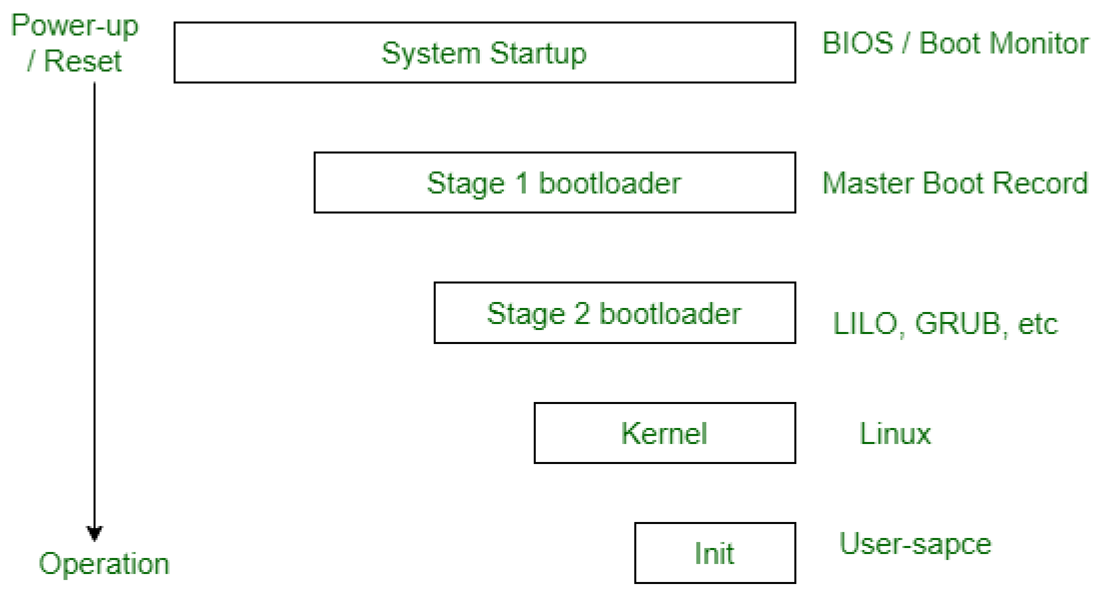
Operating System



Boot Process

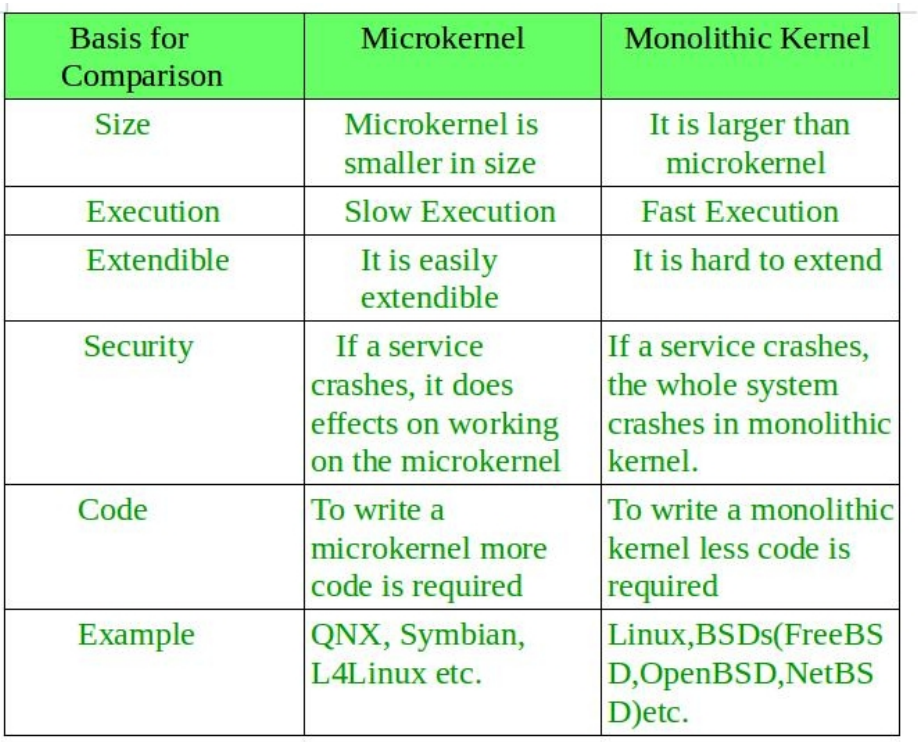
OS -

An operating system is a program on which application programs are executed and acts as an communication bridge (interface) between the user and the computer hardware.

