# Yeshwantrao Chavan College of Engineering, Nagpur

**Examination:** ESE, Exam EVEN 2021-22

**Programme:** B.E. II Year , Sem-4

**Course Code:** **IT-2255** **Course Name:** **Operating System**

| Q.No | | Step wise Marking |
|---|---|---|
| 1 (a) | **What are system calls? How are they different from API? List various system calls for process control and explain any two of them.** | 4 M |
| Ans: | **ANS:**<br>A system call is a method of interacting with the operating system via programs.<br>A system call is a request from computer software to an operating system's kernel.<br>The Application Program Interface (API) connects the operating system's functions to user programs.<br>Process control<br>• create process, terminate process<br>• end, abort<br>• load, execute<br>• get process attributes, set process attributes<br>• wait for time<br>• wait event, signal event<br>• allocate and free memory<br>• Dump memory if error<br>• **Debugger** for determining **bugs, single step** execution<br>• **Locks** for managing access to shared data between processes<br>**wait()**<br>In some systems, a process needs to wait for another process to complete its execution. This type of situation occurs when a parent process creates a child process, and the execution of the parent process remains suspended until its child process executes.<br><br>The suspension of the parent process automatically occurs with a wait() system call. When the child process ends execution, the control moves back to the parent process.<br><br>**exec()**<br>This system call runs when an executable file in the context of an already running process that replaces the older executable file. However, the original process identifier remains as a new process is not built, but stack, data, head, data, etc. are replaced by the new process. | |
| Q.No | **(B) Describe the differences between symmetric and asymmetric multiprocessing. What are three advantages and one disadvantage of multiprocessor systems?** | Step wise Marking |
| 1(B) | | |
| Ans: | **ANS:**<br>Symmetric multiprocessing treats all processors as equals, and I/O can be processed on any CPU.<br>Asymmetric multiprocessing has one master CPU and the remainder CPUs are slaves. The master distributes tasks among the slaves, and I/O is usually done by the master only.<br>**Advantage:**<br>1.Multiprocessors can save money, by not duplicating power supplies, housings, and peripherals.<br>2.They can execute programs more quickly,<br>3.Can have increased reliability.<br>**Disadvantage:**<br>They are also more complex in both hardware and software than uniprocessor systems.<br>cost is more | 3 m |

**Date: 30/06/2022** **Name: Prof. K.R.Gavhale**

**Page No. :** **Sign Every Page:**

# Yeshwantrao Chavan College of Engineering, Nagpur

**Examination:** ESE, Exam EVEN 2021-22

**Programme:** B.E. II Year , Sem-4

**Course Code:** IT-2255 **Course Name:** Operating System

| Q.No | | Step wise Marking |
|---|---|---|
| 2(A) | (A) For the processes listed in the following table, which of the following scheduling schemes will give the lowest average turnaround time? Justify your answer. | 4M |

| Process | Arrival Time | Processing Time |
|---|---|---|
| A | 0 | 3 |
| B | 1 | 6 |
| C | 4 | 4 |
| D | 6 | 2 |

A. SJF (2m)

B. RR with slice=2(2m)

**Scheduling Algorithm**:

Non pre-emptive shortest job first

**Gantt chart:**

| A | B | D | C |
|---|---|---|---|

0 3 9 11 15

**Process Table:**

| Process | Arrival Time | Processing Time | Completion time | Turnaround time |
|---|---|---|---|---|
| A | 0 | 3 | 3 | 3 |
| B | 1 | 6 | 9 | 8 |
| C | 4 | 4 | 15 | 11 |
| D | 6 | 2 | 11 | 5 |

Average turnaround time = 27/4 = 6. 75

**Scheduling Algorithm**:

Round Robin with Quantum value two

**Gantt chart:**

| A | B | A | C | B | D | C | B |
|---|---|---|---|---|---|---|---|

0 2 4 5 7 9 11 13 15

| Process | Arrival Time | Processing Time | Completion time | Turnaround time |
|---|---|---|---|---|
| A | 0 | 3 | 5 | 5 |
| B | 1 | 6 | 15 | 14 |
| C | 4 | 4 | 13 | 9 |
| D | 6 | 2 | 11 | 5 |

Average turnaround time = 33/4 = 8.25

SJF will give the lowest average turnaround time

| Q.No | | Step wise Marking |
|---|---|---|
| 2(B) | (B) Describe the differences among short-term, medium-term, and long-term scheduling. | 3 M |

