GATE PSUs

State Engg. Exams

MADE EASY
Workbook 2024

Detailed Explanations of Try Yourself *Questions*

Computer Science & IT

Operating System

MADE EASY PUBLICATIONS

1 Introduction & Background of Operating System

Detailed Explanation *of* Try Yourself Questions

T1 : Solution

(d)

When a computer is switched on the operating system is loaded in RAM and its execution starts by searching essential programs.

T2 : Solution

(b)

The process of loading the operating system into the memory of a PC is called booting.

T3 : Solution

(c)

Supervisory calls are privileged calls that are used to perform resource management functions, which are controlled by OS.

T4 : Solution

(c)

We need a seperate program for compilation of programs that program is known as Compiler.

T5 : Solution

(c)

In a multiprogramming environment more than one process resides in the memory. They run in preemptive mode. Either priority based or in time sharing mode.

■■■■

2 Processes and Threads

Detailed Explanation of Try Yourself Questions

T1 : Solution

(c)

Medium-term scheduler is involved only in the decision for selection of partially serviced jobs.

T2 : Solution

(d)

Medium-term scheduler can transit a job from 'ready' to 'suspended ready'.
If event occurs for suspend blocked job then which can move to 'suspend ready'.
Medium term scheduler can transit a job from 'suspend ready' to 'ready'.
$\therefore$ All given statements are correct.

T3 : Solution

(b)

Only the suspended processes and suspended blocked will reside in secondary memory. The processes in the remaining states will reside in main memory.

T4 : Solution

(c)

Kernel is the set of primitive functions upon which the rest of the operating system functions are build up.

T5 : Solution

(c)

Swapping out the memory image of process A to the disk typically not performed by OS when switching context from process A to B.

T6 : Solution

(d)

All the three statement are true.

T7 : Solution

(d)

When an interrupt occurs, an operating system may change the state of the interrupted process to “blocked” and schedule another process.

T8 : Solution

(32)

When $k = 5$, 1 child process is created.

When $k = 4$, 2 child process are created (1 + 1).

When $k = 3$, 4 child process are created (2 + 2).

When $k = 2$, 8 child process are created (4 + 4).

When $k = 1$, 16 child process are created (8 + 8).

and 1 parent process.

Therefore total number of processes = 1 + 16 + 8 + 4 + 2 + 1 = 32

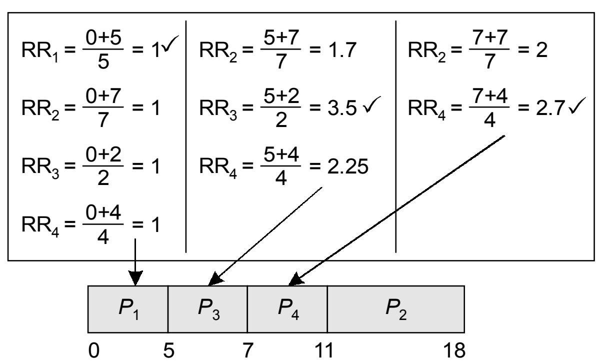
■■■■

3 CPU Scheduling

Detailed Explanation *of* Try Yourself Questions

T1 : Solution

(b)



$$\text{TAT}_1 = 5$$
$$\text{TAT}_2 = 18$$
$$\text{TAT}_3 = 7$$
$$\text{TAT}_4 = 11$$

$$\text{Average TAT} = \frac{5+18+7+11}{4} = \frac{41}{4} = 10.25$$

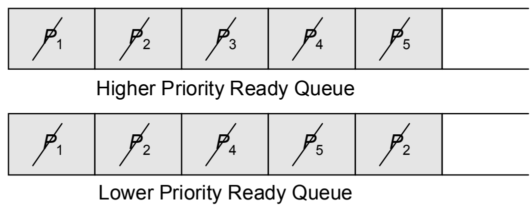
T2 : Solution

(a)

$$S(n+1) = \alpha \cdot T(n) + (1-\alpha) \cdot S(n)$$
$$S(4) = 0.8 \times T(3) + (1-0.8) \times S(3)$$
$$= 0.8 \times 4 + 0.2 \times 5$$
$$= 3.2 + 1.0 = 4.2$$

