UNIT I

1. What is an Operating system?

An operating system is a program that manages the computer hardware. It also provides a basis for application programs and act as an intermediary between a user of a computer and the computer hardware. It controls and coordinates the use of the hardware among the various application programs for the various users.

2. Why is the Operating System viewed as a resource allocator & control program?

A computer system has many resources – hardware & software that may be required to solve a problem, like CPU time, memory space, file-storage space, I/O devices & so on. The OS acts as a manager for these resources so it is viewed as a resource allocator.

The OS is viewed as a control program because it manages the execution of user programs to prevent errors & improper use of the computer.

3. What is the Kernel?

A more common definition is that the OS is the one program running at all times on the computer, usually called the kernel, with all else being application programs.

4. What are Batch systems?

Batch systems are quite appropriate for executing large jobs that need little interaction. The user can submit jobs and return later for the results. It is not necessary to wait while the job is processed. Operators batched together jobs with similar needs and ran them through the computer as a group.

5. What is the advantage of Multiprogramming?

Multiprogramming increases CPU utilization by organizing jobs so that the CPU always has one to execute. Several jobs are placed in the main memory and the processor is switched from job to job as needed to keep several jobs advancing while keeping the peripheral devices in use. Multiprogramming is the first instance where the Operating system must make decisions for the users. Therefore they are fairly sophisticated.

6. What is an Interactive computer system?

Interactive computer system provides direct communication between the user and the system. The user gives instructions to the operating system or to a program directly, using a keyboard or mouse, and waits for immediate results.

7. What do you mean by Time-sharing systems?

Time-sharing or multitasking is a logical extension of multiprogramming. It allows many users to share the computer simultaneously. The CPU executes multiple jobs

by switching among them, but the switches occur so frequently that the users can interact with each program while it is running.

8. What are multiprocessor systems & give their advantages?

Multiprocessor systems also known as parallel systems or tightly coupled systems are systems that have more than one processor in close communication, sharing the computer bus, the clock and sometimes memory & peripheral devices. Their main advantages are

- Increased throughput
- Economy of scale
- Increased reliability

9. What are the different types of multiprocessing?

Symmetric multiprocessing (SMP): In SMP each processor runs an identical copy of the Os & these copies communicate with one another as needed. All processors are peers. Examples are Windows NT, Solaris, Digital UNIX, OS/2 & Linux.

Asymmetric multiprocessing: Each processor is assigned a specific task. A master processor controls the system; the other processors look to the master for instructions or predefined tasks. It defines a master-slave relationship. Example SunOS Version 4.

10. What is graceful degradation?

In multiprocessor systems, failure of one processor will not halt the system, but only slow it down. If there are ten processors & if one fails the remaining nine processors pick up the work of the failed processor. This ability to continue providing service is proportional to the surviving hardware is called graceful degradation.

11. What is Dual- Mode Operation?

The dual mode operation provides us with the means for protecting the operating system from wrong users and wrong users from one another. User mode and monitor mode are the two modes. Monitor mode is also called supervisor mode, system mode or privileged mode. Mode bit is attached to the hardware of the computer to indicate the current mode. Mode bit is '0' for monitor mode and '1' for user mode.

12. What are privileged instructions?

Some of the machine instructions that may cause harm to a system are designated as privileged instructions. The hardware allows the privileged instructions to be executed only in monitor mode.

13. How can a user program disrupt the normal operations of a system?

A user program may disrupt the normal operation of a system by

- Issuing illegal I/O operations
- By accessing memory locations within the OS itself
- Refusing to relinquish the CPU

14. How is the protection for memory provided?

The protection against illegal memory access is done by using two registers. The base register and the limit register. The base register holds the smallest legal physical address; the limit register contains the size of the range. The base and limit registers can be loaded only by the OS using special privileged instructions.

15. What are the various OS components?

The various system components are

- Process management
- Main-memory management
- File management
- I/O-system management
- Secondary-storage management
- Networking
- Protection system
- Command-interpreter system

16. What is a process?

A process is a program in execution. It is the unit of work in a modern operating system. A process is an active entity with a program counter specifying the next instructions to execute and a set of associated resources.

It also includes the process stack, containing temporary data and a data section containing global variables.

17. What is a process state and mention the various states of a process?

As a process executes, it changes state. The state of a process is defined in part by the current activity of that process. Each process may be in one of the following states:

- New
- Running
- Waiting
- Ready
- Terminated

18. What is process control block (PCB)?

Each process is represented in the operating system by a process control block also called a task control block. It contains many pieces of information associated with a specific process. It simply acts as a repository for any information that may vary from process to process.

It contains the following information:

- Process state
- Program counter
- CPU registers
- CPU-scheduling information
- Memory-management information
- Accounting information

- I/O status information

19. What are the use of job queues, ready queues & device queues?

As a process enters a system, they are put into a job queue. This queue consists of all jobs in the system. The processes that are residing in main memory and are ready & waiting to execute are kept on a list called ready queue. The list of processes waiting for a particular I/O device is kept in the device queue.

20. What is meant by context switch?

Switching the CPU to another process requires saving the state of the old process and loading the saved state for the new process. This task is known as context switch. The context of a process is represented in the PCB of a process.

21. What is spooling?

Spooling-Simultaneous peripheral operations on line. It is a high-speed device like a disk is interposed between a running program and a low –speed device involved with the program in input/output. It disassociates a running program from the slow operation of devices like printers.

22. What are System calls?

System calls provide the interface between a process and the Operating system. System Calls are also called as Monitor call or Operating-system function call. When a system call is executed, it is treated as by the hardware as software interrupt. Control passes through the interrupt vector to a service routine in the operating system, and the mode bit is set to monitor mode.

23. List the services provided by an Operating System?

- Program execution
- I/O Operation
- File-System manipulation
- Communications
- Error detection

24. What are the two types of real time systems?

1.Hard real time system
2.Soft real time system

25. What is the difference between Hard real time system and Soft real time system?

A hard real time system guarantees that critical tasks complete on time. In a soft real time system, a critical real-time task gets priority over the other tasks, and retains that priority until it completes. Soft real time systems have more limited utility than do hard real-time systems.

26. Write the difference between multiprogramming and non-multiprogramming?

The operating system picks and begins to execute one of the jobs in the memory. Eventually, the job may have to wait for some task, such as a tape to be mounted, or an I/O operation to complete. In a non-multiprogrammed system, the CPU would sit idle. In a multiprogramming system, the operating system simply switches to and executes another job. When that job needs to wait, the CPU is switched to another job, and so on. Eventually, the first job finishes waiting and gets the CPU back. As long as there is always some job to execute, the CPU will never be idle.

27. What are the design goals of an operating system?

The requirements can be divided into two basic groups: User goals and System goals. Users desire that the system should be convenient and easy to use, easy to learn, reliable, safe and fast. The Operating system should be easy to design, implement, and maintain. Also it should be flexible, reliable, error free and efficient. These are some of the requirements, which are vague and have no general solution.

28.What are the five major categories of System Calls?

