1 Process Concepts & CPU Scheduling

For Micro Notes by the Student

1. Consider a System having 'n' CPUs ($n \geq 1$) and 'k' Processes ($k > n$). Calculate lower bound and upper bound of the number of Processes that can be in the Ready, Running and Block states.

2. Consider the following statements about process state transitions for a system using preemptive scheduling

   I. A running process can move to ready state

   II. A ready process can move to running state

   III. A blocked process can move to running state

   IV. A blocked process can move to ready state.

   Which of the above statements are TRUE?

   (a) I, II and IV only (b) I, II, III and IV

   (c) I, II and III only (d) II and III only

3. Which of the following standard C library functions will always invoke a system call when executed from a single -threaded process in a UNIX/Linux operating system?

   (a) sleep (b) malloc

   (c) strlen (d) exit

4. Which of the following statements (s) is/are correct in the context of CPU Scheduling?

   (a) The goal is to only maximize CPU utilization and minimize throughput

   (b) Turnaround time includes waiting time

   (c) Implementing preemptive scheduling needs hardware support

   (d) Round-robin policy can be used even when the CPU time required by each of the processes is not known Apriori.

For Micro Notes by the Student

5. Consider three Processes P_1, P_2, P_3 arriving in the Ready Queue at time 0 in the order P_1, P_2, P_3. Their service time requirements are 10, 20 & 30 units respectively. Each Process spends 20% of its Service time on I/O followed by 70% of its Service time on Computation at CPU and last 10% on I/O before completion.

   Assuming Concurrent I/O and negligible Scheduling Overhead. Calculate for FCFS Scheduling

   (i) Average TAT of Processes

   (ii) % CPU idleness

6. Consider the following processes, with the arrival time and the length of the CPU burst given in milliseconds. The scheduling algorithm used is preemptive Shortest Remaining-Time First (SRTF).

Process	Arrival Time	Burst Time
P1	0	10
P2	3	6
P3	7	1
P4	8	3

The average turnaround time of these processes is _____milliseconds.

For Micro Notes by the Student

7. Consider the following four processes with arrival times (in milliseconds) and their length of CPU bursts (in milliseconds) as shown below:

Process	P_1	P_2	P_3	P_4
Arrival time	0	1	3	4
CPU burst time	3	1	3	Z

These processes are run on a single processor using preemptive Shortest Remaining Time First (SRTF) Scheduling Algorithm. If the average waiting time of the processes is 1 millisecond, then the value of Z is _____.

8. Three Processes arrive at time zero with CPU bursts of 16, 20 and 10 milliseconds. If the scheduler has prior knowledge about the length of the CPU bursts, the minimum achievable average waiting time for these three processes in a Non-Preemptive Scheduler (rounded to nearest integer) is _____ milliseconds.

9. Consider a System with Preemptive Priority based Scheduling with 3 Processes P1, P2, P3 having infinite instances of them. The instances of these Processes arrive at regular intervals of 3, 7 & 20 ms respectively. The priority of the Process instances is the inverse of their periods. Each of the Process instance P1, P2, P3 consumes 1, 2 & 4 ms of CPU time respectively. The 1st instance of each Process is available at 1 ms. What is the Completion time of the 1st instance of Process P3?

For Micro Notes by the Student

10. Consider a set of 4 Processes A, B, C, D arriving in the order at time 0^+. Their Burst Time requirements are 4, 1, 8, 1 respectively using Round Robin scheduling with time quantum of 1 unit, The Completion time of Process A is _____.

11. Consider a System with 'n' Processes arriving at time 0^+ with substantially large Burst Times. The CPU scheduling overhead is 's' seconds, Time Quantum is 'q' seconds. Using Round Robin scheduling, what must be the value of Time Quantum 'q' such that each Process is guaranteed to get its turn at the CPU exactly after 't' seconds in its subsequent run-on CPU.

12. Consider the following set of Processes, assumed to have arrived at time 0. Consider the CPU scheduling algorithms Shortest Job First (SJF) and Round Robin (RR). For RR, assume that the processes are scheduled in the order P_1, P_2, P_3, P_4.

Processes	P_1	P_2	P_3	P_4
Burst time (in ms)	8	7	2	4

If the time quantum for RR is 4 ms, then the absolute value of the difference between the average turnaround times (in ms) of SJF and RR (round off to 2 decimal places) is _____.

For Micro Notes by the Student

13. Consider four Processes P, Q, R, and S scheduled on a CPU as per Round Robin Algorithm with a Time Quantum of 4 units. The Processes arrive in the order P, Q, R, S, all at time t = 0. There is exactly one context switch from S to Q, exactly one context switch from R to Q, and exactly two context switches from Q to R. There is no context switch from S to P. Switching to a ready process after the termination of another process is also considered a context switch. Which one of the following is NOT possible as CPU burst time (in time units) of these processes?

    (a) P = 4, Q = 10, R = 6, S = 2

    (b) P = 2, Q = 9, R = 5, S = 1

    (c) P = 4, Q = 12, R = 5, S = 4

    (d) P = 3, Q = 7, R = 7, S = 3

14. Consider a System using Round Robin Scheduling with 10 Processes all arriving at the time 0. Each Process is associated with 20 identical Request. Each Process request consumes 20 ms of CPU time after which it spends 10 ms of time on I/O, thereafter, initiates subsequent Request. Assuming Scheduling Overhead of 2 ms and Time Quantum of 20 ms, Calculate

    i. Response time of the 1st request of the 1st Process

    ii. Response time of the 1st request of the last Process

    iii. Response time of the subsequent request of any Process.

15. Consider a System using RR Scheduling with TQ of 'Q' seconds & CPU Scheduling overhead is 'S' seconds. Each Process on an average run for 'T' seconds before blocking on I/O. Give a formula for CPU efficiency for each of the following conditions.

    1. $Q = \infty$ 2. $Q > T$ 3. $S < Q < T$ 4. $Q = S$ 5. $Q \approx 0$

For Micro Notes by the Student

16. Consider a System using Preemptive Priority based scheduling with dynamically changing priorities. On its arrival a Process is assigned a priority of zero and Running Process Priority increases at the rate of 'β' and Priority of the Processes in the ready Q increases at the rate of 'α'. By dynamically changing the values of α and β one can achieve different Scheduling disciplines among the Processes. What discipline will be followed for the following conditions.

