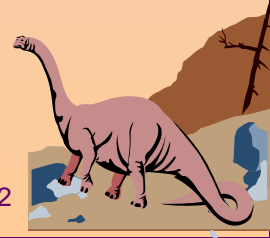




SO 1: Draw the disk structure



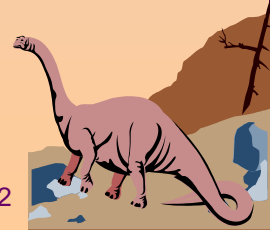
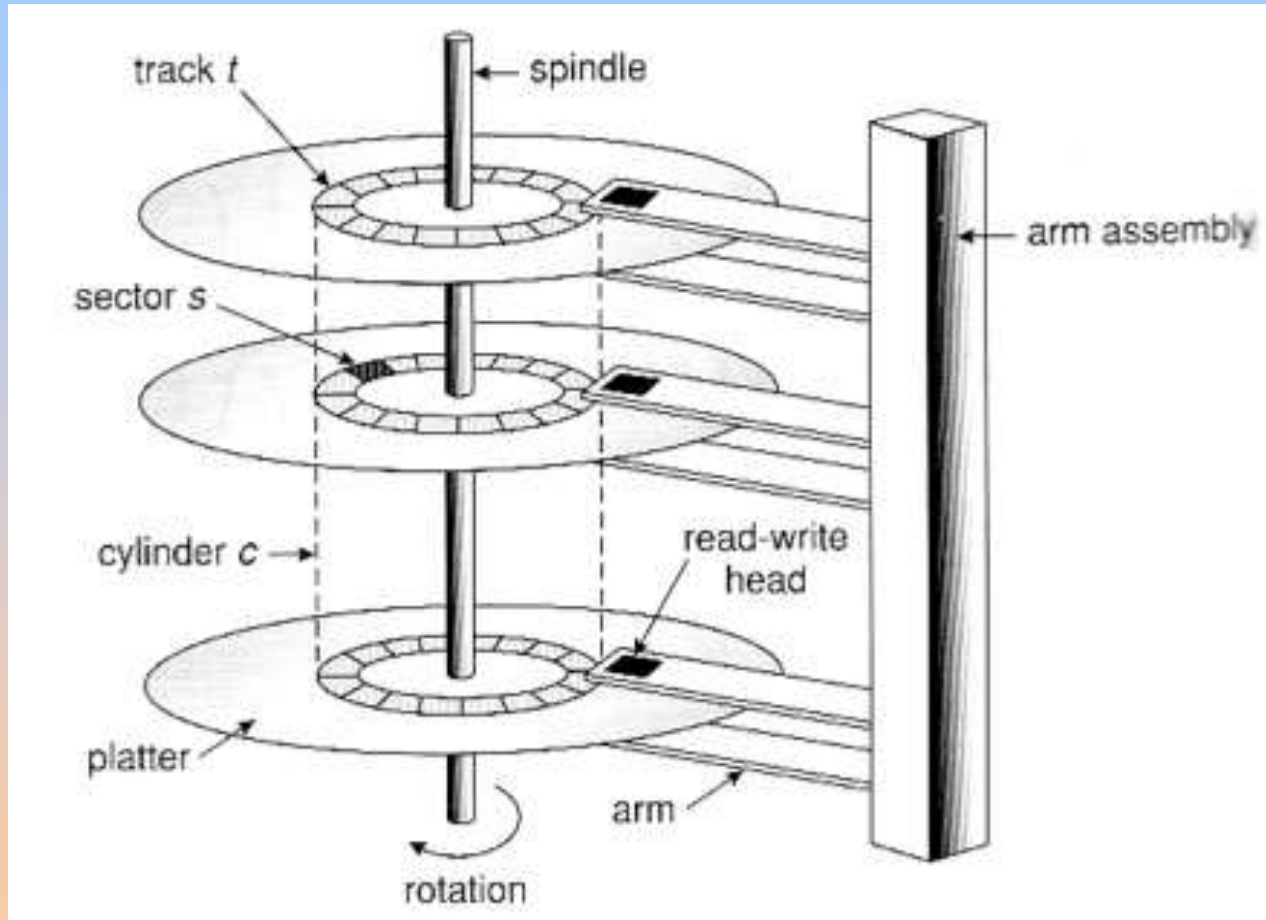


