Operating Systems, Tuitorial 1
CS30002

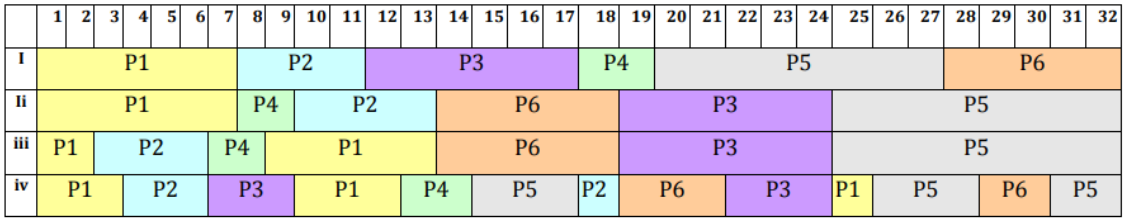
Topics - **Processes, Threads, System Calls, Inter Process Communication like Shared Memory, Pipes, and Scheduling**

1)Consider the following set of processes.Calculate the average waiting time and average turnaround time for the following scheduling algorithms:

(i) FCFS,
(ii) non-preemptive SJF,
(iii) pre-emptive SJF, and
(iv) round-robin with time quantum of 3 msec.

Process	P1	P2	P3	P4	P5	P6
Arrival Time	0	2	3	5	6	8
CPU Burst time	7	4	6	2	8	5

The Gantt chart for the four scheduling algorithms are shown below:



(i) FCFS
AWT = (0 + 5 + 8 + 12 + 13 + 19) / 6 = 9.5
ATT = (7 + 9 + 14 + 14 + 21 + 24) / 6 = 14.83

(ii) Non-preemptive SJF
AWT = (0 + 7 + 15 + 2 + 18 + 5) / 6 = 7.83
ATT = (7 + 11 + 21 + 4 + 26 + 10) / 6 = 13.17

(iii) Preemptive SJF
AWT = (6 + 0 + 15 + 1 + 18 + 5) / 6 = 7.5
ATT = (13 + 4 + 21 + 3 + 26 + 10) / 6 = 12.83

(iv) Round-robin
AWT = (18 + 12 + 15 + 7 + 18 + 17) / 6 = 14.5
ATT = (25 + 16 + 6 + 9 + 26 + 22) / 6 = 17.33

(iv)Note a change in turn around time for P3 is (24-3) = 21

2) The following program is executed in c .How many child processes are created ?

(a)

```
#include<stdio.h>
int main(){
    int i;
    for(int i=0;i<10;i++){
        If (i%2==0){
```

```
            fork();
        }
    return 0;
}
```

Fork( ) statement is executed when 'i' is even. Thus the above code is equivalent to

```
#include
int main( )
{
int i;
fork( );
fork( );
fork( );
fork( );
fork( );
}
```

n – fork statements will have 2^n-1 child.

Hence, n = 5 ⇒ We have 31 child processes.

(b)How many times will "Hello" be printed bythe following code segment?

```
if (fork() && (!fork()))
    if (fork() || fork())
        fork();
        printf ("\nHello");
```

Hello will be printed 7 times.In first statement second is not executed if first is false.In second statement second is executed only if first is false

3)Write a C/C++ code segment where a parent process creates a shared memory segment and populates it with some data.It then creates two child processes, which attaches the shared memory segment to their address spaces and carries out some computation.

4) A multiprogramming operating system uses a degree of multiprogramming 'm' which is large. It is proposed to double the throughput of the system by augmenting / replacement of its hardware components. Comment on the following four proposals in this context:

a) Replace the CPU by a CPU with twice its speed.
b) Expand the main memory to twice its present size.
c) Add new I/O devices capable of operation in DMA mode.
d) Replace the CPU by a CPU with twice the speed and expand the main memory to twice its present size.

Ans 4)

a) Here the degree of multiprogramming does not change as there may be substantial periods of time when the CPU has no work. Therefore, the throughput is limited by the availability of memory.

b) The throughput is limited by the speed of the CPU even though expanding the memory increase m. Hence, the throughput may not double.
c) The DMA implements the data transfer involved in the I/O operation without involving the CPU and raises an I/O interrupt when the data transfers completes. Thus, the DMA makes multiprogramming feasible by permitting concurrent operation of the CPU and I/O devices. So, adding I/O devices capable of operations in DMA mode make multiprogramming feasible.
d) This proposal does not have the drawbacks of only replacing CPU by the CPU with twice the speed and of only expanding the main memory to twice its present size which was discussed in parts (a) and (b). Thus, assuming a proper mix of programs, this proposal is the best.

5) Consider a parent process P that has forked a child process C in the program below.

```
int a = 5;
int fd = open(...); // opening a file
int ret = fork();
if(ret > 0) {
      close(fd);
      a=6;
      ...
}
else if(ret==0) {
      printf("a=%d\n", a);
      read(fd, something);
}
```

After forking, suppose the parent process is scheduled first before the child process. Once parent resumes after fork, it closes the file descriptor and changes the value of a as shown in the snippet. After these two changes, child process is scheduled for the first time.

a) What is the value of variable a printed in the child process? Why?
b) Will attempt to read from the file descriptor succeed in the child? Explain.

Ans 5)

a) 5. The value is only changed in the parent.
b) Yes, the file is closed in the parent.

5) The classical batch - processing system completely ignores the cost of increased waiting time for users. Consider a single batch characterized by the following parameters:

M	Average mounting time (i.e. tie to start a batch of jobs)
T	Average service time per job
N	Number of jobs
S	Unit price of service time
W	Unit price of waiting time per user

a. Find the optimal batch size that minimizes the cost of service time and waiting time per user (within a single batch)
b. In an installation in which M = 5 minutes, T = 1 minutes, and S = $200/hour. The operator chooses N = 50. Assuming that this is an optimal choice, find the unit cost of user waiting time W.

Ans 5)

a) Cost of service time = S / (total service time)
Total service time = (average service time per job) * (number of jobs) = TN
Cost of service time = S / (TN)
Cost of waiting time = ((unit price of waiting time per user) * (no. of users)) / (average mounting time) = (WN)/M
Thus, total cost = S / (TN) + (WN) / M
We need to choose a value of N which minimizes this cost.
We can use differentiation to find out the point of maxima.
d(total cost)/d(N) = -S/(TN^2) + W/M
Equating this to 0, we get: N = sqrt((MS)/(TW))
With this optimal value of N, the total cost = 2 * sqrt((SW)/(TM))

b) Using the above derived equation,
N = sqrt ((MS)/(TW))
putting the values M = 5 minutes, T = 1 minute, S = $200/hour = ($200/60) /minute and N = 50, we get
W = $0.4 per hour per user