    1) $\beta > \alpha > 0$ 2) $\alpha < \beta < 0$

17. Consider Processes P_1 & P_2 arriving in the ready queue at time 0 with following properties.

    i. P1 needs a total of 12 units of CPU time and 20 units of I/O time. After every 3 units of CPU time P1 spends 5 units on I/O.

    ii. P2 needs a total of 15 units of CPU time and no I/O. P2 arrives just after P1.

    Compute the Completion times of P_1 & P_2 using the following scheduling techniques :

    1. SRTF
    2. Round Robin with Time Quanta = 4 units

For Micro Notes by the Student

18. Three processes A, B and C each execute a loop of 100 iterations. In each iteration of the loop, a process performs a single computation that requires t_c CPU milliseconds and then initiates a single I/O operation that lasts for t_{io} milliseconds. It is assumed that the computer where the processes execute has sufficient number of I/O devices and the OS of the computer assigns different I/O devices to each process. Also the scheduling overhead of the OS is negligible. The processes have the following characteristics:

Process Id	t_c	t_{io}
A	100ms	500ms
B	350ms	500ms
C	200ms	500ms

The processes A, B, and C are started at times 0,5 and 10 milliseconds respectively in a pure time-sharing system (round robin scheduling) that uses a time slice of 50 milliseconds. The time in milliseconds at which process C would **complete** its first I/O operation is _____.

19. Threads of a process share

(a) global variables but not heap (b) heap but not global variables

(c) neither global variables nor heap (d) both heap and global variables

20. Which of the following is/are shared by all the threads in a process?

I. Program counter II. Stack

III. Address space IV. Registers

(a) I and II only (b) III only

(c) IV only (d) III and IV only

For Micro Notes by the Student

2. Process Synchronization

01. Several concurrent processes are attempting to share an I / O device.

In an attempt to achieve Mutual Exclusion each process is given the following structure. (Busy is a shared Boolean Variable)

```
<code unrelated to device use>
  Repeat
   until busy = false;
    Busy = true;
<code to access shared >
    Busy = false;
<code unrelated to device use>
```

Which of the following is (are) true of this approach?

I. It provides a reasonable solution to the problem of guaranteeing mutual exclusion.

II. It may consume substantial CPU time accessing the Busy variable.

III. It will fail to guarantee mutual exclusion.

(a) I only (b) II only (c) III only (d) I & II (e) II & III

02. Consider the following two-process synchronization solution.

Process 0

Entry: loop while (turn == 1);

(critical section)

Exit: turn = 1;

Process 1

Entry: loop while (turn == 0);

(critical section)

Exit: turn = 0;

The shared variable turn is initialized to zero.

Which one of the following is **TRUE**?

(a) This is a correct two-process synchronization solution.

(b) This solution violates mutual exclusion requirement.

(c) This solution violates progress requirement.

(d) This solution violates bounded wait requirement.

For Micro Notes by the Student

For Micro Notes by the Student

03.
```
void Dekkers_Algorithm (int i)
{
    int j = ! (i);
    while (1)
    {
        (a) Non_CS( );
          (b) flag[i] = TRUE;
        (c) while (flag [j] = = TRUE)
            {
                if (turn = = j)
                {
                 flag [i] = FALSE;
                 while (turn = = j);
                 flag [i] = TRUE;
                 }
            }
        (d) <CS>
        (e) flag [i] = FALSE;
            turn = j;
      }
    }
```

04. Consider a non-negative counting semaphore S. The operation P(S) decrements S, and V(S) increments S. During an execution, 20 P(S) operations and 12 V(S) operations are issued in some order. The largest initial value of S for which at least one P(S) operation will remain blocked is_____.

For Micro Notes by the Student

05. Bsem S = 1 ,T= 1

Pr_i

```
{
while (1)
{
   P(S);
   P(T);
   <CS>
   V(T);
   V(S);
   }
}
```

Pr_j

```
{
    while (1)
    {
        P(T);
        P(S);
        <CS>
        V(S);
        V(T);
    }
}
```

Does it guarantee ME? How about Deadlock?

06. BSem S = 1, T = 0

Pr_i

```
{
   while (1)
   {
          P(T);
          Print ('1');
          Print ('1');
          V(S);
   }
}
{
```

Pr_j

```
{
     while (1)
     {
          P(S);
          Print ('0');
          Print ('0');
          V(T);
     }
}
```

What is the Regular Expression that gets generated?

For Micro Notes by the Student

07. Bsem m [0….4] = [1];

```
Pri i =0 , 4
{
   while (1)
   {
          P(m[i]);
          P(m[(i+1) %4]);
          <CS>
          V(m[i]);
          V(m[(i+1) %4]);
   }
}
```

What is the maximum no. of processes that can be in <CS>.

08. BSem S = 1, T = 0, Z = 0

```
Pri                  Prj                  Prk
{                    {                    {
  While (1)              P(T);                P(Z);
  {                      V(S);                V(S);
      P(S);          }                    }
      Print (*);
      V(T);
      V(Z);
  }
}
```

What is the minimum and maximum no. of "*" that get printed.

For Micro Notes by the Student

09. Consider the following threads, T_1, T_2 and T_3 executing on a single processor, synchronized using three binary semaphore variables, S_1, S_2, and S_3, operated upon using standard wait() and signal(). The threads can be context switched in any order and at any time.

T_1	T_2	T_3

while(true){
wait(S_3);
print("C");
signal(S_2); }
	while(true){
	wait(S_1);
	print("B");
	signal(S_3); }
		while(true){
		wait(S_2);
		print("A");
		signal(S_1); }

Which initialization of the semaphores would print the sequence BCABCABCA….?

(a) $S_1 = 1$; $S_2 = 1$; $S_3 = 1$

(b) $S_1 = 1$; $S_2 = 1$;$S_3 = 0$

(c) $S_1 = 1$; $S_2 = 0$; $S_3 = 0$

(d) $S_1 = 0$; $S_2 = 1$; $S_3 = 1$

10. Consider a counting semaphore initialized to 2, there are 4 concurrent Processes P_i, P_j, P_k & P_L. The Processes P_i & P_j desire to increment the current value of variable C by 1 whereas P_K & P_L desire to decrement the current value of C by 1. All Processes perform their update on C under semaphore control. What can be the minimum and maximum value of C after all Processes finish their update.