- Process Control
- File-management
- Device-management
- Information maintenance
- Communications

29. What is the use of fork and execve system calls?

Fork is a System calls by which a new process is created. Execve is also a System call, which is used after a fork by one of the two processes to replace the process memory space with a new program.

30. Define Elapsed CPU time and Maximum CPU time?

Elapsed CPU Time: Total CPU time used by a process to date.
Maximum CPU Time: Maximum amount of CPU time a process may use.

UNIT II

1. What is a thread?

A thread otherwise called a lightweight process (LWP) is a basic unit of CPU utilization, it comprises of a thread id, a program counter, a register set and a stack. It shares with other threads belonging to the same process its code section, data section, and operating system resources such as open files and signals.

2. What are the benefits of multithreaded programming?

The benefits of multithreaded programming can be broken down into four major categories:

- Responsiveness
- Resource sharing
- Economy
- Utilization of multiprocessor architectures

3.Compare user threads and kernel threads.

User threads	Kernel threads
User threads are supported above the kernel and are implemented by a thread library at the user level	Kernel threads are supported directly by the operating system
Thread creation & scheduling are done in the user space, without kernel intervention. Therefore they are fast to create and manage	Thread creation, scheduling and management are done by the operating system. Therefore they are slower to create & manage compared to user threads
Blocking system call will cause the entire process to block	If the thread performs a blocking system call, the kernel can schedule another thread in the application for execution

4. Define thread cancellation & target thread.

The thread cancellation is the task of terminating a thread before it has completed. A thread that is to be cancelled is often referred to as the target thread. For example, if multiple threads are concurrently searching through a database and one thread returns the result, the remaining threads might be cancelled.

5. What are the different ways in which a thread can be cancelled?

Cancellation of a target thread may occur in two different scenarios:

- *Asynchronous cancellation:* One thread immediately terminates the target thread is called asynchronous cancellation.

- *Deferred cancellation:* The target thread can periodically check if it should terminate, allowing the target thread an opportunity to terminate itself in an orderly fashion.

6. Define CPU scheduling.

CPU scheduling is the process of switching the CPU among various processes. CPU scheduling is the basis of multiprogrammed operating systems. By switching the CPU among processes, the operating system can make the computer more productive.

7. What is pre-emptive and non-preemptive scheduling?

Under nonpreemptive scheduling once the CPU has been allocated to a process, the process keeps the CPU until it releases the CPU either by terminating or switching to the waiting state.

Preemptive scheduling can preempt a process which is utilizing the CPU in between its execution and give the CPU to another process.

8. What is a Dispatcher?

The dispatcher is the module that gives control of the CPU to the process selected by the short-term scheduler. This function involves:

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

9. What is dispatch latency?

The time taken by the dispatcher to stop one process and start another running is known as dispatch latency.

10.What are the various scheduling criteria for CPU scheduling?

The various scheduling criteria are

- CPU utilization
- Throughput
- Turnaround time
- Waiting time
- Response time

11. Define throughput?

Throughput in CPU scheduling is the number of processes that are completed per unit time. For long processes, this rate may be one process per hour; for short transactions, throughput might be 10 processes per second.

12. What is turnaround time?

Turnaround time is the interval from the time of submission to the time of completion of a process. It is the sum of the periods spent waiting to get into memory, waiting in the ready queue, executing on the CPU, and doing I/O.

13. Define race condition.

When several process access and manipulate same data concurrently, then the outcome of the execution depends on particular order in which the access takes place is called race condition. To avoid race condition, only one process at a time can manipulate the shared variable.

14. What is critical section problem?

Consider a system consists of 'n' processes. Each process has segment of code called a critical section, in which the process may be changing common variables, updating a table, writing a file. When one process is executing in its critical section, no other process can allowed to execute in its critical section.

15. What are the requirements that a solution to the critical section problem must satisfy?

The three requirements are

- Mutual exclusion
- Progress
- Bounded waiting

16. Define entry section and exit section.

The critical section problem is to design a protocol that the processes can use to cooperate. Each process must request permission to enter its critical section. The section of the code implementing this request is the entry section. The critical section is followed by an exit section. The remaining code is the remainder section.

17. Give two hardware instructions and their definitions which can be used for implementing mutual exclusion.

- TestAndSet

```
boolean TestAndSet (boolean &target)
{
        boolean rv = target;
        target = true;
        return rv;
}
```

- Swap

```
void Swap (boolean &a, boolean &b)
{
        boolean temp = a;
```

```
        a = b;
        b = temp;
}
```

18. What is semaphores?

A semaphore 'S' is a synchronization tool which is an integer value that, apart from initialization, is accessed only through two standard atomic operations; wait and signal. Semaphores can be used to deal with the n-process critical section problem. It can be also used to solve various synchronization problems.

The classic definition of 'wait'

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

The classic definition of 'signal'

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

19. Define busy waiting and spinlock.

When a process is in its critical section, any other process that tries to enter its critical section must loop continuously in the entry code. This is called as busy waiting and this type of semaphore is also called a spinlock, because the process while waiting for the lock.

20. How can we say the First Come First Served scheduling algorithm is nonpreemptive?

Once the CPU has been allocated to a process, that process keeps the CPU until it releases the CPU, either by terminating or by requesting I/O. So we can say the First Come First Served scheduling algorithm is non preemptive.

21.What is waiting time in CPU scheduling?

Waiting time is the sum of periods spent waiting in the ready queue. CPU scheduling algorithm affects only the amount of time that a process spends waiting in the ready queue.

22. What is Response time in CPU scheduling?

Response time is the measure of the time from the submission of a request until the first response is produced. Response time is amount of time it takes to start responding, but not the time that it takes to output that response.

23. Differentiate long term scheduler and short term scheduler

- The long-term scheduler or job scheduler selects processes from the job pool and loads them into memory for execution.
- The short-term scheduler or CPU scheduler selects from among the process that are ready to execute, and allocates the CPU to one of them.

24. Write some classical problems of synchronization?

The Bounded-Buffer Problem
The Readers-Writers Problem
The Dining Philosophers Problem

25. When the error will occur when we use the semaphore?

i. When the process interchanges the order in which the wait and signal operations on the semaphore mutex.
ii. When a process replaces a signal (mutex) with wait (mutex).
iii. When a process omits the wait (mutex), or the signal (mutex), or both.

26. What is Mutual Exclusion?

A way of making sure that if one process is using a shared modifiable data, the other processes will be excluded from doing the same thing. Each process executing the shared data variables excludes all others from doing so simultaneously. This is called mutual exclusion.

27. Define the term critical regions?

Critical regions are small and infrequent so that system through put is largely unaffected by their existence. Critical region is a control structure for implementing mutual exclusion over a shared variable.

28. What are the drawbacks of monitors?

1. Monitor concept is its lack of implementation most commonly used programming languages.
2. There is the possibility of deadlocks in the case of nested monitors calls.

29. What are the two levels in threads?

Thread is implemented in two ways.

