MÔN HỌC: HỆ ĐIỀU HÀNH
CÂU HỎI VÀ BÀI TẬP CHƯƠNG 4

1. Tại sao phải định thời? Có những loại bộ định thời nào?

- Trong các hệ thống đa nhiệm (multitasking), đơn vị xử lý

+ Cho phép thực thi đồng thời nhiều chương trình để làm tăng hiệu suất hệ thống (Cho phép nhiều chương trình được nạp vào bộ nhớ).

+ Tại mỗi thời điểm, chỉ có một tiến trình được thực thi.

+ Cần phải giải quyết vấn đề phân chia, lựa chọn tiến trình thực thi để đạt được hiệu quả cao nhất.

+ Cần có những phương pháp chọn lựa phù hợp.

=> *Mục đích của định thời (scheduling) là lựa chọn tiến trình phù hợp để được thực thi sao cho đạt*

được hiệu quả cao nhất.

- Các loại định thời:

+ Định thời dài (Long-term scheduling)

+ Định thời vừa (Medium-term scheduling)

+ Định thời ngắn (Short-term scheduling)

2. Định thời CPU là gì? Bộ định thời nào chịu trách nhiệm thực hiện việc này?

- Định thời CPU, còn được gọi là lập lịch CPU, là một quá trình quan trọng trong hệ điều hành đa chương trình, giúp máy tính hoạt động hiệu quả hơn. Với định thời CPU, quá trình nào sẽ được thực thi tiếp theo được xác định, giúp tối ưu hóa sử dụng tài nguyên CPU.

- Bộ định thời ngắn hạn (short-term scheduler) chịu trách nhiệm thực hiện việc định thời CPU.

3. Phí tổn gây ra khi định thời là gì?

- Phí tổn gây ra khi định thời liên quan đến việc chuyển đổi giữa các tiến trình trong hệ điều hành. Khi một tiến trình được chọn để thực thi, hệ điều hành phải lưu trữ trạng thái hiện tại của tiến trình đang chạy và khôi phục trạng thái của tiến trình mới. Quá trình này gọi là context switch.

4. Trình bày các tiêu chuẩn định thời CPU?

- Hướng người dùng (user-oriented):

+ Thời gian đáp ứng (Response time): khoảng thời gian từ lúc tiến trình gửi yêu cầu thực thi đến khi yêu cầu được đáp ứng lần đầu tiên (trong các hệ thống time-sharing, interactive system) → cực tiểu

+ Thời gian hoàn thành (Turnaround time): khoảng thời gian từ lúc một tiến trình được nạp vào hệ thống đến khi tiến trình đó kết thúc → cực tiểu

+ Thời gian đợi (Waiting time): tổng thời gian một tiến trình đợi trong ready queue

→ cực tiểu

- Hướng hệ thống (System oriented):

+ Hiệu năng sử dụng CPU (processor utilization): định thời sao cho CPU

càng bận càng tốt → cực đại

+ Tính công bằng (fairness): tất cả tiến trình phải được đối xử như nhau.

+ Thông lượng (throughput): số tiến trình hoàn tất công việc trong một đơn

vị thời gian → cực đại

5. Kể tên các giải thuật định thời CPU?

- Các giải thuật định thời:

+ First-Come, First-Served (FCFS)
+ Shortest Job First (SJF)
+ Shortest Remaining Time First (SRTF)
+ Round-Robin (RR)
+ Priority Scheduling
+ Highest Response Ratio Next (HRRN)
+ Multilevel Queue
+ Multilevel Feedback Queue

6. Mô tả và nêu ưu điểm, nhược điểm của từng giải thuật định thời sau: FCFS, SJF, SRTF, RR, Priority Scheduling, HRRN, MQ, MFQ.