For Micro Notes by the Student

11. Consider Three Processes using four Binary Semaphores a, b, c, d in the order shown below.

Which Sequence is a Deadlock Free sequence?

(I) X: P(a); P(b); P(c);
Y: P(b); P(c); P(d);
Z: P(c); P(d); P(a);

(II) X: P(b); P(a); P(c);
Y: P(b); P(c); P(d);
Z: P(a); P(c); P(d);

(III) X: P(b); P(a); P(c);
Y: P(c); P(b); P(d);
Z: P(a); P(c); P(d);

(IV) X: P(a); P(b); P(c);
Y: P(c); P(b); P(d);
Z: P(c); P(d); P(a);

12. Each of a set of n processes executes the following code using two semaphores a and b initialized to 1 and 0, respectively. Assume that count is a shared variable initialized to 0 and not used in CODE SECTION P.

CODE SECTION P

wait(a); count=count+1;
if (count==n) signal (b);
signal (a): wait (b) ; signal (b);

CODE SECTION Q

What does the code achieve ?

(a) It ensures that no process executes CODE SECTION Q before every process has finished CODE SECTION P

(b) It ensures that two processes are in CODE SECTION Q at any time

(c) It ensures that all processes execute CODE SECTION P mutually exclusively

(d) It ensures that at most n−1 processes are in CODE SECTION P at any time

13. First Reader-Writer using Semaphore with Busy Waiting

```
int R = 0, W = 0;
Bsem mutex = 1;
Void Reader (Void)
{
    L1: P(mutex);
   if (W = = 1)
   {
    ________;
      goto L1;
   }
    else
    {
          R = R+1;
          ________;
     }
  <DB_READ>
  P(mutex);
       R = R –1;
  V(mutex);
}
```

```
Void Writer ( Void)
 {
     L2: P(mutex);
     if( _____________)
     {
          V(mutex);
          goto L2;
     }
     else
     {
          W=1;
          V(mutex);
     }
     <DB_WRITE>
     P(mutex);
          W=0;
     V(mutex);
}
```

For Micro Notes by the Student

14. void P(void)
{
A;
B;
C;
}
void Q(void)
{
D;
E;
}

main ()
{
Parbegin
P();
Q();
Parend
}

Indicate the Valid Output Sequences

1. A B C D E
2. D E A B C
3. A D B E C
4. A E B D C
5. D C E B A

For Micro Notes by the Student

15. int x = 0, y = 0;
Cobegin
begin
S_1: x = 1;
S_2: y = y+x;
end
begin
S_3: y = 2;
S_4: x = x+3;
end
Coend
Final values of x & y
I) x = 1; y = 2
II) x = 1; y = 3
III) x = 4; y = 6

16. int x = 0, y = 20;
Bsem mx = 1; my = 1;
Cobegin
begin
P(mx);
x = x+1;
V(mx);
end
begin
P(mx);
x= y+1;
V(mx);
end
Coend Final possible values of x _____

For Micro Notes by the Student

For Micro Notes by the Student

17. integer B = 2;

```
P1( )
{
   C = B –1;
   B = 2*C;
}

P2( )
{
   D = 2*B;
   B = D –1;
}
main ( )
{
   Parbegin
         P1()
         P2()
   Parend
}
```

The number of distinct values of B is/are _____

18.
```
int count = 0;
void test()
{
          int i, n = 5;
          for(i = 1; i<=n; ++i)
          count = count+1;
}
main ()
{       Parbegin
                  test();
                  test();
          Parend
}
```

What is the minimum and maximum value of count?

For Micro Notes by the Student

19. N = 2
M = 2
fork L_3;
fork L_4;
S_1;
L_1: Join N
S_3;
L_2: Join M
S_5: goto next;
L_3: S_2;
goto L_1;
L_4: S_4;
goto L_2;
next : S_6;

Draw the Precedence Graph

20. Consider the following pseudocode, where S is a semaphore initialized to 5 in line#2 and **counter** is a shared variable initialized to 0 in line#1. Assume that the increment operation in line #7 is not atomic

```
1. int counter = 0
2. Semaphore S = init(5);
3. void parop(void)
4. {
5.      wait (S);
6.      wait (S);
7.     counter++;
8.     signal (S);
9.     signal (S);
10. }
```

If five threads execute the function **parop** concurrently, which of the following program behavior(s) is/are possible?

(a) There is a deadlock involving all the threads

(b) The value of **counter** is 5 after all the threads successfully complete the execution of **parop**

(c) The value of **counter** is 1 after all the threads successfully complete the execution of **parop**

(d) The value of **counter** is 0 after all the threads successfully complete the execution of **parop**

For Micro Notes by the Student

21. Consider a computer system with multiple shared resource type, with one instance per resource type. Each instance and be owned by only one process at a time. Owning and freeing of resources are done by holding a global lock (L). The following scheme is used to own a resource instance:

```
function OWN RESOURCE(Resource R)
Acquire lock L // a global lock
if R is available then
   Acquire R
   Release lock L
else
  if R is owned by another process P then
    Terminate P, after releasing all resources owned by P.
     Acquire R
     Restart P
     Release lock L
    end if
   end if
end function
```

Which of the following choice(s) about the above scheme is/are correct?

(a) The scheme ensures that deadlocks will not occur

(b) The scheme violates the mutual exclusion property

(c) The scheme may lead to live-lock

(d) The scheme may lead to starvation

For Micro Notes by the Student

22. Consider the following multi-threaded code segment (in a mix of C and pseudo-code), invoked by two processes P1 and P2, and each of the processes spawns two threads T1 and T2:

int x = 0; // global Lock L1: // global main() {

create a thread to execute foo ( ); // Thread T1 create a thread to execute foo ( ); // Thread T2 wait for the two threads to finish execution; print (x);}

foo() {

int y = 0 ; Acquire L1; x = x + 1;

y = y +1; Release L1; print (y); }

Which of the following statement (s) is/are correct?

(a) Both T1 and T2, in both the processes, will print the value of y as 1.

(b) Both P1 and P2 will print the value of x as 2.

(c) At least one of the threads will print the value of y as 2

(d) At least one of P1 and P2 will print the value of x as 4.

For Micro Notes by the Student

3. Deadlocks

01. An operating system uses the *Banker's algorithm* for deadlock avoidance when managing the allocation of three resource types X, Y, and Z to three processes P0, P1, and P2. The table given below presents the current system state. Here, the *Allocation* matrix shows the current number of resources of each type allocated to each process and the *Max* matrix shows the maximum number of resources of each type required by each process during its execution.