The main task an operating system carries out is the allocation of resources and services, such as allocation of: memory, devices, processors and information..

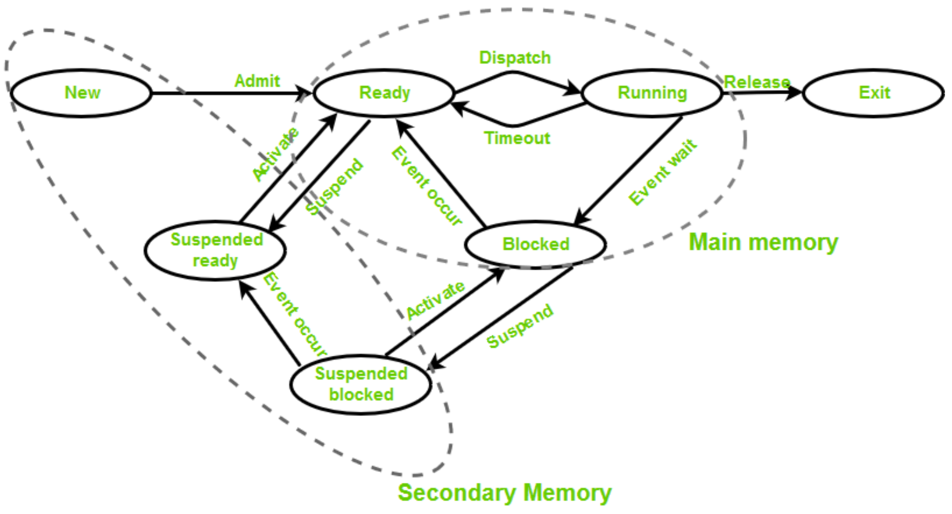
Modes -
User mode –
When the computer system is run by user applications like creating a text document or using any application program, then the system is in user mode. When the user application requests for a service from the operating system or an interrupt occurs or system call, then there will be a transition from user to kernel mode to fulfill the requests.

Kernel Mode –
When the system boots, hardware starts in kernel mode and when the operating system is loaded, it starts the user application in user mode. privileged instructions execute only in kernel mode.



Basis for Comparison	Microkernel	Monolithic Kernel
Size	Microkernel is smaller in size	It is larger than microkernel
Execution	Slow Execution	Fast Execution
Extendible	It is easily extendible	It is hard to extend
Security	If a service crashes, it does effects on working on the microkernel	If a service crashes, the whole system crashes in monolithic kernel.
Code	To write a microkernel more code is required	To write a monolithic kernel less code is required
Example	QNX, Symbian, L4Linux etc.	Linux,BSDs(FreeBSD,OpenBSD,NetBSD)etc.

A 32-bit system can access 232 memory addresses, i.e 4 GB of RAM or physical memory ideally, it can access more than 4 GB of RAM also.
A 64-bit system can access 264 memory addresses, i.e actually 18-Quintillion bytes of RAM. In short, any amount of memory greater than 4 GB can be easily handled by it.



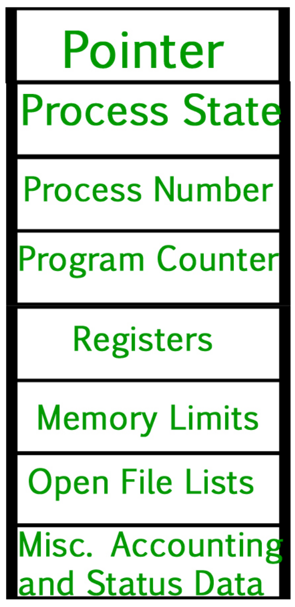
Process Management

Types of schedulers:

1. **Long term – performance –** Makes a decision about how many processes should be made to stay in the ready state, this decides the degree of multiprogramming. Once a decision is taken it lasts for a long time hence called long term scheduler.
2. **Short term – Context switching time –** Short term scheduler will decide which process to be executed next and then it will call the dispatcher. A dispatcher is a software that moves processes from ready to run and vice versa. In other words, it is context switching.
3. **Medium term – Swapping time –** Suspension decision is taken by medium term scheduler. Medium term scheduler is used for swapping that is moving the process from main memory to secondary and vice versa.