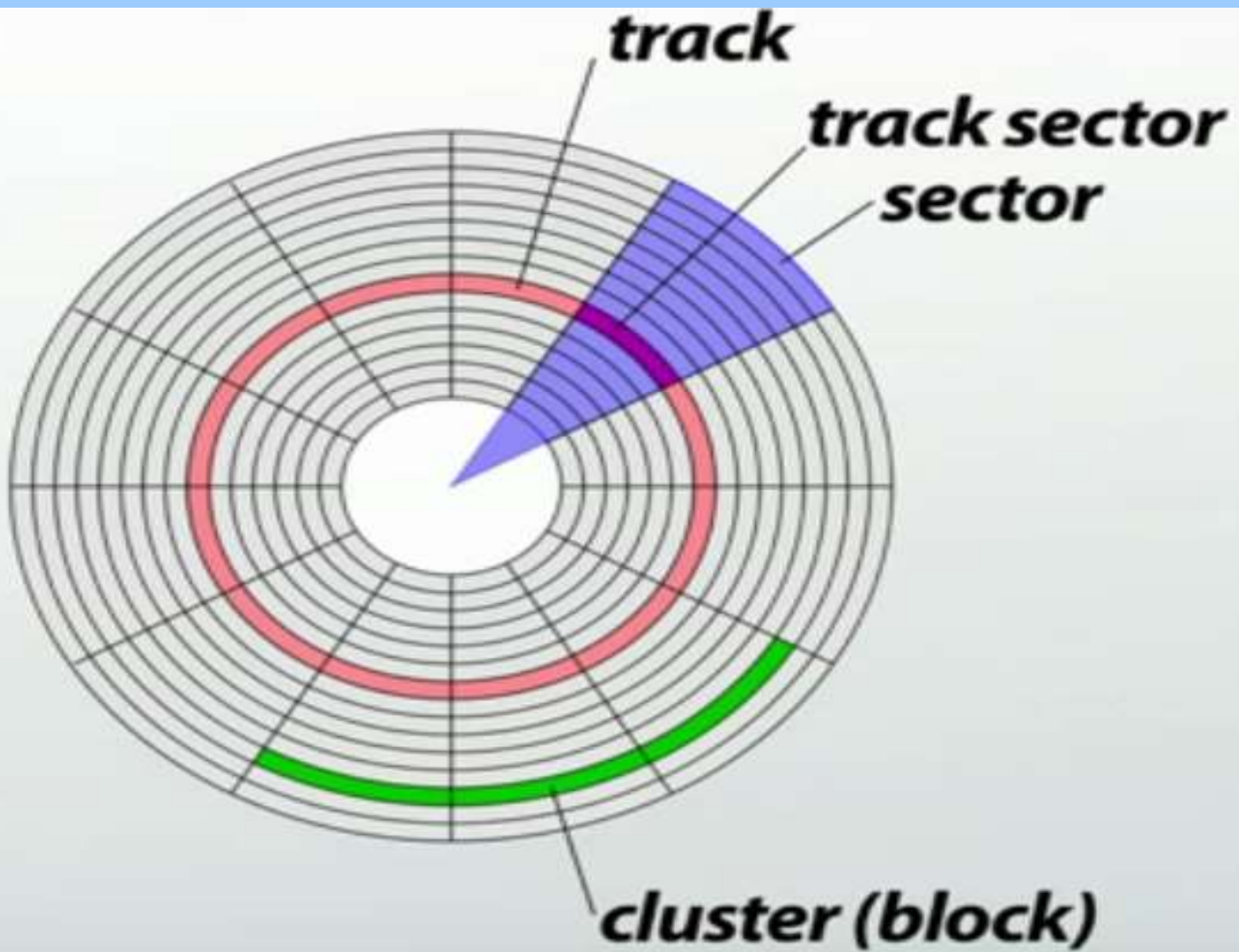
Disk Structure

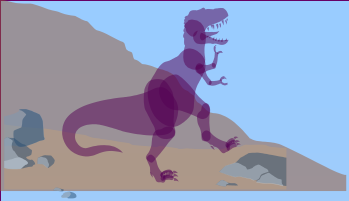
□ Magnetic Disks

- One or more **platters** in the form of disks covered with magnetic media.
- Each platter has two working **surfaces**
- Each working surface is divided into a number of concentric rings called **tracks**
- The collection of all tracks that are at the same distance from the edge of the platter is called a **cylinder**
- Each track is further divided into **sectors**
- The data on a hard drive is read by read-write **heads**
- Each head is on a separate **arm**, and controlled by a common **arm assembly**



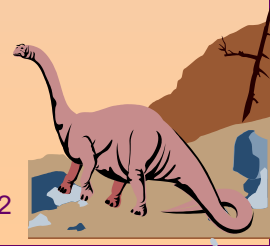






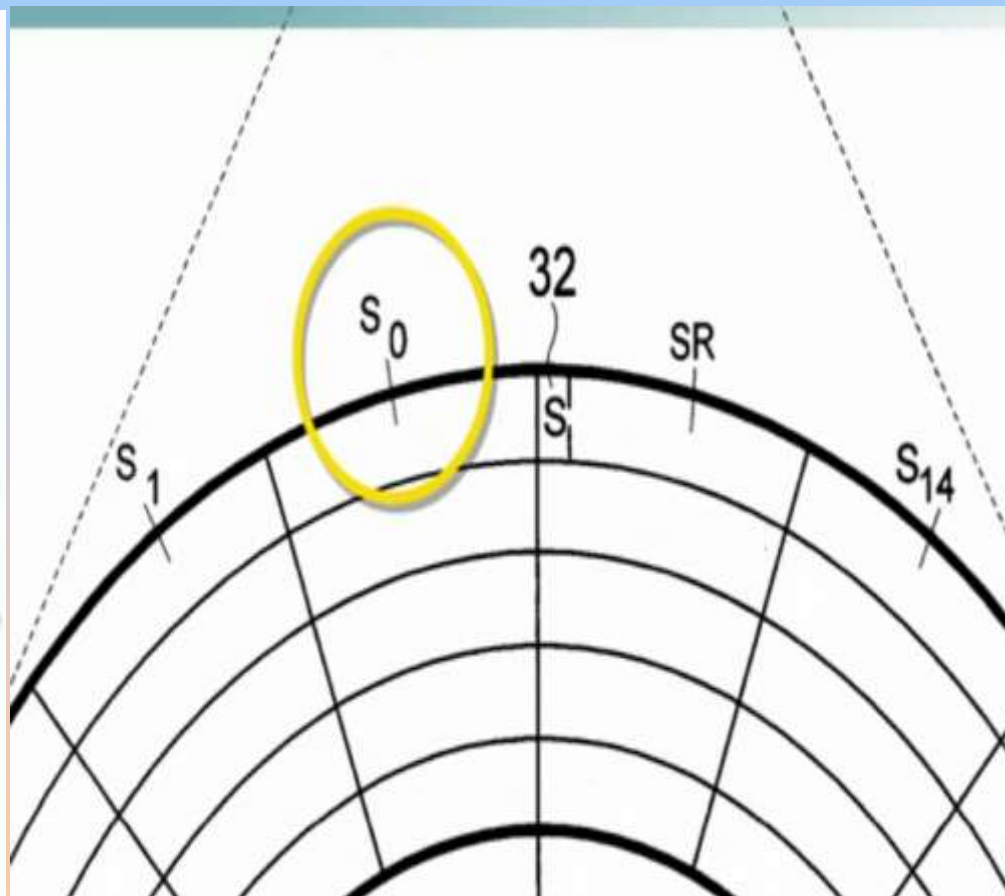
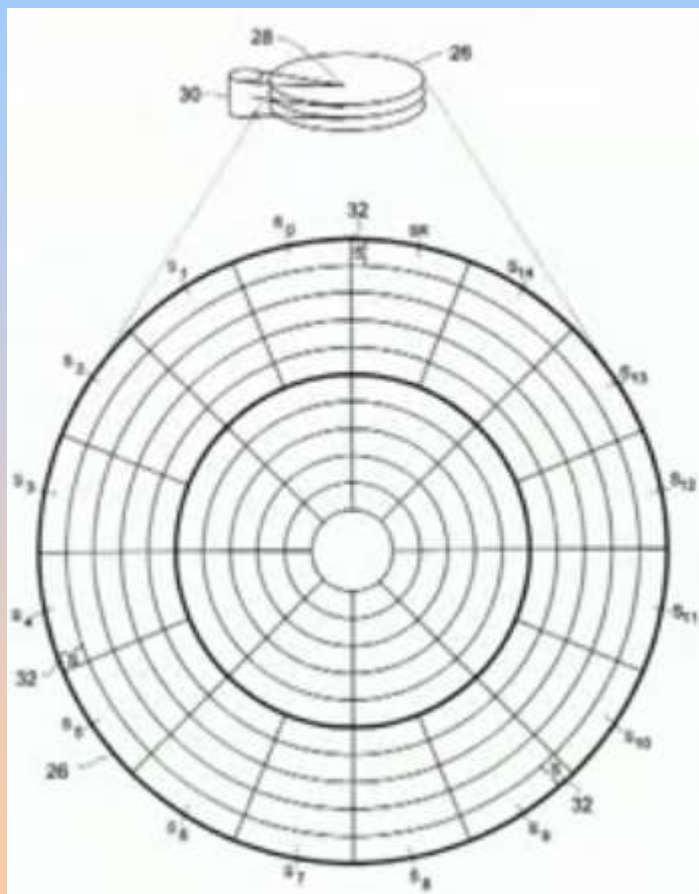
Disk Structure (cont...)