- **Short-term** (CPU scheduler) - selects from jobs in memory, those jobs which are ready to execute, and allocates the CPU to them. (1m)

- **Medium-term** - used especially with time-sharing systems as an intermediate scheduling level. A swapping scheme is implemented to remove partially run programs from memory and reinstate them later to continue where they left off. (1m)

- **Long-term** (job scheduler) - determines which jobs are brought into memory for processing. The primary difference is in the frequency of their execution. The short-term must select a new process quite often. Long-term is used much less often since it handles placing jobs in the system, and may wait a while for a job to finish before it admits another one. (1m)

**Date:** 30/06/2022 **Name:** Prof. K.R.Gavhale

**Page No. :** **Sign Every Page:**

| Q.No | | Step wise Marking |
|---|---|---|
| 3(A) | (A) Define the following with suitable example<br>i) race condition ii) critical section iii) semaphore . | 6 marks |

**ANS:**

**Race Condition** (1m)

A race condition is a situation that may occur inside a critical section. This happens when the result of multiple thread execution in critical section differs according to the order in which the threads execute.

Race conditions in critical sections can be avoided if the critical section is treated as an atomic instruction. Also, proper thread synchronization using locks or atomic variables can prevent race conditions.

(1m)A simple example of a race condition is **a light switch**. In some homes, there are multiple light switches connected to a common ceiling light. When these types of circuits are used, the switch position becomes irrelevant. If the light is on, moving either switch from its current position turns the light off.

As another example of a race condition inside computer software, picture two computing threads working with a given memory space. A user has just committed a form, and the backend software is writing this form into memory. Simultaneously, another user is reading out the fields of this form from the same memory space. Depending on what happens, the reading user may receive a partially incorrect form with partially updated information.

**Critical Section**(1m)

The critical section in a code segment where the shared variables can be accessed. Atomic action is required in a critical section i.e. only one process can execute in its critical section at a time. All the other processes have to wait to execute in their critical sections.

The critical section is given as follows: (1m)

```
do{
  Entry Section
  Critical Section
  Exit Section
  Remainder Section
} while (TRUE);
```

In the above example, the entry sections handles the entry into the critical section. It acquires the resources needed for execution by the process. The exit section handles the exit from the critical section. It releases the resources and also informs the other processes that critical section is free.

**Semaphore**(1m)

A semaphore is a signaling mechanism and a thread that is waiting on a semaphore can be signaled by another thread. This is different than a mutex as the mutex can be signaled only by the thread that called the wait function.

A semaphore uses two atomic operations, wait and signal for process synchronization.

(1m)

The wait operation decrements the value of its argument S, if it is positive. If S is negative or zero, then no operation is performed.

```
wait(S){
  while (S<=0);
  S--;
}
```

The signal operation increments the value of its argument S.

```
signal(S){
  S++;
}
```

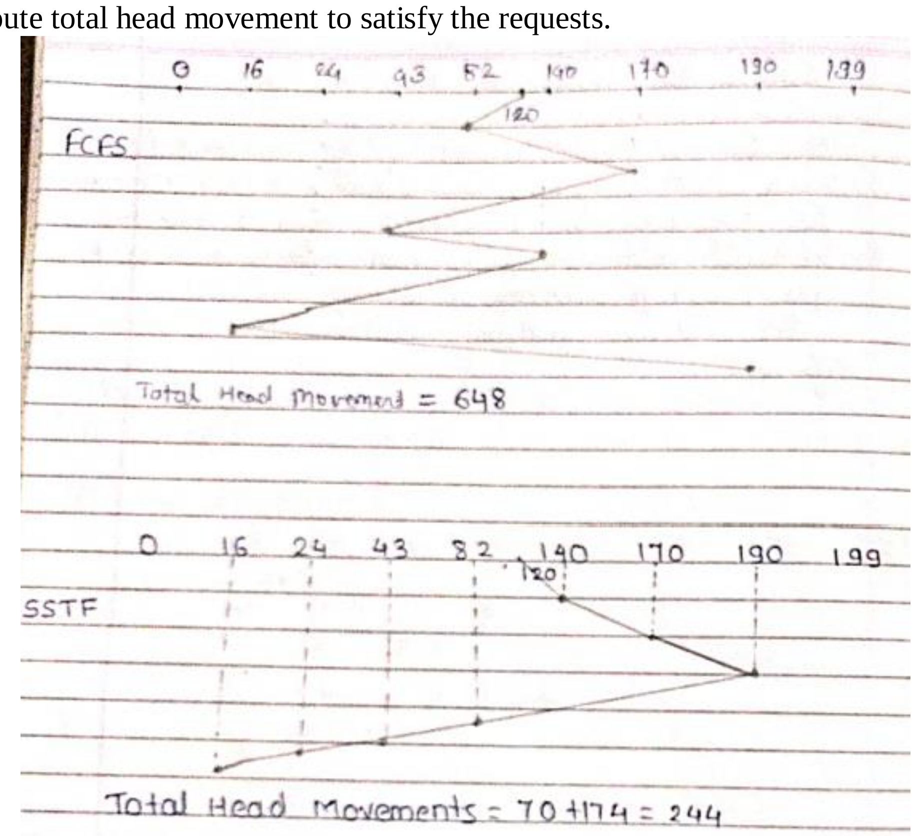
# Yeshwantrao Chavan College of Engineering, Nagpur

