



COMPUTER ENGINEERING



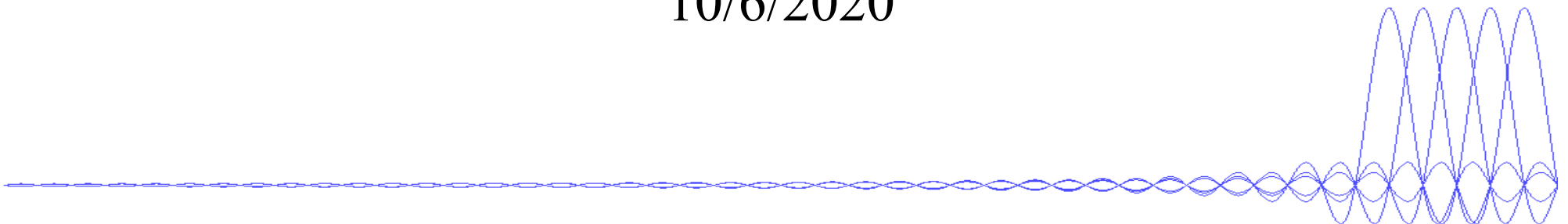
**UIT**  
TRƯỜNG ĐẠI HỌC  
CÔNG NGHỆ THÔNG TIN

# HỆ ĐIỀU HÀNH

## Chương 4 (1)

### Định thời CPU

10/6/2020





## Câu hỏi ôn tập chương 3 (tt)

- Process control block chứa những thông tin gì?
- Các tác vụ đối với tiến trình?



## Câu hỏi ôn tập chương 3

Nêu cụ thể các trạng thái của tiến trình?

```
/* test.c */
```

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

```
{
```

```
    printf("Hello world\n");
```

```
    scanf(" Nhập c = %d",&c);
```

```
    exit(0);
```

```
}
```

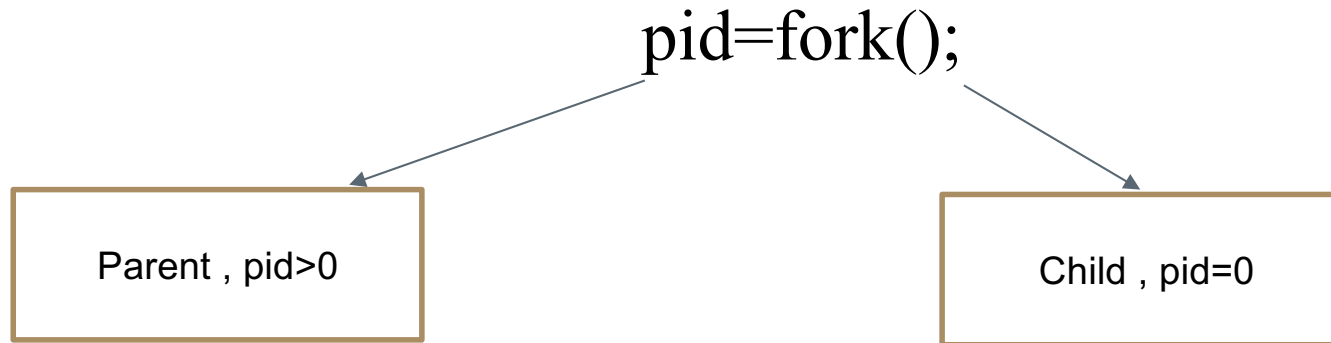
1. Đoạn ct trên in ra những gì?
2. Các trạng thái đã trải qua của chương trình trên/ có bao nhiêu lần waiting/running



# Lưu ý cho lệnh fork

## ■ Fork()

- 1. Có hai tiến trình được tạo ra từ lệnh fork().



- 2. Cả tiến trình cha và con sẽ thực thi ngay sau khi lệnh fork
- `pid=fork();`
- `printf("hello world");`



## Lưu ý cho lệnh fork

```
fork();  
fork();  
printf("Hello/n")  
fork();  
printf("World/n");
```

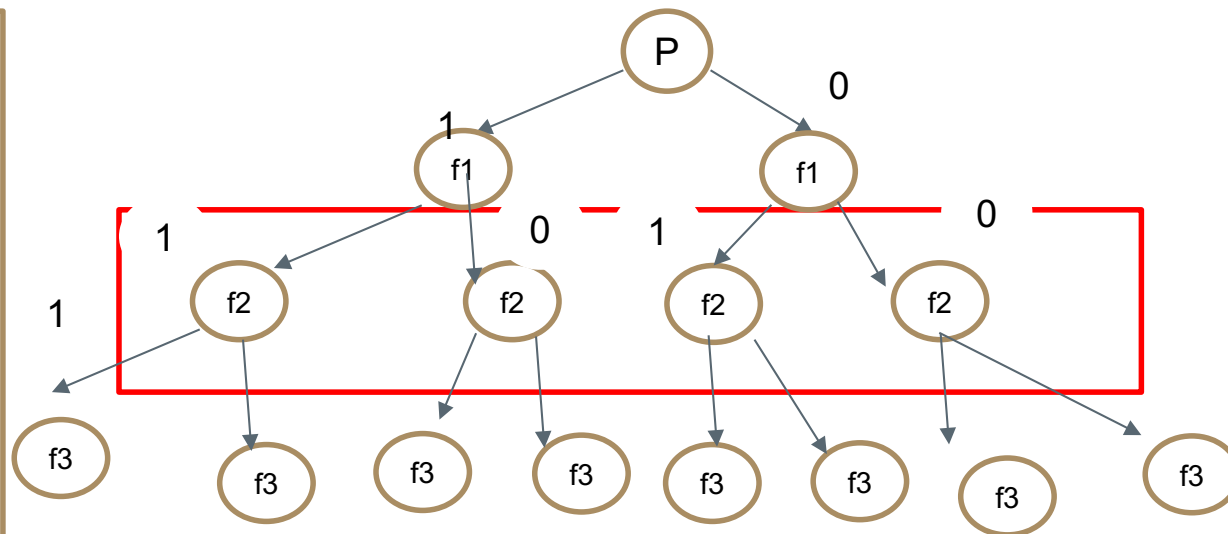
Tổng cộng có bao nhiêu tiến trình? Bao nhiêu tiến trình con?

