Name: Shankar Acharya
Roll No.: 2014SE35
Sub : AOS Assignment 2

2015 Fall

3b) Consider the following Snapshot of a System

Processes	Allocation				Max				Available			
	A	B	C	D	A	B	C	D	A	B	C	D
P_0	0	0	1	2	0	0	1	2	1	5	2	0
P_1	1	0	0	0	1	7	5	0				
P_2	1	3	5	4	2	3	5	6				
P_3	0	6	3	2	0	6	5	2				
P_4	0	0	1	4	0	6	5	6				

Answer the following questions using Banker's Algorithm:

i. What is the Content of the matrix need?

⇒ we know content of Need matrix will be

Need = Max − Allocation, So,

Need			
A	B	C	D
0	0	0	0
0	7	5	0
1	0	0	2
0	0	2	0
0	6	4	2

ii. Is the system in safe state find safe sequen

we have

Process	Allocation				Max				Need			
	A	B	C	D	A	B	C	D	A	B	C	D
P_0	0	0	1	2	0	0	1	2	0	0	0	0
P_1	1	0	0	0	1	7	5	0	0	7	5	0
P_2	1	3	5	4	2	3	5	6	1	0	0	2
P_3	0	6	3	2	0	6	5	2	0	0	2	0
P_4	0	0	1	4	0	6	5	6	0	6	4	2

A B C D

let work = (1 5 2 0)

- P_0 is complete.
- for P_2 Need$_i$ < work so, resource is provided and when finished taken back

work = (2 8 7 4)

- for P_3 Need$_i$ < work so, new work = (2. 14 10
- for P_4 Need$_i$ < work so new work = (2 14 11
- for P_1 Need$_i$ < " " " " = (3 14 11,

So, safe sequence is P_0 P_2 P_3 P_4 P_1

iii) If the request from process P_1 arrives for (0,4,2,0) can the request be granted immediately.

⇒ As request (0,4,2,0) < Available. The request can be granted but may not produce safe sequence

Process	Allocation				Max				Need				Available			
	A	B	C	D	A	B	C	D	A	B	C	D	A	B	C	D
P_0	0	4	3	2	0	4	3	2	0	0	0	0	1	1	0	0
P_1	1	0	0	0	1	7	5	0	~~0~~	~~7~~	~~5~~	~~0~~				
P_2	1	3	5	4	2	3	5	6	1	0	0	2				
P_3	0	6	3	2	0	6	5	2	0	0	2	0				
P_4	0	0	1	4	0	6	5	6	0	6	4	2				

1532

2886

So, safe sequence is P_0, P_2, P_3, P_4 & P_1

Yes, the request can be granted

2014 Fall

4(b) Describe ~~the~~ three circumstances under which blocking I/O should be used. Describe three circumtance under which non-blocking I/O should be used. Why not just implement non-blocking I/O and have processes busy-wait until their device is ready? Put your insight.

-> Generally blocking I/O is appropriate when the process will only be waiting for one specific event. Example include a disk, tape, or keyboard read by an application program. Non-blocking I/O is useful when I/O may come from more than one source and the order of the I/O arrival is not predetermined. Examples include network daemons listening to more than one network socket, window managers performance by buffering the input and output and using non-blocking I/O to keep both devices fully occupied. Non-blocking I/O is more complicated for programmers, because of the asynchronous rendezvous that is needed when an I/O occurs. Also, busy waiting is less efficient than Interrupt-driven I/O so the overall system performance would decrease.

4 fall

2.a) Give two reasons why caches are useful. What problems do they solve? What problems do they cause? If a cache can be made as large as the device for which it is caching (for instance, a cache as large as a disk), why not make it that large and eliminate device? Discuss.

→ Caches are useful when two or more components need to ex-change data, and the components perform transfers at different speeds. Caches solve the transfer problem by providing a buffer of intermediate speed between the components. If the fast device finds the data it needs in the cache, it need not wait for the slower device. The data in the cache must be kept consistent with the data in components. If a component has a data value change, and the datum is also in the cache, the cache should also be updated. This is especially a problem on multiprocessor system where more than one process may be accessing a datum.

A component may be eliminated by an equal sized cache, but only if:

(a) the cache and the component have equivalent state-saving capacity (that is, if the component retains its data when electricity is removed, the cache must retain data as well),

(b) the cache is affordable, because faster storage tends to be more expensive.

2014 Fall Q.a MS DOS provided no means of concurrent processing. Discuss three major complications that concurrent processing adds to an operating system. Discuss all important aspects.

→ The three major complications that concurrent processing adds to an operating system are

i) System resources as well as processes need protection, even from each other. Limitations need to be imposed on the processes on factors like memory they can access and operations it can perform on devices. The operating system needs to keep track of memory space allocated to all the processes, so that the processes do not harm other processes.

ii) If a process running in the system require a lot of memory, other processes need to be shifted to the hard disk, which require a lot of time, contributing to the time overhead on the operating system.

iii) All running processes must have access to the system, which should be made possible by implementing time sharing. This requires preemption of processes that do not give up CPU and kernel on their own. And switching from one process to another also contributes to the time overhead.

2016 Spring 7c) Write short notes on File Protection Methods.

→ When information is stored in a computer system we want to keep it safe from physical damage (issue of reliability) and improper access (issue of protection). Reliability is provided by duplicating the file. Protection from hardware problems (such as error in reading and writing, power surges or failures, head ~~etc~~ crashes, dirt, tempreature extremes and vandalism.