	Allocation			**Max**		
	X	**Y**	**Z**	**X**	**Y**	**Z**
P0	0	0	1	8	4	3
P1	3	2	0	6	2	0
P2	2	1	1	3	3	3

There are 3 units of type X, 2 units of type Y and 2 units of type Z still available. The system is currently in a **safe** state. Consider the following independent requests for additional resources in the current state:

REQ1: P0 requests 0 units of X, 0 units of Y and 2 units of Z
REQ2: P1 requests 2 units of X, 0 units of Y and 0 units of Z

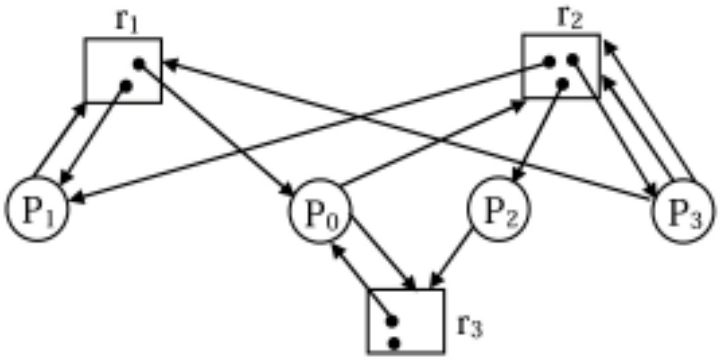
Which one of the following is **TRUE**?

(a) Only REQ1 can be permitted.
(b) Only REQ2 can be permitted.
(c) Both REQ1 and REQ2 can be permitted.
(d) Neither REQ1 nor REQ2 can be permitted.

For Micro Notes by the Student

02. Consider a System with n Processes <P_1……P_n>. Each Process is allocated x_i copies of R (resources) and makes a request for y_i copies of R. There are exactly 2 Processes A and B whose request is zero. Further there are 'k' instances of R free available. What is the condition for stating that System is not approaching deadlock (System is said to be not approaching deadlock if minimum request of Process is satisfiable? Also compute the total instances of R in System.

03. Consider the Following Resource Allocation Graph. Find if the System is in Deadlock State.



04. A system contains three programs, and each requires three tape units for its operation. The minimum number of tape units which the system must have such that deadlocks never arise is _____.

For Micro Notes by the Student

05. A system shares 9 tape drives. The current allocation and maximum requirement of tape drives for three processes are shown below:

Process	Current Allocation	Maximum Requirement
P1	3	7
P2	1	6
P3	3	5

Which of the following best describes current state of the system?

(a) Safe, Deadlocked (b) Safe, Not Deadlocked

(c) Not Safe, Deadlocked (d) Not Safe, Not Deadlocked

06. Which of the following statements is/are TRUE with respect to deadlocks?

(a) Circular wait is a necessary condition for the formation of deadlock.

(b) In a system where each resource has more than one instance, a cycle in its wait-for graph indicates the presence of a deadlock.

(c) If the current allocation of resources to processes leads the system to unsafe state, then deadlock will necessarily occur.

(d) In the resource-allocation graph of a system, if every edge is an assignment edge, then the system is not in deadlock state.

For Micro Notes by the Student

07. In a system, there are three types of resources: E, F and G. Four processes P_0, P_1, P_2 and P_3 execute concurrently. At the outset, the processes have declared their maximum resource requirements using a matrix named Max as given below. For example, Max[P_2, F] is the maximum number of instances of F that P_2 would require. The number of instances of the resources allocated to the various processes at any given state is given by a matrix named Allocation. Consider a state of the system with the Allocation matrix as shown below, and in which 3 instances of E and 3 instances of F are the only resources available.

Allocation					**Max**			
	E	**F**	**G**			**E**	**F**	**G**
P_0	1	0	1		P_0	4	3	1
P_1	1	1	2	-----	P_1	2	1	4
P_2	1	0	3		P_2	1	3	3
P_3	2	0	0		P_3	5	4	1

From the perspective of deadlock avoidance, which one of the following is true?

(a) The system is in *safe* state

(b) The system is not in *safe* state, but would be *safe* if one more instance of E were available

(c) The system is not in *safe* state, but would be *safe* if one more instance of F were available

(d) The system is not in *safe* state, but would be *safe* if one more instance of G were available

For Micro Notes by the Student

08. Consider a computer system with multiple shared resource type, with one instance per resource type.

Each instance and be owned by only one process at a time. Owning and freeing of resources are done by holding a global lock (L). The following scheme is used to own a resource instance:

```
function OWN RESOURCE(Resource R)
Acquire lock L // a global lock
if R is available then
        Acquire R
        Release lock L
else
   if R is owned by another process P then
        Terminate P, after releasing all resources  owned by P.
        Acquire R
        Restart P
        Release lock L
        end if
     end if
end function
```

Which of the following choice(s) about the above scheme is/are correct?

(a) The scheme ensures that deadlocks will not occur

(b) The scheme violates the mutual exclusion property

(c) The scheme may lead to live-lock

(d) The scheme may lead to starvation

09. A multithreaded program P executes with x number of threads and uses y number of locks for ensuring mutual exclusion while operating on shared memory locations. All locks in the program are non-reentrant, i.e., if a thread holds a lock *l*, then it cannot re-acquire lock *l* without releasing it. If a thread is unable to acquire a lock, it blocks until the lock becomes available. The minimum value of x and the minimum value of y together for which execution of P can result in a deadlock are:

(a) x = 1, y = 2 (b) x = 2, y = 1

(c) x = 2, y = 2 (d) x = 1, y = 1

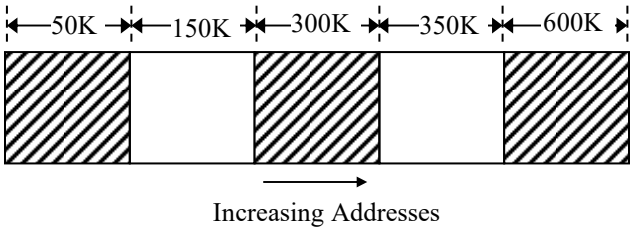
For Micro Notes by the Student

4. MEMORY MANAGEMENT

For Micro Notes by the Student

01. Consider a Memory System having 6 Partitions of sizes 200K; 400K; 600K; 500K; 300K; 250K. There are 4 Processes of sizes: 357K; 210K; 468K; 49K. Using **Best Fit Allocation Policy**, what Partitions are not allocated/ remains Unallocated?