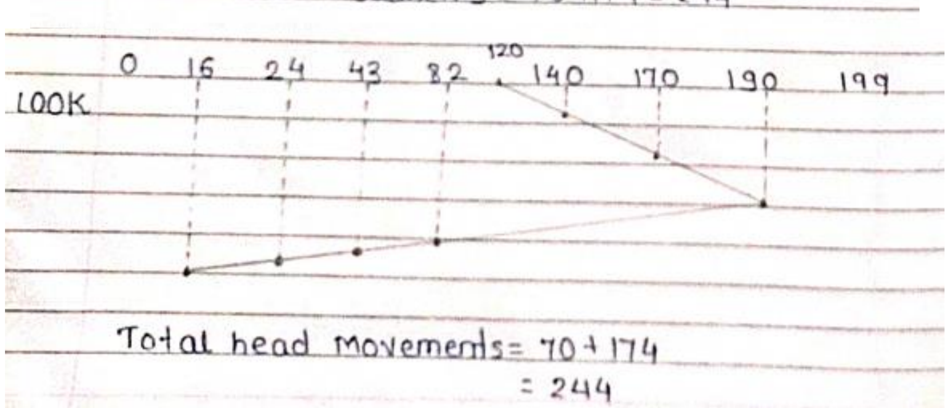
**Examination:** ESE, Exam EVEN 2021-22

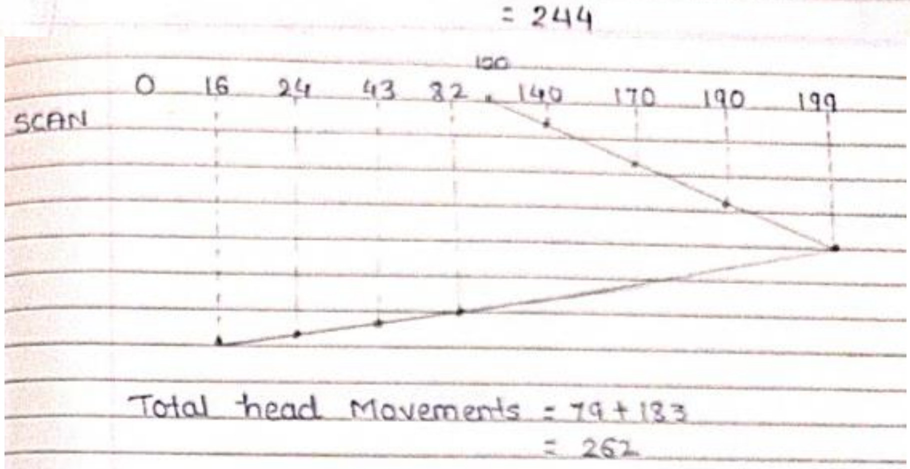
**Programme:** B.E. II Year , Sem-4

**Course Code:** IT-2255 **Course Name:** Operating System

| Q.No | | Step wise Marking |
|---|---|---|
| 4(A) | A disk has 200 cylinders (numbered 0 through 199). At a given time, it was servicing the request of reading data from cylinder 120, and the previous request, served was for cylinder 90. The pending requests (in order of their arrival) for cylinder numbers are: 82,170,43,140,24,16,190. Then for each of the following disk scheduling algorithms:<br>(I) FCFS (II) SSTF (III) LOOK, and (IV) SCAN<br>i) Find the sequence in which cylinders are served (draw graphs)<br>ii) Compute total head movement to satisfy the requests. | 4M |




