OSSP-S-2018

1. Differentiate between application program and system program.

Ans: Application software (app for short) is a program or group of programs designed for end users. System software is a type of computer program that is designed to run a computer's hardware and application programs.

a. What is the function of loader? Name and explain in brief typed of loader.

Ans: Loader is the part of an operating system that is responsible for loading programs and libraries. It is one of the essential stages in the process of starting a program, as it places programs into memory and prepares them for execution.

Type of Loaders: The different types of loaders are, absolute loader, bootstrap loader, compile and go loader, and, direct linking loader.

Absolute Loader: The operation of absolute loader is very simple. The object code is loaded to specified locations in the memory. At the end the loader jumps to the specified address to begin execution of the loaded program. The advantage of absolute loader is simple and efficient. But the disadvantages are, the need for programmer to specify the actual address, and, difficult to use subroutine libraries.

A Bootstrap Loader: When a computer is first turned on or restarted, a special type of absolute loader, called bootstrap loader is executed. This bootstrap loads the first program to be run by the computer -- usually an operating system. The bootstrap itself begins at address 0. It loads the OS starting address 0x80. Such a loader is added to the beginning of all object programs that are to be loaded into an empty and idle system.

Compile and go loader: Compile and go loader is a link editor or program loader in which the assembler itself places the assembled instruction directly into the designated memory locations for execution.

Direct linking loader: it is a relocatable loader. It has advantage of allowing programmer multiple procedure segments and multiple data segments. Complete freedom in referring data or instructions contained other segments, provide flexible intersegment referencing.

b. State and explain the seven phases of compiler.

Ans: Different phases of a compiler are

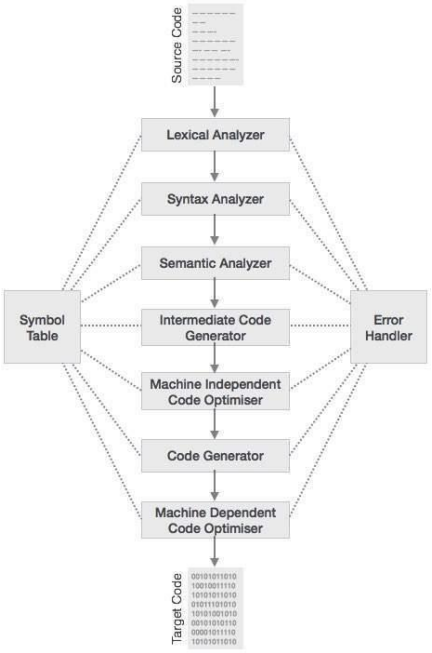
Lexical Analysis

The first phase of scanner works as a text scanner. This phase scans the source code as a stream of characters and converts it into meaningful lexemes. Lexical analyzer represents these lexemes in the form of tokens as:

<token-name, attribute-value>

Syntax Analysis

The next phase is called the syntax analysis or parsing. It takes the token produced by lexical analysis as input and generates a parse tree (or syntax tree). In this phase, token arrangements are checked against the source code grammar, i.e. the parser checks if the expression made by the tokens is syntactically correct.



Semantic Analysis

Semantic analysis checks whether the parse tree constructed follows the rules of language. For example, assignment of values is between compatible data types, and adding string to an integer. Also, the semantic analyzer keeps track of identifiers, their types and expressions; whether identifiers are declared before use or not etc. The semantic analyzer produces an annotated syntax tree as an output.

Intermediate Code Generation

After semantic analysis the compiler generates an intermediate code of the source code for the target machine. It represents a program for some abstract machine. It is in between the high-level language and the machine language. This intermediate code should be generated in such a way that it makes it easier to be translated into the target machine code.

Code Optimization

The next phase does code optimization of the intermediate code. Optimization can be assumed as something that removes unnecessary code lines, and arranges the sequence of statements in order to speed up the program execution without wasting resources (CPU, memory).

