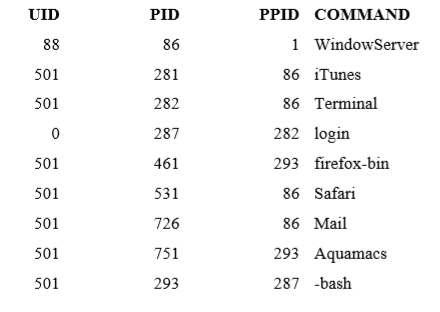
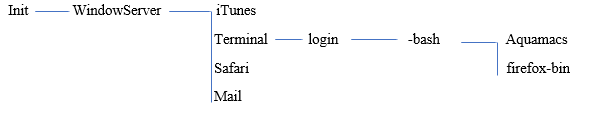
Lê Trường Danh

16520177

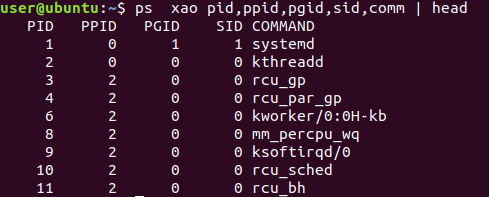
# BÀI THỰC HÀNH LAP 3

## 1. a. Mối quan hệ cha-con giữa các tiến trình a. Vẽ cây quan hệ parent-child của các tiến trình bên dưới:





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



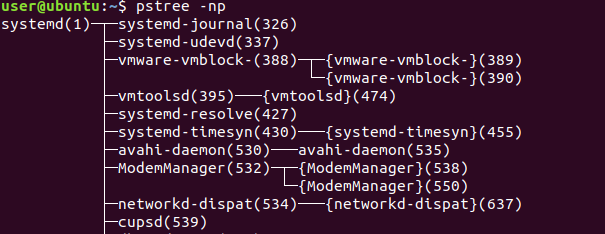
Sử dụng lệnh:

*$ ps -o ppid= -p x*

Với x là pid của tiến trình cần tìm



### c. 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 lệnh

*$ ps -s x*

Với x là pid tiến trình con cần tìm



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

#include<stdio.h>

int main(){

pid\_t pid;

int num\_coconuts = 17;

pid = fork();

if(pid == 0) {

num\_coconuts = 42;

exit(0);

} else {

wait(NULL); /\*wait until the child terminates \*/

}

printf("I see %d coconuts!\n", num\_coconuts);

exit(0);

}



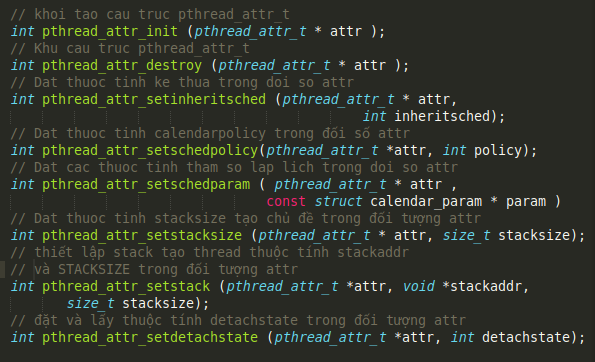
In ra màn hình: I see 17 coconuts!

Giải thích: Vì hai tiến trình cha và con không sử dụng chung dữ liệu, nên khi tiến trình con thay đổi num\_coconuts và bị tắt thì ở tiến trình cha num\_coconuts vẫn là 17.

### 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 tất cả các hàm được sử dụng để làm thay đổi thuộc tính của pthread, sau đó viết các chương trình minh họa tác động của các thuộc tính này và chú thích đầy đủ cách sử dụng hàm này trong chương trình. (Gợi ý các hàm liên quan đến thuộc tính của pthread đều bắt đầu bởi: pthread\_attr\_\*)







**#pthread\_attr\_init**

**#pthread\_attr\_destroy**

#include "pthread.h"

#include <stdio.h>

#include <stdlib.h>

#define NUM\_THREADS 2

void \*thread\_print(void \*threadid){

long tid;

tid = (long)threadid;

printf("Hello, thread #%ld\n",tid);

pthread\_exit(NULL);

}

void main(){

int check;

long i;

pthread\_t threads[NUM\_THREADS];

pthread\_attr\_t attr;

//Khoi tao doi tuong thuoc tinh luong

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

pthread\_attr\_init(&attr);