Pre-emption – Process is forcefully removed from CPU. Pre-emption is also called time sharing or multitasking.

Non pre-emption – Processes are not removed until they complete the execution.



Process Control Block

Dispatcher –

The dispatcher is the module that gives a process control over the CPU after it has been selected by the short-term scheduler. This function involves the following:

- Switching context
- Switching to user mode
- Jumping to the proper location in the user program to restart that program

Scheduling algos -

FCFS: Simplest scheduling algorithm that schedules according to arrival times of processes. First come first serve scheduling algorithm states that the process that requests the CPU first is allocated the CPU first.

Shortest Job First (SJF): Processes which have the shortest burst time are scheduled first.If two processes have the same bust time then FCFS is used to break the tie. It is a non-preemptive scheduling algorithm.

Shortest Remaining Time First (SRTF): It is a preemptive mode of the SJF algorithm in which jobs are scheduled according to the shortest remaining time.

Longest Remaining Time First (LRTF): It is a preemptive mode of LJF algorithm in which we give priority to the process having the largest burst time remaining.

Priority Based scheduling (Non-Preemptive): In this scheduling, processes are scheduled according to their priorities, i.e., highest priority process is scheduled first. If priorities of two processes match, then schedule according to arrival time.

Round Robin Scheduling: Each process is assigned a fixed time(Time Quantum/Time Slice) in a cyclic way.It is designed especially for the time-sharing system. The ready queue is treated as a circular queue. The CPU scheduler goes around the ready queue, allocating the CPU to each process for a time interval of up to 1-time quantum

Convoy Effect - One slow process slows down the performance of the entire set of processes, and leads to wastage of CPU time and other devices.

Threads

Thread is a single sequence stream within a process. Threads have the same properties as the process so they are called light weight processes. Threads are executed one after another but gives the illusion as if they are executing in parallel. Each thread has different states. Each thread has

A program counter
A register set
A stack space

User Level thread (ULT) –
Is implemented in the user level library, they are not created using the system calls. Thread switching does not need to call OS and to cause interrupt to Kernel. Kernel doesn’t know about the user level thread and manages them as if they were single-threaded processes.

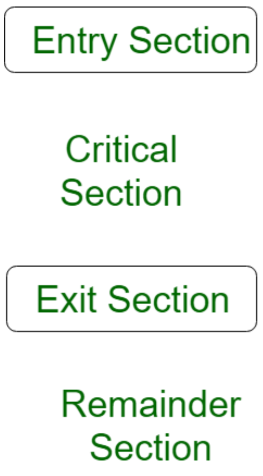
Kernel Level Thread (KLT) –
Kernel knows and manages the threads. Instead of a thread table in each process, the kernel itself has a thread table (a master one) that keeps track of all the threads in the system. In addition, the kernel also maintains the traditional process table to keep track of the processes. The OS kernel provides system calls to create and manage threads.

Fork -
Fork system call is used for creating a new process, which is called child process, which runs concurrently with the process that makes the fork() call (parent process). After a new child process is created, both processes will execute the next instruction following the fork() system call. A child process uses the same pc(program counter), same CPU registers, same open files which are used in the parent process.

It takes no parameters and returns an integer value. Below are different values returned by fork().

Negative Value: creation of a child process was unsuccessful.
Zero: Returned to the newly created child process.
Positive value: Returned to parent or caller. The value contains the process ID of the newly created child process.

Critical Section:
When more than one process accesses the same code segment that segment is known as the critical section. Critical section contains shared variables or resources which are needed to be synchronized to maintain consistency of data variable



A thread must acquire a lock prior to executing a critical section. The lock can be acquired by only one thread. There are various ways to implement locks.

Using Mutex:

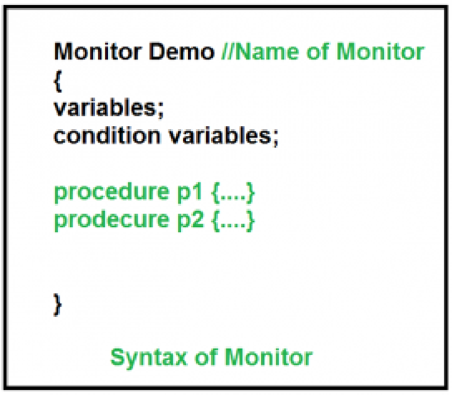
A mutex provides mutual exclusion, either producer or consumer can have the key (mutex) and proceed with their work. As long as the buffer is filled by the producer, the consumer needs to wait, and vice versa.