1.User level
2.Kernel level

30. What is a Gantt chart?

A two dimensional chart that plots the activity of a unit on the Y-axis versus the time on the X-axis. The chart quickly represents how the activities of the units are serialized.

31. Define deadlock.

A process requests resources; if the resources are not available at that time, the process enters a wait state. Waiting processes may never again change state, because the resources they have requested are held by other waiting processes. This situation is called a deadlock.

32. What is the sequence in which resources may be utilized?

Under normal mode of operation, a process may utilize a resource in the following sequence:

- Request: If the request cannot be granted immediately, then the requesting process must wait until it can acquire the resource.
- Use: The process can operate on the resource.
- Release: The process releases the resource.

33. What are conditions under which a deadlock situation may arise?

A deadlock situation can arise if the following four conditions hold simultaneously in a system:

1. Mutual exclusion
2. Hold and wait
3. No pre-emption
4. Circular wait

34. What is a resource-allocation graph?

Deadlocks can be described more precisely in terms of a directed graph called a system resource allocation graph. This graph consists of a set of vertices V and a set of edges E. The set of vertices V is partitioned into two different types of nodes; P the set consisting of all active processes in the system and R the set consisting of all resource types in the system.

35. Define request edge and assignment edge.

A directed edge from process Pi to resource type Rj is denoted by Pi→Rj; it signifies that process Pi requested an instance of resource type Rj and is currently waiting for that resource. A directed edge from resource type Rj to process Pi is denoted by Rj→Pi, it signifies that an instance of resource type has been allocated to a process Pi. A directed edge Pi→Rj is called a request edge. A directed edge Rj→Pi is called an assignment edge.

36. What are the methods for handling deadlocks?

The deadlock problem can be dealt with in one of the three ways:

- Use a protocol to prevent or avoid deadlocks, ensuring that the system will never enter a deadlock state.
- Allow the system to enter the deadlock state, detect it and then recover.
- Ignore the problem all together, and pretend that deadlocks never occur in the system.

37. Define deadlock prevention.

Deadlock prevention is a set of methods for ensuring that at least one of the four necessary conditions like mutual exclusion, hold and wait, no pre-emption and circular wait cannot hold. By ensuring that that at least one of these conditions cannot hold, the occurrence of a deadlock can be prevented.

38. Define deadlock avoidance.

An alternative method for avoiding deadlocks is to require additional information about how resources are to be requested. Each request requires the system consider the resources currently available, the resources currently allocated to each process, and the future requests and releases of each process, to decide whether the could be satisfied or must wait to avoid a possible future deadlock.

39. What are a safe state and an unsafe state?

A state is safe if the system can allocate resources to each process in some order and still avoid a deadlock. A system is in safe state only if there exists a safe sequence. A sequence of processes <P1,P2,….Pn> is a safe sequence for the current allocation state if, for each Pi, the resource that Pi can still request can be satisfied by the current available resource plus the resource held by all the Pj, with j<i. if no such sequence exists, then the system state is said to be unsafe.

40. What is banker's algorithm?

Banker's algorithm is a deadlock avoidance algorithm that is applicable to a resource-allocation system with multiple instances of each resource type. The two algorithms used for its implementation are:

Safety algorithm*:* The algorithm for finding out whether or not a system is in a safe state.

Resource-request algorithm*:* if the resulting resource-allocation is safe, the transaction is completed and process Pi is allocated its resources. If the new state is unsafe Pi must wait and the old resource-allocation state is restored.

41. Define logical address and physical address.

An address generated by the CPU is referred as logical address. An address seen by the memory unit that is the one loaded into the memory address register of the memory is commonly referred to as physical address.

42. What is logical address space and physical address space?

The set of all logical addresses generated by a program is called a logical address space; the set of all physical addresses corresponding to these logical addresses is a physical address space.

43. What is the main function of the memory-management unit?

The runtime mapping from virtual to physical addresses is done by a hardware device called a memory management unit (MMU).

44.

44.What are the methods for dealing the deadlock problem?

i. Use a protocol to ensure that the system will never enter a deadlock state.
ii. Allow the system to enter the deadlock state and then recover.
iii. Ignore the problem all together, and pretend that deadlocks never occur in the system.

45. Differentiate deadlock and starvation.

A set of processes is in deadlock state when every process in the set is waiting for an event that can be caused only by the other process in the set.

Starvation or indefinite blocking is a situation where processes wait indefinitely within the semaphore.

UNIT III

1. Define dynamic loading.

To obtain better memory-space utilization dynamic loading is used. With dynamic loading, a routine is not loaded until it is called. All routines are kept on disk in a relocatable load format. The main program is loaded into memory and executed. If the routine needs another routine, the calling routine checks whether the routine has been loaded. If not, the relocatable linking loader is called to load the desired program into memory.

2. Define dynamic linking.

Dynamic linking is similar to dynamic loading, rather that loading being postponed until execution time, linking is postponed. This feature is usually used with system libraries, such as language subroutine libraries. A stub is included in the image for each library-routine reference. The stub is a small piece of code that indicates how to locate the appropriate memory-resident library routine, or how to load the library if the routine is not already present.

3. What are overlays?

To enable a process to be larger than the amount of memory allocated to it, overlays are used. The idea of overlays is to keep in memory only those instructions and data that are needed at a given time. When other instructions are needed, they are loaded into space occupied previously by instructions that are no longer needed.

4. Define swapping.

A process needs to be in memory to be executed. However a process can be swapped temporarily out of memory to a backing store and then brought back into memory for continued execution. This process is called swapping.

48. What are the common strategies to select a free hole from a set of available holes?

The most common strategies are

5. First fit
6. Best fit
7. Worst fit

5. What do you mean by best fit?

Best fit allocates the smallest hole that is big enough. The entire list has to be searched, unless it is sorted by size. This strategy produces the smallest leftover hole.

6. What do you mean by first fit?

First fit allocates the first hole that is big enough. Searching can either start at the beginning of the set of holes or where the previous first-fit search ended. Searching can be stopped as soon as a free hole that is big enough is found.

7. How is memory protected in a paged environment?

Protection bits that are associated with each frame accomplish memory protection in a paged environment. The protection bits can be checked to verify that no writes are being made to a read-only page.

8. What is external fragmentation?

External fragmentation exists when enough total memory space exists to satisfy a request, but it is not contiguous; storage is fragmented into a large number of small holes.

9. What is internal fragmentation?

When the allocated memory may be slightly larger than the requested memory, the difference between these two numbers is internal fragmentation.

10. What do you mean by compaction?

Compaction is a solution to external fragmentation. The memory contents are shuffled to place all free memory together in one large block. It is possible only if relocation is dynamic, and is done at execution time.

11. What are pages and frames?

Paging is a memory management scheme that permits the physical-address space of a process to be non contiguous. In the case of paging, physical memory is broken into fixed-sized blocks called frames and logical memory is broken into blocks of the same size called pages.

12. What is the use of valid-invalid bits in paging?

When the bit is set to valid, this value indicates that the associated page is in the process's logical address space, and is thus a legal page. If the bit is said to invalid, this value indicates that the page is not in the process's logical address space. Using the valid-invalid bit traps illegal addresses.