Tên giải thuật	Ưu điểm	Nhược điểm
FCFS (First-Come, First-Served)	- Đơn giản và dễ triển khai. - Không có độ ưu tiên, công bằng về mặt xử lý.	- Không xem xét thời gian xử lý. - Có thể dẫn đến hiện tượng "*head starvation*".
SJF (Shortest Job First)	- Thời gian chờ trung bình thấp. - Tối ưu hóa thời gian hoàn thành.	- Không công bằng đối với các công việc dài hơn. - Có thể xảy ra hiện tượng "*starvation*".
SRTF (Shortest Remaining Time First)	- Đảm bảo tối ưu hóa thời gian hoàn thành. - Thời gian chờ trung bình thấp.	- Cần phải lập lịch thực hiện liên tục để xem xét thời gian còn lại. - Có thể xảy ra sự "*priority inversion*" cho các công việc nhỏ.
RR (Round-Robin)	- Công bằng, không ưu tiên bất kỳ công việc nào. - Đơn giản và dễ triển khai.	- Hiệu suất giảm khi *time slice* quá lớn. - Đòi hỏi lập lịch nhiều lần nếu có nhiều công việc ngắn.
Priority Scheduling	- Quản lý theo mức độ quan trọng của công việc. - Có thể ưu tiên các công việc quan trọng hơn.	- Có thể dẫn đến hiện tượng "*priority inversion*" cho công việc ưu tiên thấp. - Cần đánh giá và cập nhật thường xuyên mức độ ưu tiên.
HRRN (Highest Response Ratio Next)	- Cân bằng giữa SJF và FCFS. - Tránh hiện tượng "*head starvation*" và "*priority inversion*".	- Cần tính toán tỷ lệ phản hồi cho mỗi công việc. - Yêu cầu tính toán chi tiết và đánh giá định kỳ.
MQ (Multilevel Queue Scheduling)	- Quản lý tốt các công việc khác nhau. - Ưu tiên các công việc theo mức độ quan trọng.	- Có thể dẫn đến sự " *starvation* " cho các hàng đợi ưu tiên thấp. - Đòi hỏi cơ chế xử lý đa mức phức tạp.

MFQ (Multilevel Feedback Queue)	- Linh hoạt và hiệu quả trong việc quản lý các loại công việc khác nhau. - Độ ưu tiên được cập nhật linh hoạt.	- Đòi hỏi nhiều lần đánh giá và lập lịch. - Cơ chế phân phối tài nguyên phức tạp.

7. Đặc điểm của định thời trên hệ thống có nhiều bộ xử lý? Khi nào cần phải thực hiện cân bằng tải?

- Định thời trên hệ thống có nhiều bộ xử lý:

+ Định thời CPU trở nên phức tạp hơn khi hệ thống có nhiều bộ xử lý.

+ Khái niệm đa bộ xử lý có thể là một trong các dạng sau:

• CPU có nhiều lõi vật lý (Multicore CPUs)

• CPU có nhiều luồng xử lý trên một lõi (Multithreaded cores)

• Hệ thống NUMA (non-uniform memory access)

• Đa xử lý không đồng nhất (Heterogeneous multiprocessing)

+ Có hai cách tiếp cận phổ biến: đa xử lý bất đối xứng (asymmetric multiprocessing) và đa xử lý đối xứng (symmetric multiprocessing - SMP)

- Cần thực hiện cân bằng tải khi:

+ Một bộ xử lý có quá nhiều tải, trong khi các bộ xử lý khác rỗi => Cần đảm bảo các bộ xử lý đều được sử dụng hiệu quả.

+ Mục tiêu của cân bằng tải là phân phối khối lượng công việc (workload) đều nhau cho các CPU.

+ Có hai cách cân bằng tải:

▪ *Push migration*: Một tác vụ đặc biệt sẽ kiểm tra định kỳ tải của từng CPU. Nếu tình trạng quá tải xuất hiện, hệ thống sẽ di chuyển (đẩy) tác vụ từ CPU bị quá tải sang các CPU khác.

▪ *Pull migration:* CPU rỗi kéo (pull) tác vụ đang chờ từ CPU bận

8 Đặc điểm định thời theo thời gian thực?

- Có nhiều thách thức do yêu cầu về tính chất thời gian thực.

- Có 2 dạng hệ thống thời gian thực:

+ Soft real-time systems: Các tác vụ quan trọng sẽ được cấp độ ưu tiên lớn nhất, nhưng không đảm bảo bất cứ điều gì khác.

+ Hard real-time systems: Tác vụ phải hoàn thành trong deadline của nó.

9. Mô tả các đặc điểm cơ bản của bộ định thời CFS trên Linux?

- Nhân Linux từ 2.6.23 sử dụng bộ định thời CFS (Completely Fair Scheduler)

- Định thời theo lớp:

+ Mỗi lớp được gán một độ ưu tiên cụ thể.
+ Bộ định thời chọn tác vụ có độ ưu tiên cao nhất trong lớp có độ ưu tiên cao nhất.
+ Thời gian sử dụng CPU của mỗi tác vụ không dựa trên quantum time cố định mà dựa trên tỷ lệ giờ CPU.
+ Nhân Linux cài đặt sẵn 2 lớp: default và real-time. Các lớp khác có thể được thêm vào.

- Thời gian sử dụng CPU:

+ Được tính dựa trên giá trị nice được gán cho mỗi tác vụ, có giá trị từ -20 đến 19.
+ Giá trị thấp hơn có độ ưu tiên cao hơn.
+ Target latency – khoảng thời gian mà một tiến trình cần được chạy ít nhất một lần.
+ Target latency có thể tăng lên nếu số lượng tiến trình tăng lên.

