18CS10048 Sahil Jindal Date...../...../.....

Page.....

Ans1

(a) True. Since the time slice is greater than CPU burst of each process, we won't need to switch b/w different processes without completing ongoing process which is similar to traditional multiprogramming.

1.1

(b) True. Otherwise a user programs may hijack the CPU.

1.2

(c) False. In multithreading the data stack is shared automatically b/w various threads.

1.3

(d) False. Shortest job first theoretically gives shortest average waiting time and it is non pre-emptive CPU scheduling algo.

1.4

(e)

(f) True. Each process will have separate copies of the variable defined within a functions in case of parent-child process. However in multithreading the data stack is same for all threads.

1.6

(e) False. It is used by a user-level process to send signal to the kernel instead of different process.

1.5

(g) Assuming a is a local variable, it is false since different copies for the parent & child is made.

1.7

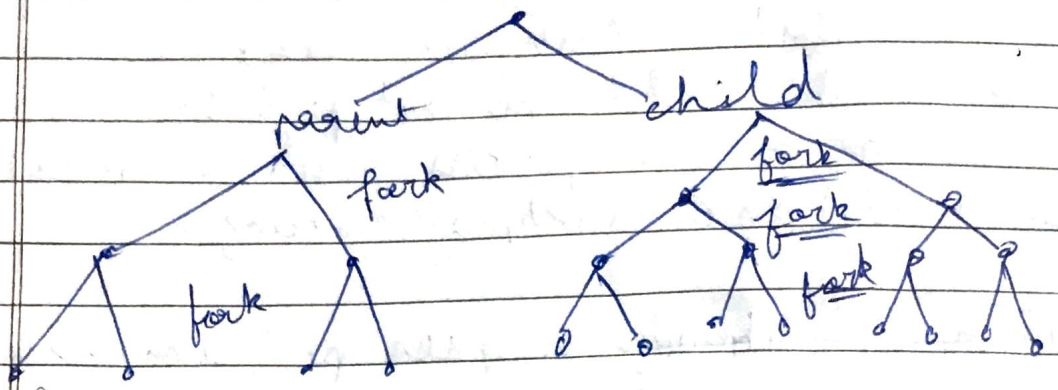
Archies

Date...../...../.....

Page.....

(h)

1.8 False. It is printed $2^2 + 2^3 = 12$ times



A2

2.1 ① We need to make sure that the ker... provides some basic functionalities acr... all the platforms. If we could treat it like a normal function call, it won't be ab...

A2 2.1 ① kernel provides various functionalities in privileged mode which we cannot give to a normal function.

② Kernel is interrupt driven which won't be supported by a normal function.

2.2 Yes using shared memory we can achieve this functionality.

2.3

Archies

Date...../...../.....

Page.....

2.4

(i) Running to Ready

In case of interrupt, the state is transitioned task to ready from running

(ii) Waiting to Ready

When a process is completed on an I/O task is completed, state transitions to ready from waiting.

A3 3.1

	P1	P2	P3
Arrival	0	0	0
Execution	10	20	30
first 20% I/O	2	4	6
CPU time-out	7	14	21
Output I/O out	1	2	3

. For first 2 sec, I/O is taking place for all the three process struck of I/O. From 2-9 unit P1 is processed & I/O is finished for P1, P2, P3.