**Date: 30/06/2022** **Name: Prof. K.R.Gavhale**

**Page No. :** **Sign Every Page:**

# Yeshwantrao Chavan College of Engineering, Nagpur

**Examination:** ESE, Exam EVEN 2021-22

**Programme:** B.E. II Year , Sem-4

**Course Code:** **IT-2255** **Course Name:** **Operating System**

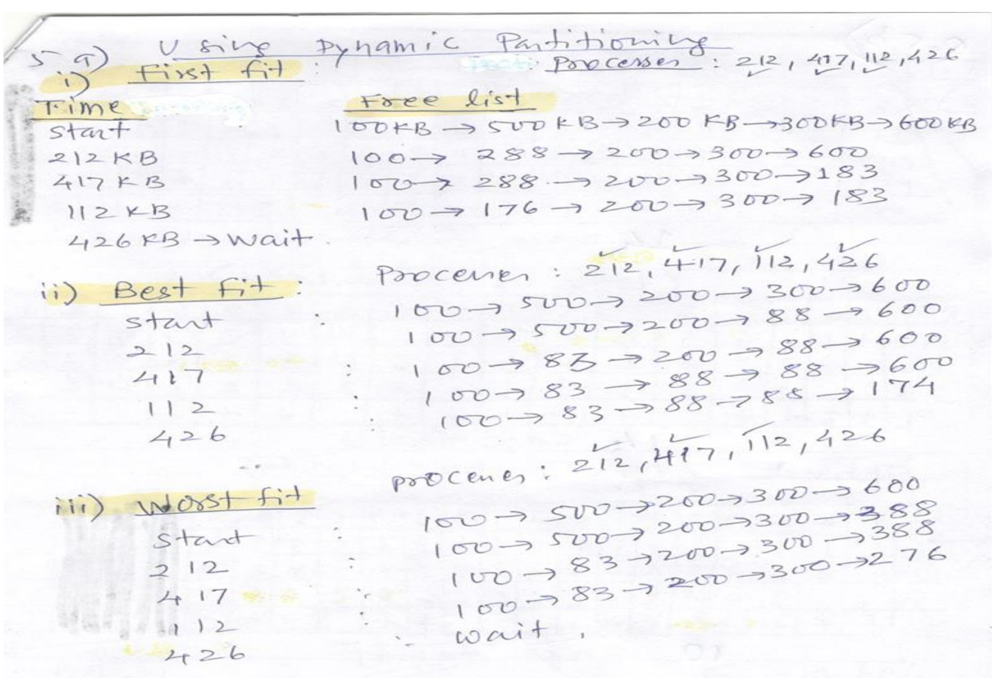
| Q.No | | Step wise Marking |
|---|---|---|
| | A) Enlist various types of fragmentation in Memory Management. Also give various memory management schemes that suffer from each type and the remedy for the same. | |
| 5(A) | **ANS:** | 4 M |
| Ans: | Types of Fragmentation in Memory Management: (1m) | |

i) Internal Fragmentation
ii) External Fragmentation
iii) Table Fragmentation

Removing / Minimizing Fragmentation: (Summary given here, detailed explanation required) (3m)

| Type of Fragmentation | MM Schemes suffering from this fragmentation | Technique to Remove/ Minimize fragmentation |
|---|---|---|
| Internal Fragmentation | Static Memory Partitioning, Paging, Paged Segmentation, Demand paging | Use Dynamic Memory Partitioning / Segmentation |
| External Fragmentation | Dynamic Memory Partitioning, Segmentation | Use Compaction and Coalescing of holes, Use paging |
| Table Fragmentation | All memory management schemes (as every MM scheme requires data structures to manage memory) | No way |

**5(B)** (B) Given main memory with partitions (initially all free) of 100 KB,500 KB,200 KB, 300KB, and 600KB (in order), then what will be the free list after application of first fit, Best fit and the worst fit algorithm(assuming Dynamic Memory Management) on the processes of sizes 212 KB, 417 KB, 112 KB and 426KB (in order)? Also show the allocations to partitions. **3 M**