13. What is the basic method of segmentation?

Segmentation is a memory management scheme that supports the user view of memory. A logical address space is a collection of segments. The logical address consists of segment number and offset. If the offset is legal, it is added to the segment base to produce the address in physical memory of the desired byte.

14. A Program containing relocatable code was created, assuming it would be loaded at address 0. In its code, the program refers to the following addresses: 50, 78, 150,

152, 154. If the program is loaded into memory starting at location 250, how do those addresses have to be adjusted?

All addresses need to be adjusted upward by 250.So the adjusted addresses would be 300, 328, 400, 402, and 404.

15. What is virtual memory?

Virtual memory is a technique that allows the execution of processes that may not be completely in memory. It is the separation of user logical memory from physical memory. This separation provides an extremely large virtual memory, when only a smaller physical memory is available.

16. What is Demand paging?

Virtual memory is commonly implemented by demand paging. In demand paging, the pager brings only those necessary pages into memory instead of swapping in a whole process. Thus it avoids reading into memory pages that will not be used anyway, decreasing the swap time and the amount of physical memory needed.

17. Define lazy swapper.

Rather than swapping the entire process into main memory, a lazy swapper is used. A lazy swapper never swaps a page into memory unless that page will be needed.

18. What is a pure demand paging?

When starting execution of a process with no pages in memory, the operating system sets the instruction pointer to the first instruction of the process, which is on a non-memory resident page, the process immediately faults for the page. After this page is brought into memory, the process continues to execute, faulting as necessary until every page that it needs is in memory. At that point, it can execute with no more faults. This schema is pure demand paging.

19. Define effective access time.

Let p be the probability of a page fault ($0 \leq p \leq 1$). The value of p is expected to be close to 0; that is, there will be only a few page faults. The effective access time is

Effective access time = (1-p) * ma + p * page fault time.

ma : memory-access time

20. Define secondary memory.

This memory holds those pages that are not present in main memory. The secondary memory is usually a high speed disk. It is known as the swap device, and the section of the disk used for this purpose is known as swap space.

21. What is the basic approach of page replacement?

If no frame is free is available, find one that is not currently being used and free it. A frame can be freed by writing its contents to swap space, and changing the page table to indicate that the page is no longer in memory. Now the freed frame can be used to hold the page for which the process faulted.

22. What are the various page replacement algorithms used for page replacement?

- FIFO page replacement
- Optimal page replacement
- LRU page replacement
- LRU approximation page replacement
- Counting based page replacement
- Page buffering algorithm.

23. What are the major problems to implement demand paging?

The two major problems to implement demand paging is developing

1. Frame allocation algorithm
2. Page replacement algorithm

24. What is a reference string?

An algorithm is evaluated by running it on a particular string of memory references and computing the number of page faults. The string of memory reference is called a reference string.

UNIT IV

1. What is a file?

A file is a named collection of related information that is recorded on secondary storage. A file contains either programs or data. A file has certain "structure" based on its type.

- File attributes: Name, identifier, type, size, location, protection, time, date
- File operations: creation, reading, writing, repositioning, deleting, truncating, appending, renaming
- File types: executable, object, library, source code etc.

2. List the various file attributes.

A file has certain other attributes, which vary from one operating system to another, but typically consist of these: Name, identifier, type, location, size, protection, time, date and user identification

3.What are the various file operations?

The six basic file operations are

- Creating a file
- Writing a file
- Reading a file
- Repositioning within a file
- Deleting a file
- Truncating a file

4. What are the information associated with an open file?

Several pieces of information are associated with an open file which may be:

- File pointer
- File open count
- Disk location of the file
- Access rights

5. What are the different accessing methods of a file?

The different types of accessing a file are:

- Sequential access: Information in the file is accessed sequentially
- Direct access: Information in the file can be accessed without any particular order.
- Other access methods: Creating index for the file, indexed sequential access method (ISAM) etc.

6. What is Directory?

The device directory or simply known as directory records information- such as name, location, size, and type for all files on that particular partition. The directory can be viewed as a symbol table that translates file names into their directory entries.

7. What are the operations that can be performed on a directory?

The operations that can be performed on a directory are

- Search for a file
- Create a file
- Delete a file
- Rename a file
- List directory
- Traverse the file system

8. What are the most common schemes for defining the logical structure of a directory?

The most common schemes for defining the logical structure of a directory

- Single-Level Directory
- Two-level Directory
- Tree-Structured Directories
- Acyclic-Graph Directories
- General Graph Directory

9. Define UFD and MFD.

In the two-level directory structure, each user has her own user file directory (UFD). Each UFD has a similar structure, but lists only the files of a single user. When a job starts the system's master file directory (MFD) is searched. The MFD is indexed by the user name or account number, and each entry points to the UFD for that user.

10.What is a path name?

A pathname is the path from the root through all subdirectories to a specified file. In a two-level directory structure a user name and a file name define a path name.

11. What is access control list (ACL)?

The most general scheme to implement identity-dependent access is to associate with each file and directory an access control unit.

12. Define Equal allocation.

The way to split '*m*' frames among '*n*' processes is to give everyone an equal share, *m/n* frames. For instance, if there are 93 frames and 5 processes, each process will get 18 frames. The leftover 3 frames could be used as a free-frame buffer pool. This scheme is called equal allocation.

13. What is the cause of thrashing? How does the system detect thrashing? Once it detects thrashing, what can the system do to eliminate this problem?

Thrashing is caused by under allocation of the minimum number of pages required by a process, forcing it to continuously page fault. The system can detect thrashing by evaluating the level of CPU utilization as compared to the level of multiprogramming. It can be eliminated by reducing the level of multiprogramming.

14. If the average page fault service time of 25 ms and a memory access time of 100ns.Calculate the effective access time.

Effective access time = (1-p)*ma + p*page fault time
= (1-p)*100+p*25000000
= 100-100p+25000000*p
= 100 + 24999900p

15. What is Belady's Anomaly?

For some page replacement algorithms, the page fault rate may increase as the number of allocated frames increases.

16. What are the different types of access?

Different types of operations may be controlled in access type. These are

1.Read
2.Write
3.Execute.
4.Append
5.Delete
6.List

17. What are the types of path names?

Path names can be of two types.

a) Absolute path name.
b) Relative path name.

Absolute path name: Begins at the root and follows a path down to the specified file, giving the directory names on the path.

Relative path name: Defines a path from the current directory.

18. What is meant by Locality of Reference?

The locality model states that, as a process executes, it moves from locality to locality. Locality are of two types.

1) Spatial locality

2) Temporal locality.

19. What are the various layers of a file system?

The file system is composed of many different levels. Each level in the design uses the feature of the lower levels to create new features for use by higher levels.