Số lần mà từ Hello/World được in ra? Liệt kê Output của đoạn chương trình trên?



# Lưu ý cho lệnh fork

```
fork();    (f1)
fork();    (f2)
printf("Hello/n")
fork();    (f3)
printf("World/n");
```



Tổng cộng có bao nhiêu tiến trình? Bao nhiêu tiến trình con?

Số lần mà từ Hello/World được in ra? Liệt kê Output của đoạn chương trình trên?



## Mục tiêu chương 4

- Biết được các khái niệm cơ bản về định thời
- Biết được các tiêu chuẩn định thời CPU
- Hiểu được các giải thuật định thời
- Vận dụng các giải thuật định thời để làm bài tập và mô phỏng



## Nội dung chương 4

- Các khái niệm cơ bản về định thời
- Các bộ định thời
- Các tiêu chuẩn định thời CPU
- Các giải thuật định thời

First-Come, First-Served (FCFS)

Shortest Job First (SJF)

Shortest Remaining Time First (SRTF)

Priority Scheduling





# Khái niệm cơ bản

## ■ Trong các hệ thống multitasking

Thực thi nhiều chương trình đồng thời làm tăng hiệu suất hệ thống

Tại mỗi thời điểm, chỉ có một process được thực thi don chuong

## ■ => Cần phải giải quyết vấn đề phân chia, lựa chọn process thực thi sao cho được hiệu quả nhất

## ■ => Chiến lược định thời CPU

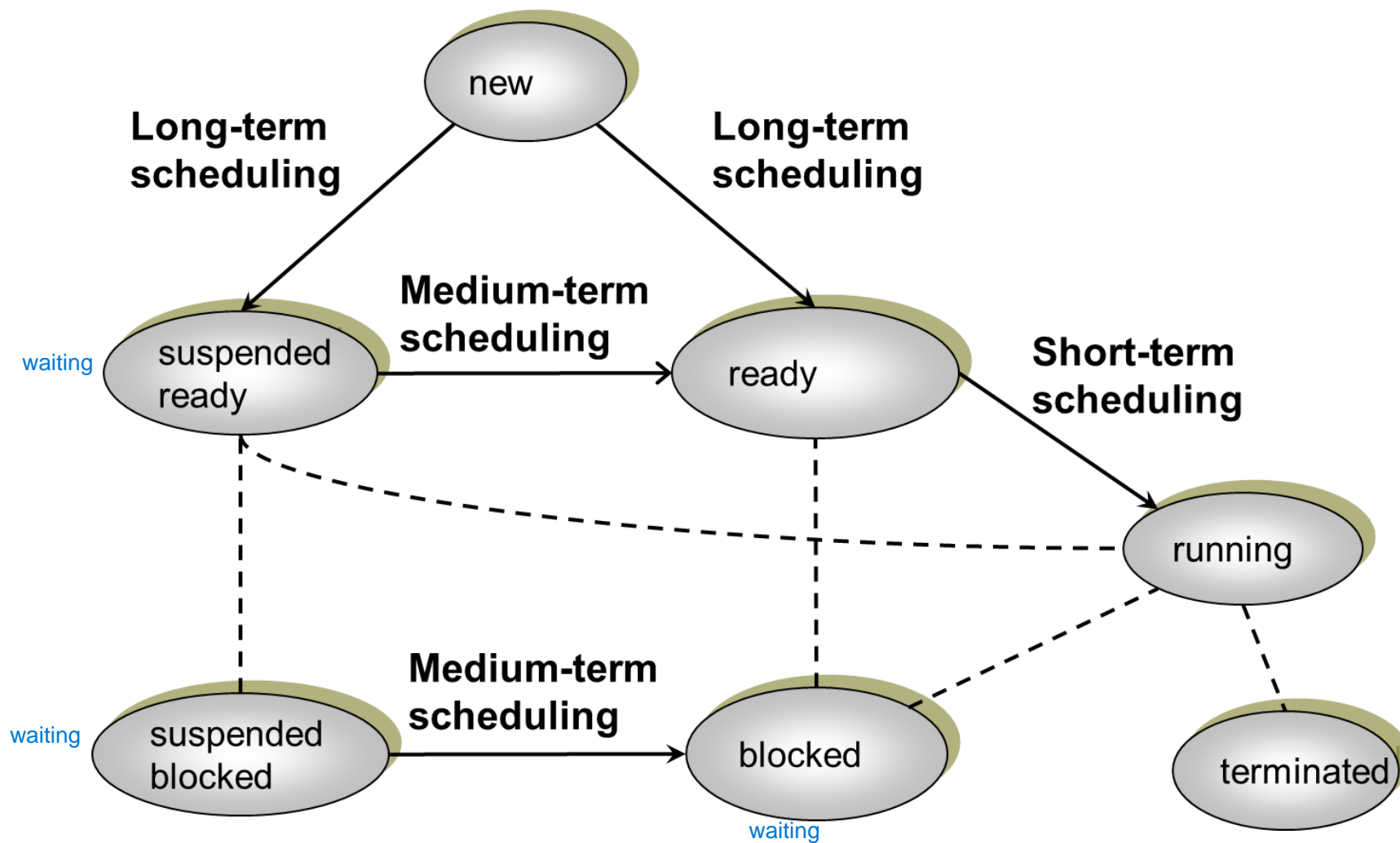
## ■ Định thời CPU

Chọn một process (từ ready queue) thực thi

Với một multithreaded kernel, việc định thời CPU là do OS chọn kernel thread được chiếm CPU



# Các bộ định thời





## ■ Long-term scheduling

1. Xác định chương trình nào được chấp nhận nạp vào hệ thống để thực thi
2. Điều khiển mức độ multiprogramming của hệ thống
3. Long term scheduler thường cố gắng duy trì xen lẫn CPU-bound và I/O-bound process

## ■ Medium-term scheduling

1. Process nào được đưa vào (swap in), đưa ra khỏi (swap out) bộ nhớ chính xem tiến trình trước (tiến trình này) đang dùng ở đâu
2. Được thực hiện bởi phần quản lý bộ nhớ và được thảo luận ở phần quản lý bộ nhớ



# Các bộ định thời (tt)

## ■ Short-term scheduling

Xác định process nào trong ready queue sẽ được chiếm CPU để thực thi kế tiếp (còn được gọi là định thời CPU, CPU scheduling)

Bộ định thời short-term được gọi mỗi khi có một trong các sự kiện/interrupt sau xảy ra:

- Ngắt thời gian (clock interrupt)
- Ngắt ngoại vi (I/O interrupt)
- Lời gọi hệ thống (operating system call)
- Signal



# Bộ định thời

- Bộ định thời sẽ chuyển quyền điều khiển CPU về cho process được chọn.