- CFS xác định tác vụ được thực thi kế tiếp qua virtual run time:

+ Mỗi tác vụ có giá trị virtual run time riêng, được kết hợp với một hệ số đặc biệt dựa trên độ ưu tiên.
+ Các tiến trình có độ ưu tiên bình thường có virtual run time tương đương với thời gian chạy thực tế.
+ Chọn tiến trình có virtual run time nhỏ nhất để thực thi tiếp.

10. Mô tả các đặc điểm cơ bản của định thời trên Windows?

- Định thời theo độ ưu tiên với chế độ trưng dụng.
- Tác vụ có độ ưu tiên cao nhất luôn được chạy tiếp.
- Tiến trình sẽ được thực thi cho đến khi (1) block bởi system call, (2) hết quantum time, (3) bị thay thế bởi một tiến trình khác có độ ưu tiên cao hơn.
- Sử dụng 32 độ ưu tiên, được chia thành 2 lớp: variable (1-15) và real-time (16-31). Độ ưu tiên 0 dành cho quản lý bộ nhớ.
- Mỗi độ ưu tiên có hàng đợi riêng.
- Idle thread được chạy nếu không có bất cứ tác vụ nào trong hàng đợi.

- Các hàm thư viện Windows API cung cấp cho tiến trình các lớp ưu tiên sau:

  REALTIME_PRIORITY_CLASS, HIGH_PRIORITY_CLASS,

  ABOVE_NORMAL_PRIORITY_CLASS,NORMAL_PRIORITY_CLASS,

  BELOW_NORMAL_PRIORITY_CLASS, IDLE_PRIORITY_CLASS.

- Tiến trình có thể có các độ ưu tiên tương đối sau:

  TIME_CRITICAL, HIGHEST, ABOVE_NORMAL, NORMAL,

  BELOW_NORMAL, LOWEST, IDLE

- Lớp ưu tiên và độ ưu tiên tương đối có thể kết hợp để xác định giá trị ưu tiên.
- Độ ưu tiên cơ sở (lúc khởi tạo) là NORMAL bên trong lớp.
- Khi hết quantum, độ ưu tiên có thể giảm nhưng không nhỏ hơn độ ưu tiên cơ sở.
- Windows 7 có thêm user-mode scheduling (UMS):
  + Ứng dụng tạo và quản lý tiểu trình độc lập với nhân.
  + Hiệu quả hơn trong trường hợp có nhiều tiểu trình.
  + Định thời UMS được thực hiện với sự hỗ trợ của các thư viện như C++ Concurrent Runtime (ConcRT).

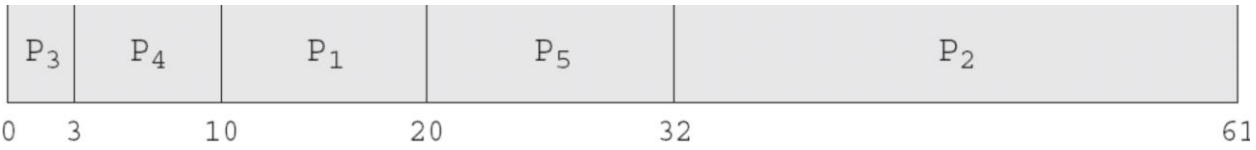
11. (Bài tập mẫu) Cho các tiến trình với thông tin ở bảng bên dưới. Biết rằng tất cả các tiến trình đều đến ở thời điểm 0 theo thứ tự từ P1 đến P5. Vẽ giản đồ Gantt, tính thời gian đợi trung bình và thời gian lưu lại trong hệ thống (turnaround time) trung bình cho các giải thuật sau:

Process	Burst Time
P1	10
P2	29
P3	3
P4	7
P5	12

a. FCFS

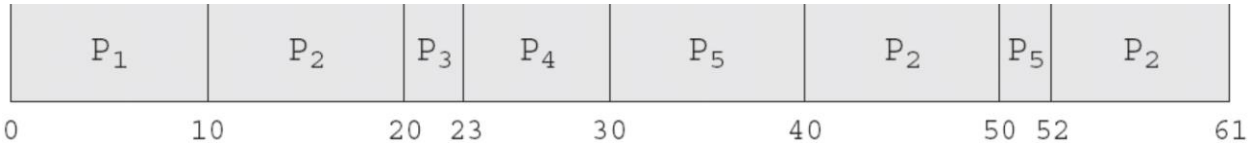
- Thời gian đợi trung bình: (0 + 10 + 39 + 42 + 49)/5 = 28
- Thời gian lưu lại trong hệ thống trung bình: (10 + 39 + 42 + 49 + 61)/5 = 40.2