T3 : Solution

(d)



P_1	P_2	P_3	P_4	P_5	P_1	P_2	P_4	P_5	P_2

0 2 4 6 8 10 11 13 15 16 17

$$\text{TAT} = \text{Completion time} - \text{Arrival time}$$

$$\text{TAT}_1 = 11 - 0 = 11$$

$$\text{TAT}_2 = 17 - 2 = 15$$

$$\text{TAT}_3 = 6 - 3 = 3$$

$$\text{TAT}_4 = 15 - 5 = 10$$

$$\text{TAT}_5 = 16 - 6 = 10$$

$$\text{Average TAT} = \frac{11 + 15 + 3 + 10 + 10}{4} = \frac{49}{4} = 9.8$$

T4 : Solution

(a)

$$\text{Response ratio} = \frac{\text{Wait time} + \text{Service time}}{\text{Service time}}$$

A process which has highest Response Ratio is selected to schedule next. It increases response to processes.

T5 : Solution

(c)

FCFS: The process run in the order they arrived.

RR: Every process get a chance to execute.

SRTF: Minimizes the average waiting time.

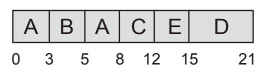
Priority: Important processes get execute first.

T6 : Solution

(7.2)

	A.T	E.T
A	0	6
B	3	2
C	5	4
D	7	6
E	10	3

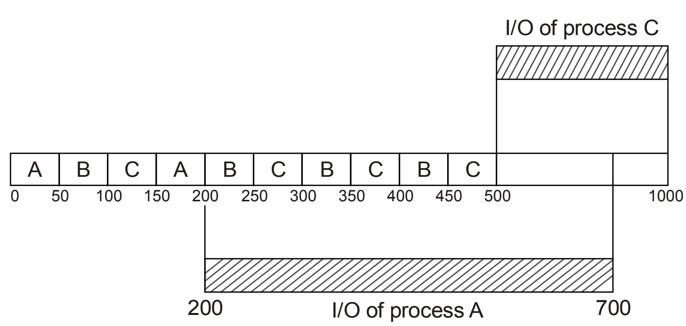
Using SRTF:



$$\text{T.A.T (A)} = 8 - 0 = 8$$
$$\text{T.A.T (B)} = 5 - 3 = 2$$
$$\text{T.A.T (C)} = 12 - 5 = 7$$
$$\text{T.A.T (D)} = 21 - 7 = 14$$
$$\text{T.A.T (E)} = 15 - 10 = 5$$
$$\text{Average T.A.T} = \frac{8+2+7+14+5}{5} = 7.2$$

T7 : Solution

(1000)



∴ At 1000 time units C completes its I/O

T8 : Solution

(12)

Periodic arrival times of T_1 : 0, 3, 6, 9, 12, 15, 18, 21,...

Priority of $T_1 = \frac{1}{3}$, service time of $T_1 = 1$.

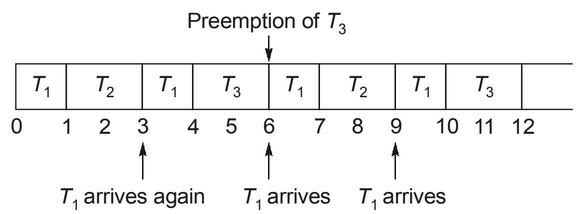
Periodic arrival times of T_2 : 0, 7, 14, 21,...

Priority of $T_2 = \frac{1}{7}$, service time of $T_2 = 2$.

Periodic arrival times of T_3 : 0, 20, 40,...

Priority of $T_3 = \frac{1}{20}$, service time of $T_3 = 4$.