//Tao pthread

for(i=0;i<NUM\_THREADS;i++){

check = pthread\_create(

&threads[i],

&attr,

thread\_print,

NULL

);

if(check != 0){

printf("Error. Can't create Thread #%ld", i);

exit(-1);

}

}

//Huy doi tuong thuoc tinh luong

//pthread\_attr\_destroy(pthread\_attr\_t \*attr);

pthread\_attr\_destroy(&attr);

//Dung pthread

pthread\_exit(NULL);

}

**#pthread\_attr\_setinheritsched**

**# pthread\_attr\_setschedpolicy**

#define \_MULTI\_THREADED

#include <pthread.h>

#include <stdio.h>

#include <except.h>

#include "check.h"

int main(int argc, char \*\*argv)

{

pthread\_t thread;

pthread\_attr\_t attr;

pthread\_attr\_init(&attr);

//#include <pthread.h>

//#include <sched.h>

//int pthread\_attr\_setinheritsched(pthread\_attr\_t \*attr,int \*inheritsched);

pthread\_attr\_setinheritsched(&attr, PTHREAD\_INHERIT\_SCHED);

pthread\_attr\_setschedpolicy (&attr, SCHED\_RR);

pthread\_attr\_destroy(&attr);

return 0;

}

**pthread\_attr\_setdetachstate**

#include <stdlib.h>

#include <stdio.h>

#include <pthread.h>

void \*\*stat;

void \*thread\_print(void \*arg)

{

printf("Hello\n");

pthread\_exit((void \*)0);

}

int main()

{

int ds, rc;

pthread\_attr\_t attr;

pthread\_t tid;

//Khoi tao doi tuong thuoc tinh luong

rc = pthread\_attr\_init(&attr);

if (rc != 0) {

perror("error");

exit(-1);

}

ds = 0;

//Dat thuoc tinh tach roi cho attrs

rc = pthread\_attr\_setdetachstate(

&attr,

&ds);

if (rc != 0) {

perror("error");

exit(-1);

}

rc = pthread\_create(&tid,

&attr,

thread\_print,

NULL);

if (rc != 0) {

perror("error");

exit(-1);

}

rc = pthread\_join(tid, stat);

exit(0);

}

**pthread\_attr\_setschedparam**

#include <stdio.h>

#include <stdlib.h>

#include <pthread.h>

int main(void)

{

pthread\_attr\_t attr;

struct sched\_param param;

param.sched\_priority = 999;

if (pthread\_attr\_init(&attr) == -1) {

perror("error");

exit(1);

}

//Thiet lap thuoc tinh cua doi tuong luong attr den cac gia tri duoc

//chi dinh duoc tro boi param

//pthread\_attr\_setschedparam (pthread\_attr\_t \* attr , const struct calendar\_param \* param );

pthread\_attr\_setschedparam(&attr, &param);

printf("Set schedpriority to %d\n", param.sched\_priority);

pthread\_attr\_destroy(&attr);

exit(0);

}

**pthread\_attr\_setstacksize**

#include <pthread.h>

#include <stdio.h>

#include <limits.h>

void main(){

pthread\_attr\_t attr;

size\_t size;

int ret;

//PTHREAD\_STACK\_MIN luong khong gian ngan xep can

//de bat dau cuoi

size = (PTHREAD\_STACK\_MIN + 0x4000);

//Dat kich thuoc moi

//pthread\_attr\_setstacksize tra ve 0 sau khi hoan thanh thanh cong

ret = pthread\_attr\_setstacksize(&attr, size);

if(ret != 0){

printf("Error\n");

}

}

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

### a. In ra dòng chữ: “Welcome to IT007, I am <your\_Student\_ID>!”

### b. Mở tệp abcd.txt bằng vim editor

### c. Tắt vim editor khi người dùng nhấn CRTL+C

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

#include <stdio.h>

#include <stdlib.h>

#include <signal.h>

#include <unistd.h>

int loop\_forever = 1;

void kill\_vim(){

system("pkill vim");

}

void on\_sigint(){

printf("You are pressed CTRL+C! Goodbye!\n");

loop\_forever = 0;

}

int main(){

loop\_forever = 1;

printf("Welcom to IT007, I am <16520177>\n");

system("vim abc");

signal(SIGINT, kill\_vim);

signal(SIGINT, on\_sigint);

while(loop\_forever){}

return 0;

}