b. SJF



- Thời gian đợi trung bình: (10 + 32 + 0 + 3 + 20)/5 = 13
- Thời gian lưu lại trong hệ thống trung bình: (20 + 61 + 3 + 10 + 32)/5 = 25.2

c. RR với quantum time = 10



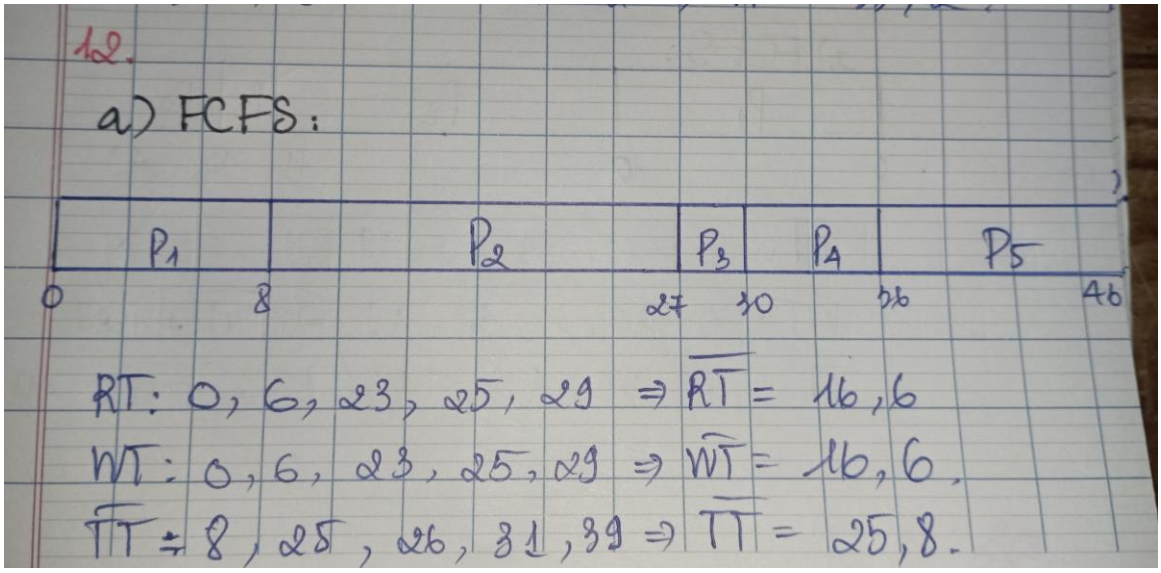
- Thời gian đợi trung bình: (0 + (10 + 20 + 2) + 20 + 23 + (30 + 10))/5 = 23
- Thời gian lưu lại trong hệ thống trung bình: (10 + 61 + 23 + 30 + 52)/5 = 35.2

12. Cho 5 tiến trình P1, P2, P3, P4, P5 với thời gian vào hàng đợi ready và thời gian cần CPU tương ứng như bảng sau:

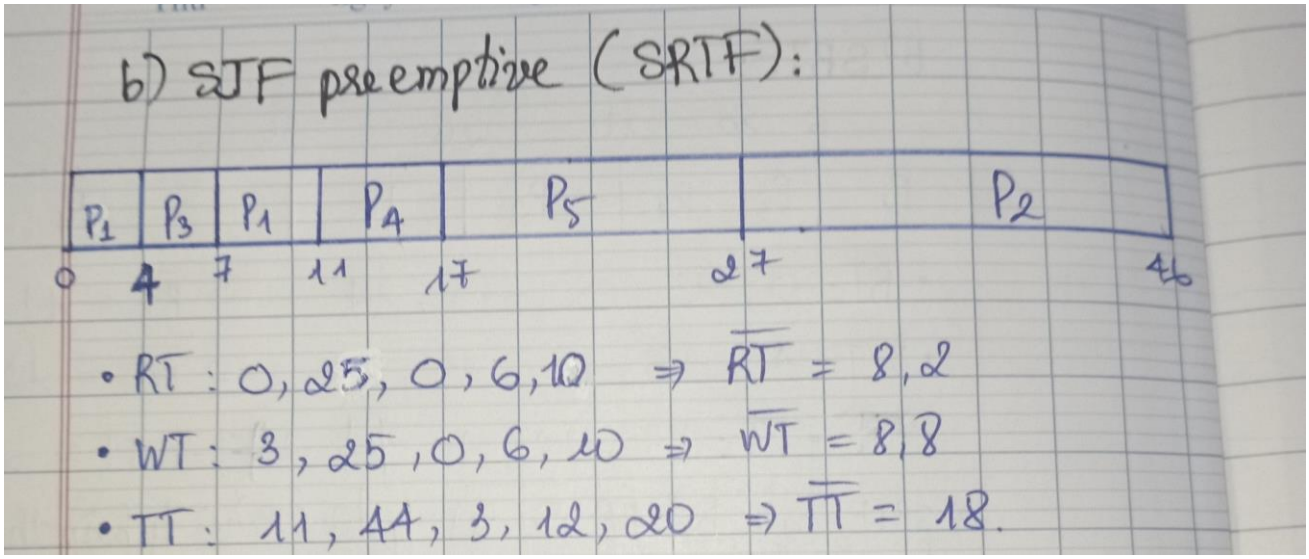
Process	Arrival Time	CPU Burst Time
P1	0	8
P2	2	19
P3	4	3
P4	5	6
P5	7	10