- Disk drives are addressed as large 1-dimensional arrays of logical blocks, where the logical block is the smallest unit of transfer.
- The 1-dimensional array of logical blocks is mapped into the sectors of the disk sequentially.
 - Sector 0 is the first sector of the first track on the outermost cylinder.
 - Mapping proceeds in order through that track, then the rest of the tracks in that cylinder, and then through the rest of the cylinders from outermost to innermost.





FAT





Disk Structure (cont...)

- The rate at which data can be transferred from the disk to the computer is composed of several steps:
 - **Seek time** or **random access time** Seek time is the time taken for a hard disk controller to locate a specific piece of stored data.
 - When anything is read or written to a disc drive, the read/write head of the disc needs to move to the right position. The actual physical positioning of the read/write head of the disc is called seeking.
 - The amount of time that it takes the read/write head of the disc to move from one part of the disk to another is called the seek time



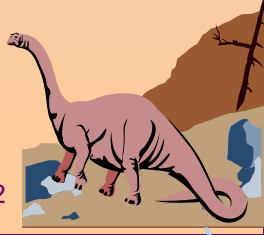


- The ***rotational latency*** is the amount of time required for the desired sector to rotate around and come under the read-write head
- The ***transfer rate***, which is the time required to move the data electronically from the disk to the computer





SO 2: Demonstrate the disk scheduling algorithms





Disk Scheduling

- ❑ The operating system is responsible for using hardware efficiently — for the disk drives, this means having a fast access time and disk bandwidth.
- ❑ Access time has two major components
 - ❑ Seek time is the time for the disk arm to move the heads to the cylinder containing the desired sector.
 - ❑ Rotational latency is the additional time waiting for the disk to rotate the desired sector to the disk head.
- ❑ Minimize seek time
- ❑ Disk bandwidth is the total number of bytes transferred, divided by the total time between the first request for service and the completion of the last transfer.



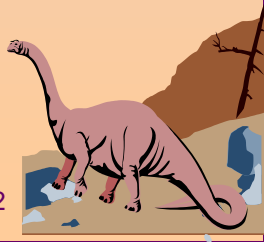


Disk Scheduling (Cont.)

- Several algorithms exist to schedule the servicing of disk I/O requests.
- We illustrate them with a request queue (0-199).