Code Generation

In this phase, the code generator takes the optimized representation of the intermediate code and maps it to the target machine language. The code generator translates the intermediate code into a sequence of (generally) re-locatable machine code. Sequence of instructions of machine code performs the task as the intermediate code would do.

Symbol Table

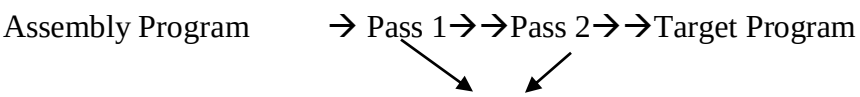
It is a data-structure maintained throughout all the phases of a compiler. All the identifier's names along with their types are stored here. The symbol table makes it easier for the compiler to quickly search the identifier record and retrieve it. The symbol table is also used for scope management.

2. Name two operating system.

Ans: windows, linux

a. What is the function of assembler? Explain.

Ans: An assembler is a program that converts assembly language into machine code. It takes the basic commands and operations from assembly code and converts them into binary code that can be recognized by a specific type of processor. Assemblers are similar to compilers in that they produce executable code. It generates instructions by evaluating the mnemonics (symbols) in operation field and find the value of symbol and literals to produce machine code. Now, if assembler do all this work in one scan then it is called single pass assembler, otherwise if it does in multiple scans then called multiple pass assembler.



Here assembler divide tasks in two passes:

Pass-1:

i. Define symbols and literals and remember them in symbol table and literal table respectively.
ii. Keep track of location counter
iii. Process pseudo-operations

Pass-2:

iv. Generate object code by converting symbolic op-code into respective numeric op-code
v. Generate data for literals and look for values of symbols

b. What is file ? state file organization and explain file access methods.

Ans: A f*ile* is a computer resource for recording data discretely in a computer storage device.

- File organization: There are four methods of organizing files on a storage media. This include:

- sequential,
- random,
- serial and
- indexed-sequential

1. Sequential file organization

- Records are stored and accessed in a particular order sorted using a key field.
- Retrieval requires searching sequentially through the entire file record by record to the end.
- Because the record in a file are sorted in a particular order, better file searching methods like the binary search technique can be used to reduce the time used for searching a file .
- Since the records are sorted, it is possible to know in which half of the file a particular record being searched is located, Hence this method repeatedly divides the set of records in the file into two halves and searches only the half on which the records is found.

Disadvantages of sequential file organization

- The sorting does not remove the need to access other records as the search looks for particular records.
- Sequential records cannot support modern technologies that require fast access to stored records.
- The requirement that all records be of the same size is sometimes difficult to enforce.

1. Random or direct file organization

- Records are stored randomly but accessed directly.
- To access a file stored randomly, a record key is used to determine where a record is stored on the storage media.
- Magnetic and optical disks allow data to be stored and accessed randomly.

Advantages of random file access

- Quick retrieval of records.
- The records can be of different sizes.

1. Serial file organization

- Records in a file are stored and accessed one after another.
- The records are not stored in any way on the storage medium this type of organization is mainly used on magnetic tapes.

Advantages of serial file organization

- It is simple
- It is cheap

Disadvantages of serial file organization

- It is cumbersome to access because you have to access all proceeding records before retrieving the one being searched.
- Wastage of space on medium in form of inter-record gap.
- It cannot support modern high speed requirements for quick record access.

1. Indexed-sequential file organization method

- Almost similar to sequential method only that, an index is used to enable the computer to locate individual records on the storage media. For example, on a magnetic drum, records are stored sequential on the tracks. However, each record is assigned an index that can be used to access it directly.

3. What is spooling?

b. Ans: Spooling is a process in which data is temporarily held to be used and executed by a device, program or the system. Data is sent to and stored in memory or other volatile storage until the program or computer requests it for execution. "Spool" is technically an acronym for simultaneous peripheral operations online.

a. Define device management. Explain the function of I/O scheduler and I/O device handler.