Vẽ sơ đồ Gantt và tính thời gian chờ trung bình, thời gian đáp ứng trung bình, thời gian lưu lại trong hệ thống (turnaround time) trung bình cho các giải thuật sau:

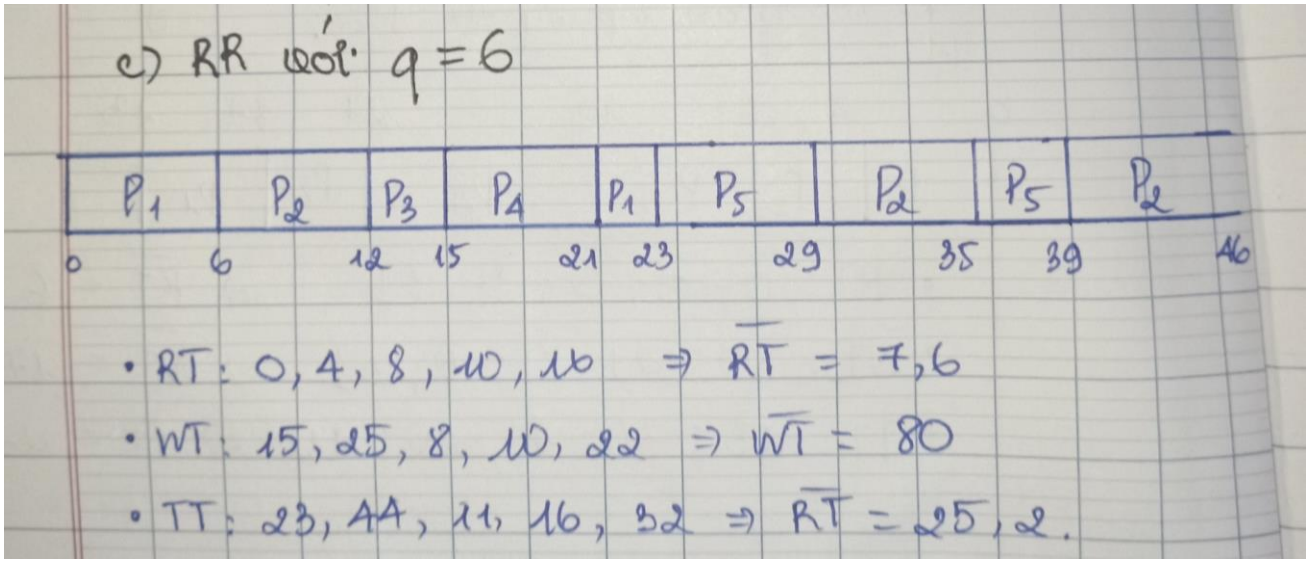
a. FCFS



b. SJF preemptive



c. RR với quantum time = 6.



13. (Bài tập mẫu) Cho 5 tiến trình P1, P2, P3, P4, P5 với thời gian vào hàng đợi ready và thời gian cần CPU tương ứng như bảng sau:

Process	Arrival Time	Burst Time
P1	0	13
P2	4	9
P3	6	4
P4	7	20
P5	12	10

Vẽ giản đồ Gantt và tính thời gian đợi trung bình, thời gian đáp ứng trung bình, thời gian lưu lại trong hệ thống (turnaround time - thời gian hoàn thành) trung bình khi thực hiện các giải thuật định thời sau:

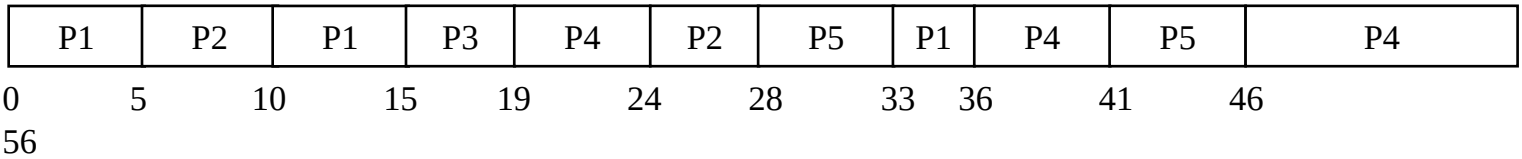
a) ***Round Robin với quantum time = 5***

b) ***SRTFo***

Có nhận xét gì về tính hiệu quả của hai giải thuật trên?