98, 183, 37, 122, 14, 124, 65, 67

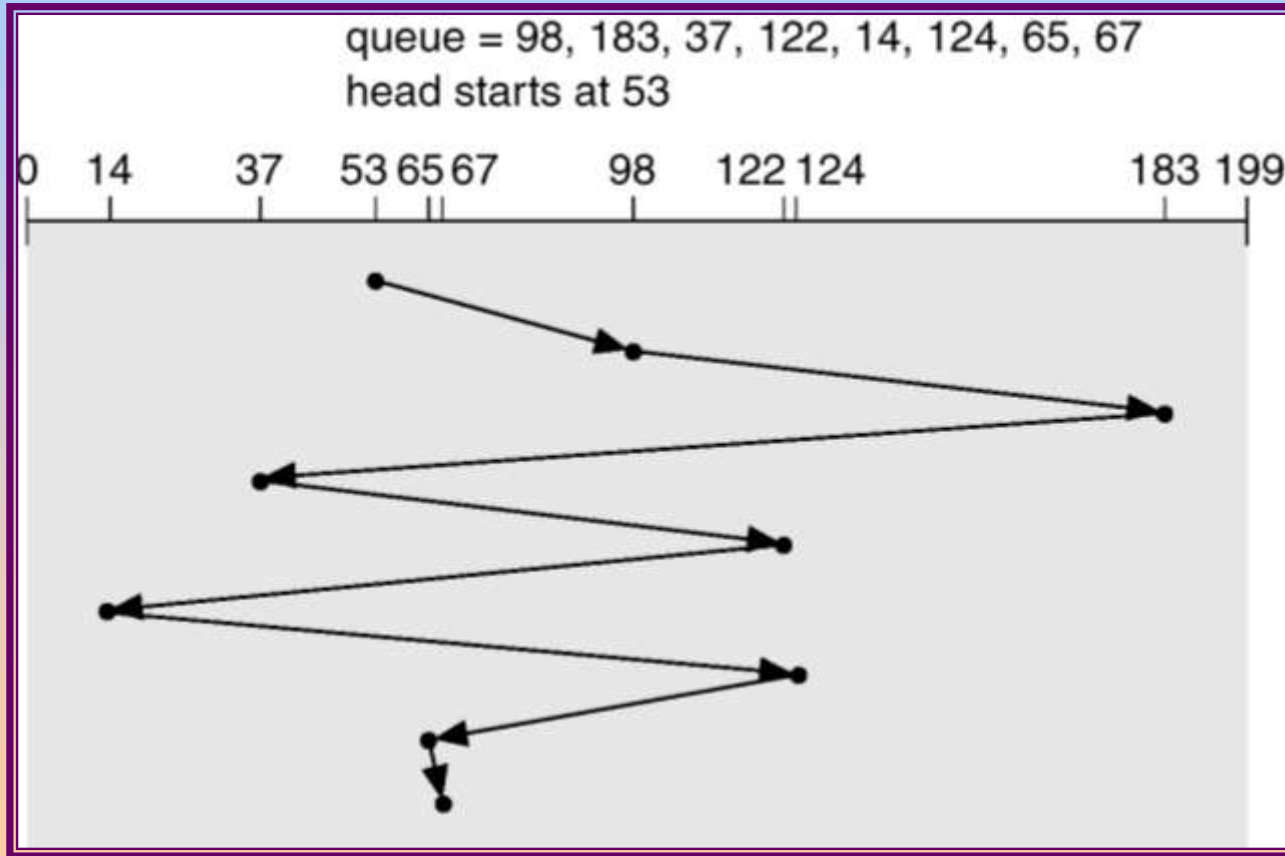
Head pointer 53





FCFS

Illustration shows total head movement of 640 cylinders.

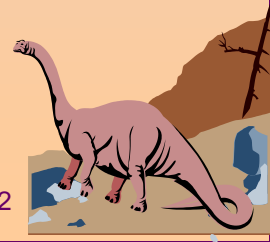


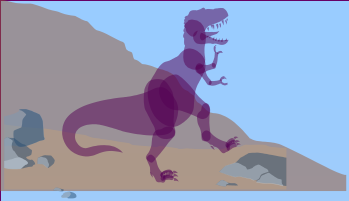


□ The order of request serviced will be:

53 -> 98 -> 183 -> 37 -> 122 -> 14 -> 124 -> 65 -> 67

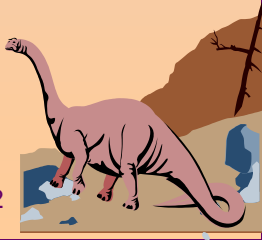
Thus, the total head movement is $|98-53|+|183-98|+|37-183|+|122-37|+|14-122|+|124-14|+|65-124|+|67-65| = 45+85+146+85+108+110+59+2=640$





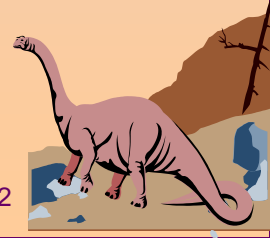
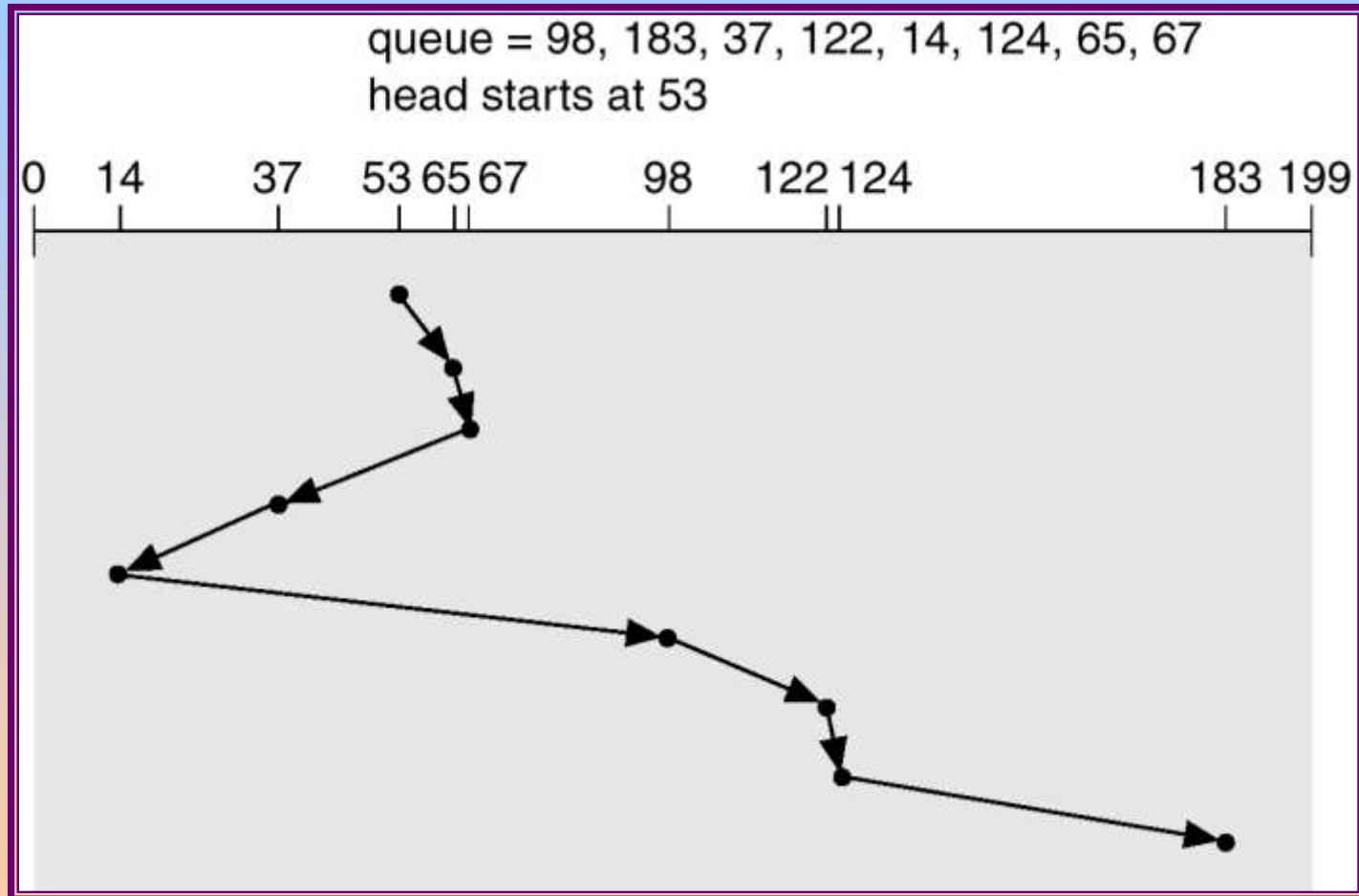
SSTF

- ❑ Selects the request with the minimum seek time from the current head position.
- ❑ SSTF scheduling is a form of SJF scheduling; may cause starvation of some requests.
- ❑ Illustration shows total head movement of 236 cylinders.