Ans: Device management is the process of managing the implementation, operation and maintenance of a physical and/or virtual device.

i/o scheduler: An I/O scheduler works by managing a block device's request queue. It decides the order of requests in the queue and at what time each request is dispatched to the block device. It manages the request queue with the goal of reducing seeks, which results in greater *global throughput*. An I/O scheduler, very openly, is unfair to some requests at the expense of improving the *overall* performance of the system.

I/O schedulers perform two primary actions to minimize seeks: merging and sorting. Merging is the coalescing of two or more requests into one. Consider an example request that is submitted to the queue by a file system say, to read a chunk of data from a file. (At this point, of course, everything is occurring in terms of sectors and blocks and not files, but presume that the requested blocks originate from a chunk of a file.) If a request is already in the queue to read from an adjacent sector on the disk (for example, an earlier chunk of the same file), the two requests can be merged into a single request operating on one or more adjacent on-disk sectors. By merging requests, the I/O scheduler reduces the overhead of multiple requests down to a single request. More importantly, only a single command needs to be issued to the disk and servicing the multiple requests can be done without seeking. Consequently, merging requests reduces overhead and minimizes seeks.

- i/o device handler: Device Handlers are software that interface between OS and I/O devices. When I/O instruction is received, Device Handler converts generic instruction and internal character character code into format required by specific device. When I/O instruction is received, Device Handler converts generic instruction and internal character character code into format required by specific device.To convert generic instructions in specific commands, device handler needs to know the device's characteristics.

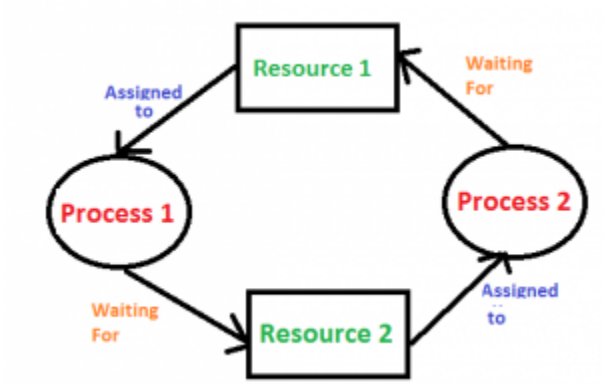
b. State the condition when dead lock occurs. Explain how it is detected, recovered and prevented.

Ans: A deadlock is a situation in which two computer programs sharing the same resource are effectively preventing each other from accessing the resource, resulting in both programs ceasing to function. The earliest computer operating systems ran only one program at a time.

Deadlock detection and recovery: Deadlock Detection

1. If resources have single instance:

In this case for Deadlock detection we can run an algorithm to check for cycle in the Resource Allocation Graph. Presence of cycle in the graph is the sufficient condition for deadlock.



2. In the above diagram, resource 1 and resource 2 have single instances. There is a cycle R1 → P1 → R2 → P2. So, Deadlock is Confirmed.

3. If there are multiple instances of resources:

Detection of the cycle is necessary but not sufficient condition for deadlock detection, in this case, the system may or may not be in deadlock varies according to different situations.

Deadlock Recovery

A traditional operating system such as Windows doesn't deal with deadlock recovery as it is time and space consuming process. Real-time operating systems use Deadlock recovery.

Recovery method

1. Killing the process: killing all the process involved in the deadlock. Killing process one by one. After killing each process check for deadlock again keep repeating the process till system recover from deadlock.
2. Resource Preemption: Resources are preempted from the processes involved in the deadlock, preempted resources are allocated to other processes so that there is a possibility of recovering the system from deadlock. In this case, the system goes into starvation.
Deadlock Preventation: Deadlock prevention is a set of methods for ensuring that at least one of these necessary conditions cannot hold.
Mutual Exclusion: The mutual exclusion condition holds for non sharable devices. Sharable resources do not require mutual exclusive access and thus cannot be involved in a dead lock.

Hold and wait: To ensure that the hold and wait condition never occurs in the system, we must guaranty that whenever a process requests a resource it does not hold any other resources. There are two protocols to handle these problems such as one protocol that can be used requires each process to request and be allocated all its resources before it begins execution. The other protocol allows a process to request resources only when the process has no resource. These protocols have disadvantages like resource utilization may be low. And also starvation may be possible.