Using Semaphore:

A semaphore is a generalized mutex. In lieu of a single buffer, we can split the 4 KB buffer into four 1 KB buffers (identical resources). A semaphore can be associated with these four buffers. Semaphore is signaling mechanism ("I am done, you can carry on" kind of signal).

Monitor -

The monitor is one of the ways to achieve Process synchronization. The monitor is supported by programming languages to achieve mutual exclusion between processes.



```
Monitor Demo //Name of Monitor
{
variables;
condition variables;

procedure p1 {....}
prodecure p2 {....}


}
```

Syntax of Monitor

Critical Section Algos -

Dekker's algorithm was the first provably-correct solution to the critical section problem. It allows two threads to share a single-use resource without conflict, using only shared memory for communication. It avoids the strict alternation of a naïve turn-taking algorithm, and was one of the first mutual exclusion algorithms to be invented.

Peterson's algorithm - The idea is that first a thread expresses its desire to acquire a lock and sets flag[self] = 1 and then gives the other thread a chance to acquire the lock. If the thread desires to acquire the lock, then, it gets the lock and passes the chance to the 1st thread. If it does not desire to get the lock then the while loop breaks and the 1st thread gets the chance.

Bakery algorithm is one of the simplest known solutions to the mutual exclusion problem for the general case of N process. Bakery Algorithm is a critical section solution for N processes. The algorithm preserves the first come first serve property.

Before entering its critical section, the process receives a number. Holder of the smallest number enters the critical section.

```
If processes Pi and Pj receive the same number,
if i < j
        Pi is served first;
else
        Pj is served first.
```

Deadlock is a state in which each member of a group of actions is waiting for some other member to release a lock.

Livelock occurs when two or more processes continually repeat the same interaction in response to changes in the other processes without doing any useful work. These processes are not in the waiting state, and they are running concurrently. This is different from a deadlock because in a deadlock all processes are in the waiting state.

Starvation is a problem which is closely related to both Livelock and Deadlock. In a dynamic system, requests for resources keep on happening. Thereby, some policy is needed to make a decision about who gets the resource when. This process, being reasonable, may lead to some processes never getting serviced even though they are not deadlocked.

Deadlock has following characteristics.

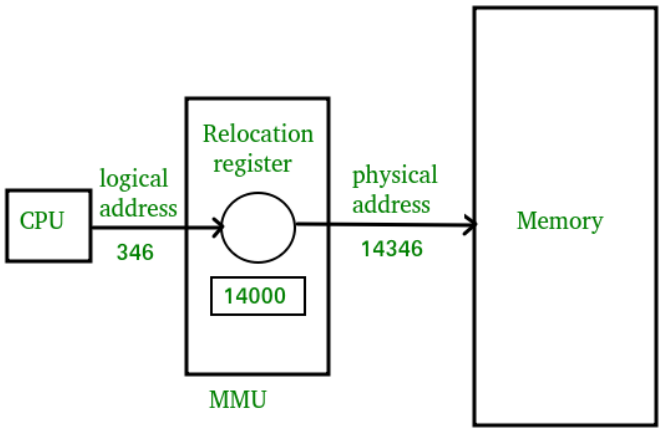
1. Mutual Exclusion
2. Hold and Wait
3. No preemption
4. Circular wait

Livelock occurs when two or more processes continually repeat the same interaction in response to changes in the other processes without doing any useful work. These processes are not in the waiting state, and they are

Memory Management

Logical Address is generated by the CPU while a program is running. The logical address is virtual address as it does not exist physically, therefore, it is also known as Virtual Address.

Physical Address identifies a physical location of required data in a memory. The user never directly deals with the physical address but can access by its corresponding logical address. the logical address must be mapped to the physical address by MMU before they are used.



Memory allocation techniques -
The main memory interacts with both the user processes and the operating system.So we need to efficiently use the main memory.Main memory is divided into non-overlapping memory regions called partitions.