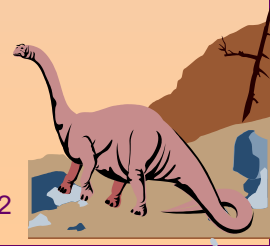


SSTF (Cont.)





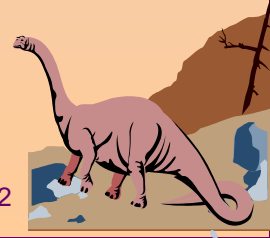
- e.g. for the same question (as in FCFS), the order of request serviced using SSTF will be:
53 -> 65 -> 67 -> 37 -> 14 -> 98 -> 122 -> 124 -> 183
Thus, the total head movement is $|65-53|+|67-65|+|37-67|+|14-37|+|98-14|+|122-98|+|124-122|+|183-124| = 12+2+30+23+84+24+2+59 = 236$





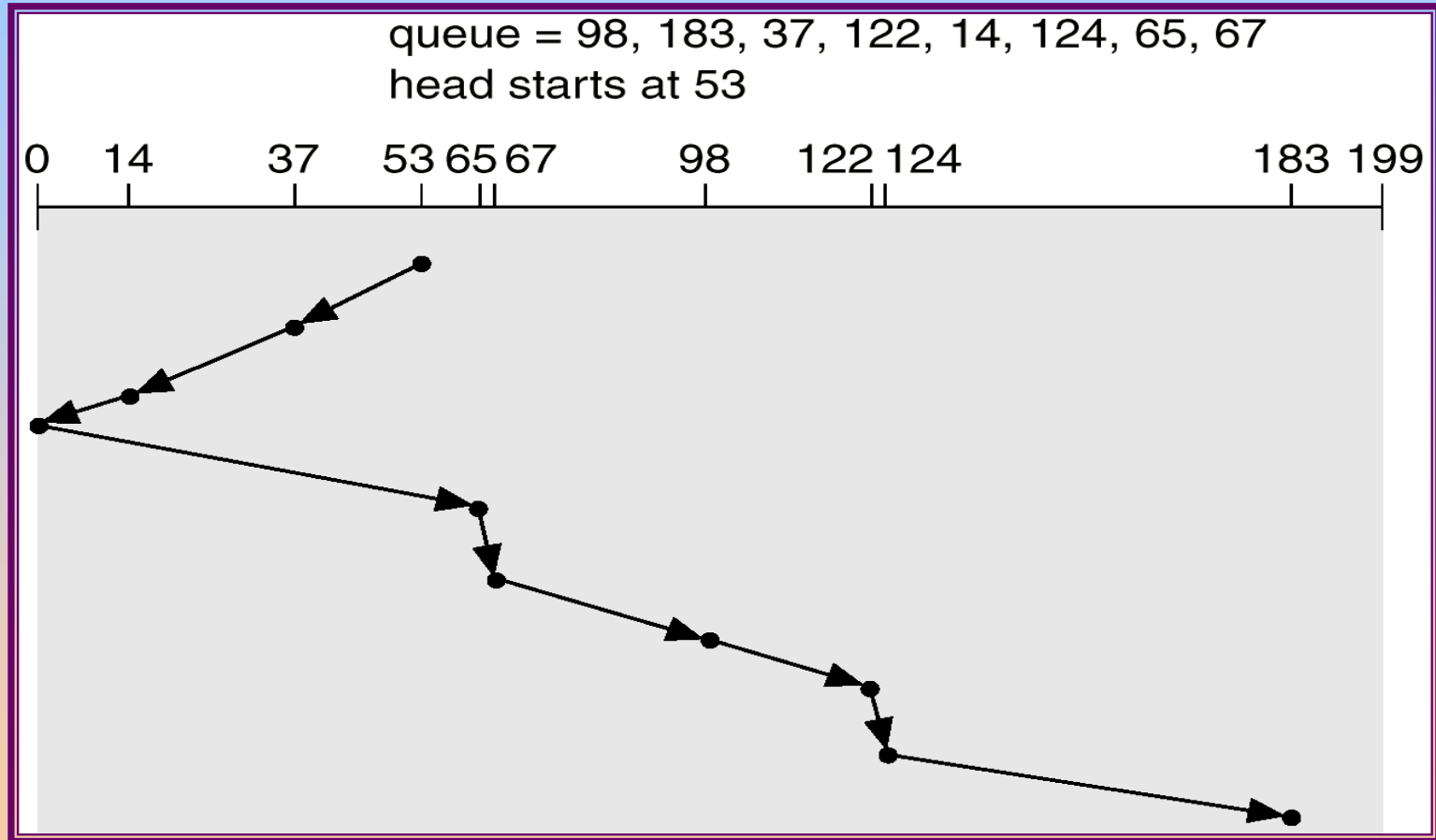
SCAN

- ❑ The disk arm starts at one end of the disk, and moves toward the other end, servicing requests until it gets to the other end of the disk, where the head movement is reversed and servicing continues.
- ❑ Sometimes called the *elevator algorithm*.
- ❑ Illustration shows total head movement of 208 cylinders.





SCAN (Cont.)