- Application programs
- Logical file system
- File-organization module
- Basic file system
- I/O control
- Devices

20. What are the structures used in file-system implementation?

Several on-disk and in-memory structures are used to implement a file system

i. On-disk structure include

1. Boot control block
2. Partition block
3. Directory structure used to organize the files
4. File control block (FCB)

ii .In-memory structure include

1. In-memory partition table
2. In-memory directory structure
3. System-wide open file table
4. Per-process open table

21. What are the functions of virtual file system (VFS)?

It has two functions

i. It separates file-system-generic operations from their implementation defining a clean VFS interface. It allows transparent access to different types of file systems mounted locally.

ii.VFS is based on a file representation structure, called a vnode. It contains a numerical value for a network-wide unique file .The kernel maintains one vnode structure for each active file or directory.

22. Define seek time and latency time.

The time taken by the head to move to the appropriate cylinder or track is called seek time. Once the head is at right track, it must wait until the desired block rotates under the read- write head. This delay is latency time.

23. What are the allocation methods of a disk space?

Three major methods of allocating disk space which are widely in use are

- Contiguous allocation
- Linked allocation
- Indexed allocation

24. What are the advantages of Contiguous allocation?

The advantages are

- Supports direct access
- Supports sequential access
- Number of disk seeks is minimal.

25. What are the drawbacks of contiguous allocation of disk space?

The disadvantages are

- Suffers from external fragmentation
- Suffers from internal fragmentation
- Difficulty in finding space for a new file
- File cannot be extended
- Size of the file is to be declared in advance

26. What are the advantages of Linked allocation?

The advantages are

- No external fragmentation
- Size of the file does not need to be declared

27. What are the disadvantages of linked allocation?

The disadvantages are

- Used only for sequential access of files.
- Direct access is not supported
- Memory space required for the pointers.
- Reliability is compromised if the pointers are lost or damaged

28. What are the advantages of Indexed allocation?

The advantages are

- No external-fragmentation problem
- Solves the size-declaration problems.
- Supports direct access

29. How can the index blocks be implemented in the indexed allocation scheme?

The index block can be implemented as follows

- Linked scheme
- Multilevel scheme

- Combined scheme

30. Define rotational latency and disk bandwidth.

Rotational latency is the additional time waiting for the disk to rotate the desired sector to the disk head. The disk bandwidth is the total number of bytes transferred, divided by the time between the first request for service and the completion of the last transfer.

31. How free-space is managed using bit vector implementation?

The free-space list is implemented as a bit map or bit vector. Each block is represented by 1 bit. If the block is free, the bit is 1; if the block is allocated, the bit is 0.

32. Define buffering.

A buffer is a memory area that stores data while they are transferred between two devices or between a device and an application. Buffering is done for three reasons

- To cope with a speed mismatch between the producer and consumer of a data stream
- To adapt between devices that have different data-transfer sizes
- To support copy semantics for application I/O

UNIT V

1. Define caching.

A cache is a region of fast memory that holds copies of data. Access to the cached copy is more efficient than access to the original. Caching and buffering are distinct functions, but sometimes a region of memory can be used for both purposes.

2. Define spooling.

A spool is a buffer that holds output for a device, such as printer, that cannot accept interleaved data streams. When an application finishes printing, the spooling system queues the corresponding spool file for output to the printer. The spooling system copies the queued spool files to the printer one at a time.

3. What are the various disk-scheduling algorithms?

The various disk-scheduling algorithms are

- First Come First Served Scheduling
- Shortest Seek Time First Scheduling
- SCAN Scheduling
- C-SCAN Scheduling
- LOOK scheduling

4. What is low-level formatting?

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.

5. What is the use of boot block?

For a computer to start running when powered up or rebooted it needs to have an initial program to run. This bootstrap program tends to be simple. It finds the operating system on the disk loads that kernel into memory and jumps to an initial address to begin the operating system execution. The full bootstrap program is stored in a partition called the boot blocks, at fixed location on the disk. A disk that has boot partition is called boot disk or system disk.

6. What is sector sparing?

Low-level formatting also sets aside spare sectors not visible to the operating system. 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.

7. What are the techniques used for performing I/O.

1. Programmed I/O.

2. Interrupt driven I/O.
3. Direct Memory Access (DMA).

8. Give an example of an application in which data in a file should be accessed in the following order:

a. Sequentially

b. Randomly

a. Print the content of the file.

b. Print the content of record *i*. This record can be found using hashing or index techniques.

9. What problems could occur if a system allowed a file system to be mounted simultaneously at more than one location?

There would be multiple paths to the same file, which could confuse users or encourage mistakes. (Deleting a file with one path deletes the file in all the other paths.)

10. Why must the bit map for file allocation be kept on mass storage rather than in main memory?

In case of system crash (memory failure), the free-space list would not be lost as it would be if the bit map had been stored in main memory.

11. What criteria should be used in deciding which strategy is best utilized for a particular file?

Contiguous—if file is usually accessed sequentially, if file is relatively small.

Linked—if file is large and usually accessed sequentially.

Indexed—if file is large and usually accessed randomly.

12. What is meant by RAID?

"RAID" is now used as an umbrella term for computer data storage schemes that can divide and replicate data among multiple hard disk drives. The different schemes/architectures are named by the word RAID followed by a number, as in RAID 0, RAID 1, etc. RAID's various designs involve two key design goals: increase data reliability and/or increase input/output performance. When multiple physical disks are set up to use RAID technology, they are said to be *in* a *RAID* array.

13. What is meant by Stable storage?

Stable storage is a classification of computer data storage technology that guarantees atomicity for any given write operation and allows software to be written that is robust against some hardware and power failures. To be considered atomic, upon reading back a just written-to portion of the disk, the storage subsystem must return either the write data or the data that was on that portion of the disk before the write operation.

14. What is meant by Tertiary storage?

Tertiary storage or **tertiary memory**,[3] provides a third level of storage. Typically it involves a robotic mechanism which will *mount* (insert) and *dismount* removable mass storage media into a storage device according to the system's demands; this data is often copied to secondary storage before use.

15. Write a note on descriptor?

UNIX processes use *descriptors* to reference I/O streams. Descriptors are small unsigned integers obtained from the *open* and *socket* system calls.. A *read* or *write* system call can be applied to a descriptor to transfer data. The *close* system call can be used to deallocate any descriptor. Descriptors represent underlying objects supported by the kernel, and are created by system calls specific to the type of object. In 4.4BSD, three kinds of objects can be represented by descriptors: files, pipes, and sockets.

16. Write short notes on pipes?

A *pipe* is a linear array of bytes, as is a file, but it is used solely as an I/O stream, and it is unidirectional. It also has no name, and thus cannot be opened with *open*. Instead, it is created by the *pipe* system call, which returns two descriptors, one of which accepts input that is sent to the other descriptor reliably, without duplication, and in order. The system also supports a named pipe or FIFO. A FIFO has properties identical to a pipe, except that it appears in the filesystem; thus, it can be opened using the *open* system call. Two processes that wish to communicate each open the FIFO: One opens it for reading, the other for writing.