a. Round Robin với quantum time = 5

Giản đồ Gantt:



0 5 10 15 19 24 28 33 36 41 46 56

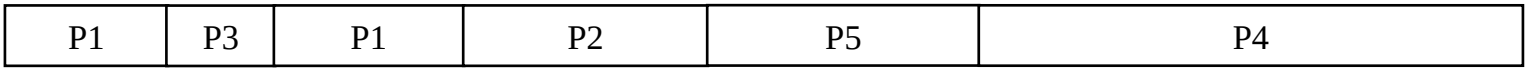
Thời gian đáp ứng trung bình: (0 + 1 + 9 + 12 + 16)/5 = 7.6

Thời gian đợi trung bình: ((5 + 18) + (1 + 14) + 9 + (12 + 12 + 5) + (16 + 8))/5 = 20

Thời gian hoàn thành trung bình: (36 + 24 + 13 + 49 + 34)/5 = 31.2

b. SRTF

Giản đồ Gantt:



0 6 10 17 26 36 56

Thời gian đáp ứng trung bình: (0 + 13 + 0 + 29 + 14)/5 = 11.2

Thời gian đợi trung bình: (4 + 13 + 0 + 29 + 14)/5 = 12

Thời gian hoàn thành trung bình: (17 + 22 + 4 + 49 + 24)/5 = 23.2

Nhận xét về hai giải thuật trên:

- SRTF hiệu quả hơn (tốt hơn) Round Robin nếu xét trên các tiêu chuẩn thời gian đợi (trung bình) và thời gian hoàn thành (trung bình).

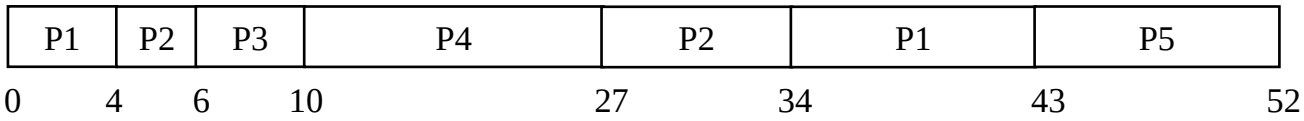
- Round Robin cho thời gian đáp ứng (trung bình) tốt hơn SRTF.

14. (Bài tập mẫu) Cho 5 tiến trình P1, P2, P3, P4, P5 với thời gian vào hàng đợi ready và thời gian cần CPU tương ứng như bảng sau:

Process	Arrival Time	Burst Time	Priority
P1	0	13	4
P2	4	9	3
P3	6	4	1
P4	7	17	2
P5	12	9	5

Vẽ giản đồ Gantt và tính thời gian đợi trung bình, thời gian đáp ứng trung bình, thời gian lưu lại trong hệ thống (turnaround time - thời gian hoàn thành) trung bình khi thực hiện giải thuật định thời Preemptive Priority (độ ưu tiên 1 > 2 > 3 …)

Giản đồ Gantt:



Thời gian đáp ứng trung bình: (0 + 0 + 0 + 3 + 31)/5 = 6.8

Thời gian đợi trung bình: (30 + 21 + 0 + 3 + 31)/5 = 17

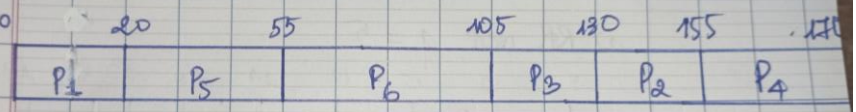
Thời gian hoàn thành trung bình: (43 + 30 + 4 + 20 + 40)/5 = 27.4

15. Sử dụng các giải thuật FCFS, SJF, SRTF, Priority -Pre, RR (10) để tính các giá trị thời gian đợi, thời gian đáp ứng, thời gian hoàn thành trung bình và vẽ giản đồ Gantt cho các tiến trình sau:

Process	Arrival Time	Burst Time	Priority
P1	0	20	20
P2	25	25	30
P3	20	25	15
P4	35	15	35
P5	10	35	5
P6	15	50	10

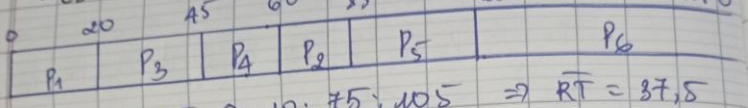
15.