- Bao gồm:

Chuyển ngữ cảnh (sử dụng thông tin ngữ cảnh trong PCB)

Chuyển chế độ người dùng vì có tiến trình hệ thống và tiến trình người dùng, nên có thể chuyển về tiến trình user

Nhảy đến vị trí thích hợp trong chương trình ứng dụng để khởi động lại chương trình (chính là program counter trong PCB)

- Công việc này gây ra phí tổn

Dispatch latency: thời gian mà bộ định thời dừng một process và khởi động một process khác



# Các tiêu chuẩn định thời CPU

## ■ Hướng người dùng (User-oriented)

không để user đợi quá lâu

1. Thời gian đáp ứng (Response time): khoảng thời gian process nhận yêu cầu đến khi yêu cầu đầu tiên được đáp ứng (time-sharing, interactive system) → cực tiểu
2. Thời gian quay vòng (hoàn thành) (Turnaround time): khoảng thời gian từ lúc một process được nạp vào hệ thống đến khi process đó kết thúc → cực tiểu
3. Thời gian chờ (Waiting time): tổng thời gian một process đợi trong ready queue → cực tiểu



# Các tiêu chuẩn định thời CPU (tt)

## ■ Hướng hệ thống (System-oriented)

1. Sử dụng CPU (processor utilization): định thời sao cho CPU càng bận càng tốt → cực đại
2. Công bằng (fairness): tất cả process phải được đối xử như nhau
3. Thông lượng (throughput): số process hoàn tất công việc trong một đơn vị thời gian → cực đại



# Hai yếu tố của giải thuật định thời

- Hàm chọn lựa (selection function): dùng để chọn process nào trong ready queue được thực thi (thường dựa trên độ ưu tiên, yêu cầu về tài nguyên, đặc điểm thực thi của process,...)
- Chế độ quyết định (decision mode): chọn thời điểm thực hiện hàm chọn lựa để định thời





# Hai yếu tố của giải thuật định thời (tt)

## ■ Có hai chế độ quyết định:

### Không trung dụng (**Non-preemptive**)

- Khi ở trạng thái running, process sẽ thực thi cho đến khi kết thúc hoặc bị blocked do yêu cầu I/O

### Trung dụng (**Preemptive**)

- Process đang thực thi (trạng thái running) có thể bị ngắt nửa chừng và chuyển về trạng thái ready
- Chi phí cao hơn non-preemptive nhưng đánh đổi lại bằng thời gian đáp ứng tốt hơn vì không có trường hợp một process độc chiếm CPU quá lâu



# Preemptive và Non-preemptive

## ■ Hàm định thời được thực hiện khi

- (1) Chuyển từ trạng thái running sang waiting
- (2) Chuyển từ trạng thái running sang ready
- (3) Chuyển từ trạng thái waiting, new sang ready
- (4) Kết thúc thực thi

(1) và (4) không cần lựa chọn loại định thời biểu, (2) và (3) cần

## ■ Trường hợp 1, 4 được gọi là định thời nonpreemptive

## ■ Trường hợp 2, 3 được gọi là định thời preemptive

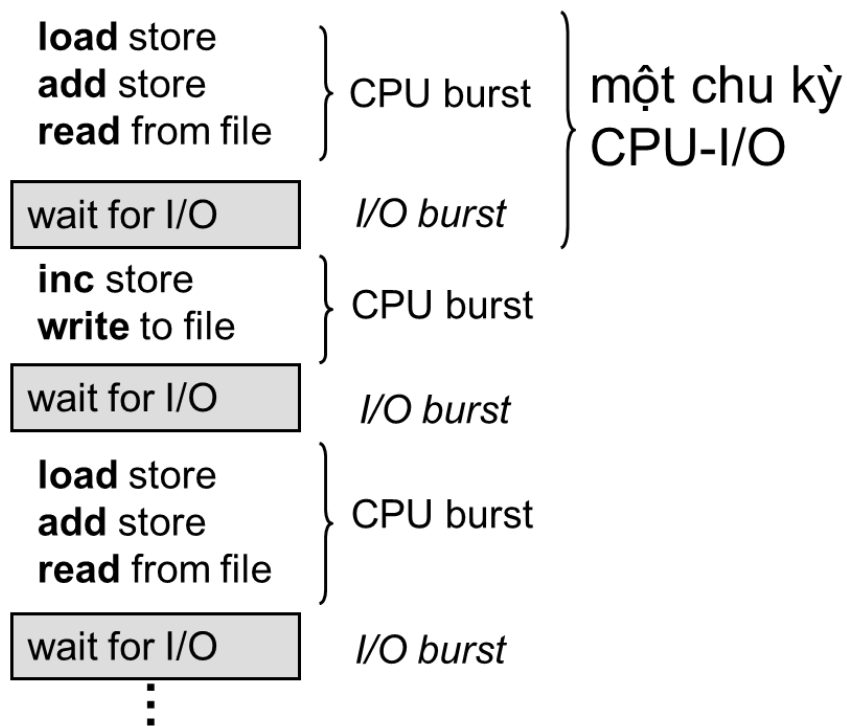
Thực hiện theo cơ chế nào khó hơn? Tại sao?

preemptive khó hơn, vì phải xét nhiều tiêu chí (VD: khi cường chế 1 tiến trình phải xem xét tiến trình do có độ ưu tiên cao hơn không?)



