

operating system

1) Turn around Time (A) = Finish Time (A) - Arrival Time (A) = 3.000 - 0.000 = 3.000
 Turn around Time (B) = Finish Time (B) - Arrival Time (B) = 9.000 - 1.001 = 7.999
 Turn around Time (C) = Finish Time (C) - Arrival Time (C) = 13.000 - 4.001 = 8.999
 Turn around Time (D) = Finish Time (D) - Arrival Time (D) = 15.000 - 6.001 = 8.999

Average Turn around Time (FCFS) = $(3.000 + 7.999 + 8.999 + 8.999) / 4$
 = 7.249875

waiting time (A) = (3.000 - 3.000) = 0.000

waiting time (B) = 7.999 - 6 = 1.999

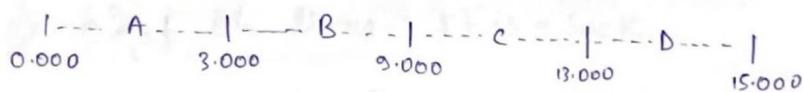
waiting time (C) = 8.999 - 4 = 4.999

waiting time (D) = 6.999

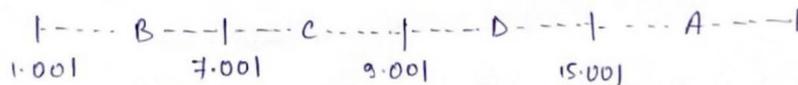
Average waiting

Time = $(1.999 + 4.999 + 6.999) / 4$
 = 3.49925

Gantt chart for FCFS



Gantt chart for LRTF



Turn around time (B) = 7.001 - 1.001 = 6.000

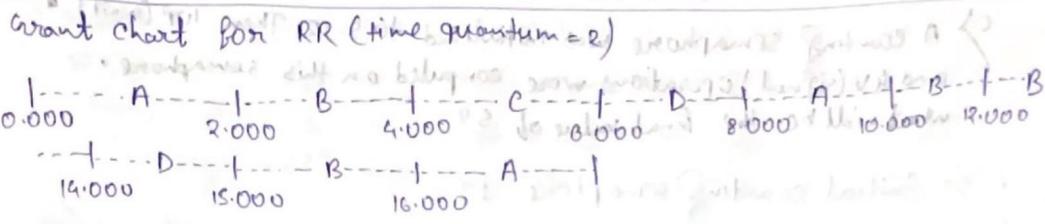
Turn around time (C) = 9.001 - 4.001 = 5.000

Turn around time (D) = 15.001 - 6.001 = 9.000

Turn around time (A) = 15.001 - 0.000 = 15.001

Turn around time for LRTF = $(6.000 + 5.000 + 9.000 + 15.001) / 4 = 8.75025$

all waiting time for LRTF is 0. so average waiting time is 0.



waiting time (A) = 16.000 - 3 = 13.000
 waiting time (B) = 10.999 - 6 = 4.999
 waiting time (C) = 1.999
 waiting time (D) = 5.999

Turn around time (A) = 13.000
 Turn around time (B) = 10.999
 Turn around time (C) = 5.999
 Turn around time (D) = 7.999

Average waiting time for RR = $\frac{2 \times 5.999}{4}$
 = 6.99925

Average Turn around time RR = $\frac{40.999}{4}$
 = 10.24925

a) Difference between short term & long term scheduler

- | <u>short-term</u> | <u>Long-term</u> |
|--|--|
| i) It is CPU scheduler. | i) It is a Job scheduler. |
| ii) It selects processes from ready queue which are ready to execute & allocates CPU to one of them. | ii) It selects processes from job pool & loads them into memory for execution. |
| iii) Speed is fast. | iii) Speed is less than short-term scheduler. |
| iv) Access ready queue & CPU. | iv) Access job pool & ready queue. |
| v) It executes frequently. It executes when CPU is available for allocation. | v) It executes much less frequently. It executes when memory has space to accommodate new process. |

b) Difference between synchronous & asynchronous Multi-processing

- | <u>synchronous</u> | <u>Asynchronous</u> |
|--|---|
| i) sequential eqn - In synchronous multiprocessing task or processes are executed one after other in pre-determined order. | i) Parallel eqn - In Asynchronous Multi-processing tasks run independent or concurrently. |
| ii) Blocking in nature. | ii) non-blocking. |
| iii) eqn order is predictable & easy to understand. | iii) eqn order of task if not predetermined & may vary each time program runs. |
| iv) Straight forward implement. | iv) well situated for tasks that involve I/O operation. |

c) A counting semaphore was initialized to 5-12. Then 10P (wait) and 4V (signal) operations were computed on this semaphore, what will be the final value of S?

⇒ Initial counting semaphore 12

wait operation = 10P

signal = 4V

Final value $F = 12 + 10 \times P + 4 \times V$

$$= 12 + 10 \times (-1) + 4 \times (+1)$$

$$= 12 - 10 + 4 = 6$$

d) Match the following.

A. NUMA → Multicore processor (5)

B. Aging → Priority scheduling (3)

C. context switching Overhead → Round Robin (2)

D. Convoy Effect → FCFS (1)

E. Mutual Exclusion → Peterson's solution (4)

e) suppose a computing unit consists with 50% Parallel and 50% serial components. Also let this contains 8 processing core. what will be the max speed-up?

⇒ The parallelizable 50 units happen in $50/8 = 6.25$ unit of time.

$$\text{total duration} = (50 + 6.25) = 56.25$$

$$\text{speed up} = \frac{(\text{unparallelizable total runtime})}{(\text{partially parallelized total runtime})}$$

$$= \frac{100}{56.25} = 1.778$$

3.a.i) True

ii) False

iii) False

iv) True.

b. Fill in the blanks.

i) Execution Time

ii) Preemption

iii) NUMA

iv) Thread control Block

Q) suppose $n = \lambda * w$ (where, λ is the average no. of processors arrive into the ready queue; and w is the avg waiting time) what is n ? In which situation the system remains in steady state? when the system may become unsteady?

⇒ Little law states that $L = \lambda * w$

n is the average number of process in ready queue.
A steady state situation occurs when the system is in balance, meaning that the rate of which process arrive into the ready queue is equal to the rate of process are being executed & removed from the ready queue.

In an unsteady state n will be fluctuate & the system may experience backlogs.

Q) If func1 execute first -

$$A \text{ becomes } 10 - 5 = 5$$

$$C \text{ " } 5 * 5 = 25$$

If func2 execute first -

$$A \text{ becomes } 5 * 10 = 50$$

$$C \text{ " } 50 - 5 = 45$$

If func2 execute after func1 -

$$A \text{ becomes } 5 * 10 = 50$$

$$B \text{ " } 50 - 5 = 45$$

If func1 executes after func2 -

$$A \text{ becomes } 45 - 5 = 40$$

$$C \text{ " } 5 * 40 = 200$$

∴ The sum of these distinct value's = $25 + 45 + 200 = 270$