- FCFS:



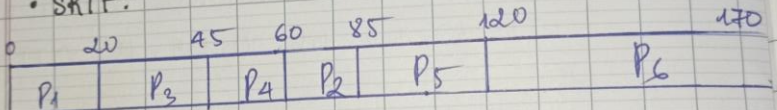
- RT: 0; 105; 85; 120; 10; 40 $\Rightarrow \overline{RT} = 60$
- WT: 0; 105; 85; 120; 10; 40 $\Rightarrow \overline{WT} = 60$
- TT: 20; 130; 110; 135; 45; 90 $\Rightarrow \overline{TT} = 88,3$

- SJF:



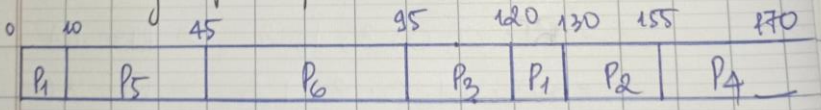
- RT: 0; 35; 0; 10; 75; 105 $\Rightarrow \overline{RT} = 37,5$
- WT: 0; 35; 0; 10; 75; 105 $\Rightarrow \overline{WT} = 37,5$
- TT: 20; 60; 85; 110; 155 $\Rightarrow \overline{TT} = 61,6$

- SRTF:



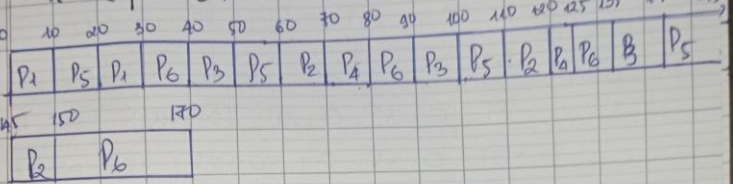
- $\overline{RT} = 37,5$
- $\overline{WT} = 37,5$
- $\overline{TT} = 61,6$

- Priority - preemptive:



- RT: 0; 105; 75; 120; 0; 30 $\Rightarrow \overline{RT} = 55$
- WT: 110; 105; 75; 120; 0; 30 $\Rightarrow \overline{WT} = 73,3$
- TT: 130; 130; 100; 135; 35; 80 $\Rightarrow \overline{TT} = 101,7$.

- RR (q = 10):

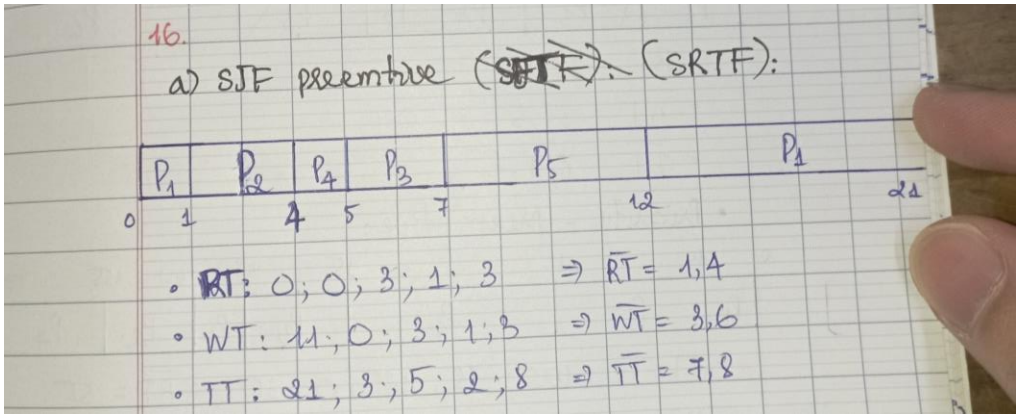


- RT: 0, 35, 20, 35, 0, 15 $\Rightarrow \overline{RT} = 17,5$
- WT: 10, 100, 75, 40, 100, 90 $\Rightarrow \overline{WT} = 69,2$
- TT: 30, 125, 120, 90, 135, 155 $\Rightarrow \overline{TT} = 109,2$.

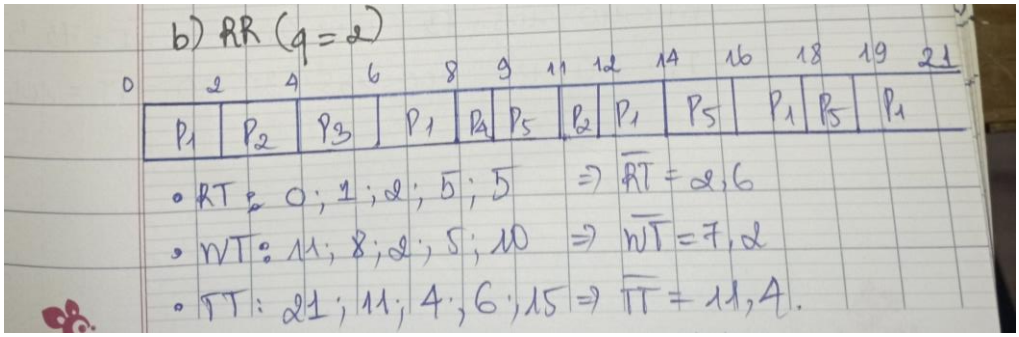
16. Xét tập các tiến trình sau (với thời gian yêu cầu CPU và độ ưu tiên kèm theo). Vẽ giản đồ Gantt và tính thời gian đợi trung bình và thời gian lưu lại trong hệ thống trung bình (turnaround time) cho các giải thuật sau:

Process	Arrival Time	Burst Time	Priority
P1	0	10	3
P2	1	3	2
P3	2	2	1
P4	3	1	2
P5	4	5	4

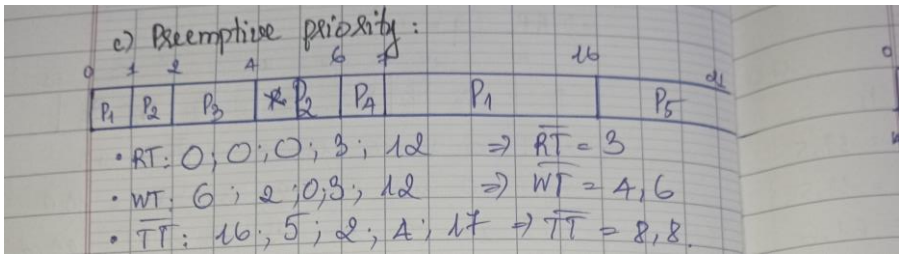
a. SJF Preemptive



b. RR với quantum time = 2



c. Preemptive Priority (độ ưu tiên 1 > 2 > ...)

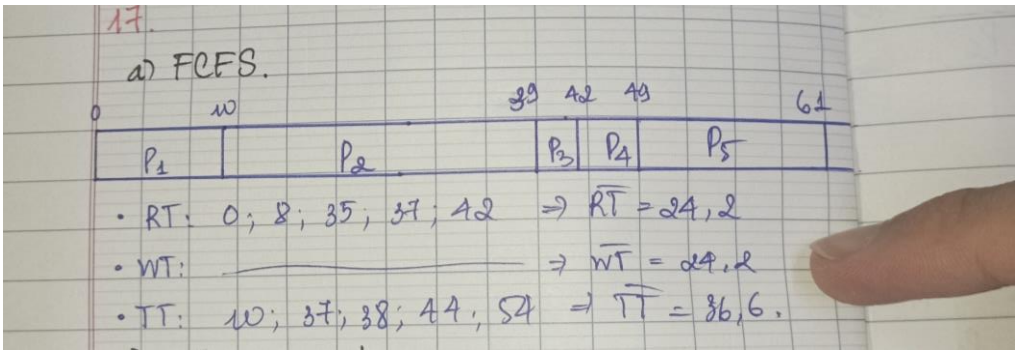


17. Cho 5 tiến trình với thời gian vào hàng đợi ready và thời gian cần CPU tương ứng như bảng sau:

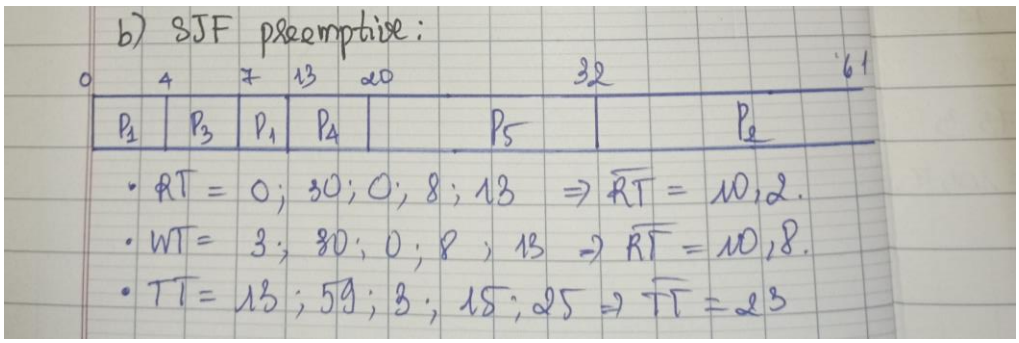
Process	Arrival Time	Burst Time
P1	0	10
P2	2	29
P3	4	3
P4	5	7
P5	7	12

Vẽ giản đồ Gantt và tính thời gian đợi trung bình, thời gian đáp ứng trung bình và thời gian lưu lại trong hệ thống (turnaround time) trung bình cho các giải thuật sau:

a. FCFS



b. SJF preemptive



c. RR với quantum time = 10