# Khảo sát giải thuật định thời

- Service time = thời gian process cần CPU trong một chu kỳ CPU-I/O
- Process có service time lớn là các CPU-bound process



Process	Arrival Time	Service Time
1	0	3
2	2	6
3	4	4
4	6	5
5	8	2



# 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



# First-Come, First-Served (FCFS)

## ■ Hàm lựa chọn:

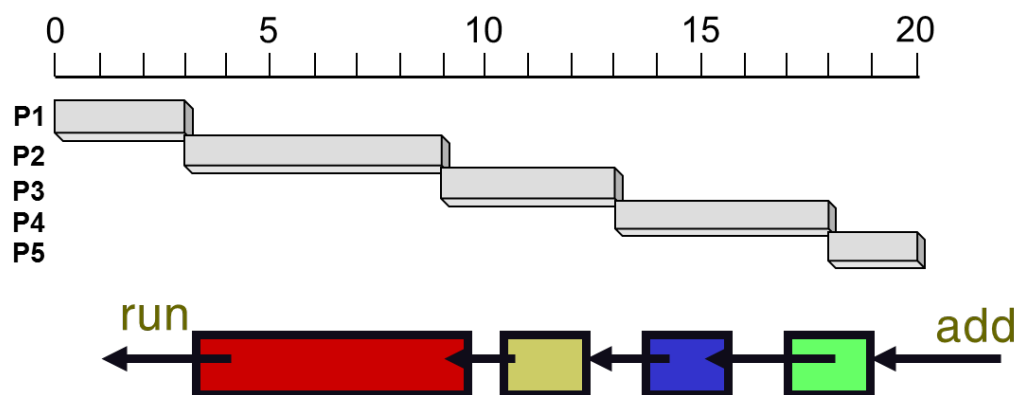
Tiến trình nào yêu cầu CPU trước sẽ được cấp phát CPU trước  
Process sẽ thực thi đến khi kết thúc hoặc bị blocked do I/O

## ■ Chế độ quyết định: non-preemptive algorithm

## ■ Hiện thực: sử dụng hàng đợi FIFO (FIFO queues)

Tiến trình đi vào được thêm vào cuối hàng đợi

Tiến trình được lựa chọn để xử lý được lấy từ đầu của queues

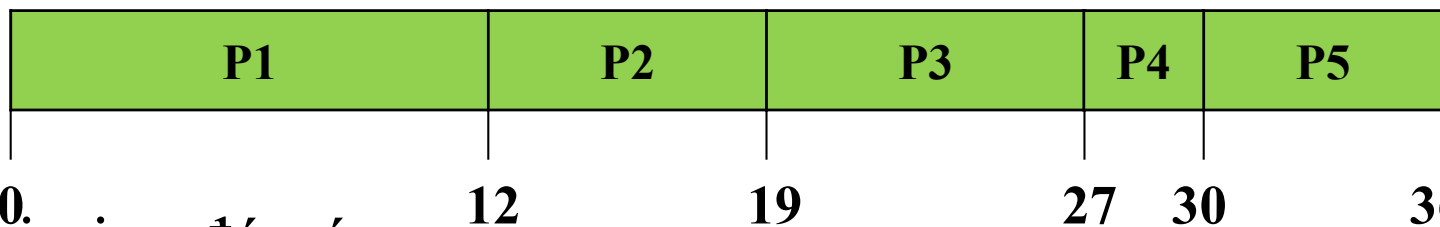




# First-Come, First-Served (FCFS)

Process	Arrival Time	Burst Time
P1	0	12
P2	2	7
P3	5	8
P4	9	3
P5	12	6

## ■ Giải đồ Gantt



## ■ Thời gian đáp ứng: = Thời gian được cấp CPU - thời gian bắt đầu cho

$$P1 = 0, P2 = 10, P3 = 14, P4 = 18, P5 = 18$$

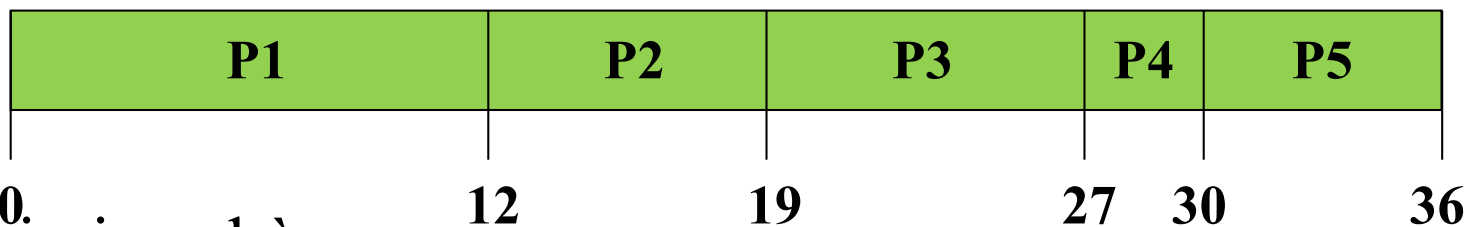
$$\text{Thời gian đáp ứng trung bình: } (0 + 10 + 14 + 18 + 18)/5 = 12$$



# First-Come, First-Served (FCFS)

Process	Arrival Time	Burst Time
P1	0	12
P2	2	7
P3	5	8
P4	9	3
P5	12	6

## ■ Giải đồ Gantt



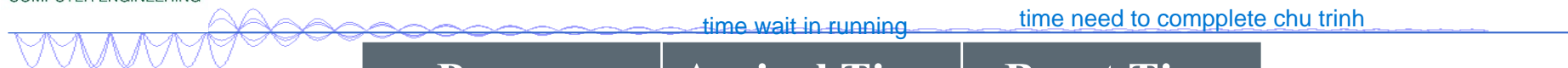
## ■ Thời gian chờ: vi không trung dung nen cho = dap ung