T_1 has highest priority and T_3 has lowest priority.

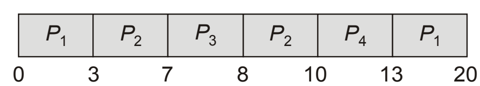


First instance of T_3 (4 units) completed at the end of 12 ms.

T9 : Solution

(8.25)

Gantt chart



Process Number	Arrival time	Burst time	Completion time	TAT
1	0	10	20	20
2	3	6	10	7
3	7	1	8	1
4	8	3	13	5
Average turn around time : 33/4 = 8.25				

Average turn around time is 8.25.

■■■■

4 Process Synchronization

Detailed Explanation of Try Yourself Questions

T1 : Solution

(b)

Process-3 depends on shared variable *B*. Initial value of *A* is 3. Process-1 can execute atmost three times successfully *P*(*A*), then fourth time will be blocked. So *V*(*B*) is executed maximum 3 times. Now process-3 can execute *P*(*B*) atmost 3 times successfully and fourth time it will be blocked. So process-3 prints "3" maximum three times in the execution.

T2 : Solution

(a)

(*i*) $A = 3, B = 0$
First process-1 execute three times and process-1 is blocked ($A = 0, B = 3$).
(*ii*) Process-3 executes next three times and process-3 is blocked ($A = 0, B = 0$).
(*iii*) Process-2 executes last then it will be blocked by executing *P*(*B*) first time.
∴ No '1' is printed by the execution of three processes and all are blocked.
Number of 1's printed = 0

T3 : Solution

(c)
Mutual exclusion guaranteed but deadlock occurs.
(*i*) P_1 executes $P(S1)$ then preempts
(*ii*) P_2 executes $P(S2)$ then preempts
(*iii*) P_3 executes $P(S3)$ then preempts
(*iv*) P_1 executes $P(S2)$ then P_1 blocked
(*v*) P_2 executes $P(S3)$ then P_2 blocked
(*vi*) P_3 executes $P(S1)$ then P_3 blocked
∴ Deadlock occurs.

T4 : Solution

(d)

$$S_1 = 0, S_2 = 1, S_3 = 0 \Rightarrow (0, 1, 0)$$

2	1	0	2	1	0	2	1	0	2	1
(0, 1, 0) $P_3 \Rightarrow$ "2" (1, 0, 0)	(1, 0, 0) $P_1 \Rightarrow$ "1" (0, 0, 1)	(0, 0, 1) $P_1 \Rightarrow$ "0" (0, 1, 0)	same as 1	same as 2	same as 3	same as 1	same as 2	same as 3	same as 1	same as 2

T5 : Solution

(b)

Start	7*V*	5*P*	14*V*	10*P*	21*V*	15*P*
S = 1	1	0	1	0	1	0
# Blocked = 0	0	4	0	9	0	14

Number of blocked processes = 14

T6 : Solution

(d)

P and Q can execute in any sequence therefore the string generated by two processes is (1+0)*.

T7 : Solution

(d)

Case-1: $1^+ 0^*$ that is any number of times P followed by Q any number of times.

Case-2: Alternate execution of P followed by Q every time (10)*.

Combining case 1 and 2 the string generated by two processes is $(1^+0^* + (10)^*)$

T8 : Solution

(b)

All processes are in deadlock condition i.e., not even a single process can enter into critical section.

T9 : Solution

(10)

Everytime when process P_{10} enters it performs the 'UP' operation so after that one more process can enter into it, so this phenomena continue till end.

So 10 processes at a time can enter into critical section.

T10 : Solution

(c)

Consumer executes wait(S) then wait(n) and goes to sleep by decreasing n value.

After consumer sleep, producer goes to the sleep by executing wait(S).

T11 : Solution

(a)

It satisfies the mutual exclusion, so only one process can be in the critical section at any time.

