – ready – running – waiting – ready – running – waiting – ready – running – terminated

Câu 8: Cho đoạn chương trình sau:

```
/* test.c */
int main(int argc, char** argv)
{
        int a;
        for (int i = 1; i < 5; i++)
        {
                if (i % 2 == 0)
                        printf("Hello world\n");
                else a = 5*9;
        }
        exit(0);
}
```

Hỏi trong quá trình thực thi thì tiến trình khi chạy từ chương trình trên đã trải qua những trạng thái nào? Vẽ sơ đồ chuyển trạng thái trong quá trình thực thi?

Chương trình đã trải qua các trạng thái:

new-ready-running-waiting-ready-running-waiting-ready-running-terminated

Câu 9: Cho đoạn chương trình sau:

```
int main (int argc, char** argv)
{
        int i = 2;
        while (i < =5)
        {
                i++;
                if (i % 2 == 0)
                {
                        printf ("Hello");
                        printf ("Hi");
                }
                else
                {
```

```
                printf (“Bye”);
            }
        }
        exit (0);
}
```

Hỏi trong quá trình thực thi thì tiến trình khi chạy từ chương trình trên đã trải qua những trạng thái nào? Vẽ sơ đồ chuyển trạng thái trong quá trình thực thi?

Chương trình đã trải qua các trạng thái:

new-ready-running-waiting-ready-running-waiting-ready-running-waiting-ready-running-waiting-ready-running-terminated

Câu 10: Cho đoạn chương trình sau:

```
int main (int argc, char** argv)
{
        int a, b, i;
        for (i = 16, i >=6; i --)
        {
            if (i % 3 == 0)
            {
                printf (“Số %d chia hết cho 3”, i);
            }
            else
            {
                a = b + i;
            }
        }
        exit (0);
}
```

Hỏi trong quá trình thực thi thì tiến trình khi chạy từ chương trình trên đã trải qua những trạng thái nào? Vẽ sơ đồ chuyển trạng thái trong quá trình thực thi?

Chương trình đã trải qua các trạng thái:

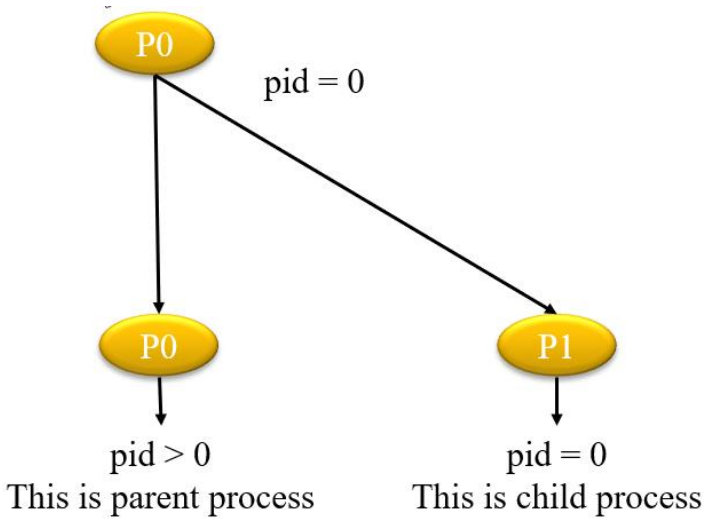
new-ready-running-waiting-ready-running-waiting-ready-running-waiting-ready-running-waiting-ready-running-terminated

Câu 11: (Bài tập mẫu) Cho đoạn code sau, hỏi khi chạy, bao nhiêu process được sinh ra và chương trình sẽ in ra những gì? Vẽ cây tiến trình khi thực thi đoạn chương trình sau

```
#include <stdio.h>
#include <unistd.h>
int main (int argc, char *argv[]){
        int     pid;
        /* create a new process */
        pid = fork();
        if (pid > 0){
                printf("This is parent process");
                wait(NULL);
                exit(0);}
        else if (pid == 0)    {
                printf("This is child process");
                execlp("/bin/ls", "ls", NULL);
                exit(0);}
        else {   // pid < 0
                printf("Fork error\n");
                exit(-1);
        }
}
```

Trả lời:

Khi chạy đoạn chương trình trên, khi chạy hết sẽ có 2 process được sinh ra bao gồm 1 tiến trình cha và 1 tiền trình con. Theo chương trình trên thì tiến trình cha sẽ in ra dòng chữ "This is parent process"; và tiến trình con sẽ in ra dòng chữ "This is child process". Cây tiến trình khi thực thi đoạn chương trình trên như sau:



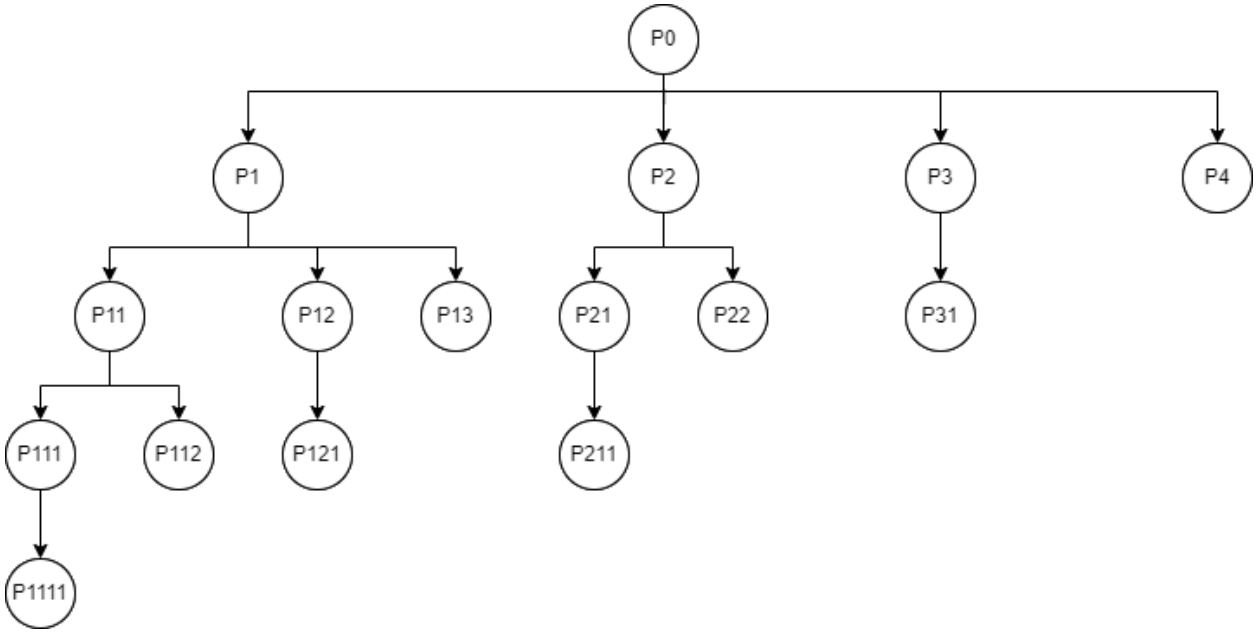
Câu 12: Cho đoạn code sau, hỏi khi chạy, bao nhiêu process (kể cả cha) được sinh ra? Vẽ cây tiến trình khi thực thi đoạn chương trình sau

```
int main()
```

```
{
	fork();
	fork();
	fork();
	fork();
	return 0;
}
```

Sau khi chạy hết chương trình trên thì có 16 tiến trình được sinh ra bao gồm 1 tiến trình cha và 15 tiến trình con

Cây tiến trình:



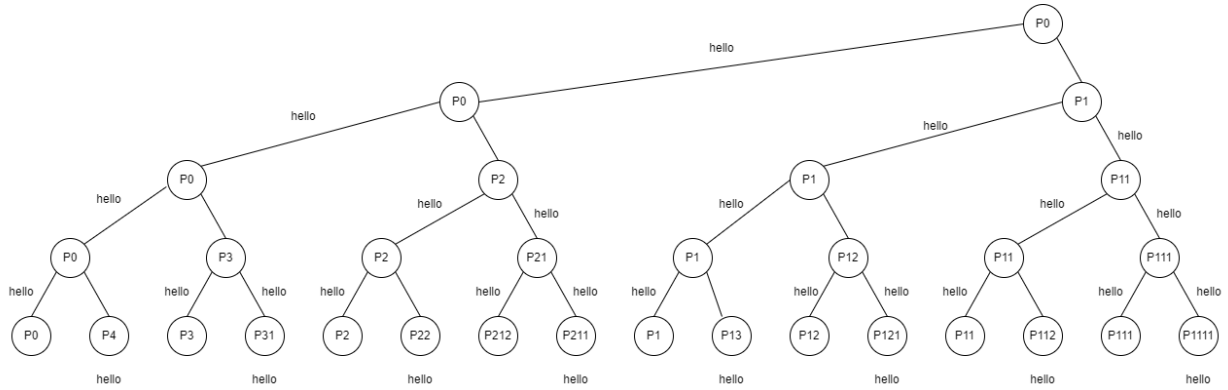
Câu 13: Cho đoạn code sau, hỏi khi chạy thì tiến trình được tạo ra từ chương trình trên sẽ in ra màn hình những gì? Vẽ cây tiến trình và những từ được in ra khi thực thi đoạn chương trình sau?

```
#include <stdio.h>
#include <unistd.h>
int main()
{
	int i;
	for (i = 0; i < 4; i++)
	{
		fork();
		printf("hello\n");
	}
	return 0;
}
```