$$P1 = 0, P2 = 10, P3 = 14, P4 = 18, P5 = 18$$

$$\text{Thời gian chờ trung bình: } (0 + 10 + 14 + 18 + 18)/5 = 12$$

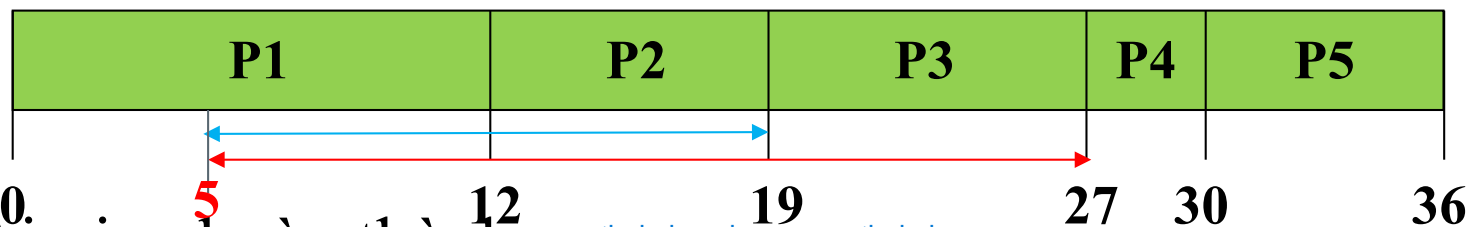


# First-Come, First-Served (FCFS)



Process	Arrival Time	Burst Time
P1	0	12
P2	2	7
P3	5	8
P4	9	3
P5	12	6

## ■ Giải đồ Gantt



## ■ Thời gian hoàn thành: = thời gian gian xong - thời gian nạp vào

$$P1 = 12, P2 = 17, P3 = 22, P4 = 21, P5 = 24$$

$$\text{Thời gian hoàn thành trung bình: } (12 + 17 + 22 + 21 + 24)/5 = 19.2$$





# Shortest-Job-First (SJF)

- Định thời biểu công việc ngắn nhất trước
- Khi CPU được tự do, nó sẽ cấp phát cho tiến trình yêu cầu ít thời gian nhất để kết thúc (tiến trình ngắn nhất)
- Liên quan đến chiều dài thời gian sử dụng CPU cho lần tiếp theo của mỗi tiến trình
- Sử dụng những chiều dài này để lập lịch cho tiến trình với thời gian ngắn nhất



# Shortest-Job-First (SJF) (tt)

## ■ Scheme 1: Non-preemptive

tu lúc c cấp cpu ên khi kết thúc, hđh kh c lấy lại cpu

Khi CPU được trao cho quá trình nó không nhường cho đến khi nó kết thúc chu kỳ xử lý của nó

## ■ Scheme 2: Preemptive

hđh c lấy lại cpu

Nếu một tiến trình mới được đưa vào danh sách với chiều dài sử dụng CPU cho lần tiếp theo nhỏ hơn thời gian còn lại của tiến trình đang xử lý, nó sẽ dừng hoạt động tiến trình hiện hành  
→ Shortest-Remaining-Time-First (SRTF)

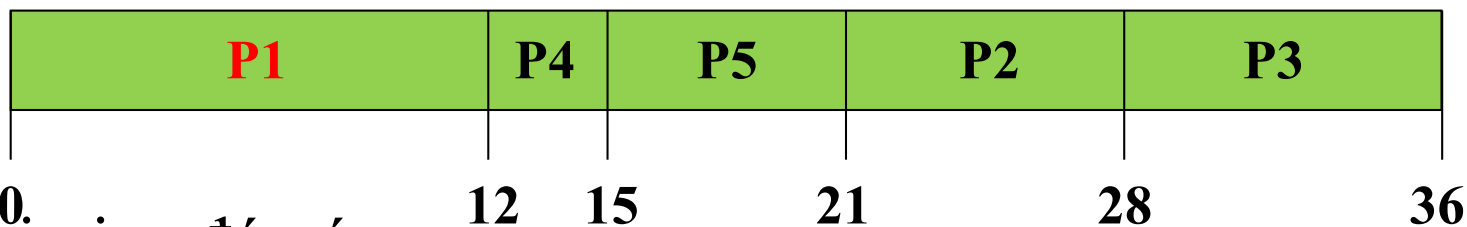
## ■ SJF là tối ưu – cho thời gian chờ đợi trung bình tối thiểu với một tập tiến trình cho trước



# Non-Preemptive SJF

Process	Arrival Time	Burst Time
P1	0	12
P2	2	7
P3	5	8
P4	9	3
P5	12	6

## ■ Giải đồ Gantt



## ■ Thời gian đáp ứng: time cap cpu - time cho

$$P1 = 0, P2 = 19, P3 = 23, P4 = 3, P5 = 3$$

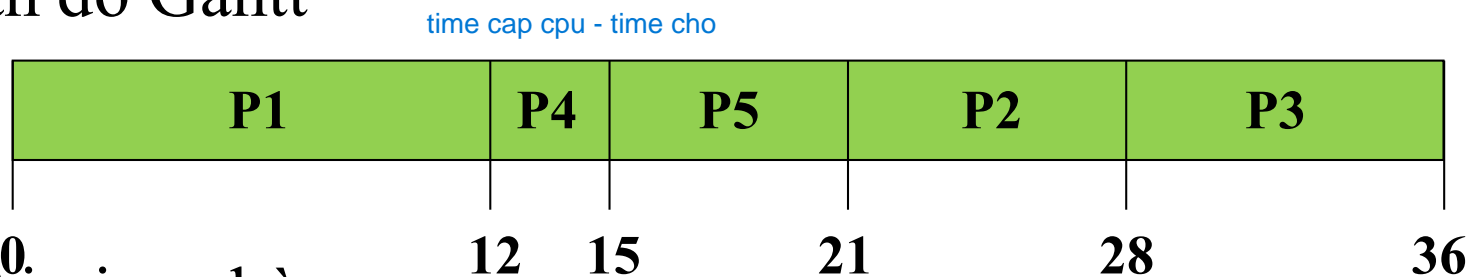
$$\text{Thời gian đáp ứng trung bình: } (0 + 19 + 23 + 3 + 3)/5 = 9.6$$