T12 : Solution

(7)

$$S - 20 + 12 = -1$$
$$S - 8 = -1$$
$$S = -1 + 8 = +7$$

So, the initial values of the semaphore should be '7'.

■■■■

5 Deadlocks

Detailed Explanation of Try Yourself Questions

T1 : Solution

(d)

(a) X = 40, Y = 20 ⇒ (P_1, P_2, P_3, P_4) = Need (25, 20, 20, 20)

Total resource units = 150

Currently allocated 45 + 40 + 40 + 20 = 145

⇒ Available = 5 units

No process can satisfy its need

(b) X = 50, Y = 10 ⇒ Need = (25, 20, 10, 30)

Current allocation = 45 + 40 + 50 + 10 = 145

Available = 5 units

⇒ No process can satisfy its need.

(c) X = 30, Y = 20

Allocation = (45, 40, 30, 20)

Need = (25, 20, 30, 20)

Available = 150 – (45 + 40 + 30 + 20) = 15

No process can satisfy its need.

(d) X = 20, Y = 30

Allocation = (45, 40, 20, 30)

Need = (25, 20, 40, 10)

Available = 150 – (45 + 40 + 20 + 30) = 15

P_4 can satisfy its need

Available = 15 + 30 = 45

Now P_1 or P_2 or P_3 can satisfy and they can finish in any order.

∴ Safe sequence exist for X = 20, Y = 30.

T2 : Solution

(a)

	Need		
	R_1	R_2	R_3
P_1	1	1	0
P_2	1	0	0
P_3	1	0	1
P_4	1	1	0
P_5	2	0	0

Available = $(1, x, 1)$. If $x = 0$, the system will be in safe state

$$
\begin{aligned}
\text{Available} &= (1,\ 0,\ 1) \\
P_2 &\rightarrow \underline{(0,\ 1,\ 0)} \\
&\quad\ (1,\ 1,\ 1) \\
P_1 &\rightarrow \underline{(0,\ 0,\ 1)} \\
&\quad\ (1,\ 1,\ 2) \\
P_3 &\rightarrow \underline{(1,\ 2,\ 3)} \\
&\quad\ (2,\ 3,\ 5) \\
P_4 &\rightarrow \underline{(0,\ 1,\ 1)} \\
&\quad\ (2,\ 4,\ 6) \\
P_5 &\rightarrow \underline{(1,\ 0,\ 1)} \\
&\quad\ \underline{(3,\ 4,\ 7)}
\end{aligned}
$$

There is safe sequence.

$\therefore$ Minimum zero units of R_2 is guarantee deadlock free.

■■■■

6 Memory Management

Detailed Explanation of Try Yourself Questions

T1 : Solution

(a)

In buddy system allocation ,the size of memory blocks allocated is in power of 2 (smallest sufficient to service the memory allocation request).

Hence the first six allocations are 64 KB, 256 KB, 128 KB, 256 KB, 128 KB, 128 KB. This adds upto 960 KB and hence the next request of 120 KB will not be allocated (the first request to fail).

T2 : Solution

(c)

$$16\text{ KB} = 2^{14}\text{ bytes} \Rightarrow 14\text{ bit offset}$$

$$2^{11} * 2^{11} * 2^{11} * 2^{14} = 2^{47}\text{ bytes}$$

11	11	11	14
Level1	Level2	Level3	Offset

⇒ 3 levels

|← 47 →|

T3 : Solution

(d)

158 = 100 11110

Page 4 in frame 2 ⇒ 010 11110 (94)

53 = 001 10101

Page 1 in frame 7 ⇒ 111 10101 (245)

125 = 011 11101

Page 3 is invalid ⇒ Page fault

167 = 101 00111

Page 5 in frame 1 ⇒ 001 00111 (39)

T4 : Solution

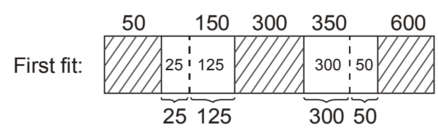
(d)

Link editor is another name of 'linker'. Linker is a program which links the object code with it's library.