No Preemption:Alternatively if a process requests some resources, the operating system first check whether they are available. If they are, the operating system allocate them. If they are not available, operating system check whether they are allocated to some other process that is waiting for additional resources. If so, operating system preempt the desired resources from the waiting process and allocate them to the requesting process. If the resources are not either available or held by a waiting process, the requesting process must wait.

Circular Wait: Let R = {R1, R2, …….Rn}be the set of resource types. We assign to each resource type a unique integer number, which allows us to compare two resources and to determine whether one precedes another in our ordering. This can be ensure that this condition never holds by ordering of all resource type and to require that each process requests resource in an increasing order of enumeration.

4. What is page fault?

Ans: A page fault is a type of exception raised by computer hardware when a running program accesses a memory page that is not currently mapped by the memory management unit (MMU) into the virtual address space of a process.

a. State and explain banker's Algorithm.

Ans: The Banker algorithm is a resource allocation and deadlock avoidance algorithm that tests for safety by simulating the allocation of predetermined maximum possible amounts of all resources, and then makes an "safty-state" check to test for possible deadlock conditions for all other pending activities, before deciding whether allocation should be allowed to continue.
The Data structures used by the Banker's Algorithm are:
Let 'n' be the number of processes in the system and 'm' be the number of resources types.
Available :

- It is a 1-d array of size 'm' indicating the number of available resources of each type.
- Available[j] = k means there are 'k' instances of resource type R_j

Max :

- It is a 2-d array of size 'n*m' that defines the maximum demand of each process in a system.
- Max[i, j] = k means process P_i may request at most 'k' instances of resource type R_j.

Allocation :

- It is a 2-d array of size 'n*m' that defines the number of resources of each type currently allocated to each process.
- Allocation[i, j] = k means process P_i is currently allocated 'k' instances of resource type R_j

Need :

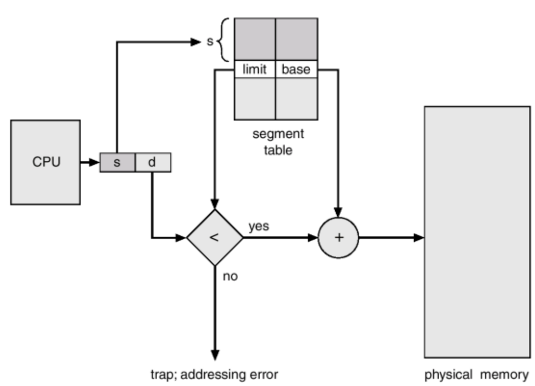
- It is a 2-d array of size ‘n*m’ that indicates the remaining resource need of each process.
- Need [i, j] = k means process P_i currently need ‘k’ instances of resource type R_j for its execution.
- Need [i, j] = Max [i, j] – Allocation [i, j]

$Allocation_i$ specifies the resources currently allocated to process P_i and $Need_i$ specifies the additional resources that process P_i may still request to complete its task.

b. What is segmentation? Explain memory management with segmentation.

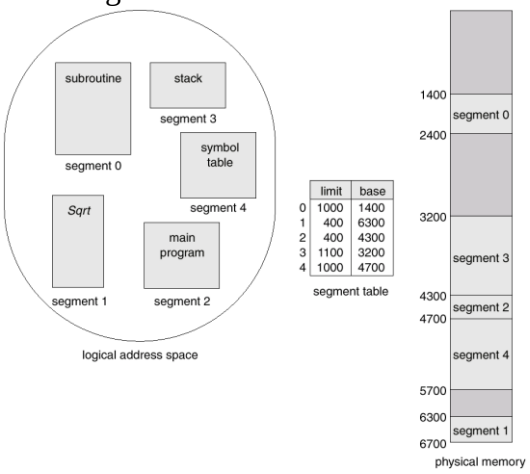
Ans: Segmentation is a memory management technique in which, the memory is divided into the variable size parts. Each part is known as segment which can be allocated to a process. The details about each segment are stored in a table called as segment table.
Segmentation is a memory management scheme that supports this user view of memory.
· A logical address space is a collection of segments. Each segment has a name and a length.
· The addresses specify both the segment name and the offset within the segment.
· The user therefore specifies each address by two quantities such as segment name and an offset.
For simplicity of implementation, segments are numbered and are referred to by a segment number, rather than by a segment name.
· Logical address consists of a two tuples:
$<$segment-number, offset$>$
· Segment table – maps two-dimensional physical addresses; each table entry has:
o Base – contains the starting physical address where the segments reside in memory.
o Limit – specifies the length of the segment.
· Segment-table base register (STBR) points to the segment table’s location in memory.
· Segment-table length register (STLR) indicates number of segments used by a program;