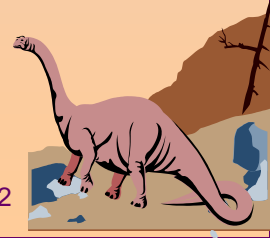




- e.g. for the same question (as in FCFS), the order of request serviced using SCAN will be:

53 -> 37 -> 14 -> 0 -> 65 -> 67 -> 98 -> 122 -> 124 -> 183

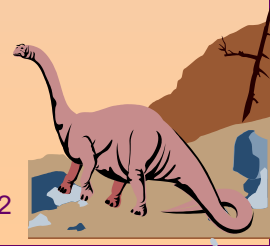
Thus, the total head movement is $|37-53|+|14-37|+|0-14|+|65-0|+|67-65|+|98-67|+|122-98|+|124-122|+|183-124|$
 $= 16+23+14+65+2+31+24+2+59=236$





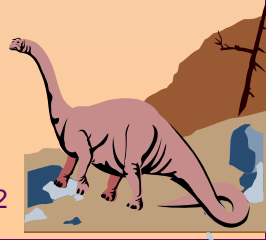
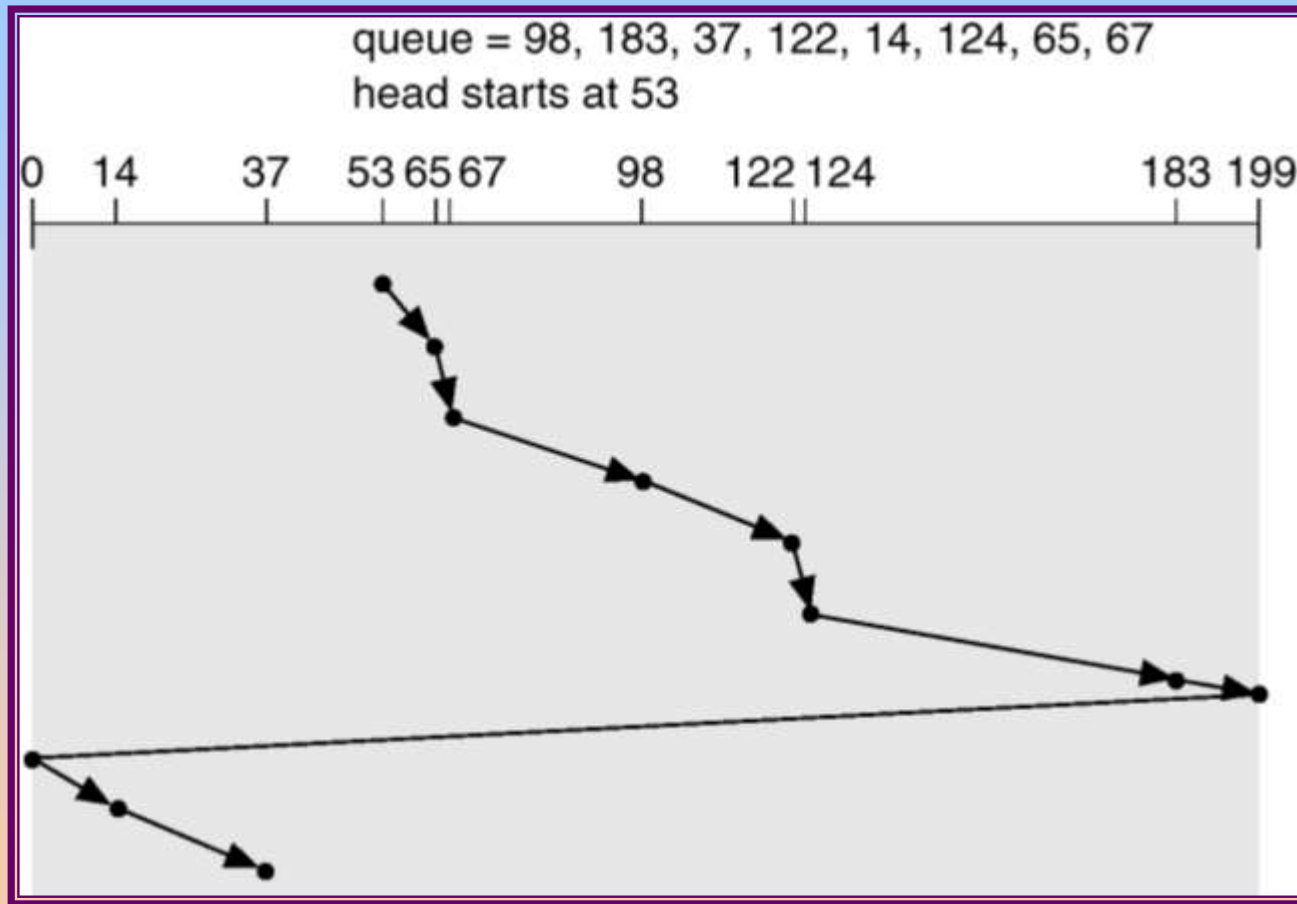
C-SCAN

- ❑ Provides a more uniform wait time than SCAN.
- ❑ The head moves from one end of the disk to the other, servicing requests as it goes. When it reaches the other end, however, it immediately returns to the beginning of the disk, without servicing any requests on the return trip.
- ❑ Treats the cylinders as a circular list that wraps around from the last cylinder to the first one.



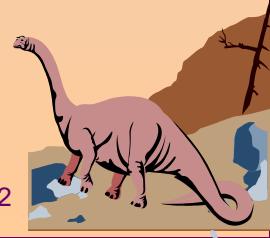


C-SCAN (Cont.)





- Total head movements incurred while servicing these requests
- $= (65 - 53) + (67 - 65) + (98 - 67) + (122 - 98) + (124 - 122) + (183 - 124) + (199 - 183) + (199 - 0) + (14 - 0) + (41 - 14)$
- $= 12 + 2 + 31 + 24 + 2 + 59 + 16 + 199 + 14 + 27$
- $= 386$