02. Consider the following Memory Map in which blank regions are *not in use* and hatched regions are in use. Using Variable Partitions with no Compaction:



The sequence of requests for blocks of sizes 300K, 25K, 125K, 50K can be satisfied if we use:

(a) Either first fit or best fit policy (any one)

(b) First fit but not best fit policy

(c) Best fit but not first fit policy

(d) None of the above.

03. Consider a System with Memory of size **1000KBytes**.It uses Variable Partitions with no Compaction. Presently there are 2 partitions of sizes 200K & 260K respectively.

(i) What is the allocation request of the Process which would always be denied?

(a) 131K (b) 151K (c) 181K (d) 541K

(ii) The smallest Allocation Request which could be denied is:

(a) 131K (b) 151K (c) 181K (d) 541K

For Micro Notes by the Student

04. Consider a System having Memory of size 2^{46} Bytes, uses **Fixed Partitioning**. It is divided into fixed size Partitions each of size 2^{24} Bytes. The OS maintains a Process Table with one entry per Process. Each entry has, two fields: First, is a pointer pointing to Partition in which the Process is loaded and Second, Field is Process ID(PID). The Size of PID is 4Bytes. Calculate

(a) The **Size of Pointer** to the nearest Byte.

(b) Size of **Process Table** in Bytes if the System has 500 Processes.

05. Consider a System Using Variable Partition with no Compaction

Free holes	4K; 8K; 20K; 2K
Program size	2K; 14K; 3K; 6K; 10K; 20K; 2K
Time for Execution (B.T)	4; 10; 2; 1; 4; 1; 8

Using **Best Fit Allocation Policy** and **FCFS CPU Scheduling Technique**, Find the Time of Loading & Time of Completion of each program. The Burst Times are in Seconds.

06. Consider allocation of memory to a new process. Assume that none of the existing holes in the memory will exactly fit the process's memory requirement. Hence, a new hole of smaller size will be created if allocation is made in any of the existing holes. Which one of the following statements is TRUE?

(a) The hole created by next fit is never larger than the hole created by best fit

(b) The hole created by worst fit is always larger than the hole created by first fit

(c) The hole created by first fit is always larger than the hole created by next fit

(d) The hole created by best fit is never larger than the hole created by first fit

For Micro Notes by the Student

07. A Computer System using **Paging Technique** implements an 8KB Page with a **Page Table of Size** 24MB. The Page Table Entry is 24 bits. What is the **length of Virtual Address** in this System?

08. A Computer System using Paging technique has a Virtual Address of length 'l'. The Number of Pages in the Address Space are 'Z'. There are 'H' Frames in PAS. Calculate the number of bits in Page Offset and the size of PAS.

09. Consider a System using **Simple Paging Technique** with Logical Address (LA) of 32 bits. Page Table Entry (PTE) of 32 bits. What must be the Page Size in bytes, such that the Page Table of the Process Exactly fits in one Frame of Memory (PAS)?

10. Consider a System using **Paging** with **TLB**. What Hit Ratio is required to reduce the EMAT from 'D' ns to 'Z' ns using **TLB**. Assume that **TLB** access time is 'K' ns.

11. A Machine has a 32-bit Address Space and an 8KB Page. The Page Table is entirely in hardware, with one 32-bit word per entry. When a Process starts, the Page Table is copied to the hardware from memory, at one word every 100 nsec. If each Process runs for 100 msec (including the time to load the page table), what fraction of the CPU time is devoted to loading the Page Tables?

For Micro Notes by the Student

12. Consider a System using Paging technique with an Address Space of 65,536 Bytes. The Page Size in this System is 4096 Bytes. The Program Consists of Text, Data and Stack Sections as per the specifications given below:

    Text: 32,768 Bytes

    Data: 16,386 Bytes

    Stack: 15,870 Bytes

    A Page of the program contains portion of only one section i.e either Text or Data or Stack.

    a) Does the Program Fit in the given Address Space?

    b) What is the Maximum Page Size in Bytes such that the Program Fits in the Given Address Space?

13. Consider a System using 2 Level Paging Architecture. The Top level 9 bits of the Virtual Address are used to index into the outer Page Table. The next 7 bits of the Address are used to index into next level Page Table. If the size of Virtual Address is 28 bits. Then

    (i) How large are the Pages and How many are there in Virtual Address Space?

    (ii) If P.T.E at both levels is 32 bits in size then what is the Space Overhead needed to translate Virtual Address to Physical Address of an Instruction or Data Unit?

For Micro Notes by the Student

14. Consider a Computer System using 3 Level Paging Architecture with a uniform Page Size at all levels of Paging. The size of Virtual Address is 46 bits. Page Table Entries at all levels of Paging is 32 bits. What must be the Page Size in Bytes such that the Outer Page Table exactly fits in one frame of Memory. Assume Page Size is power of 2 in Bytes. Show the Virtual Address format indicating the number of bits required to access all the three levels of Page Tables and the Page offset of Virtual Address Space.

15. Consider a System using Segmented-Paging Architecture with V.A.S = P.A.S = 2^{16}Bytes. The VAS is divided into 8 equal sized non-overlapping Segments. Paging is applied on Segment, with a Page size being a power of 2 in Bytes. The Page Table Entry size of the Page Tables of the Segment is 16 bits. What must be the Page Size of the Segment such that the Page Table of the Segment exactly fits in one frame of Memory.

16. Consider a System using Segmented-Paging Architecture. Paging is applied on Segment. The System maintains a 256 entry Page-Table per Segment. The Page Size of Segment is 8Kbytes. The Virtual Address Space supports 2K Segments. Page Table Entry Size is 16 bits while the Segment Table entry is 32 bits in size.

    Calculate

    a) Size of Virtual Address Space

    b) Address Translation Space Overhead in Bytes.

    c) The number of levels of Memory accesses required for Address Translation.

17. Suppose the time to service a Page fault is on the average 10 milliseconds, while a Memory Access takes 1 microsecond. Then a 99.99% Hit ratio results in Average Memory Access Time of _____.

For Micro Notes by the Student

18. If an Instruction takes 'i' microseconds and a Page Fault takes an additional 'j' microsecond, the Effective Instruction Time if on the average a page fault occurs every 'k' instruction is _____.

