OS - Class question 2

1. Kể tên 1 hệ thống chỉ cho phép một tiến trình thực thi tại một thời điểm thời gian? 1 hệ thống cho phép nhiều tiến trình thực thi đồng thời?

-Hệ thống chỉ cho phép 1 tiến trình thực thi tại 1 thời điểm là quản lí tiến trình - Hệ thống cho phép nhiều tiến trình thực thi đồng thời là quản lí bộ nhớ chính

1. Trong Unix, lời gọi hệ thống nào được sử dụng để tạo tiến trình.
2. Một tiến trình sẽ bị kết thúc trong các trường hợp nào?
3. Trạng thái Sẵn sàng của tiến trình là gì?
4. Các tiến trình liên lạc với nhau để làm gì?
5. Khi mở một tab mới trên trình duyệt chrome, trình duyệt này có tạo ra tiến trình mới không?
6. Tiến trình và luồng khác nhau gì?
7. Tại sao các hệ điều hành hiện đại hỗ trợ môi trường đa nhiệm ?
8. Sự khác biệt, mối quan hệ giữa tiến trình và tiểu trình ?
9. Tìm một số ứng dụng thích hợp với mô hình đa tiến trình; và một số ứng dụng thích hợp với mô hình đa tiểu trình.
10. Lệnh sau tạo ra bao nhiêu tiến trình. Hãy vẽ cây tiến trình tạo ra bởi chương trình sau

include <stdio.h>

include <unistd.h>

int main()

{

fork();wait(NULL)

fork();wait(NULL)

fork();wait(NULL)

fork();wait(NULL)

return 0;

}

1. Cho chương trình sau, xác định giá trị của pid tại các dòng A, B, C, và D. Giả sủ rằng pid của tiến trình cha la 1500, pid của tiến trình con la 1506

#include <sys/types.h>

#include <stdio.h>

#include <unistd.h>

int main()

{

pid\_t pid, pid1;

//fork a child process

pid = fork();

if (pid< 0) //error occurred

{

fprintf(stderr, "Fork Failed");

return 1;

}

else if (pid == 0) //child process

{

pid1 = getpid();

printf("child: pid = %d", pid); //line A

printf("child: pid1 = %d", pid1); //line B

}

else //parent process

{

pid1 = getpid();

printf("parent: pid = %d", pid); //line C

printf("parent: pid1 = %d", pid1); //line D

wait(NULL);

}

return 0;

}

1. Xác định giá trị được in ra tại dòng A

#include <sys/types.h>

#include <stdio.h>

#include <unistd.h>

int value = 5;

int main()

{

pid\_t pid;

pid = fork();

if (pid == 0) //child process

{

value += 15;

return 0;

}

else (pid > 0) //parent process

{

wait(NULL);

printf("Parent: value = %d", value); //Line A

return 0;

}

}

1. Viết chương trình “prog” tạo ra các cây tiến trình sau
2. Cây tiến trình A



1. Cây tiến trình B



Lệnh nào trong các lệnh sau nên là lệnh đặc quyền?

Đặt giá trị cho timer

Đọc đồng hồ

Xóa bộ nhớ

Đưa ra lệnh tạo bẫy

Tắt các ngắt

Thay đổi các mục trong bảng trạng thái thiết bị

Chuyển từ chế độ người dùng sang chế độ hạt nhân

Truy nhập thiết bị I/O

1. Xét tập các tiến trình sau (với thời gian yêu cầu CPU và độ ưu tiên kèm theo) :

|  |  |  |  |
| --- | --- | --- | --- |
| Tiến trình | Thời điểm vào RL | Thời gian CPU | Độ ưu tiên |
| P1 | 0 | 10 | 3 |
| P2 | 1 | 1 | 1 |
| P3 | 2 | 2 | 3 |
| P4 | 3 | 1 | 4 |
| P5 | 4 | 5 | 2 |