5 a) Using Dynamic Partitioning
i) First fit : Processes : 212, 417, 112, 426
Time — Free list
Start 100KB → 500KB → 200KB → 300KB → 600KB
212KB 100 → 288 → 200 → 300 → 600
417KB 100 → 288 → 200 → 300 → 183
112KB 100 → 176 → 200 → 300 → 183
426KB → wait.
Processes : 212, 417, 112, 426
ii) Best fit :
start : 100 → 500 → 200 → 300 → 600
212 : 100 → 500 → 200 → 88 → 600
417 : 100 → 83 → 200 → 88 → 600
112 : 100 → 83 → 88 → 88 → 600
426 : 100 → 83 → 88 → 88 → 174
processes : 212, 417, 112, 426
iii) Worst fit
start : 100 → 500 → 200 → 300 → 600
212 : 100 → 500 → 200 → 300 → 388
417 : 100 → 83 → 200 → 300 → 388
112 : 100 → 83 → 200 → 300 → 276
426 : wait.

| Algorithm | Allocations: Process -> Partition | Free list after allocation |
|---|---|---|
| First Fit | 212->500, 417->600, 112->200, 426->wait | 100->176->200-.300->183 |
| Best Fit | 212->600, 417->500, 112->388, 426->600 | 100->83->88->88->174 |
| Worst Fit | 212->300, 417->500, 112->200, 426->wait | 100->83->200->300->276 |

| Q.No | | Step wise Marking |
|---|---|---|
| 6(A) | (A) Calculate number pf Page faults for FIFO, LRU and Optimal Algorithms for the Page Trace: 1, 2, 3 ,4, 2, 1 ,5 ,6 , 2 , 1 ,2, 3 ,7, 6, 3 ,2, 1 with number of Page Frames=3. | 4 m |

The given page reference string is: 1,2,3,4,2,1,5,6,2,1,2,3,7,6,3,2,1

No. of page frames allocated =3

a) Finding page faults for **FIFO algorithm**

| MRS:- | 1 | 2 | 3 | 4 | 2 | 1 | 5 | 6 | 2 | 1 | 2 | 3 | 7 | 6 | 3 | 2 | 1 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| PF1 | 1 | 1 | 1 | 4 | 4 | 4 | 4 | 6 | 6 | 6 | 6 | 3 | 3 | 3 | 3 | 2 | 2 |
| PF2 | | 2 | 2 | 2 | 2 | 1 | 1 | 1 | 2 | 2 | 2 | 2 | 7 | 7 | 7 | 7 | 1 |
| PF3 | | | 3 | 3 | 3 | 3 | 5 | 5 | 5 | 1 | 1 | 1 | 1 | 6 | 6 | 6 | 6 |
| Page Faults (+) | + | + | + | + | | + | + | + | + | + | | + | + | + | | + | + |

No. of page faults=14

b) Finding page faults for **LRU algorithm**

| MRS:- | 1 | 2 | 3 | 4 | 2 | 1 | 5 | 6 | 2 | 1 | 2 | 3 | 7 | 6 | 3 | 2 | 1 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| PF1 | 1 | 1 | 1 | 4 | 4 | 4 | 5 | 5 | 5 | 1 | 1 | 1 | 7 | 7 | 7 | 2 | 2 |
| PF2 | | 2 | 2 | 2 | 2 | 2 | 2 | 6 | 6 | 6 | 6 | 3 | 3 | 3 | 3 | 3 | 3 |
| PF3 | | | 3 | 3 | 3 | 1 | 1 | 1 | 2 | 2 | 2 | 2 | 2 | 6 | 6 | 6 | 1 |
| Page Faults (+) | + | + | + | + | | + | + | + | + | + | | + | + | + | | + | + |

No. of page faults=14

c) Finding page faults for **Optimal algorithm**

| MRS:- | 1 | 2 | 3 | 4 | 2 | 1 | 5 | 6 | 2 | 1 | 2 | 3 | 7 | 6 | 3 | 2 | 1 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| PF1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 3 | 3 | 3 | 3 | 2 | 1 |
| PF2 | | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 7 | 7 | 7 | 7 | 7 |
| PF3 | | | 3 | 4 | 4 | 4 | 5 | 6 | 6 | 6 | 6 | 6 | 6 | 6 | 6 | 6 | 6 |
| Page Faults (+) | + | + | + | + | | | + | + | | | | + | + | | | + | + |

No. of page faults=10

| | | |
|---|---|---|
| | | |
| 6(B) | (B) What do you mean by thrashing ? State the ways to prevent it.<br>**ANS:**<br>If most of the processes in the system are above their upper page fault rate threshold, a system may exhibit a behavior known as thrashing. It is characterized by intensive paging activity. (1m)<br><br>Ways to prevent Thrashing: (1m)<br>• Use local replacement policies<br>• Provide a process with as many frames as it needs using Working Set Model<br>• Use Page Fault frequency strategy to measure and control page fault rate | **2m** |