19. Assume that we have a Demand-Paged memory. It takes 8 milliseconds to service a page fault if an empty frame is available or if the replaced page is not modified, and 20 milliseconds if the replaced page is modified. Memory-access time is 100 nanoseconds. Assume that the page to be replaced is modified 70 percent of the time. What is the acceptable page-fault rate for an effective access time of no more than 200 nanoseconds?

20. Consider a process executing on an operating system that uses demand paging. The average time for a memory access in the system is M units if the Corresponding memory page is available in memory, and D units if the memory Access causes a page fault. It has been experimentally measured that the average Time taken for a memory access in the process is X units. Which one of the Following is the correct expression for the page fault rate experienced by the Process?

    (a) (D – M)/(X – M) (b) (X – M)/(D – M)

    (c) (D – X)/(D – M) (d) (X – M)/(D – X)

21. Consider a Process in Demand Page environment having a reference string of length 'L' in which 'K' unique Pages occur. Calculate the lower bound and upper bound of the number of Page Fault for this Process. Assume Process is allocated 'Z' frames.

22. A Process refers 'n' unique Pages numbered 1 to n is the order and then it refers them back in the reverse order. Process is allocated 'k' Frames. Calculate the number of Page Faults using FIFO replacement with Pure Demand Paging.

For Micro Notes by the Student

23. Consider a System using Demand Paging with Page size of 100 records. Process is allocated one frame and uses pure Demand Paging. The Address sequence generated by the Process is:
0100, 0200, 0430, 0499, 0510, 0530, 0560, 0120, 0220, 0240, 0260, 0320, 0370

    (1) What is the Length of Reference String?

    (2) Calculate the number of Page Faults?

24. Consider a demand paging system with four page frames (initially empty) and LRU page replacement policy. For the following page reference string
7, 2,7,3, 2,5,3, 4,6,7,7,1,5,6,1
the page fault rate, defined as the ratio of number of page faults to the number of memory accesses (rounded off to one decimal place) is_____.

25. Consider a System with V.A.S = P.A.S = 2^{16} Bytes. Page Size is 512 Bytes. The size of Page Table entry is 32 bits. If the Page Table Entry contains besides other information 1 V/I bit, 1 Reference, 1 Modified bit, 3 bits for Page Protection. How many bits can be assigned for storing other attributes of the Page. Also compute Page Table Size in Bytes?

26. Let the Page Reference and the Delta (▲) be "c c d b c e c e a d" and 4, respectively. The initial Working Set a time t = 0 contains the pages {a, d, e , where 'a' was referenced at time t = 0, 'd' was referenced at time t = -1, and 'e' was referend at time t = -2. Determine the total number of page faults and the average number of page frames used by computing the working set at each reference.

For Micro Notes by the Student

27. Consider a computer system with 40-bit virtual addressing and page size of sixteen kilobytes. If the computer system has a one-level page table per process and each page table entry requires 48 bits, then the size of the per-process page table is megabytes.

28. Recall that Belady's anomaly is that the page-fault rate may increase as the number of allocated frames increases. Now, consider the following statements:

S1: Random page replacement algorithm (where a page chosen at random is replaced) suffers from Belady's anomaly

S2: LRU page replacement algorithm suffers from Belady's anomaly

Which of the following is CORRECT?

(a) S1 is true, S2 is true (b) S1 is true, S2 is false

(c) S1 is false, S2 is true (d) S1 is false, S2 is false

29. Assume that in a certain computer, the virtual addresses are 64 bits long and the physical addresses are 48 bits long. The memory is word addressable. The page size is 8 kB and the word size is 4 bytes. The Translation Look-aside Buffer (TLB) in the address translation path has 128 valid entries. At most how many distinct virtual addresses can be translated without any TLB miss?

(a) 4×2^{20} (b) 16×2^{10}

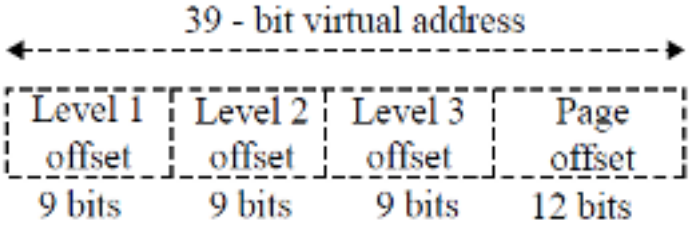
(c) 256×2^{10} (d) 8×2^{20}

For Micro Notes by the Student

30. Consider a paging system that uses I-level page table residing in main memory and a TLB for address translation. Each main memory access takes 100 ns and TLB lookup takes 20ns. Each page transfer to/ from the disk takes 5000 ns. Assume that the TLB hit ratio is 95%, page fault rate is 10%. Assume that for 20% of the total page faults, a dirty page has to be written back to disk before the required page is read in from disk. TLB update time is negligible. The average memory access time in ns (round off to 1 decimal places) is _____

31. In the context of operating systems, which of the following statements is/are correct with respect to paging?
(a) Page size has no impact on internal fragmentation.
(b) Paging helps solve the issue of external fragmentation.
(c) Paging incurs memory overheads
(d) Multi-level paging is necessary to support pages of different sizes

32. Consider a three-level page table to translate a 39-bit virtual address to a physical address as shown below.



The page size is 4KB (1KB = 2^{10} bytes) and page table entry size at every level is 8 bytes. A process P is currently using 2GB (1GB = 2^{30} bytes) virtual memory which is mapped to 2 GB of physical memory, The minimum amount of memory required for the page table of P across all levels is_____ KB.

For Micro Notes by the Student

33. Which one of the following statements is FALSE?

(a) The TLB performs an associative search in parallel on all its valid entries using page number of incoming virtual address.

(b) If the virtual address of a word given by CPU has a TLB hit, but the subsequent search for the word results in a cache miss, then the word will always be present in the main memory.

(c) The memory access time using a given inverted page table is always same for all incoming virtual addresses.

(d) In a system that uses hashed page tables, if two distinct virtual addresses V1 and V2 map to the same value while hashing, then the memory access time of these addresses will not be the same.

