Disk requests come to a disk driver for cylinders in the order 10, 22, 20, 2, 40, 6 and 38, at a time when the disk drive is reading from cylinder 20. The seek time is 6 ms/cylinder. The total seek time, if the disk arm scheduling algorithms is first-comefirst-served is

1. 360 ms

2. 850 ms

3. 900 ms

4. None of the above

Data:

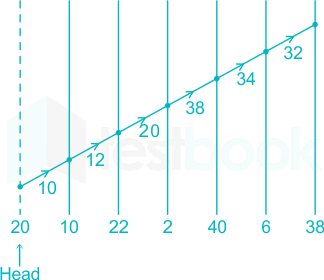
**Seek time = t = 6 ms per cylinder**

**Head = 20**

Concept:

**Since, it is a first-come-first-serve scheduling requests will be served in the given sequence: 10, 22, 20, 2, 40, 6 and 38**

Diagram



Formula:

Total head movement = (20 - 10) + (22 - 10) + (22 - 2) + (40 - 2) + (40 - 6) + (38 - 6) = = 10 + 12 + 20 + 38 + 34 + 32 = 146 Total Seek time = 146 × 6 = 876 ms

Consider a disk system having 60 cylinders. Disk requests are received by a disk drive for cylinders 10, 22, 20, 2, 40, 6, and 38, in that order. Assuming the disk head is currently at cylinder 20, what is the time taken to satisfy all the requests if it takes 2 milliseconds to move from one cylinder to an adjacent one and Shortest Seek Time First (SSTF) algorithm is used ?

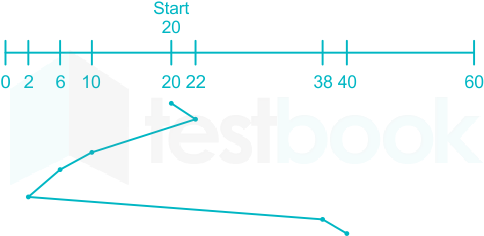
1. 240 milliseconds

2. 96 milliseconds

3. 120 milliseconds

4. 112 milliseconds

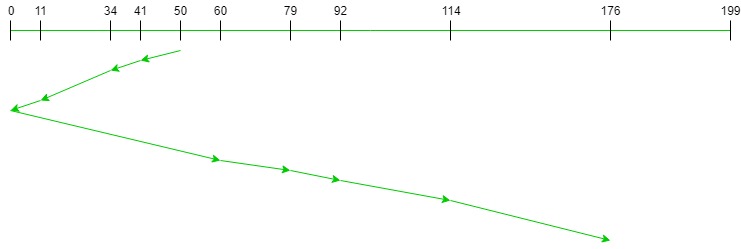
**Shortest Seek Time First (SSTF)** 10, 22, 20, 2, 40, 6, and 38

****

**SSTF =(22-20+22-10+10-6+6-2+38-2+40-38)=2+12+4+4+36+2=60It takes 2 milliseconds to move from one cylinder to adjacent one = 60x2 =120 milliseconds**

**Input:   
Request sequence = {176, 79, 34, 60, 92, 11, 41, 114}  
Initial head position = 50  
Direction = left (We are moving from right to left)  
Output:  
Total number of seek operations = 226  
Seek Sequence is  
41  
34  
11  
0  
60  
79  
92  
114  
176**

**The following chart shows the sequence in which requested tracks are serviced using SCAN(elevator).**

****

***SCAN Disk Scheduling Algorithm***

**Therefore, the total seek count is calculated as:**

**= (50-41) + (41-34) + (34-11) + (11-0) + (60-0) + (79-60) + (92-79) + (114-92) + (176-114)  
= 226**

Top of Form

Bottom of Form

Top of Form

A computer has four page frames. The time of loading, time of last access, and the R and M bits for each page are as shown below (the times are in clock ticks):

# A computer has four page frames. The time of loading, time of last access, and the R and M bits for each page are shown below (the times are in clock ticks): (2 points) https://gateoverflow.in/?qa=blob&qa_blobid=13069318226263648489 a) Which page will second chance replace? b) Which page will LRU replace? c) Which page will NRU replace? d) Which page will FIFO replace?

