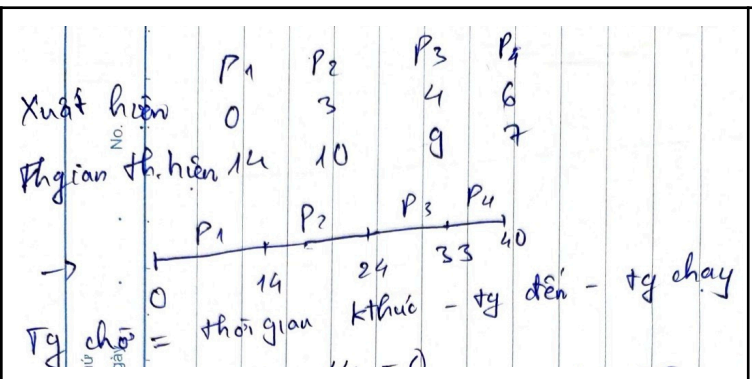
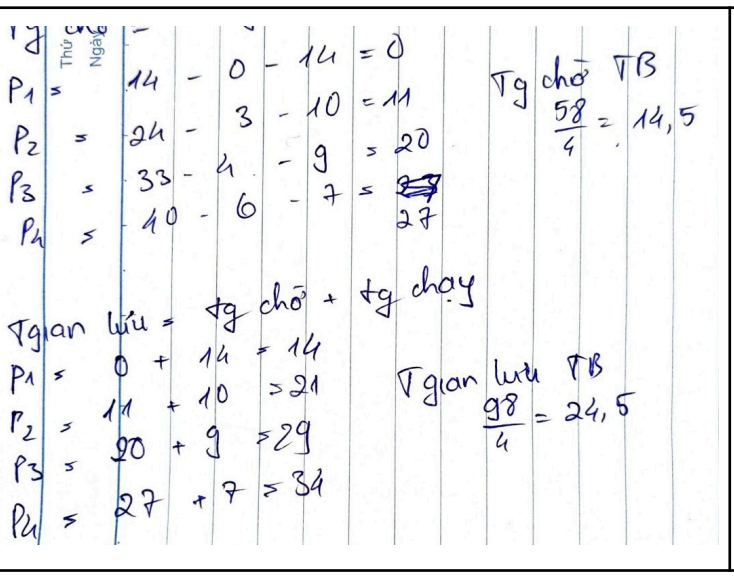
Chương 3: Quản lý tiến trình
First Come First Serve

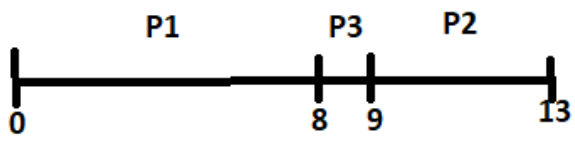




Shortest Job Next (Shortest Job First non-Preemptive - không chiếm đoạt)

- Khi có nhiều hơn 1 tiến trình đang chờ:
+ Tiến trình có thời gian chạy ít hơn sẽ được ưu tiên chạy trước
+ Nếu 2 tiến trình có th.gian chạy bằng nhau thì ưu tiên tiến trình đến trước sẽ chạy trước

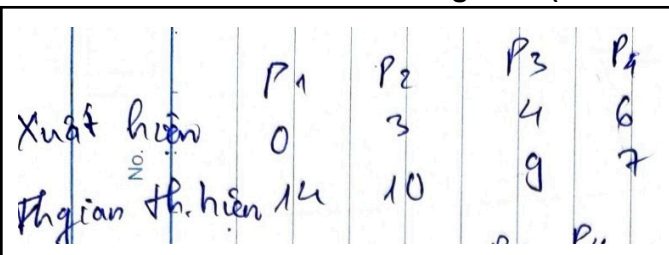
Process	Arrival Time	Burst Time
P1	0.0	8
P2	0.4	4
P3	1.0	1

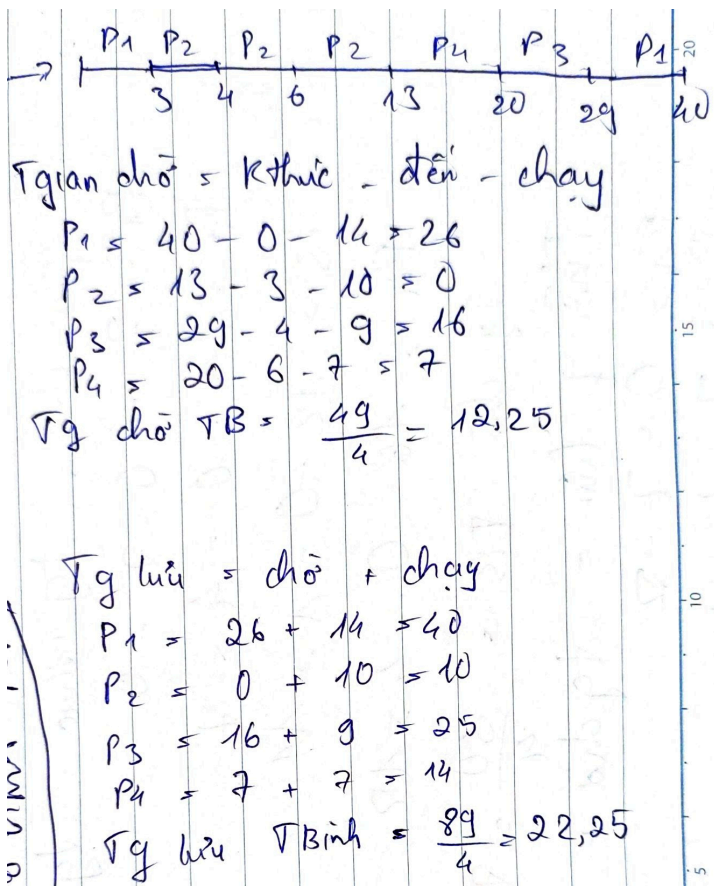


Thời gian lưu của từng tiến trình
P1 = 8 - 0 = 8
P2 = 13 - 0.4 = 12.6
P3 = 9 - 1 = 8