For Micro Notes by the Student

5. FILE SYSTEM & DEVICE MANAGEMENT

01. Consider the following Disk Specifications:

Number of Platters = 16

Number of Tracks/Surface = 512

Number of Sectors/Track = 2048

Sector offset = 12 bits

Average Seek Time = 30 ms

Disk RPM = 3600

Calculate the Following:

(a) Unformatted Capacity of Disk.

(b) IO-Time/Sector

(c) Data Transfer Rate

(d) Sector Address

02. Consider a Disk with the following Specifications:

Number of surfaces = 64

Outer diameter = 16 cm

Inner diameter = 4 cm

Inter Track space = 0.1 mm

Max Density = 8000 bits/cm

Calculate the Unformatted Capacity of Disk.

03. How long does it take to load a 64 Kbytes Program from a disk whose Average Seek time is 30 ms Rotation time is 20 ms, Track Size is 32 Kbytes, Page Size is 4 Kbytes. Assume that Pages of the Program are distributed randomly around the disk. What will be the % saving in time if 50% of the Pages of program are Contiguous?

For Micro Notes by the Student

04. An Application requires 100 libraries at startup. Each library requires 1 disk access. Seek Time is 10 ms, Disk RPM is 6000. All 100 libraries are at random locations. 50% of Libraries requires transfer time of ½ Rotation, while for the remaining 50% it is negligible. How long does it take to load all 100 libraries?

05. Consider a linear list based directory implementation in a file system. Each directory is a list of nodes, where each node contains the file name along with the file metadata, such as the list of pointers to the data blocks. Consider a given directory foo.

Which of the following operations will necessarily require a full scan of foo for successful completion?

(a) Opening of an existing file in foo

(b) Creation of a new file in foo

(c) Renaming of an existing file in foo

(d) Deletion of an existing file from foo

06. In a file allocation system, which of the following allocation scheme(s) can be used if no external fragmentation is allowed?

I. Contiguous

II. Linked

III. Indexed

(a) I and III only (b) II only

(c) III only (d) II and III only

For Micro Notes by the Student

07. Consider a Unix I-node structure that has 8 direct Disk Block Addresses and 3 Indirect Disk Block Addresses, namely Single, Double & Triple. Disk Block Size is 1Kbytes & each Block can hold 128 Disk Block Addresses.

Calculate (i) Maximum File Size with this I-Node Structure?
(ii) Size of Disk Block Address?
(iii) Is this File Size possible over the given Disk?

08. Consider a File System that stores 128 Disk Block Addresses in the index table of the Directory. Disk Block Size is 4 Kbytes. If the file size is less than 128 Blocks, then these addresses act as direct Data Block addresses. However, if the File Size is more than 128 Blocks, then these 128 addresses in the Index table point to next level Index Blocks, each of which contain 256 Data block addresses. What is the Max File Size in this File System?

09. The index node (inode) of a Unix-like file system has 12 direct, one single-indirect and one double-indirect pointers. The disk block size is 4 kB, and the disk block address is 32- bits long. The maximum possible file size is ____ GB (rounded off to 1 decimal place)

10. A File System with 300 G Byte Disk uses a File descriptor with 8 Direct Block Addresses, 1 Indirect Block Address and 1 Doubly Indirect Block Address. The size of each Disk Block is 128 Bytes and the size of each Disk Block Address is 8 Bytes. The maximum possible File Size in this file System is

(a) 3 K Bytes
(b) 35 K Bytes
(c) 280 K Bytes
(d) Dependent on the size of the disk

For Micro Notes by the Student

11. The Data Blocks of a very large file in the Unix File System are allocated using
    (a) Contiguous allocation (b) Linked allocation
    (c) Indexed allocation (d) An extension of indexed allocation.

12. Using a Larger Block size in a Fixed Block Size File System leads to
    (a) Better Disk Throughput but Poorer Disk Space Utilization.
    (b) Better Disk Throughput and Better Disk Space Utilization
    (c) Poorer Disk Throughput but Better Disk Space Utilization
    (d) Poorer Disk Throughput and Poorer Disk Space Utilization

13. A FAT (File allocation table) based file System is being used and the total overhead of each entry in the FAT is 4 bytes in size. Given a 100×10^6 bytes' disk on which the file System is stored and data block size is 10^3 bytes, the maximum size of a file that can be stored on this disk in units of 10^6 bytes is _____.

14. A File System with a One-level Directory structure is implemented on a disk with Disk Block Size of 4 Kbytes. The disk is used as follows:
    Disk Block 0 : Boot Control Block
    Disk Block 1 : File Allocation Table, consisting of one 10-bit entry per Data Block, representing the Data Block Address of the next Data Block in the files.
    Disk Block 2, 3 : Directory with 32-bit entry per File.
    Disk block 4 : Data block 1.
    Disk Block 5 : Data Block 2,3 etc;
    a) What is the Maximum possible number of Files?
    b) What is the Maximum Possible File size in Bytes?

For Micro Notes by the Student

15. Consider a Disk with 'B' Blocks, 'F' of which are free. Disk Block Address is 'D' bits, Disk Block Size is 'X' Bytes.

    (A) Calculate

    (i) Given Disk Size.

    (ii) Maximum possible Disk Size.

    (iii) Relation between 'B' & 'D' .

    (B) What is the condition in which Free List uses less space than Bit Map?

16. Consider two files systems A and B , that use contiguous allocation and linked allocation, respectively. A file of size 100 blocks is already stored in A and also in B. Now, consider inserting a new block in the middle of the file (between 50^{th} and 51^{st} block), whose data is already available in the memory. Assume that there are enough free blocks at the end of the file and that the file control blocks are already
    in memory. Let the number of disk accesses required to insert a block in the middle
    of the file in A and B are n_A and n_B respectively, then the value of $n_A + n_B$ is _____.

17. The beginning of a free space Bit-Map looks like this after the Disk Partition is first formatted: 1000 0000 0000 0000 (the first block is used by the Root Directory). The System always searches for free blocks starting at the lowest numbered block, so after writing file A, which uses 6 blocks, the bitmap looks like this: 1111 1110 0000 0000. Show the Bit-Map after each of the following additional actions as HEX Code:

    (a) File B is written, using 5 blocks

    (b) File deleted

    (c) File C is written, using 8 blocks

    (d) File B is deleted.

For Micro Notes by the Student

18. Consider a disk queue with requests for I/O to blocks on cylinders 47, 38, 121, 191, 87, 11, 92, 10. The C-LOOK scheduling algorithm is used. The head is initially at cylinder number 63, moving towards larger cylinder numbers on its servicing pass. The cylinders are numbered from 0 to 199. The total head movement (in number of cylinders) incurred while servicing these requests is _____.