A **replacement algorithm** is a method used by computer**programs** to determine which data should be replaced when the memory becomes full. Examples include LRU and FIFO.

Let's analyze each page replacement algorithm:  
a) Second Chance: In this method, we give a second chance to the page with the R **bit** set to 1. Here, page 0 and 3 have R bits set to 1. We then look at the oldest loaded page with R bit set to 0. Page 2 is the oldest with R bit set to 0. So, Second Chance will replace page 2.  
b) LRU (Least Recently Used): We replace the page that was accessed the least recently. Here, the "last ref" value shows the last access time. Page 1 has the smallest last ref value (265). Therefore, **LRU**will replace page 1.

c) NRU (Not Recently Used): We classify pages based on R and M bits. The priority order is: (0,0), (0,1), (1,0), and (1,1). Page 2 has the lowest priority (0,0). So, NRU will replace page 2. Bottom of Form

d) FIFO (First In, First Out): We replace the oldest(cũ) loaded page. Here, the **"loaded"** value shows the load time. Page 3 has the smallest loaded value (110). Therefore, FIFO will replace page 3.

Assume jobs A-D arrive in quick succession in the READY queue. Using round robin scheduling (quantum=4), the turnaround time for job D is \_\_\_\_.

 Arrival time: 0 1 2 3  
Job: A B C D  
CPU cycle: 8 4 9 5 round robin scheduling. the turnaround time

A B C D A C D C

4 4 4 4 4 4 1 1

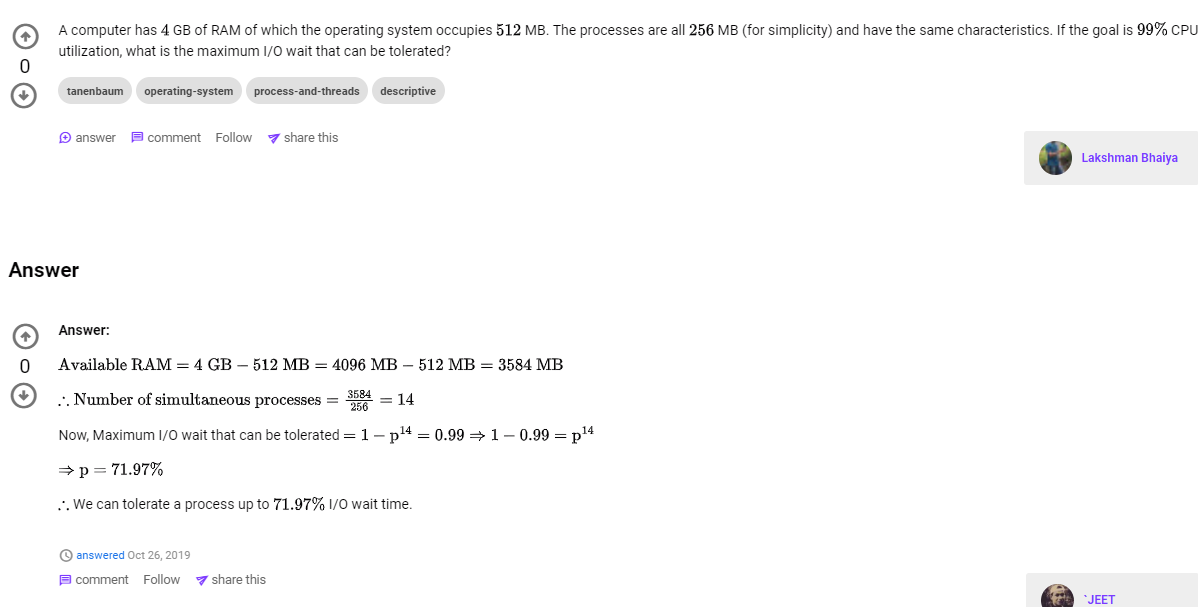
**To calculate the turnaround time for job D using round-robin scheduling with a quantum of 4, we need to determine the order in which the jobs are executed.**