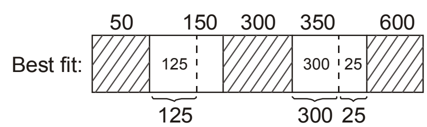
T5 : Solution

(b)

Requests: 300, 25, 125, 50



⇒ First fit can satisfy all the requests.



⇒ 50 can not be satisfied by best fit

■■■■

7 Virtual Memory

Detailed Explanation of Try Yourself Questions

T1 : Solution

(1.76×10^{-4})

$$
\begin{aligned}
10^{-6} &= (1-F)\,120 \times 10^{-9} + F \times 5 \times 10^{-3} \\
10^{-6} &= (1-F)\,120 \times 10^{-6} + 5000\,F \times 10^{-6} \\
1 &= 0.120 - 0.120\,F + 5000\,F \\
0.88 &= -120\,F + 5000\,F \\
0.88 &= 4999.88\,F \\
F &= 1.76 \times 10^{-4}
\end{aligned}
$$

T2 : Solution

(c)

3 page frame

0	9	0	1	8	1	8	7	8	7	1	2	8	2	7	8	2	3	8	3
			1	1	1	1	1	1	1	1	2	2	2	2	2	2	2	2	2
	9	9	9	9	9	9	7	7	7	7	7	7	7	7	7	7	7	8	8
0	0	0	0	8	8	8	8	8	8	8	8	8	8	8	8	8	3	3	3
F	F		F	F		F					F						F	F	

FIFO = 8

Since FIFO is 8 so option (a) and (b) can be answer.

0	9	0	1	8	1	8	7	8	7	1	2	8	2	7	8	2	3	8	3
			1	1	1	1	1	1	1	1	1	1	1	7	7	7	3	3	3
	9	9	9	8	8	8	8	8	8	8	2	2	2	2	2	2	2	2	2
0	0	0	0	0	0	0	7	7	7	7	7	8	8	8	8	8	8	8	8
F	F		F	F			F				F	F		F			F		

LRU = 9

So option (c) matching. No need of check for optimal algorithm same for (e) and (d) option.

T3 : Solution

(c)

$$500\ \mu\text{sec} = 150\text{ ns} \times 0.9 + 0.1 \times S$$
$$500\ \mu\text{sec} = 0.150\ \mu\text{s} \times 0.9 + 0.1 \times S$$
$$500 = 0.135 + 0.1\ S$$
$$499.865 = 0.1\ S$$
$$S = 4998.65\ \mu\text{sec} = 4.99865\text{ msec}$$

T4 : Solution

(b)

Virtual memory fetch strategies determine when a page or segment should be moved from secondary storage to main memory.

T5 : Solution

(a)

Reference string order: $\underbrace{r_1, r_2 \dots r_{n-k+1}, \dots r_k, \dots r_{n-1}, r_n,}_{\text{First } n \text{ frames page faults}}\ \underbrace{r_n, r_{n-1}, \dots r_k, \dots r_{n-k+1},}_{k\text{-frames no page fault}}\ \underbrace{r_{n-k}, \dots r_2, r_1}_{n-k \text{ frames page faults}}$

Total number of page faults = $n + n - k = 2n - k$

T6 : Solution

(16384)

$$\text{Page table size} = \text{Number of entries} \times \text{PTE size}$$
$$4 \times 2^x = \frac{2^{28}}{2^x} \times 2^2$$
$$2^{x+2} = 2^{30-x} \quad \because \ 4 \text{ page frame}$$
$$1 \text{ page frame} = 2^x$$

So, $4 \text{ page frame} = 4.2^x$

$$x + 2 = 30 - x$$
$$2x = 28$$
$$x = 14$$

So, $\text{Page size} = 2^x = 2^{14} = 16384$

T7 : Solution

(122)

Effective memory access time = 0.6 ∗ (10 + 80) + 0.4 ∗ (10 + 80 + 80) = 122

T8 : Solution

(6)

✓	✓	✓	✓	×	×	×	✓	✓	×
		6	6	6	6	6	6	7	7
	7	7	7	7	7	7	2	2	2
4	4	4	1	1	1	1	1	1	1
4	7	6	1	7	6	1	2	7	2

∴ 6 page faults will occur using LRU

8 Disk Scheduling

Detailed Explanation of Try Yourself Questions

T1 : Solution

(a)

Data cannot be written to secondary storage unless written within a file.