19. Consider a Disk with the Following pertinent details:
    Track-Track Time = 1 ms
    Current Head Position = 65
    Direction: moving towards higher tracks with highest Track being 199.
    Current Clock time = 160 ms
    Consider the Following Data Set:

Serial No.	Track No	Time of Arrival
1	**12**	**65 ms**
2	**85**	**80 ms**
3	**40**	**110 ms**
4	**100**	**120 ms**
5	**75**	**175 ms**

Calculate the *Time of Decision, Pending Requests, Head Position, Selected Request, Seek Time* using FCFS, SSTF, SCAN, LOOK, C-SCAN, C-LOOK Algorithms.

For Micro Notes by the Student

20. Disk requests come to disk driver for cylinders 10,22,20,2,40,06 and 38, in that order at a time when the disk drive is reading from cylinder 20. The seek time is 6 msec per cylinder.

Compute the total seek time if the disk arm scheduling algorithm is:

(a) First come first served

(b) Closest cylinder next.

21. Consider a storage disk with 4 platters (numbered as 0,1,2 and 3), 200 cylinders (numbered as 0, 1, ., 199), and 256 sectors per track (numbered as 0, 1, , 255). The following 6 disk requests of the form [sector number, cylinder number, platter number] are received by the disk controller at the same time:

[120, 72, 2], [180, 134, 1], [60, 20, 0] [212, 86, 3], [56, 116, 2], [118, 16, 1]

Currently the head is positioned at sector number 100 of cylinder 80, and is moving towards higher cylinder numbers. The average power dissipation in moving the head over 100 cylinders is 20 milliwatts and for reversing the direction of the head movement once is 15 milliwatts.

Power dissipation associated with rotational latency and switching of head between different platters is negligible. The total power consumption in milliwatts to satisfy all of the above disk requests using the Shortest Seek Time First disk scheduling algorithms is_____

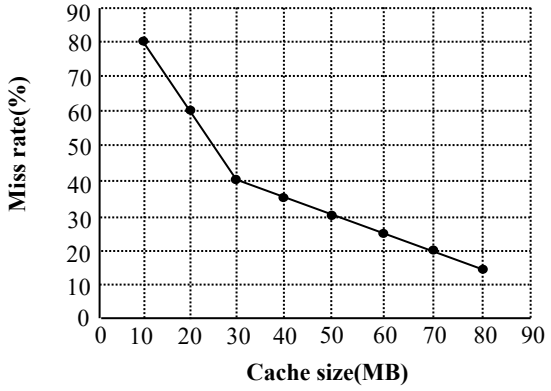
22. Suppose a disk has 201 cylinders, numbered from 0 to 200. At some time the disk arm is at cylinder 100, and there is a queue of disk access requests for cylinders 30, 85, 90, 100, 105, 110, 135 and 145. If Shortest-Seek Time First (SSTF) is being used for scheduling the disk access, the request for cylinder 90 is serviced after servicing _____ number of requests.

For Micro Notes by the Student

23. Suppose the following disk request sequence (track numbers) for a disk with 100 tracks is given: 45, 20, 90, 10, 50, 60, 80, 25, 70. Assume that the initial position of the R/W head is on track 50. The additional distance that will be traversed by the R/W head when the Shortest Seek Time First (SSTF) algorithm is used compared to the SCAN (Elevator) algorithm (assuming that SCAN algorithm moves towards 100 when it starts execution) is _____ tracks.

24. Consider an operating System capable of loading and executing a single sequential user Process at a time. The disk head scheduling algorithm used is First Come First Served (FCFS). If FCFS is replaced by Shortest Seek Time First (SSTF), claimed by the vendor to give 50 % better benchmark results, what is the expected improvement in the I/O performance of user programs?

25. A File System uses an in-memory cache to cache disk blocks. The miss rate of the cache is shown in the figure. The latency to read a block from the cache is 1 ms and to read a block from the disk is 10 ms. Assume that the cost of checking whether a block exists in the cache is negligible. Available cache sizes are in multiples of 10 MB.



The smallest cache size required to ensure an average read latency of less than 6 ms is _____MB.

26. The amount of Disk Space that must be available for Page storage is related to Maximum number of Processes 'N' , the number of Bytes in Virtual Address Space 'B' and the number of Bytes in RAM 'R'. Give an expression for the worst case Disk Space required.

27. Consider the following five disk access requests of the form(request id, cylinder number) that are present in the disk scheduler queue at a given time.

(P, 155), (Q, 85), (R, 110), (S,30), (T,115)

Assume the head is positioned at cylinder 100. The scheduler follows Shortest Seek Time First Scheduling to service the requests. Which one of the following statements is FALSE?

(a) R is serviced before P.

(b) T is serviced before P.

(c) Q is serviced after S, but before T

(d) The head reverses its direction of movement between servicing of Q and P.

For Micro Notes by the Student

6. Fork System Call

For Micro Notes by the Student

01.
```
main ()
{
    int i, n;
    for (i = 1; i < = n; + + i)
    if (fork () = = 0)
        print (“*”);
}
```

02.
```
main ()
{
    int i, n;
    for (i = 1; i< = n; ++i)
    {
     fork ();
     print (“*”);
    }
}
```

03.
```
main ()
{
    int i, n;
    for (i = 1; i < = n; ++ i)
    {
        print (“*”);
        fork ();
    }
}
```

For Micro Notes by the Student

04.
```
main ()
{
    int a = 1, b = 2, c = 3;
    a + = b+ = + + c;
    print (a, b, c);
    if (fork () = = 0)
    {
    int d;
    + + a; + + b; – – c;
    print (a, b, c);
    if (fork () = = 0)
    {
            d = a + b + c;
            print (a, b, c, d);
    }
    else
    {
             – – a; – – b;
            c = a + b; d = a + b + c;
            print (a, b, c, d);
    }
    }
    else
    {
    c + = b + = + + a;
    print (a, b, c) ;
    }
    print (d);
}
```

05. The following C program is executed on a Unix/Linux system:

```
#include <unistd. h>
int main( )
{       int i;
        for (i = 0; i < 10; i + +)
            if (i % 2 = = 0) fork( );
}
```

The total number of child processes created is _____

For Micro Notes by the Student