**Based on the arrival times and CPU cycles provided, the order of execution would be as follows:**

1. **Job A starts at time 0 and runs for 4 cycles.**
2. **Job B starts at time 4 and runs for 4 cycles.**
3. **Job C starts at time 8 and runs for 4 cycles.**
4. **Job D starts at time 12 and runs for 4 cycles.**
5. **Job A resumes at time 16 and runs for the remaining 4 cycles.**
6. **Job C resumes at time 20 and runs for the remaining 4 cycles.**
7. **Job D resumes at time 24 and runs for 1 cycles**
8. **Job C resumes at time 25 and runs for the remaining 1 cycles.**

**Therefore, the turnaround time for job D is the time it takes from its arrival until it completes execution. In this case, job D completes execution at time 25 - 3 = 22.**

**So, the turnaround time for job D is 22.**



A computer has 4 GB of RAM of which the operating system occupies 512 MB. The processes are all 256 MB (for simplicity) and have the same characteristics. If the goal is 99% CPU utilization, what is the maximum I/O wait that can be tolerated?

### [Answer](https://github.com/dsifford/CSC_4420/blob/master/assignment_02/README.md#answer)

Available RAM = 4096 MB - 512MB = 3584 MB

3584 / 256 = 14 simultaneous processes

Maximum I/O wait that can be tolerated...

1 - p^14 = 0.99

1 - 0.99 = p^14

root14(1 - 0.99) = root14(p^14)

71.97% = p

Assume that process A-D make up the set of runnable processes on memory as  
B1 B2 B3 A1 A2 A5 A7 D3 D4 D6 C1 C6 C5.  
Suppose D gets a page fault.  
Which page is replaced using the local policy? Assume that the replaced page is always a last page.

To determine which page is replaced using the local policy, we need to understand the local page replacement algorithm. In this case, the algorithm replaces the last page that was brought into memory.

Given the set of runnable processes on memory: B1 B2 B3 A1 A2 A5 A7 D3 D4 D6 C1 C6 C5, and assuming that process D gets a page fault, we can determine which page will be replaced.

Since the algorithm replaces the last page that was brought into memory, we need to identify the last page that was brought in before process D. Looking at the list, we can see that the last page brought in before process D is D6.

Therefore, if process D gets a page fault, the page that will be replaced using the local policy is D6.

1. Suppose a virtual address space of 2^32 words and the page size is 2^12 words. If the virtual address is 12345678 in Hexadecimal, what would be the page number in Hexadecimal?

It is given that virtual address is 32 bit long.

Hence, there are 2^32 addresses in the virtual address space.

Page Size is given to be 4 KB ( there are 4K (4 \* (2 ^ 10) )addresses in a page), so the number of pages will be ( 2^32 ) / ( 2 ^ 12 ) = 2 ^ **20.**

To address each page 20 bits are required.

The most significant 20 bits in the virtual address will denote the page number being referred and the remaining 32-20=**12 bits** **will be the page offset.**

One thing to remember is page size (in the virtual address space ) is always same as the frame size in the main memory. Hence the last 12 bits will remain same in the physical address as that of the virtual address.

To get the frame address in the main memory just use the **first 20 bits**.

Example: Consider the virtual address 0x12345678

Here 12345 in **12345**678 denotes the page number .

**A. 12345**

B. 1234

C. 123

D. 123456

1. How large is the block size, if the maximum partition size is 8 MB and the FAT type is FAT-12?

In FAT12 the address is 12-bit long, so there are maximum ~212 clusters (slightly less since some addresses are reserved for special purposes). For example with 2KB clusters the maximum volume size will be

~212 × ?Bytes = 8MB=23+20 bytes =>?Kb = 211byte

~212 × KBytes = 8MB=23+10 =>Kb = 2

~212 × KBytes = 256MB=28+10 =>Kb = 2^6

1. 8 KB
2. 4 KB
3. **2 KB**
4. 1KB
5. Five batch jobs A through E, arrive at a computer center at almost the same time. They have estimated running times of 8, 6, 2 10, and 4 minutes. Determine the average waiting time for SJF (Shortest job first) scheduling Ignore process switching overhead.