❑ LOOK Disk Scheduling Algorithm-



❑ LOOK Algorithm is an improved version of the SCAN ALGORITHM

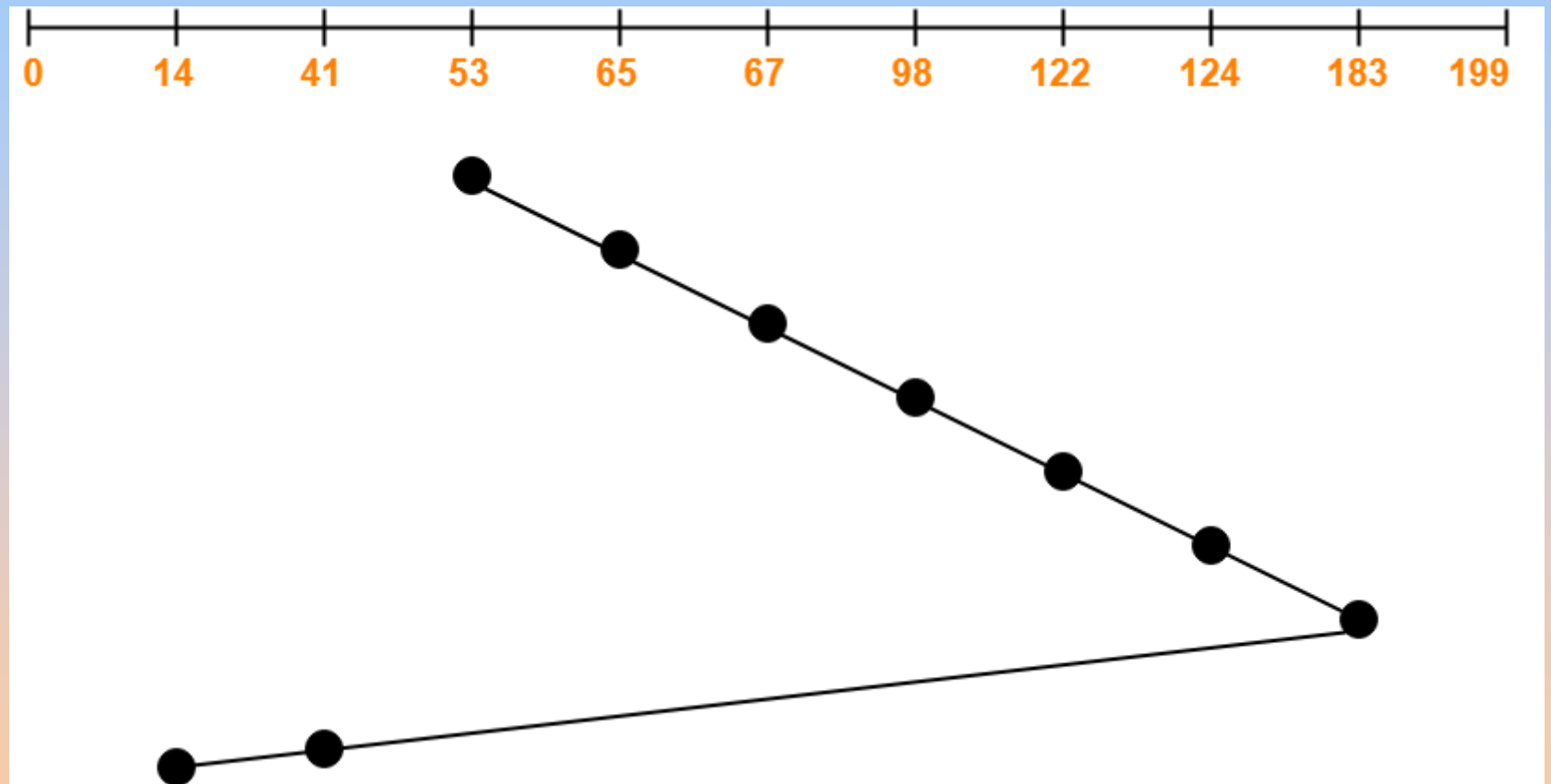
❑ Head starts from the first request at one end of the disk and moves towards the last request at the other end servicing all the requests in between.

❑ After reaching the last request at the other end, head reverses its direction.

❑ It then returns to the first request at the starting end servicing all the requests in between.

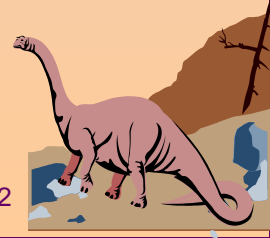
❑ The same process repeats.







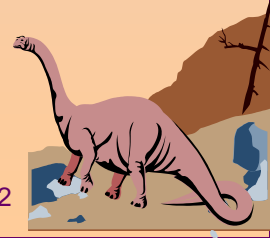
- Total head movements incurred while servicing these requests
- $= (65 - 53) + (67 - 65) + (98 - 67) + (122 - 98) + (124 - 122) + (183 - 124) + (183 - 41) + (41 - 14)$
- $= 12 + 2 + 31 + 24 + 2 + 59 + 142 + 27$
- $= 299$