Giả sử các tiến trình cùng được đưa vào hệ thống tại thời điểm 0

1. Cho biết kết quả điều phối hoạt động của các tiến trình trên theo thuật toán FIFO; SJF; điều phối theo độ ưu tiên độc quyền (độ ưu tiên 1 > 2 > ...); và RR (quantum=2).
2. Cho biết thời gian lưu lại trong hệ thống (turnaround time) của từng tiến trình trong từng thuật toán điều phối ở câu a.
3. Cho biết thời gian chờ trong hệ thống (waiting time) của từng tiến trình trong từng thuật toán điều phối ở câu a.
4. Thuật toán điều phối nào trong các thuật toán ở câu a cho thời gian chờ trung bình là cực tiểu ?
5. Giả sử có các tiến trình sau trong hệ thống :

|  |  |  |
| --- | --- | --- |
| Tiến trình | Thời điểm vào RL | Thời gian CPU |
| P1 | 0.0 | 8 |
| P2 | 0.4 | 4 |
| P3 | 1.0 | 1 |

Sử dụng nguyên tắc điều phối độc quyền và các thông tin có được tại thời điểm ra quyết định để trả lời các câu hỏi sau đây :

1. Cho biết thời gian lưu lại trung bình trong hệ thống (turnaround time) của các tiến trình trong thuật toán điều phối FIFO.
2. Cho biết thời gian lưu lại trung bình trong hệ thống (turnaround time) của các tiến trình trong thuật toán điều phối SJF.
3. Thuật toán SJF dự định cải tiến sự thực hiện của hệ thống , nhưng lưu ý chúng ta phải chọn điều phối P1 tại thời điểm 0 vì không biết rằng sẽ có hai tiến trình ngắn hơn vào hệ thống sau đó . Thử tính thời gian lưu lại trung bình trong hệ thống nếu để CPU nhàn rỗi trong 1 đơn vị thời gian đầu tiên và sau đó sử dụng SJF để điều phối. Lưu ý P1 và P2 sẽ phải chờ trong suốt thời gian nhàn rỗi này, do vậy thời gian chờ của chúng tăng lên. Thuật toán điều phối này được biết đến như điều phối dựa trên thông tin về tương lai.
4. Phân biệt sự khác nhau trong cách tiếp cận để ưu tiên cho tiến trình ngắn trong các thuật toán điều phối sau :
5. FIFO.
6. RR
7. Điều phối với độ ưu tiên đa cấp
8. Giả sử một hệ điều hành áp dụng giải thuật điều phối multilevel feedback với 5 mức ưu tiên (giảm dần). Thời lượng quantum dành cho hàng đợi cấp 1 là 0,5s. Mỗi hàng đợi cấp thấp hơn sẽ có thời lượng quantum dài gấp đôi hàng đợi ứng với mức ưu tiên cao hơn nó. Một tiến trình khi vào hệ thống sẽ được đưa vào hàng đợi mức cao nhất, và chuyển dần xuống các hàng đợi bên dưới sau mỗi lượt sử dụng CPU. Một tiến trình chỉ có thể bị thu hồi CPU khi đã sử dụng hết thời lượng quantum dành cho nó. Hệ thống có thể thực hiện các tác vụ xử lý theo lô hoặc tương tác, và mỗi tác vụ lại có thể hướng xử lý hay hướng nhập xuất.
9. Giải thích tại sao hệ thống này hoạt động không hiệu quả ?
10. Cần phải thay đổi (tối thiểu) như thế nào để hệ thống điều phối các tác vụ với những bản chất khác biệt như thế tốt hơn ?
11. Phân biệt Concurrency (đồng thời) và parallelism (song song).
12. Xem xét giải thuật Banker với ma trận yêu cầu cấp phát như sau:

|  |  |  |  |
| --- | --- | --- | --- |
| **Processes** | **Allocation** **A B C** | **Max** **A B C** | **Available** **A B C** |
| **P0** | **1 1 2** | **4 3 3** | **2 1 0** |
| **P1** | **2 1 2** | **3 2 2** |  |
| **P2** | **4 0 1** | **9 0 2** |  |
| **P3** | **0 2 0** | **7 5 3** |  |
| **P4** | **1 1 2** | **1 1 2** |  |

1. Tính ma trận Need?

2. Kiểm tra xem hệ thống có ở trạng thái an toàn hay không?

3. Xác định tổng tổng của từng loại tài nguyên?