2 4 6 8 10

2 6 12 20 30 = 70/5

1. 14 minutes
2. 8 minutes
3. 6 minutes
4. 18.8 minutes
5. Concept:
6. Least Recently Used page replacement:
7. The Least Recently Used page replacement method tracks page consumption over a short time period. It operates on the premise that the pages that have been widely utilized in the past are likely to be heavily used in the future as well.
8. Explanation:
9. There are four-page frames are given and an LRU page replacement policy is used.
10. The given page size= 4,
11. The given page reference string is =
12. **2-> 3-> 6-> 4-> 6-> 3-> 1-> 2-> 4-> 6**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Page**  **frames**  **size=**  **4** | **2** | **2** | **2** | **2** | **2** | **2** | **1** | **1** | **1** | **1** |
|  | **3** | **3** | **3** | **3** | **3** | **3** | **3** | **3** | **6** |
|  |  | **6** | **6** | **6** | **6** | **6** | **6** | **4** | **4** |
|  |  |  | **4** | **4** | **4** | **4** | **2** | **2** | **2** |
| **Page fault** | **F** | **F** | **F** | **F** | **H** | **H** | **F** | **F** | **F** | **F** |

BT. Five batch jobs A through E, arrive at a computer center at almost the same time. They have estimated running times of 11, 6, 2, 4, and 8 minutes. Their (externally

determined) priorities are 3, 5, 2, 1, and 4, respectively, with 5 being the highest priority. For each of the following scheduling algorithms, determine the mean process turnaround time.

Ignore process switching overhead.

1. Round-robin
2. Priority scheduling 6 8 11 2 4

6 14 25 27 31 =103/5

1. First come, First served (run in order 11, 6, 2, 4, 8)
2. Shortest job first

For (a), assume that the system is multiprogrammed, and that each job gets its fair share of the CPU. For (b) through (d) assume that only one job runs at a time, until it finishes. All jobs are completely CPU bound. Assume that the time quantum of the scheduler is 1 minute.

Answer:

Remember that the turnaround time is the amount of time that elapses between the job arriving and the job completing. Since we assume that all jobs arrive at time 0, the turnaround time will simply be the time that they complete.

1. Round Robin: The table below gives a break down of which jobs will be processed during each time quantum. A \* indicates that the job completes during that quantum.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
| A | B | C | D | E | A | B | C\* | D | E | A | B | D | E | A |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | 31 |
| B | D\* | E | A | B | E | A | B\* | E | A | E | A | E\* | A | A | A\* |

Average turnaround = (8 + 17 + 23 + 28 + 31)/5 = 107/5 = 21.4 minutes

1. Priority Scheduling:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 1-6 | 7-14 | 15-25 | 26-27 | 28-31 |
| B | E | A | C | D |

Avg. turnaround = (6 + 14 + 25 + 27 + 31)/5 = 103/5 = 20.6 minutes

1. FCFS

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 1-11 | 12-17 | 18-19 | 20-23 | 24-31 |
| A | B | C | D | E |

Avg. turnaround = (11 + 17 + 19 + 23 + 31)/5 = 101/5 = 20.2 minutes

1. SJF

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 1-2 | 3-6 | 7-12 | 13-20 | 21-31 |
| C | D | B | E | A |

Avg. turnaround = (2 + 6 + 12 + 20 + 31)/5 = 71/5 = 14.2 minutes

Finding the average process turnaround time can also be done by multiplying the number of unfinished processes by the time they remain unfinished dividing by number of processes. This makes finding solutions for parts b, c, and d particularly efficient. For example, the calculation for part C can be represented by (5\*11 + 4\*6 + 3\*2 + 2\*4 + 1\*8)/5.