****

UNIT-1

1. What are the system components of an operating system and explain them?

Common System Components

- Process Management
- Main Memory Management
- File Management
- I/O System Management
- Secondary Management
- Networking
- Protection System
- Command-Interpreter System

2. Define system calls. Write about the various system calls.

Introduction

Types of System Calls

- Process control
- File management
- Device management
- Information maintenance
- Communications

3. What is a process? Explain the process control block and the various process states.

Introduction

- An operating system executes a variety of programs:
  - Batch system – jobs
  - Time-shared systems – user programs or tasks
- Textbook uses the terms *job* and *process* almost interchangeably.
- Process – a program in execution; process execution must progress in sequential fashion.
- A process includes:
  - program counter
  - stack
  - data section

Process State

- new: The process is being created.
- running: Instructions are being executed.
- waiting: The process is waiting for some event to occur.
- ready: The process is waiting to be assigned to a process.
- terminated: The process has finished execution.

4. Explain process creation and process termination

Process Creation

- Parent process create children processes, which, in turn create other processes, forming a tree of processes.
- Resource sharing
  - Parent and children share all resources.
  - Children share subset of parent's resources.
  - Parent and child share no resources.
- Execution
  - Parent and children execute concurrently.
  - Parent waits until children terminate.
- Address space
  - Child duplicate of parent.
  - Child has a program loaded into it.
- UNIX examples
  - fork system call creates new process
  - exec system call used after a fork to replace the process' memory space with a new program.

Process Termination

- Process executes last statement and asks the operating system to decide it (exit).
  - Output data from child to parent (via wait).
  - Process' resources are deallocated by operating system.
- Parent may terminate execution of children processes (abort).
  - Child has exceeded allocated resources.
  - Task assigned to child is no longer required.
  - Parent is exiting.
    - Operating system does not allow child to continue if its parent terminates.
    - Cascading termination.

5. Explain about interprocess communication.

- Definition
- Message Passing System
- Naming
  - Direct Communication
  - Indirect Communication
- Synchronization
- Buffering

UNIT-II

1. Write about the various CPU scheduling algorithms.

- Optimization Criteria
- First-Come, First-Served (FCFS) Scheduling
- Shortest-Job-First (SJF) Scheduling
- Priority Scheduling
- Round Robin (RR)
- Multilevel Queue
- Multilevel Feedback Queue

2. Explain the classical problem on synchronization.

Classical Problems

- Bounded-Buffer Problem
- Readers and Writers Problem
- Dining-Philosophers Problem

3.Explain about monitors.

Introduction

- **High-level synchronization construct that allows the safe sharing of an abstract data type among concurrent processes.**

```
monitor monitor-name
{
        shared variable declarations
        procedure body P1 (…) {
                . . .
        }
        procedure body P2 (…) {
                . . .
        }
        procedure body Pn (…) {
                . . .
        }
        {
                initialization code
        }
}
```

4. Monitor Implementation Using Semaphores

- **Variables**

```
semaphore mutex;  // (initially  = 1)
semaphore next;     // (initially  = 0)
int next-count = 0;
```

- **Each external procedure *F* will be replaced by**

```
wait(mutex);
    …
  body of F;
    …
if (next-count > 0)
        signal(next)
else
        signal(mutex);
```

- **Mutual exclusion within a monitor is ensured.**

- **For each condition variable *x*, we have:**

```
semaphore x-sem; // (initially  = 0)
int x-count = 0;
```

- **The operation** x.wait **can be implemented as:**

```
x-count++;
if (next-count > 0)
        signal(next);
else
        signal(mutex);
wait(x-sem);
x-count--;
```

- **The operation** x.signal **can be implemented as:**

```
if (x-count > 0) {
        next-count++;
        signal(x-sem);
        wait(next);
        next-count--;
}
```

5. Give a detailed description about deadlocks and its characterization

- Deadlock Characterization
- Necessary Conditions

- Mutual exclusion: only one process at a time can use a resource.
- Hold and wait: a process holding at least one resource is waiting to acquire additional resources held by other processes.
- No preemption: a resource can be released only voluntarily by the process holding it, after that process has completed its task.
- Circular wait: there exists a set $\{P_0, P_1, \ldots, P_0\}$ of waiting processes such that P_0 is waiting for a resource that is held by P_1, P_1 is waiting for a resource that is held by

$P_2, \ldots, P_{n-1}$ is waiting for a resource that is held by
P_n, and P_0 is waiting for a resource that is held by P_0.

6. Explain about the methods used to prevent deadlocks

Deadlock Prevention

- Mutual Exclusion – not required for sharable resources; must hold for nonsharable resources.
- Hold and Wait – must guarantee that whenever a process requests a resource, it does not hold any other resources.
- No Preemption –
- Circular Wait – impose a total ordering of all resource types, and require that each process requests resources in an increasing order of enumeration.

7. Write in detail about deadlock avoidance.

- Multiple instances.
- Each process must a priori claim maximum use.
- When a process requests a resource it may have to wait.
- When a process gets all its resources it must return them in a finite amount of time.
- Data Structures for the Banker's Algorithm
- Safety Algorithm
- Resource-Request Algorithm for Process P_i
- Example of Banker's Algorithm

UNIT-III

1. Explain Dynamic Storage-Allocation Problem

- First-fit: Allocate the *first* hole that is big enough.
- Best-fit: Allocate the *smallest* hole that is big enough; must search entire list, unless ordered by size. Produces the smallest leftover hole.
- Worst-fit: Allocate the *largest* hole; must also search entire list. Produces the largest leftover hole.

First-fit and best-fit better than worst-fit in terms of speed and storage utilization.

2. Explain about fragmentation

Fragmentation

- External Fragmentation – total memory space exists to satisfy a request, but it is not contiguous.
- Internal Fragmentation – allocated memory may be slightly larger than requested memory; this size difference is memory internal to a partition, but not being used.
- Reduce external fragmentation by compaction

- Shuffle memory contents to place all free memory together in one large block.
- Compaction is possible *only* if relocation is dynamic, and is done at execution time.
- I/O problem
  - Latch job in memory while it is involved in I/O.
  - Do I/O only into OS buffers.

3. Explain the concept of Paging

Basic method

- Logical address space of a process can be non-contiguous; process is allocated physical memory whenever the latter is available.
- Divide physical memory into fixed-sized blocks called frames (size is power of 2, between 512 bytes and 8192 bytes).
- Divide logical memory into blocks of same size called pages.
- Keep track of all free frames.
- To run a program of size *n* pages, need to find *n* free frames and load program.
- Set up a page table to translate logical to physical addresses.
- Internal fragmentation.

Address Translation Scheme

- Address generated by CPU is divided into:

- *Page number (p)* – used as an index into a *page table* which contains base address of each page in physical memory.
  Page offset (d) – combined with base address to define the physical memory address that is sent to the memory unit.

4. Explain the types of Page Table Structure

- Hierarchical Paging
- Hashed Page Tables
- Inverted Page Tables

5.Explain about segmentation in detail.

Basic method

- Memory-management scheme that supports user view of memory.

Segmentation Architecture

a. Logical address
b. *Segment table*
   - base
   - *limit*
   - *Segment-table base register (STBR)*
     Segment-table length register (STLR)
   - Relocation.