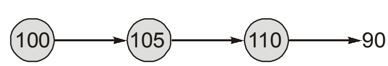
T2 : Solution

(d)

File attributes consist of name, type and identifier.

T3 : Solution

(3)



90 is serviced after servicing the 3 requests.

T4 : Solution

(99.60)

Overhead of each entry in FAT = 4 bytes

$$\text{Block size} = 10^3 \text{ bytes}$$

$$\text{Total size for each entry} = 1004 \text{ bytes}$$

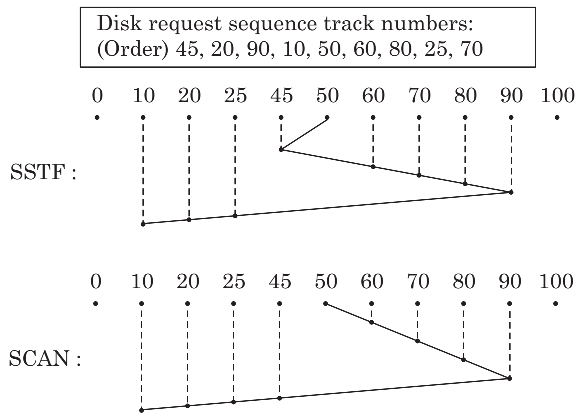
$$\text{Number of entries in FAT} = \frac{100 \times 10^6}{1004} = 0.099601$$

$$\text{Maximum size of a file} = 0.099601 \times 10^3 \text{ bytes}$$

$$= 99.601 \times 10^6 \text{ bytes}$$

T5 : Solution

(10)



$$\text{SSTF distance} = (50 - 45) + (90 - 45) + (90 - 10)$$
$$= 5 + 45 + 80 = 130$$
$$\text{SCAN distance} = (90 - 50) + (90 - 10)$$
$$= 40 + 80 = 120$$

$\therefore$ SSTF distance – SCAN distance = 130 – 120 = 10

■■■■

9 File System

Detailed Explanation of Try Yourself Questions

T1 : Solution

(b)

1.
$$8192 - 3209 = 4983$$
$$= \frac{4983}{8192} \times 100 = 60.82\,\%$$

2.
$$24{,}576 - (8192 \times 3) = 0\%$$

$$3.2{,}328{,}432{,}002 - (8192 \times 284232) = 3458$$
$$8192 - 3458 = 4734$$
$$\frac{4734}{8192} \times 100 = 57.7\%$$

$$4.2{,}328{,}927{,}678 - (8192 \times 284292) = 7614$$
$$8192 - 7614 = 578$$
$$\frac{578}{8192} \times 100 = 7.05\%$$

T2 : Solution

(b)

Maximum possible size = [(Address pointed by doubly indirect block)^2 + (Address pointed by single address) + address points by single direct address] × [block size]

$$= \left[\left(\frac{128}{8}\right)^2 + \left(\frac{128}{8}\right) + 8\right] \times 128\text{B}$$

$$= [2^8 + 2^4 + 2^3] \times 128 \text{ B}$$

$$= 32 \text{ KB} + 2 \text{ KB} + 1 \text{ KB} = 35 \text{ KB}$$

T3 : Solution

(c)

$$\frac{\text{Data Block Size}}{\text{DBA}} = \text{Number of DBA's possible in one disk block.}$$

■■■■