Chương trình sẽ in ra: (hello 30 lần)

hello
hello
hello
hello
hello
hello
hello
hello
hello
hello
hello
hello
hello
hello
hello
hello
hello
hello
hello
hello
hello
hello
hello
hello
hello
hello
hello
hello
hello
hello
hello
hello

Cây tiến trình:



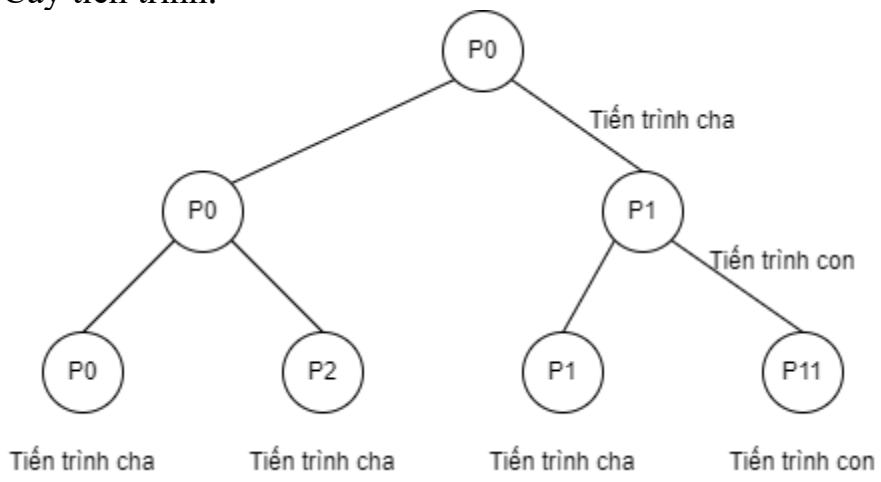
Câu 14: Cho đoạn code sau, hỏi khi chạy thì tiến trình được tạo ra từ chương trình trên sẽ in ra màn hình những gì? Vẽ cây tiến trình và những từ được in ra khi thực thi đoạn chương trình sau?

```
int main (int argc, char **argv)
{
        int pid;
        printf("Tiến trình cha \n");
        pid = fork();
        if (pid > 0)
        {
                fork();
                printf("Tiến trình cha \n");
        }
        else
        {
                printf("Tiến trình con \n");
                if(fork() > 0 )
                        printf("Tiến trình cha \n");
                else
                        printf("Tiến trình con \n");
        }
}
```

Chương trình sẽ in ra:

Tiến trình cha
Tiến trình con
Tiến trình con
Tiến trình cha
Tiến trình cha
Tiến trình cha

Cây tiến trình:



Câu 15: Cho đoạn code chương trình sau:

```
if (fork() == 0)
{
   a = a + 5;
   printf("%d,%d\n", a, &a);
}
else
{
    a = a –5;
    printf("%d, %d\n", a, &a);
}
```

Giả sử u, v là các giá trị được in ra bởi process cha, và x, y là các giá trị được in ra bởi process con. Tìm mối quan hệ giữa u, v và x, y?

Ta có:

- Process cha:

  u = a – 5

  v = &a

- Process con:

  x = a + 5

  y = &a

Địa chỉ vật lý của biến a trong process cha và process con phải khác nhau. Nhưng chương trình lại truy cập bộ nhớ ảo (giả sử chương trình chạy trên hệ điều hành sử dụng bộ nhớ ảo). Nên tiến trình con sẽ sao chép tiến trình tiến trình và địa chỉ ảo trong biến a không thay đổi trong tiến trình con(vậy địa chỉ biến a của tiến trình cha và con là giống nhau).

Suy ra mối quan hệ giữa u, v và x, y:

x = u + 10

v = y