1. First Fit
This method keeps the free/busy list of jobs organized by memory location, low-ordered to high-ordered memory. In this method, the first job claims the first available memory with space more than or equal to its size.

2. Next Fit
The next fit is a modified version of 'first fit'. It begins as the first fit to find a free partition but when called next time it starts searching from where it left off, not from the beginning.

3. Worst Fit
Worst Fit allocates a process to the partition which is largest sufficient among the freely available partitions available in the main memory.

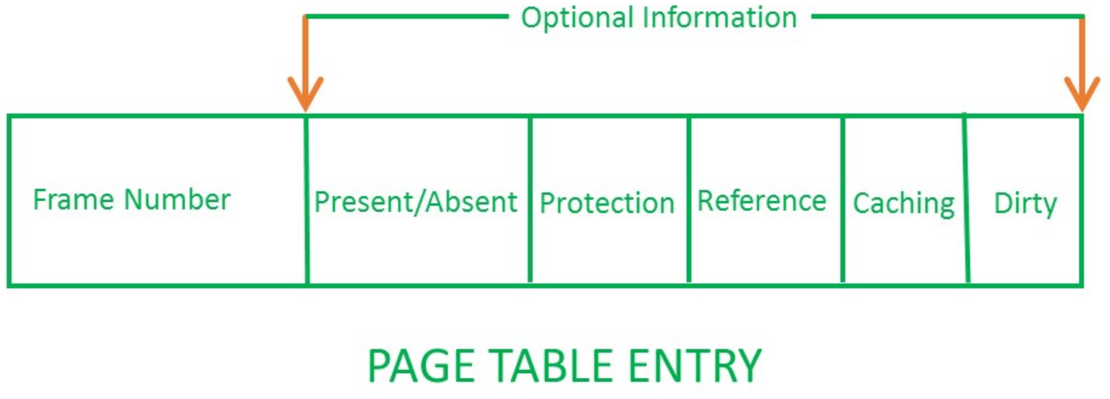
4. Best Fit
This method keeps the free/busy list in order by size – smallest to largest. In this method, the operating system first searches the whole of the memory according to the size of the given job and allocates it to the closest-fitting free partition in the memory, making it able to use memory efficiently.

Paging -

Paging is a memory management scheme that eliminates the need for contiguous allocation of physical memory. This scheme permits the physical address space of a process to be non – contiguous.

The mapping from virtual to physical address is done by the memory management unit (MMU) which is a hardware device and this mapping is known as paging technique.

Page table has page table entries where each page table entry stores a frame number and optional status (like protection) bits. Many of the status bits are used in the virtual memory system. The most important thing in PTE is frame Number.



Virtual Memory
Virtual Memory is a storage allocation scheme in which secondary memory can be addressed as though it were part of main memory. The addresses a program may use to reference memory are distinguished from the addresses the memory system uses to identify physical storage sites, and program generated addresses are translated automatically to the corresponding machine addresses.

Demand Paging :
The process of loading the page into memory on demand (whenever page fault occurs) is known as demand paging.

Page Fault Service Time :
The time taken to service the page fault is called as page fault service time. The page fault service time includes the time taken to perform all the above six steps.

Swapping:

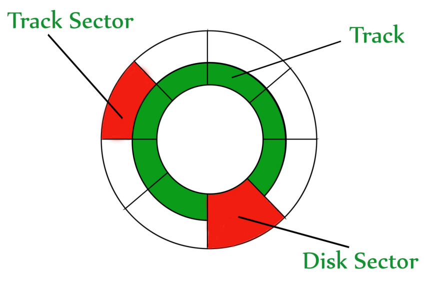
Swapping a process out means removing all of its pages from memory, or marking them so that they will be removed by the normal page replacement process. Suspending a process ensures that it is not runnable while it is swapped out. At some later time, the system swaps back the process from the secondary storage to main memory. When a process is busy swapping pages in and out then this situation is called thrashing.

Thrashing :

In the steady state practically, all of the main memory will be occupied with process's pages, so that the processor and OS has direct access to as many processes as possible. Thus when the OS brings one page in, it must throw another out. If it throws out a page just before it is used, then it will just have to get that page again almost immediately. Too much of this leads to a condition called Thrashing.

File/Disk Management

A **hard disk** is a memory storage device which looks like this.



The disk is divided into **tracks**. Each track is further divided into **sectors**.