P_{TB} = (8 + 12,6 + 8) / 3 = 9,533

Shortest Remaining Next (Shortest Job First Preemptive - chiếm đoạt)





- Khi có nhiều hơn 1 tiến trình đang chờ:
+ Tiến trình

- Shortest remaining time first

Lập lịch với độ ưu tiên

Chương 5: Bế tắc
Banker Algorithm

Chương 6: Quản lý bộ nhớ
Chuyển đổi địa chỉ ảo ↔ Địa chỉ vật lý

<table><tr><td>0</td><td>6</td></tr><tr><td>1</td><td>9</td></tr><tr><td>2</td><td>8</td></tr><tr><td>3</td><td>7</td></tr><tr><td>4</td><td>1</td></tr><tr><td>5</td><td>2</td></tr></table>	Địa chỉ ảo 4567 có địa chỉ vật lý là bao nhiêu Page size = Frame size = 1KB = 1024B - Bài làm: Tìm số hiệu trang = 4567 / 1024 = 4 (lấy phần nguyên) → frame số 1 (Nhìn bảng trang, cột trái là số hiệu trang, cột phải là số hiệu frame) Độ lệch trang = 4567 % 1024 = 471 Địa chỉ vật lý = số hiệu frame * kích thước frame + độ lệch = 1 * 1024 + 471 = 1495	Cho địa chỉ vật lý 6789 tìm địa chỉ ảo (địa chỉ luận lý) Tìm số hiệu frame = 6789 / 1024 = 6 → số hiệu trang là 0 Độ lệch 6789 % 1024 = 645 Địa chỉ ảo = 0 * 1024 + 645 = 645
		5*1024 + 709 = 5765

1, Vẽ bảng kết quả phân phối bộ nhớ cho các tiến trình theo 3 phương pháp là FF, BF, WF
2, Tìm địa chỉ ảo/ thật
3, Tìm địa chỉ vật lý theo phương pháp phân đoạn

Segment	Base	Length
0	219	600
1	2300	14
2	90	100
3	1327	580
4	1952	96

Cho biết địa chỉ vật lý tương ứng với các địa chỉ logic sau đây:
a, 0, 430
b. 1, 10
c. 2, 500
d. 3, 400
e. 4, 112

FF	BF
a, 649 (430 + 219) b, 2310 (10 + 2300) c, 1827 (500 + 1327) d, x e, x	a, 1757 (430 + 1327) b, 2310 (10 + 2300) c, 1827 (500 + 1327) d, x e, x

Chương 7: Bộ nhớ ảo
Thuật toán vào trước ra trước (FIFO)

	1	2	3	4	3	5	7	1	4	7	5	2	1	6	3	2	4	7	3	5
F1	1	1	1	4	4	4	4	1	1	1	1	2	2	2	3	3	3	7	7	7
F2		2	2	2	2	5	5	5	4	4	4	4	1	1	1	2	2	2	3	3
F3			3	3	3	3	7	7	7	7	5	5	5	6	6	6	4	4	4	5
F	x	x	x	x		x	x	x	x		x	x	x	x	x	x	x	x	x	x

- Để ý chuỗi liên tục của các trang. Tại lần xuất hiện của số 4 (vàng), trang số 1 đang có chuỗi xuất hiện dài nhất → trang 4 sẽ thay trang số 1. Trang số 3 (vàng) xuất hiện và thấy trang 3 cũng đang có frame rồi thì tiếp tục chạy trang 3 trên frame đó

Thuật toán tối ưu (Optimal Algorithm)

	1	2	3	4	1	2	5	1	2	3	4	5
F1	1	1	1	1	1	1	1	1	1	3	3	3

Thuật toán LRU (Least Recently Used)

	1	2	3	4	1	2	5	1	2	3	4	5
F1	1	1	1	4	4	4	5	5	5	3	3	3

F2		2	2	2	2	2	2	2	2	2	4	4
F3			3	4	4	4	5	5	5	5	5	5
F	x	x	x	x			x			x	x	

Tại thời điểm có trang mới vào mà không có frame trống, trang được thay sẽ là trang xa nhất nhìn từ trái sang phải tại trang đang được xét. Nhìn vào lần xuất hiện đầu tiên của trang 4 (màu vàng), khi nhìn từ trái sang phải ta thấy trang số 3 ở xa nhất (1 → 2 → … → 3). Nhìn vào lần xuất hiện của trang 3 (màu vàng), nhìn sang phải ta chỉ thấy trang số 5, thì coi trang 1, 2 là rất xa (trường hợp này thì chọn trang nào cũng được, thầy Toàn quy ước chọn trang có số nhỏ hơn)

F2		2	2	2	1	1	1	1	1	1	4	4
F3			3	3	3	2	2	2	2	2	2	5

- Tương tự Optimal Algorithm, nhưng sẽ nhìn từ phải sang trái

Ví dụ khác về LRU

	1	2	3	4	3	5	7	1	4	7	5	2	1	6	3	2	4	7	3	5
F1	1	1	1	4	4	4	7	7	7	7	7	7	1	1	1	2	2	2	3	3
F2		2	2	2	2	5	5	5	4	4	4	2	2	2	3	3	3	7	7	7
F3			3	3	3	3	3	1	1	1	5	5	5	6	6	6	4	4	4	5
F	x	x	x	x		x	x	x	x		x	x	x	x	x	x	x	x	x	x