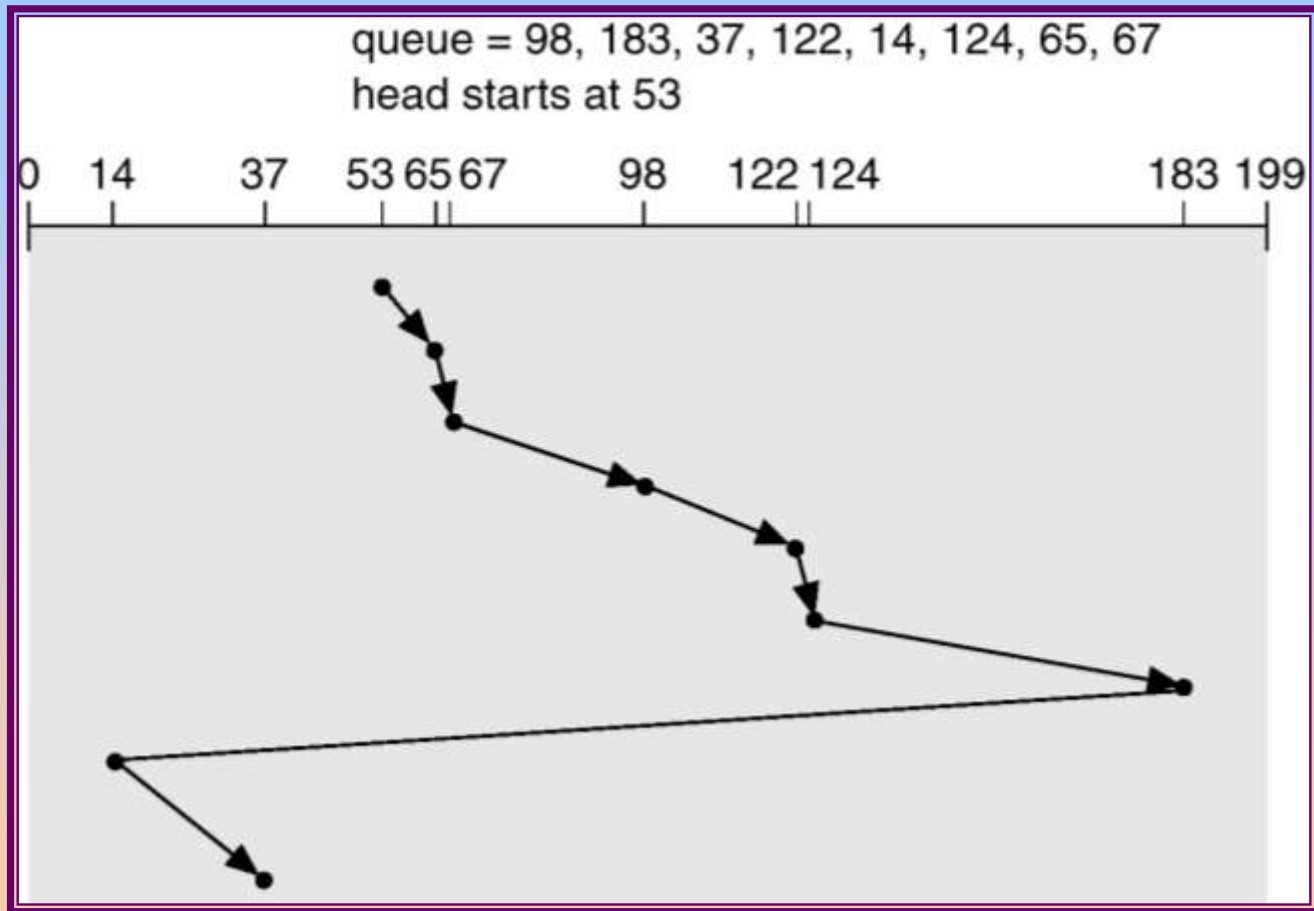
C-LOOK

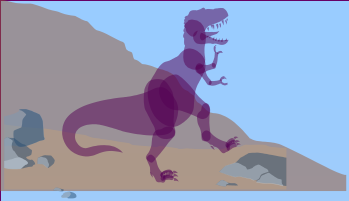
- Version of C-SCAN
- Arm only goes as far as the last request in each direction, then reverses direction immediately, without first going all the way to the end of the disk.



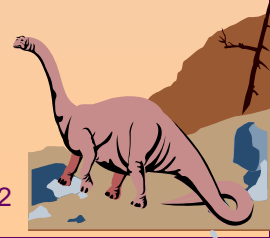


C-LOOK (Cont.)





- Total head movements incurred while servicing these requests
- $= (65 - 53) + (67 - 65) + (98 - 67) + (122 - 98) + (124 - 122) + (183 - 124) + (183 - 14) + (41 - 14)$
- $= 12 + 2 + 31 + 24 + 2 + 59 + 169 + 27$
- $= 326$





Selecting a Disk-Scheduling Algorithm

- ❑ SSTF is common and has a natural appeal
- ❑ SCAN and C-SCAN perform better for systems that place a heavy load on the disk.
- ❑ Performance depends on the number and types of requests.
- ❑ Requests for disk service can be influenced by the file-allocation method.
- ❑ The disk-scheduling algorithm should be written as a separate module of the operating system, allowing it to be replaced with a different algorithm if necessary.
- ❑ Either SSTF or LOOK is a reasonable choice for the default algorithm.





- Suppose the order of request is- (0-199)
 - (82,170,43,140,24,16,190)
- And current position of Read/Write head is : 50





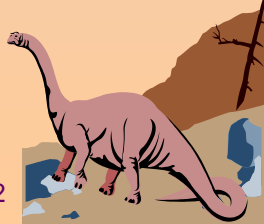
SO 3: Summarize the steps in disk management





Disk Management

- ❑ The operating system is responsible for several aspects of disk management.
- ❑ **Disk Formatting**
- ❑ A new magnetic disk is a blank slate. It is just platters of a magnetic recording material. Before a disk can store data, it must be divided into sectors that the disk controller can read and write. This process is called low-level formatting (or **physical formatting**).
- ❑ **Low-level formatting** fills the disk with a special data structure for each sector. The data structure for a sector consists of a header, a data area, and a trailer. The header and trailer contain information used by the disk controller, such as a sector number and an **error-correcting code (ECC)**.








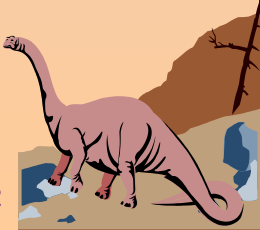
- ❑ To use a disk to hold files, the operating system still needs to record its own data structures on the disk. It does so in two steps. The first step is to **partition** the disk into one or more groups of cylinders.
- ❑ The operating system can treat each partition as though it were a separate disk. For instance, one partition can hold a copy of the operating system's executable code, while another holds user files.
- ❑ After partitioning, the second step is **logical formatting** (or creation of a file system). In this step, the operating system stores the initial file-system data structures onto the disk.





Boot Block

-  When a computer is powered up or rebooted, it needs to have an initial program to run. This initial program is called the bootstrap program. It initializes all aspects of the system (i.e. from CPU registers to device controllers and the contents of main memory) and then starts the operating system.
-  To do its job, the bootstrap program finds the operating system kernel on disk, loads that kernel into memory, and jumps to an initial address to begin the operating-system execution.
-  For most computers, the bootstrap is stored in read-only memory (**ROM**). This location is convenient because ROM needs no initialization and is at a fixed location that the processor can start executing when powered up or reset. And since ROM is read-only, it cannot be infected by a computer virus. The problem is that changing this bootstrap code requires changing the ROM hardware chips.





- For this reason, most systems store a tiny bootstrap loader program in the boot ROM, whose only job is to bring in a full bootstrap program from disk. The full bootstrap program can be changed easily:
- A new version is simply written onto the disk. The full bootstrap program is stored in a partition (at a fixed location on the disk) is called **the boot blocks**. A disk that has a boot partition is called **a boot disk or system disk**.





Bad Blocks

- Since disks have moving parts and small tolerances, they are prone to failure. Sometimes the failure is complete, and the disk needs to be replaced, and its contents restored from backup media to the new disk.
- More frequently, one or more sectors become defective. Most disks even come from the factory with bad blocks. Depending on the disk and controller in use, these blocks are handled in a variety of ways.
- The controller maintains a list of bad blocks on the disk. The list is initialized during the low-level format at the factory and is updated over the life of the disk. The controller can be told to replace each bad sector logically with one of the spare sectors. This scheme is known as sector sparing or forwarding.