- Sharing.

- shared segments
- same segment number

- Allocation.

- first fit/best fit
- external fragmentation

- Protection. With each entry in segment table associate:

- validation bit = 0 ⇒ illegal segment
- read/write/execute privileges

- Protection bits associated with segments; code sharing occurs at segment level.
- Since segments vary in length, memory allocation is a dynamic storage-allocation problem.
- A segmentation example is shown in the following diagram

UNIT-IV

1. Explain the file system structure in detail

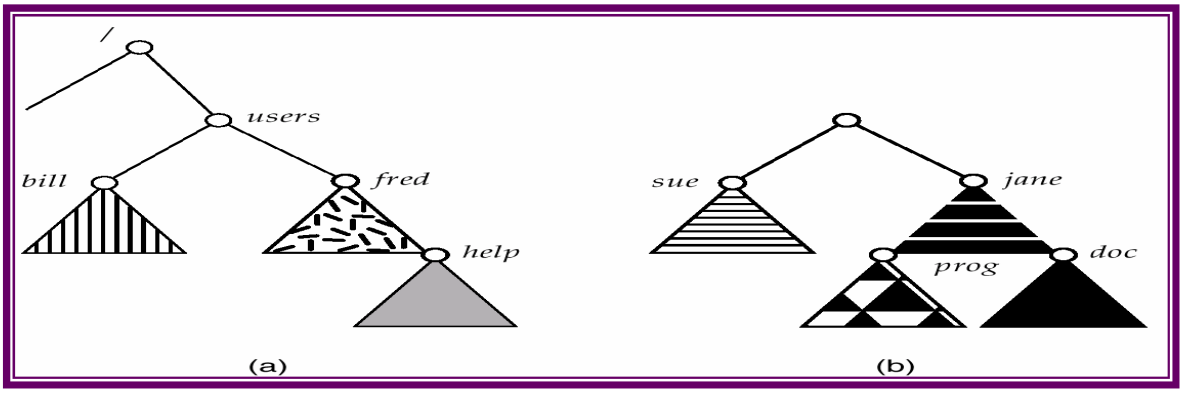
- None - sequence of words, bytes
- Simple record structure

- Lines
- Fixed length
- Variable length
  - Complex Structures
- Formatted document
- Relocatable load file
  - Can simulate last two with first method by inserting appropriate control characters.
  - Who decides:
- Operating system
- Program

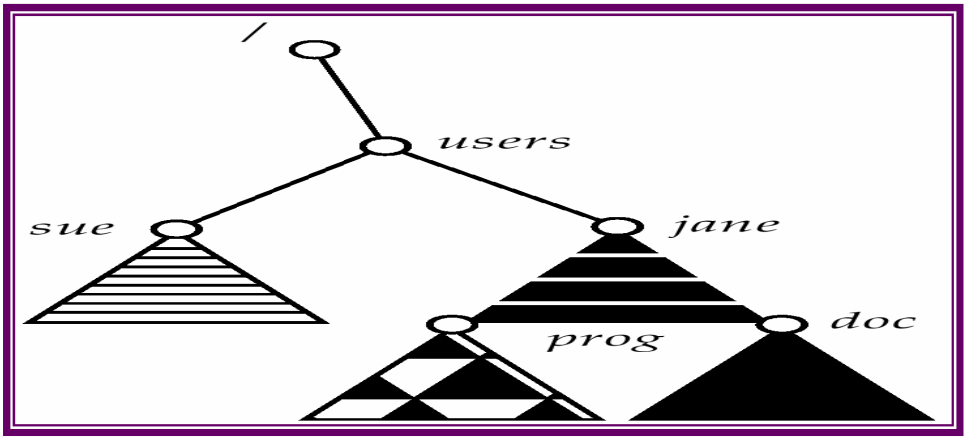
2. Discuss the file system organization and file system mounting.

- A file system must be mounted before it can be accessed.
- A unmounted file system is mounted at a mount point.

(a) Existing. (b) Unmounted Partition



Mount Point



3. Explain about file sharing.

- Introduction

- File Sharing – Remote File Systems
- File Sharing – Failure Modes
- File Sharing – Consistency Semantics

4. File System Implementation

- File System Structure
- File System Implementation
- Directory Implementation
- Allocation Methods
- Free-Space Management
- Efficiency and Performance
- Recovery
- Log-Structured File Systems
- NFS

5. Explain about various allocation methods.

An allocation method refers to how disk blocks are allocated for files:

- Contiguous allocation
- Linked allocation
- Indexed allocation

UNIT-V

1. Explain about disk scheduling technique

1. FCFS
2. SSTF
3. SCAN
4. C-SCAN
5. C-LOOK

2. Overview of Mass Storage Structure

- Magnetic disks provide bulk of secondary storage of modern computers

- Drives rotate at 60 to 200 times per second
- Transfer rate is rate at which data flow between drive and computer
- Positioning time (random-access time) is time to move disk arm to desired cylinder (seek time) and time for desired sector to rotate under the disk head (rotational latency)

- Head crash results from disk head making contact with the disk surface
  - That's bad
    - Disks can be removable
    - Drive attached to computer via I/O bus
- Busses vary, including EIDE, ATA, SATA, USB, Fibre Channel, SCSI
- Host controller in computer uses bus to talk to disk controller built into drive or storage array
  - Magnetic tape
- Was early secondary-storage medium
- Relatively permanent and holds large quantities of data
- Access time slow
- Random access ~1000 times slower than disk
- Mainly used for backup, storage of infrequently-used data, transfer medium between systems
- Kept in spool and wound or rewound past read-write head
- Once data under head, transfer rates comparable to disk
- 20-200GB typical storage
- Common technologies are 4mm, 8mm, 19mm, LTO-2 and SDLT

3. Explain in detail about Raid

RAID 1
RAID 2
RAID 3
RAID 4
RAID 5

QUESTION BANK (16 MARKS)

UNIT-I

1. Explain the various types of computer systems.
2. Explain how protection is provided for the hardware resources by the operating system.
3. What are the system components of an operating system and explain them?
4. Define system calls. Write about the various system calls.
5. What are the various process scheduling concepts
6. Explain about interprocess communication.
7. Explain the essential properties of the following types of Operating System.
    i) Multiprogrammed Systems
    ii) Time-sharing Systems

8. Explain the following:

    i) Real-time Systems
    ii) Distributed Systems

9. What is a process? Explain the process control block and the various process states.

10. Explain process creation and process termination
11. What are the important issue involved in the Design and Implementation of operating systems?
12. Discuss the design issues of distributed operating system.
13. Distinguish between multiprogramming (multi-tasking) and multiprocessing.
14. What are operating system services? Explain each one of them.
15. Explain Co-operating process with example.

UNIT-II

2. Give an overview about threads.
3. Explain in detail about the threading issues.
4. a) Write about the various CPU scheduling algorithms.

b) Explain the various scheduling criteria used for comparing CPU scheduling algorithms.