# Non-Preemptive SJF

Process	Arrival Time	Burst Time
P1	0	12
P2	2	7
P3	5	8
P4	9	3
P5	12	6

## ■ Giải đồ Gantt



## ■ Thời gian chờ:

$$P1 = 0, P2 = 19, P3 = 23, P4 = 3, P5 = 3$$

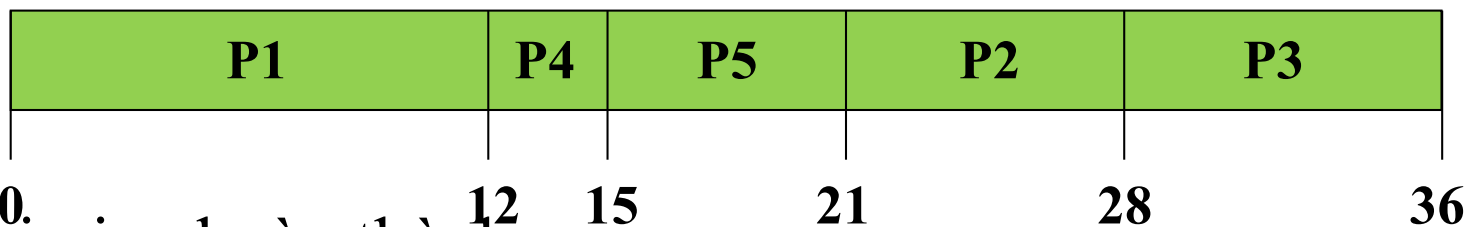
$$\text{Thời gian chờ trung bình: } (0 + 19 + 23 + 3 + 3)/5 = 9.6$$



# Non-Preemptive SJF

Process	Arrival Time	Burst Time
P1	0	12
P2	2	7
P3	5	8
P4	9	3
P5	12	6

## ■ Giải đồ Gantt



## ■ Thời gian hoàn thành:

time hoàn thành xong - time cap cpu

$$P1 = 12, P2 = 26, P3 = 31, P4 = 6, P5 = 9$$

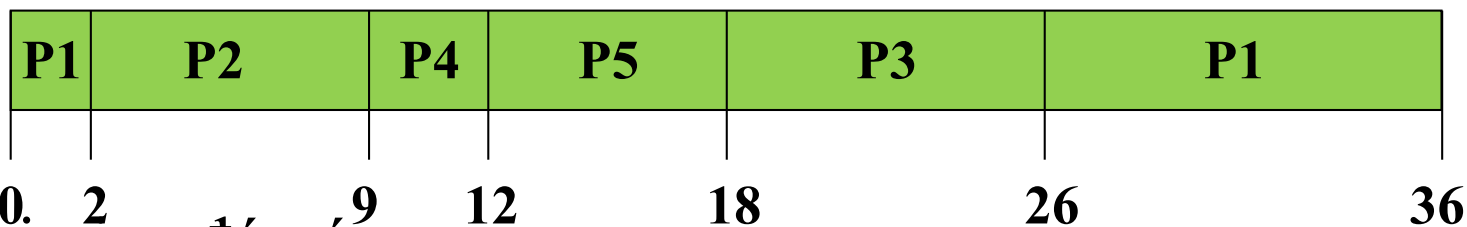
$$\text{Thời gian hoàn thành trung bình: } (12 + 26 + 31 + 6 + 9)/5 = 16.8$$



# Preemptive SJF (SRTF)

Process	Arrival Time	Burst Time
P1	0	12
P2	2	7
P3	5	8
P4	9	3
P5	12	6

## ■ Giải đồ Gantt



## ■ Thời gian đáp ứng: t/g cấp cpu lan dau tien - t/g xếp hàng

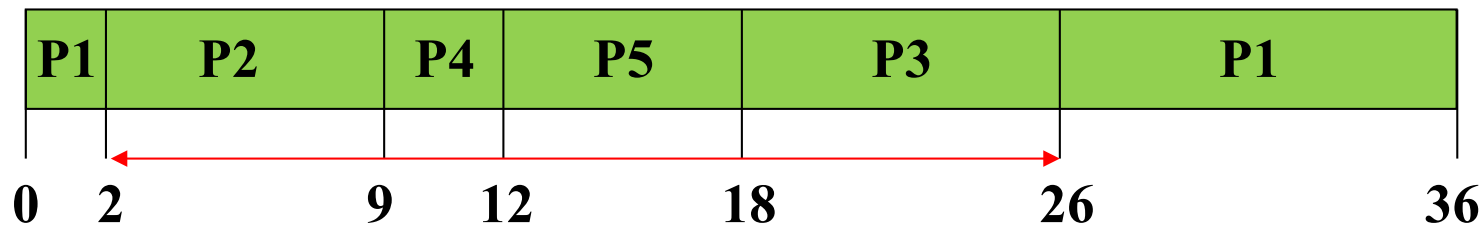
$$P1 = 0, P2 = 0, P3 = 13, P4 = 0, P5 = 0$$

$$\text{Thời gian chờ trung bình: } (0 + 0 + 13 + 0 + 0)/5 = 2.6$$



# Preemptive SJF (SRTF)

## ■ Giải đồ Gantt



## ■ Thời gian đáp ứng:

$$P1 = 0, P2 = 0, P3 = 13, P4 = 0, P5 = 0$$

$$\text{Thời gian chờ trung bình: } (0 + 0 + 13 + 0 + 0)/5 = 2.6$$

## ■ Thời gian chờ: ( t/g cap cpu - luc xep hang ) + ( t/g cap lai cpu - t/g lay lai cpu )

$$P1 = 24, P2 = 0, P3 = 13, P4 = 0, P5 = 0$$

$$\text{Thời gian chờ trung bình: } (24 + 0 + 13 + 0 + 0)/5 = 7.4$$

1. Thời gian đáp ứng (Response time): khoảng thời gian process nhận yêu cầu đến khi yêu cầu đầu tiên được đáp ứng (time-sharing, interactive system) → cực tiểu