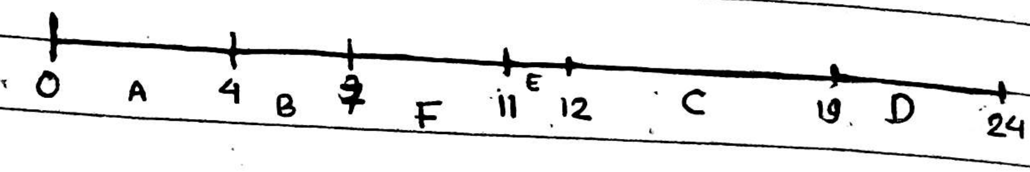
Protection can be provided by defining Access Control List (ACL), using seperate user accounts and specifying permissions for each users.

16 spring 3.b) Consider following set of processes along with their burst time, arrival time and priorities. Calculate the average waiting time, and average turnaround time using following scheduling algorithms

i. FCFS ii. SJF iii. HRRN iv. Priority (Preemptive)

Process	Arrival Time	Burst Time	Priority
A	0	4	5
B	2	3	4
C	5	7	1
D	7	5	3
E	4	1	2
F	3	4	1

i) FCFS



So, Avg. waiting time = ~~0+4+7~~

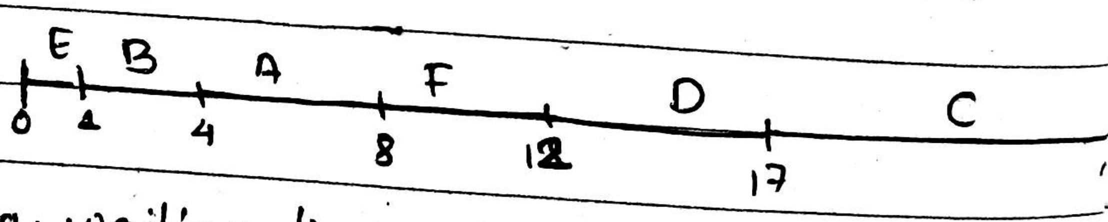
$$= \frac{0+(4-2)+(7-3)+(11-4)+(12-5)+(19-7)}{6}$$

$$= 5.33$$

Avg. turn around time $= \frac{(4-0)+(7-2)+(11-3)+(12-4)+(19-5)+(24-7)}{6}$

$$= 9.33$$

ii) SJF



Avg. waiting time $= \frac{0+1+4+8+12+17}{6} = 7$

Avg. Turn around time $= \frac{1+4+8+12+17+24}{6} = 11$

iii) Priority (preemptive)

A	B	F	C	E	D	B	A
0–2	2–3	3–7	7–14	14–15	15–20	20–22	22–24

so,

$$\text{Avg. waiting time} = \frac{(22-2) + [(20-3)] + (7-5) + (15-7) + (14-4) + (3-3)}{6}$$

$$= 9.5$$

$$\text{Avg. turnaround time} = \frac{24 + (22-2) + (14-5) + (20-7) + (15-4) + (7-3)}{6}$$

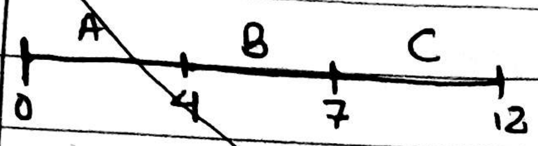
$$= 13.5$$

iv) HRRN (Highest response ratio next)

we have,

$$\text{new Priority for HRRN} = 1 + \frac{\text{waiting time}}{\text{burst time}}$$

so



~~at time 7, both C & D have Arrived~~

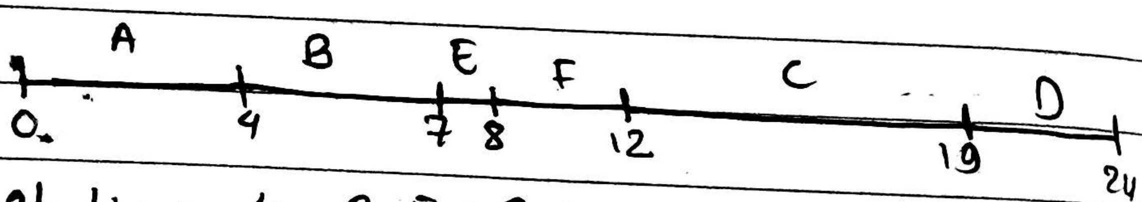
~~Priority of C = $1+\frac{2}{7}$ = 1.285 ✓~~

~~Priority of D = $1+\frac{0}{5}$ = 1~~

~~at time 12, D, E, F have arrived~~

~~Priority of D = 1+ 5/5 = 2~~

~~" " E = 1+~~



at time 4, B, E & F have arrived

so, Priority of B = $1+\frac{2}{3}$ = 1.75 ✓

" of E = $1+\frac{0}{1}$ = 1

" " F = 1 + 1/4 = 1.25

At time = 7 C, D, E, & F have arrived

so, Priority of C = 1+ 12/7 = 1.02

D = 1 + 0/5 = 1

E = 1 + 3/1 = 4 ✓

F = 1 + 4/4 = 2

At time = 8

Priority of C = $1 + 3/7$

" " D = $1 + 1/5$

" " F = $1 + 5/4 = 2.25$ ✓

At time = 12,

priority of C = $1 + 7/7 = 2$

" " D = $1 + 5/5 = 2$

So, both have equal priority any one can be choosed.

So, Avg. waiting time = $\dfrac{0+(\overset{4}{\cancel{7}}-2)+(1\overset{2}{\cancel{8}}-5)+(19-7)+(7-4)+(8-3)}{6}$

$= 4.833$

Avg. Turnaround time = $\dfrac{4+(7-2)+(19-5)+(24-7)+(8-4)+(12-3)}{6}$

$= 8.833$