Segment number s is legal if s< STLR.



When the user program is compiled by the compiler it constructs the segments.
· The loader takes all the segments and assigned the segment numbers.
· The mapping between the logical and physical address using the segmentation technique is shown in above figure.
· Each entry in the segment table as limit and base address.
· The base address contains the starting physical address of a segment where the limit address specifies the length of the segment.
· The logical address consists of 2 parts such as segment number and offset.

· The segment number is used as an index into the segment table. Consider the example is given below.

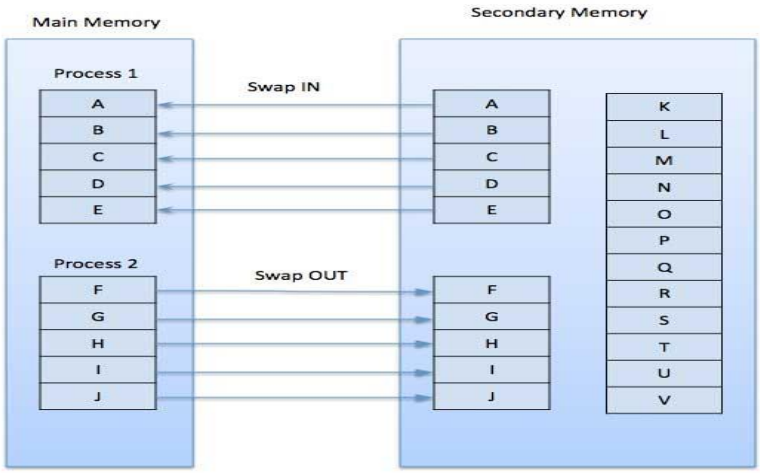


5. What is interacting process?

Ans: Process Interaction is a model of managing parallel or concurrent processes by defining how data between these processes is exchanged and how the processes are synchronized with each other.

a. State and explain demand paging concepts.

Ans: A demand paging system is quite similar to a paging system with swapping where processes reside in secondary memory and pages are loaded only on demand, not in advance. When a context switch occurs, the operating system does not copy any of the old program's pages out to the disk or any of the new program's pages into the main memory Instead, it just begins executing the new program after loading the first page and fetches that program's pages as they are referenced.



While executing a program, if the program references a page which is not available in the main memory because it was swapped out a little ago, the processor treats this invalid memory reference as a page fault and transfers control from the program to the operating system to demand the page back into the memory.

Advantages

Following are the advantages of Demand Paging −

- Large virtual memory.
- More efficient use of memory.
- There is no limit on degree of multiprogramming.

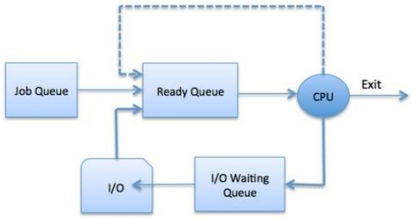
Disadvantages

- Number of tables and the amount of processor overhead for handling page interrupts are greater than in the case of the simple paged management techniques.

b. Write the types of scheduling. State and explain job scheduling by taking suitable example.