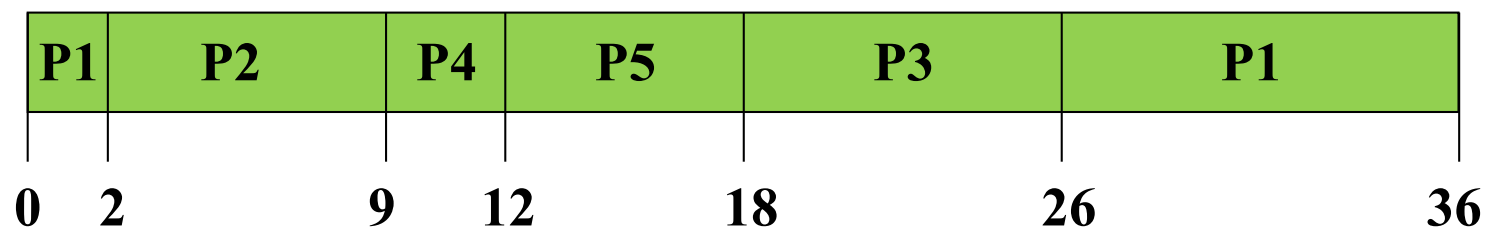
2. Thời gian chờ (Waiting time): tổng thời gian một process đợi trong ready queue → cực tiểu



# Preemptive SJF (SRTF)

Process	Arrival Time	Burst Time
P1	0	12
P2	2	7
P3	5	8
P4	9	3
P5	12	6

## ■ Giải đồ Gantt



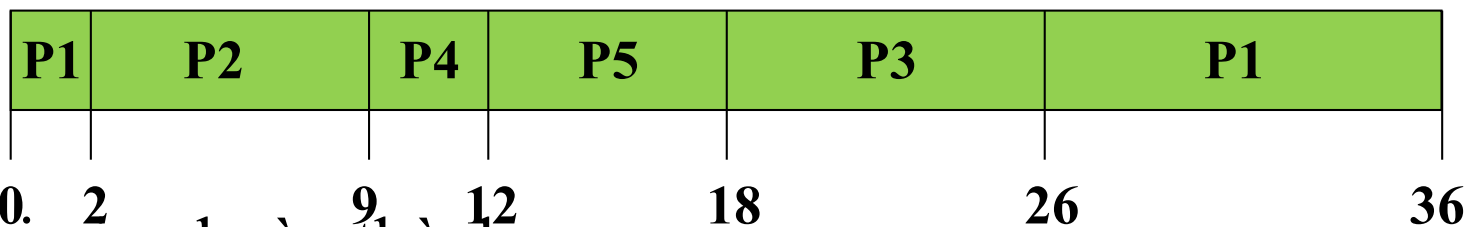




# Preemptive SJF (SRTF)

Process	Arrival Time	Burst Time
P1	0	12
P2	2	7
P3	5	8
P4	9	3
P5	12	6

## ■ Giải đồ Gantt



## ■ Thời gian hoàn thành: time lúc xong - time xếp hàng

$$P1 = 36, P2 = 7, P3 = 21, P4 = 3, P5 = 6$$

$$\text{Thời gian hoàn thành trung bình: } (36 + 7 + 21 + 3 + 6)/5 = 14.6$$



# Nhận xét về giải thuật SJF

- Có thể xảy ra tình trạng “đói” (starvation) đối với các process có CPU-burst lớn khi có nhiều process với CPU-burst nhỏ đến hệ thống. =
- Cơ chế non-preemptive không phù hợp cho hệ thống time sharing (interactive).
- Giải thuật SJF ngầm định ra độ ưu tiên theo burst time.
- Các CPU-bound process có độ ưu tiên thấp hơn so với I/O-bound process, nhưng khi một process không thực hiện I/O được thực thi thì nó độc chiếm CPU cho đến khi kết thúc.



## Nhận xét về giải thuật SJF (tt)

- Tương ứng với mỗi process cần có độ dài của CPU burst tiếp theo
- Hàm lựa chọn: chọn process có độ dài CPU burst nhỏ nhất
- Chứng minh được: SJF tối ưu trong việc giảm thời gian đợi trung bình
- Nhược điểm: Cần phải ước lượng thời gian cần CPU tiếp theo của process
- Giải pháp cho vấn đề này?



## Nhận xét về giải thuật SJF (tt)

- Thời gian sử dụng CPU chính là độ dài của CPU burst  
Trung bình tất cả các CPU burst đo được trong quá khứ  
Nhưng thông thường những CPU burst càng mới càng phản ánh đúng hành vi của process trong tương lai
- Một kỹ thuật thường dùng là sử dụng trung bình hàm mũ (exponential averaging)