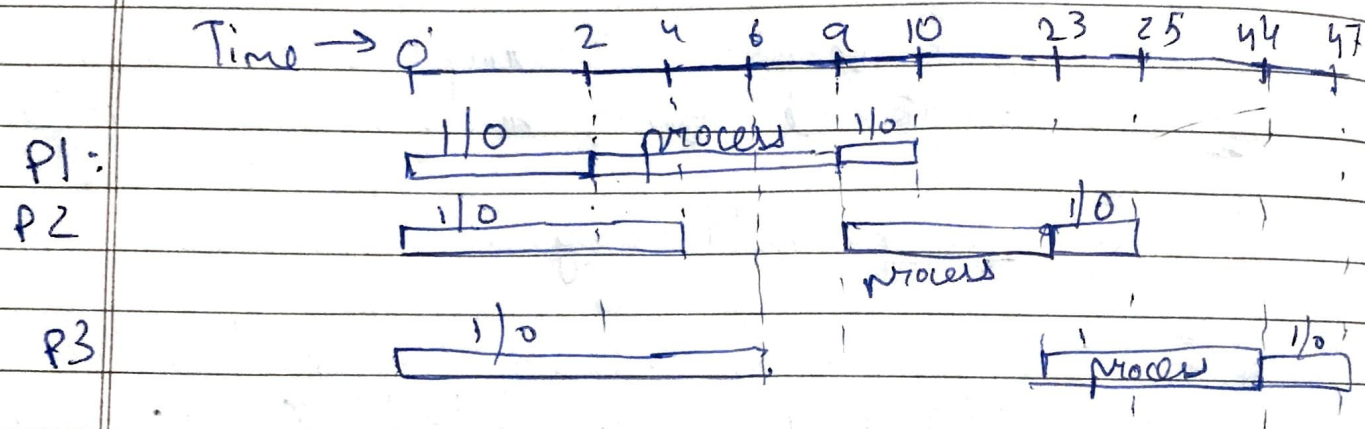
Archies

Date...../...../.....

Page.....

from 9-10 P1 completes its output & in parallel, 9-23 P2 is processed. 23-25 P2's output is completed & in parallel 23-44 P3 is processed. 44-47 P3's output is completed.

In a diagram:



Here CPU is idle in 0-2 & 44-47.
i.e. for 4 units.

3.2 Each process waits for

$(100-1) * (\text{time quantum} + \text{scheduling overhead})$

$\therefore 99 * (h + 5 \times 10^{-6}) = 1$

$h + 5 \times 10^{-6} = \frac{1}{99} = 0.01010101$

$h = 0.010101 - 0.000005$

$h = 0.010096$ seconds

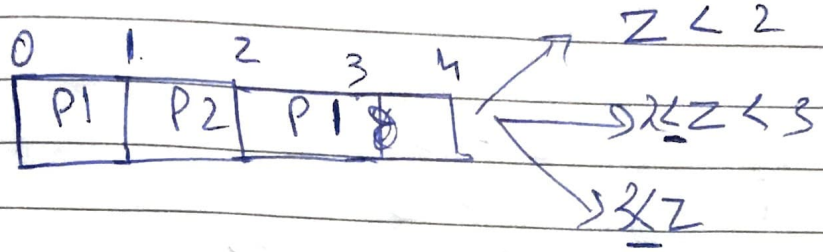
Archies

Date..../..../....

Page.....

Thus the time quantum must be ≥ 0.010096 seconds

3.3



for the first 4 seconds, P1 is in 0-1 , P2 in 1-2 & P1 in 2-4.
Now when P4 arrived, remaining time for P1 is 2, P3 is 3 & P4 is 2.
Three cases:

$z < 2$:

P4 in $4 - 4+z$
P1 in $4+z - 6+z$
P3 in $6+z - 9+z$

avg waiting time $= \frac{(1+z)+(0)+(3+z)+0}{4}$

$2 = \frac{4+2z}{4}$

$\Rightarrow z = 2$

Since z must be ≤ 2, considered

$2 < z < 3$. P1 in $2-5$
P4 in $5 - 5+2$
P3 in $5+2 - 8+z$

avg wait time $= \frac{(1)+(0)+(1)+(2+z)}{4}$

Archies

Date...../...../.....

Page.....

$$2 = \frac{4+z}{4}$$

$$\Rightarrow \quad z = 4$$

Since $2 \leq z < 3$, rejected.

$3 \leq z$

P1 in 2 - 5

P3 in 5 - 8

P4 in 8 - 8 + 2.

$$\text{Avg wait} = \frac{(1)+(0)+(2)+4}{4}$$

$\Rightarrow$ nosol.

~~Hence~~ 2.

Hence z must be 2 millisecond