Ans: Job scheduling is the process of allocating system resources to many different tasks by an operating system (OS). The system handles prioritized job queues that are awaiting CPU time and it should determine which job to be taken from which queue and the amount of time to be allocated for the job. The Operating System maintains the following important process scheduling queues −

- Job queue − This queue keeps all the processes in the system.
- Ready queue − This queue keeps a set of all processes residing in main memory, ready and waiting to execute. A new process is always put in this queue.
- Device queues − The processes which are blocked due to unavailability of an I/O device constitute this queue.



Job Schedulers are special system software which handle process scheduling in various ways. Their main task is to select the jobs to be submitted into the system and to decide which process to run. Job Schedulers are of three types −

- Long-Term Scheduler
- Short-Term Scheduler
- Medium-Term Scheduler

Long Term Scheduler

It is also called a job scheduler. A long-term scheduler determines which programs are admitted to the system for processing. It selects processes from the queue and loads them into memory for execution. Process loads into the memory for CPU scheduling.

The primary objective of the job scheduler is to provide a balanced mix of jobs, such as I/O bound and processor bound. It also controls the degree of multiprogramming. If the degree of multiprogramming is stable, then the average rate of process creation must be equal to the average departure rate of processes leaving the system.

Short Term Scheduler

It is also called as CPU scheduler. Its main objective is to increase system performance in accordance with the chosen set of criteria. It is the change of ready state to running state of the process. CPU scheduler selects a process among the processes that are ready to execute and allocates CPU to one of them.

Short-term schedulers, also known as dispatchers, make the decision of which process to execute next. Short-term schedulers are faster than long-term schedulers.

Medium Term Scheduler

Medium-term scheduling is a part of swapping. It removes the processes from the memory. It reduces the degree of multiprogramming. The medium-term scheduler is in-charge of handling the swapped out-processes.

A running process may become suspended if it makes an I/O request. A suspended processes cannot make any progress towards completion. In this condition, to remove the process from memory and make space for other processes, the suspended process is moved to the secondary storage. This process is called swapping, and the process is said to be swapped out or rolled out. Swapping may be necessary to improve the process mix.

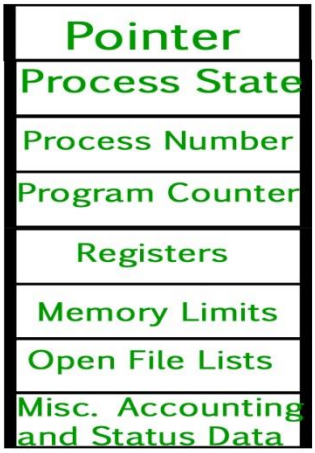
6. Define kernel.

Ans: The kernel is the central module of an operating system which is responsible for memory management, process and task management, and disk management. The kernel connects the system hardware to the application software. Every operating system has a kernel.

a. What is PCB? Explain different fields stored in PCBs.

Ans: A process control block is a data structure used by computer operating systems to store all the information about a process.

A process control block (PCB) contains information about the process, i.e. registers, quantum, priority, etc. The process table is an array of PCB's, that means logically contains a PCB for all of the current processes in the system.



Process Control Block

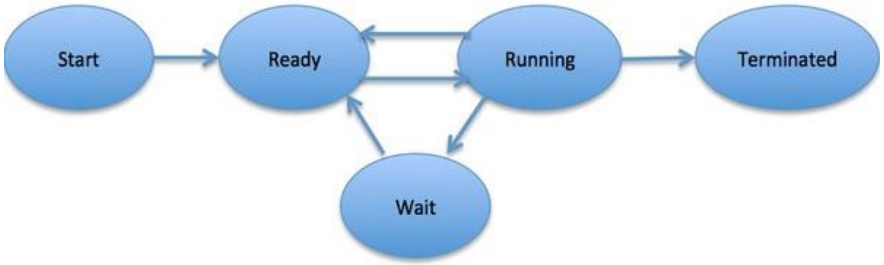
- Pointer – It is a stack pointer which is required to be saved when the process is switched from one state to another to retain the current position of the process.
- Process state – It stores the respective state of the process.
- Process number – Every process is assigned with a unique id known as process ID or PID which stores the process identifier.
- Program counter – It stores the counter which contains the address of the next instruction that is to be executed for the process.
- Register – These are the CPU registers which includes: accumulator, base, registers and general purpose registers.
- Memory limits – This field contains the information about memory management system used by operating system. This may include the page tables, segment tables etc.
- Open files list – This information includes the list of files opened for a process.