5. Write notes about multiple-processor scheduling and real-time scheduling.
6. What is critical section problem and explain two process solutions and multiple process solutions?
7. Explain what semaphores are, their usage, implementation given to avoid busy waiting and binary semaphores.
8. Explain the classic problems of synchronization.
9. Write about critical regions and monitors.

9. Explain about Dining Philosopher's problem of Synchronization

Explain- Dinning philosopher Problem, algorithm

10. Explain about Producer Consumer Problem

Explain- Producer Consumer Problem, algorithm.

11. What is the job of a scheduling algorithm? State the objective of a good scheduling algorithm. Explain round robin, priority and shortest job first scheduling algorithms.
12. a) Explain the criteria for scheduling.

b) Describe in detail the Multi-level feedback queue scheduling.

13. What is the advantage of having different time-quantum size at different levels in Multi-level Feedback Queue (MFQ) based scheduling.
14. Explain the various CPU scheduling techniques with Gantt charts clearly as indicated by (process name, arrival time, process time) for the following

(A, 0, 4), (B, 2, 7), (C, 3, 2) and (D, 3, 2) for FCFS,SJF,SRT,RR with quantum 1 and 2.

15. A system uses the following preemptive priority-scheduling algorithm (processes with larger priority numbers have high priority). Processes enter the system with a priority of 0. While waiting on the ready queue, a process priority changes at rate α. While running, a process priority change at rate β.

(1). what is the algorithm that results from $\beta > \alpha > 0$?

(2). what is the algorithm that results from $\alpha < \beta < 0$?

Justify your statement.

16. a) Explain process scheduler.

b) What is context switching? Explain briefly.

17. What are p-threads? Explain with the code.

UNIT-III

1. Give a detailed description about deadlocks and its characterization
2. Explain about the methods used to prevent deadlocks
3. Write in detail about deadlock avoidance.
4. Explain the Banker's algorithm for deadlock avoidance.
5. Give an account about deadlock detection.
6. What are the methods involved in recovery from deadlocks?
7. Explain about contiguous memory allocation.
8. Give the basic concepts about paging.
9. Write about the techniques for structuring the page table.
10. Explain the basic concepts of segmentation.
11. Describe the consequence of multiprogramming with fixed partitioning and variable partitioning. Also explain the swapping process.
12. Explain how memory can be dynamically allocated using first fit, best fit, and worst fit strategies.
13. Explain how sharing of pages can be achieved.
14. Define the deadlock problem. Explain in detail the prevention and recovery methods of deadlock.
15. Discuss the salient features and merits of multilevel paging and inverted page tables.
16. Explain in detail about External and Internal fragmentation.
17. Why paging and segmentation combined in one scheme for virtual memory.
18. What are the aspects for comparison of different MMV strategies?
19. Explain the concept of inverted page-tables.
20. Write short notes on "working set" and "Degree of multiprogramming".

UNIT-IV

1. What is demand paging and what is its use?
2. Explain the various page replacement strategies.
3. What is thrashing and explain the methods to avoid thrashing?
4. Explain about Access Methods.
5. What are files and explain the access methods for files?
6. Explain the schemes for defining the logical structure of a directory.
7. Write notes about the protection strategies provided for files.
8. Explain in detail the WS clock page replacement algorithm.
9. What is page fault? Explain the procedure for handling the page fault using Demand paging?
10. Explain I/O Interlock.
11. Explain the following Page Replacement Algorithms:

    i. Optimal Algorithm
    ii. LRU approximation Algorithm
    iii. Page Buffering Algorithm
    iv. Counting Algorithm

12. Explain in thrashing in detail.
13. Explain Allocation of frames.
14. Give references to the following pages by a program, 0,9,0,1,8, 1,8,7,8,7, 1,2,8,2,7, 8,2,3,8,3.

    How many page faults will occur if the program has three page frames available to it and uses:

    (a). FIFO replacement? (8 Faults)
    (b). LRU replacement? (9 Faults)
    (c). Optimal replacement? (7 Faults)

15. Consider the following page reference string:

    1, 2, 3, 4, 2, 1, 5, 6, 2, 1, 2, 3, 7, 6, 3, 2, 1, 2, 3, 6.

    How many page faults would occur for the following replacement algorithms, assuming one, two, three, four, five, six, or seven frames? Remember all frames are initially empty, so your first unique pages will all cost one fault each.

    (a). LRU replacement
    (b). FIFO replacement
    (c). Optimal replacement

Answer:

Number of frames	LRU	FIFO	Optimal
1	20	20	20
2	18	18	15
3	15	16	11
4	10	14	8
5	8	10	7
6	7	10	7
7	7	7	7

16. A page-replacement algorithm should minimize the number of page faults. We can do this minimization by distributing heavily used pages evenly over all of memory, rather than

having them compete for a small number of page frames. We can associate with each page frame a counter of the number of pages that are associated with that frame. Then, to replace a page, we search for the page frame with the smallest counter.

(a). Define a page-replacement algorithm using this basic idea. Specifically address the problems of (1) what the initial value of the counters is, (2) when counters are increased,(3) when counters are decreased, and (4) how the page to be replaced is selected.

(b). How many page faults occur for your algorithm for the following reference string, for four page frames?

1, 2, 3, 4, 5, 3, 4, 1, 6, 7, 8, 7, 8, 9, 7, 8, 9, 5, 4, 5, 4, 2.

(c). what is the minimum number of page faults for an optimal page-replacement strategy for the reference string in part b with four page frames?

Answer:

(a). Define a page-replacement algorithm addressing the problems of:

i. Initial value of the counters—0.
ii. Counters are increased—whenever a new page is associated with that frame.
iii. Counters are decreased—whenever one of the pages associated with that frame is no longer required.
iv. How the page to be replaced is selected—find a frame with the smallest counter. Use FIFO for breaking ties.

(b). 14 page faults

(c). 11 page faults

UNIT-V

1. Explain the allocation methods for disk space.
2. What are the various methods for free space management?
3. Write about the kernel I/O subsystem.
4. Explain the various disk scheduling techniques
5. Write notes about disk management and swap-space management.
6. Explain in detail the allocation and freeing the file storage space.
7. Explain the backup and recovery of files.
8. Discuss with diagrams the following three disk scheduling: FCFS, SSTF, C-SCAN.
9. Compare and contrast the FREE SPACE and SWAP SPACE management.
10. Explain the disk scheduling algorithms
11. Discuss the file system organization and file system mounting.
12. Explain the merits and demerits of any one-file system allocation method.
13. Describe the most common schemes for defining the logical structure of a Directory.
14. Explain the life cycle of an I/O request with flowchart.
15. Discuss about the UNIX file system in detail.
16. Discuss briefly about Memory Management in UNIX.
17. Explain the process management under LINUX OS.
18. Compare and contrast “free-space management” and “swap-space management”.
19. What is meant by “symmetric multiprocessing” and “asymmetric multiprocessing” in LINUX OS.
20. In what ways the directory is implemented?
21. Explain linked allocation in detail.
22. Write the indexed allocation with its performance.
23. Explain the I/O hardware.