Oed here is that outer tracks are bigger in size than the inner tracks but they contain the same number of sectors and have equal storage capacity.

This is because the storage density is high in sectors of the inner tracks whereas the bits are sparsely arranged in sectors of the outer tracks.

Some important terms must be noted here:

1. **Seek time –** The time taken by the R-W head to reach the desired track from its current position.
2. **Rotational latency –** Time taken by the sector to come under the R-W head.
3. **Data transfer time –** Time taken to transfer the required amount of data. It depends upon the rotational speed.
4. **Controller time –** The processing time taken by the controller.
5. **Average Access time –** seek time + Average Rotational latency + data transfer time + controller tim

A **file** is a collection of related information that is recorded on secondary storage. Or file is a collection of logically related entities.

FILE DIRECTORIES:

Collection of files is a file directory. The directory contains information about the files, including attributes, location and ownership. Much of this information, especially that is concerned with storage, is managed by the operating system.

FILE ALLOCATION METHODS

1. Contiguous Allocation: A single continuous set of blocks is allocated to a file at the time of file creation. Thus, this is a pre-allocation strategy, using variable size portions. The file allocation table needs just a single entry for each file, showing the starting block and the length of the file.

2. Linked Allocation(Non-contiguous allocation) : Allocation is on an individual block basis. Each block contains a pointer to the next block in the chain. Again the file table needs just a single entry for each file, showing the starting block and the length of the file.

3. Indexed Allocation:

It addresses many of the problems of contiguous and chained allocation. In this case, the file allocation table contains a separate one-level index for each file: The index has one entry for each block allocated to the file. Allocation may be on the basis of fixed-size blocks or variable-sized blocks.

File Access Methods -

When a file is used, information is read and accessed into computer memory and there are several ways to access this information of the file.

Sequential Access –

It is the simplest access method. Information in the file is processed in order, one record after the other. This mode of access is by far the most common; for example, editors and compilers usually access the file in this fashion.

Direct Access –

Another method is *direct access method* also known as *relative access method*. A field-length logical record that allows the program to read and write records rapidly. in no particular order. The direct access is based on the disk model of a file since disk allows random access to any file block.

Index sequential method –

It is the other method of accessing a file which is built on the top of the sequential access method. These methods construct an index for the file. The index, like an index in the back of a book, contains the pointer to the various blocks.

Disk Scheduling Algos -

Disk scheduling is done by operating systems to schedule I/O requests arriving for the disk. Disk scheduling is also known as I/O scheduling.

Disk Access Time = Seek Time + Rotational Latency + Transfer Time.

Seek Time: Seek time is the time taken to locate the disk arm to a specified track where the data is to be read or written. So the disk scheduling algorithm that gives minimum average seek time is better.

Rotational Latency: Rotational Latency is the time taken by the desired sector of disk to rotate into a position so that it can access the read/write heads.

Transfer Time: Transfer time is the time to transfer the data. It depends on the rotating speed of the disk and number of bytes to be transferred.

Disk Response Time: Response Time is the average of time spent by a request waiting to perform its I/O operation.

1. **FCFS:** FCFS is the simplest of all the Disk Scheduling Algorithms. In FCFS, the requests are addressed in the order they arrive in the disk queue.Let us understand this with the help of an example.

2. **SSTF:** In SSTF (Shortest Seek Time First), requests having the shortest seek time are executed first. So, the seek time of every request is calculated in advance in the queue and then they are scheduled according to their calculated seek time.

3. **SCAN:** In SCAN algorithm the disk arm moves into a particular direction and services the requests coming in its path and after reaching the end of disk, it reverses its direction and again services the request arriving in its path. So, this algorithm works as an elevator and hence is also known as **elevator algorithm.**

4. **C-SCAN:** the disk arm instead of reversing its direction goes to the other end of the disk and starts servicing the requests from there. So, the disk arm moves in a circular fashion and this algorithm is also similar to SCAN algorithm and hence it is known as C-SCAN .

5. **LOOK:** It is similar to the SCAN disk scheduling algorithm except for the difference that the disk arm in spite of going to the end of the disk goes only to the last request to be serviced in front of the head and then reverses its direction from there only.

6. **CLOOK**: the disk arm in spite of going to the end goes only to the last request to be serviced in front of the head and then from there goes to the other end’s last request.