Miscellaneous accounting and status data – This field includes information about the amount of CPU used, time constraints, jobs or process number, etc.
The process control block stores the register content also known as execution content of the processor when it was blocked from running. This execution content architecture enables the operating system to restore a process's execution context when the process returns to the running state. When the process made transitions from one state to another, the operating system update its information in the process's PCB. The operating system maintains pointers to each process's PCB in a process table so that it can access the PCB quickly.

b. Define process and process state. Draw a suitable diagram to explain different state of a process.

Ans: A process or a task is an instance of a program running in a computer. A process can initiate a subprocess, which is a called a child process. When a process executes, it passes through different states. These stages may differ in different operating systems, and the names of these states are also not standardized.

In general, a process can have one of the following five states at a time.



Start

This is the initial state when a process is first started/created.

Ready

The process is waiting to be assigned to a processor. Ready processes are waiting to have the processor allocated to them by the operating system so that they can run. Process may come into this state after Start state or while running it by but interrupted by the scheduler to assign CPU to some other process.

Running

Once the process has been assigned to a processor by the OS scheduler, the process state is set to running and the processor executes its instructions.

Waiting

Process moves into the waiting state if it needs to wait for a resource, such as waiting for user input, or waiting for a file to become available.

Terminated or Exit

Once the process finishes its execution, or it is terminated by the operating system, it is moved to the terminated state where it waits to be removed from main memory.

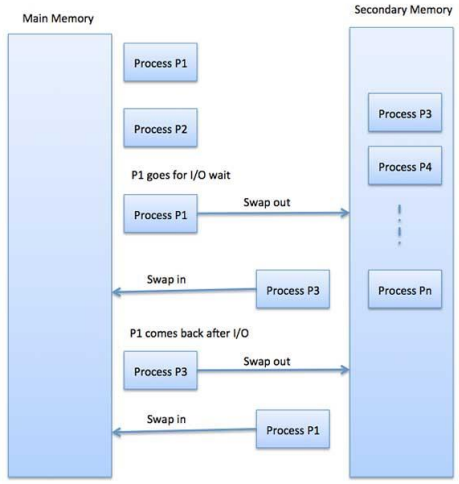
7. Define semaphore.

c. Ans: Semaphore is simply a variable that is non-negative and shared between threads. A semaphore is a signaling mechanism, and a thread that is waiting on a semaphore can be signaled by another thread. It uses two atomic operations, 1)wait, and 2) signal for the process synchronization.

a. What do you mean by swapping? Explain swapping with suitable diagram.

Ans: Swapping is a method in which the process should be swapped temporarily from the main memory to the backing store. It will be later brought back into the memory for continue execution.

Backing store is a hard disk or some other secondary storage device that should be big enough inorder to accommodate copies of all memory images for all users. It is also capable of offering direct access to these memory images.



Benefits of Swapping

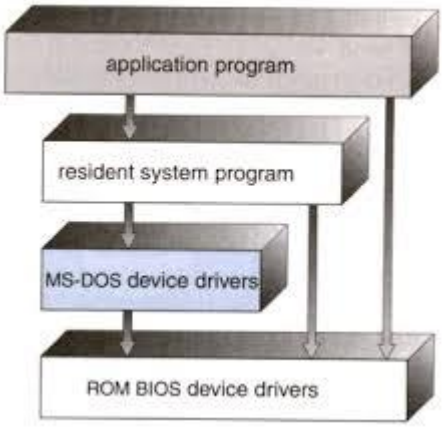
Here, are major benefits/pros of swapping:

- It offers a higher degree of multiprogramming.
- Allows dynamic relocation. For example, if address binding at execution time is being used, then processes can be swap in different locations. Else in case of compile and load time bindings, processes should be moved to the same location.
- It helps to get better utilization of memory.
- Minimum wastage of CPU time on completion so it can easily be applied to a priority-based scheduling method to improve its performance.

b. State and explain structure of operating system.