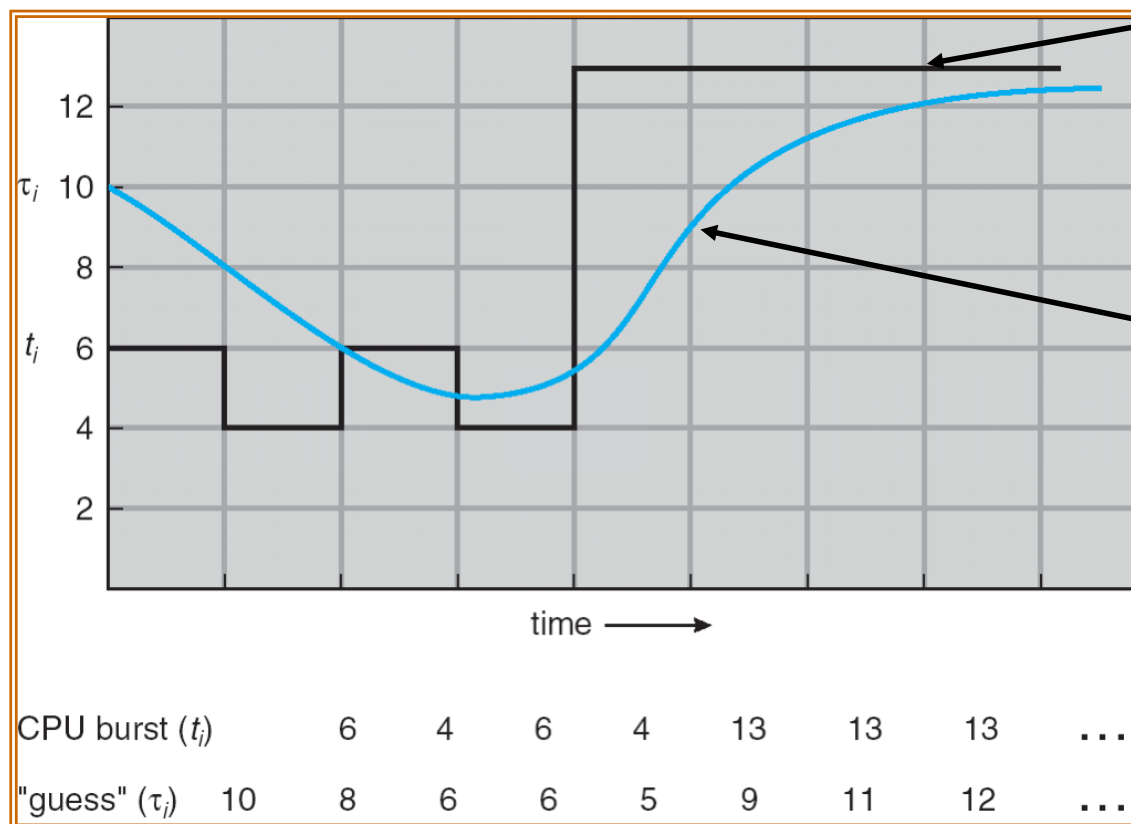
$$\tau_{n+1} = a \tau_n + (1 - a) \tau_n, \quad 0 \leq a \leq 1$$

$$\tau_{n+1} = a \tau_n + (1 - a) a \tau_{n-1} + \dots + (1 - a)^j a \tau_{n-j} + \dots + (1 - a)^{n+1} a \tau_0$$

Nếu chọn  $a = \frac{1}{2}$  thì có nghĩa là trị đo được  $\tau_n$  và trị dự đoán  $\tau_n$  được xem quan trọng như nhau.



# Dự đoán thời gian sử dụng CPU



Độ dài CPU burst  
đo được

Độ dài CPU burst  
dự đoán, với  
 $a = \frac{1}{2}$  và  $\tau_0 = 10$



# Priority Scheduling

- Mỗi process sẽ được gán một độ ưu tiên
- CPU sẽ được cấp cho process có độ ưu tiên cao nhất
- Định thời sử dụng độ ưu tiên có thể:
  - Preemptive hoặc
  - Non-preemptive



## Priority Scheduling (tt)

- SJF là một giải thuật định thời sử dụng độ ưu tiên với độ ưu tiên là thời-gian-sử-dụng-CPU-dự-đoán
- Gán độ ưu tiên còn dựa vào:
  - Yêu cầu về bộ nhớ
  - Số lượng file được mở
  - Tỉ lệ thời gian dùng cho I/O trên thời gian sử dụng CPU
  - Các yêu cầu bên ngoài ví dụ như: số tiền người dùng trả khi thực thi công việc



## Priority Scheduling (tt)

- Vấn đề: trì hoãn vô hạn định – process có độ ưu tiên thấp có thể không bao giờ được thực thi
- Giải pháp: làm mới (aging) – độ ưu tiên của process sẽ tăng theo thời gian

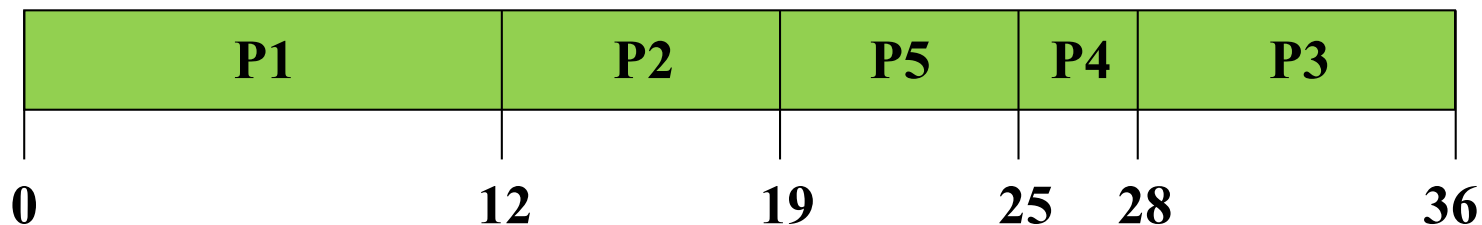




# Non-preemptive Priority Scheduling

Process	Arrival Time	Burst Time	Priority
P1	0	12	2
P2	2	7	1
P3	5	8	5
P4	9	3	4
P5	12	6	3

## ■ Giải đồ Gantt



## ■ Thời gian chờ, đáp ứng, hoàn thành?



# Tóm tắt lại nội dung buổi học

- Các khái niệm cơ bản về định thời
- Các bộ định thời
- Các tiêu chuẩn định thời CPU
- Các giải thuật định thời

First-Come, First-Served (FCFS)

Shortest Job First (SJF)

Shortest Remaining Time First (SRTF)

Priority Scheduling



# Bài tập 1

- Sử dụng các giải thuật FCFS, SJF, SRTF, Priority để tính các giá trị thời gian đợi, thời gian đáp ứng và thời gian hoàn thành trung bình

Process	Arrival Time	Burst Time	Priority
P1	0	4	3
P2	5	1	2
P3	3	8	1
P4	10	2	4
P5	8	7	3



## Bài tập 2

- Sử dụng các giải thuật FCFS, SJF, SRTF, Priority để tính các giá trị thời gian đợi, thời gian đáp ứng và thời gian hoàn thành trung bình

Thread	Priority	Burst	Arrival
$P_1$	40	20	0
$P_2$	30	25	25
$P_3$	30	25	30
$P_4$	35	15	60
$P_5$	5	10	100
$P_6$	10	10	105



COMPUTER ENGINEERING



**UIT**  
TRƯỜNG ĐẠI HỌC  
CÔNG NGHỆ THÔNG TIN

# THẢO LUẬN