Ans: System structure:
1. Simple structure: There are several commercial system that don‘t have a well- defined structure such operating systems begins as small, simple & limited systems and then grow beyond their original scope.

MS-DOS is an example of such system. It was not divided into modules carefully. Another example of limited structuring is the UNIX operating system.



(MS DOS Structure)

2. Layered approach: In the layered approach, the OS is broken into a number of layers (levels) each built on top of lower layers. The bottom layer (layer o) is the hardware & top

most layer (layer N) is the user interface.

The main advantage of the layered approach

is modularity.

The layers are selected such that each users

functions (or operations) & services of only

lower layer.

This approach simplifies debugging & system verification, i.e. the first layer can be debugged without concerning the rest of the system. Once the first layer is debugged, its correct functioning is assumed while the 2nd layer is debugged & so on.

If an error is found during the debugging of a particular layer, the error must be on that layer because the layers below it are already debugged. Thus the design & implementation of the system are simplified when the system is broken down into layers.

Each layer is implemented using only operations provided by lower layers. A layer doesn‘t need to know how these operations are implemented; it only needs to know what these operations do.

8. Functions of operating system.

Ans: Following are some of important functions of an operating System.

Memory Management

An Operating System does the following activities for memory management −

- Keeps tracks of primary memory, i.e., what part of it are in use by whom, what part are not in use.
- In multiprogramming, the OS decides which process will get memory when and how much.
- Allocates the memory when a process requests it to do so.
- De-allocates the memory when a process no longer needs it or has been terminated.

Processor Management

An Operating System does the following activities for processor management −

- Keeps tracks of processor and status of process. The program responsible for this task is known as traffic controller.
- Allocates the processor (CPU) to a process.
- De-allocates processor when a process is no longer required.

Device Management

It does the following activities for device management −

- Keeps tracks of all devices. Program responsible for this task is known as the I/O controller.
- Decides which process gets the device when and for how much time.
- Allocates the device in the efficient way.
- De-allocates devices.

File Management

An Operating System does the following activities for file management −

- Keeps track of information, location, uses, status etc. The collective facilities are often known as file system.
- Decides who gets the resources.
- Allocates the resources.
- De-allocates the resources.

a. Process synchronization.

Ans: Process Synchronization means sharing system resources by processes in a such a way that, Concurrent access to shared data is handled thereby minimizing the chance of inconsistent data. Maintaining data consistency demands mechanisms to ensure synchronized execution of cooperating processes. The need for synchronization originates when processes need to execute concurrently. The main purpose of synchronization is the sharing of resources without interference using mutual exclusion. The other purpose is the coordination of the process interactions in an operating system.

b. Techniques of device management.

Ans: An Operating System manages device communication via their respective drivers. It does the following activities for device management −

- Keeps tracks of all devices. Program responsible for this task is known as the I/O controller.
- Decides which process gets the device when and for how much time.
- Allocates the device in the efficient way.
- De-allocates devices.

c. Secondary storage management.

Ans: Secondary storage devices are non-volatile devices where the data is stored for long-term storage. Disks are the mainly used secondary storage devices. They provide the bulk of secondary storage in operating systems today. The main activity that is performed in secondary storage management is disk scheduling. Secondary storage management is a classical feature of database management systems. It is usually supported through a set of mechanisms. ... The application programmer should not have to write code to maintain indices, to allocate disk storage, or to move data between disk and main memory.

d. Interpreter

Ans: It translates program one statement at a time. It takes less amount of time to analyze the source code but the overall execution time is slower. No intermediate object code is generated, hence are memory

efficient. Continues translating the program until the first error is met, in which case it stops. Hence debugging is easy.