OSSP-W-2019

1. Answer ALL questions.

(a) Write the functions of a Loader.

Ans: In computer systems a 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.

(b) What is semaphore?

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.

(c)What do you mean by segmentation?

Ans: In Operating Systems, 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.

(d) Define Deadlock.

Ans: Deadlock is a situation where a set of processes are blocked because each process is holding a resource and waiting for another resource acquired by some other process.

(e)Distinguish between File and Directory.

Ans: A file is a collection of data that is stored on disk and that can be manipulated as a single unit by its name. A directory is a file that acts as a folder for other files.

(f) What is Assembler?

Ans; An Assembler is a program that translates mnemonics into binary data that can be executed by a processor.

(g) Write the definition of Swapping.

Ans: Swapping is a mechanism in which a process can be swapped temporarily out of main memory (or move) to secondary storage (disk) and make that memory available to other processes. At some later time, the system swaps back the process from the secondary storage to main memory.

(h) Name various types of allocating Devices.

Ans: different types of allocating devices are preempting devices, non-preempting devices and virtual devices.

(i) Define virtual memory.

Ans: Virtual memory is a memory management capability of an operating system which uses hardware and software to allow a computer to compensate for physical memory shortages, by temporarily transferring data from random access memory (RAM) to disk storage.

(j) What do you mean by Spooling?

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

2. Answer any FIVE questions

(a)**Distinguish between paging and Segmentation**.

Ans:

Sr. No.	Key	Paging	Segmentation
1	Memory Size	In Paging, a process address space is broken into fixed sized blocks called pages.	In Segmentation, a process address space is broken in varying sized blocks called sections.

Sr. No.	Key	Paging	Segmentation
2	Accountability	Operating System divides the memory into pages.	Compiler is responsible to calculate the segment size, the virtual address and actual address.
3	Size	Page size is determined by available memory.	Section size is determined by the user.
4	Speed	Paging technique is faster in terms of memory access.	Segmentation is slower than paging.
5	Fragmentation	Paging can cause internal fragmentation as some pages may go underutilized.	Segmentation can cause external fragmentation as some memory block may not be used at all.
6	Logical Address	During paging, a logical address is divided into page number and page offset.	During segmentation, a logical address is divided into section number and section offset.
7	Table	During paging, a logical address is divided into page number and page offset.	During segmentation, a logical address is divided into section number and section offset.
8	Data Storage	Page table stores the page data.	Segmentation table stores the segmentation data.

(b) Explain the methods of deadlock prevention.

Ans: 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.

(c)What are the main functions of Operating system? Explain in brief.

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

In multiprogramming environment, the OS decides which process gets the processor when and for how much time. This function is called process scheduling. 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

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.

File Management

A file system is normally organized into directories for easy navigation and usage. These directories may contain files and other directions.

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.

Other Important Activities

Following are some of the important activities that an Operating System performs −

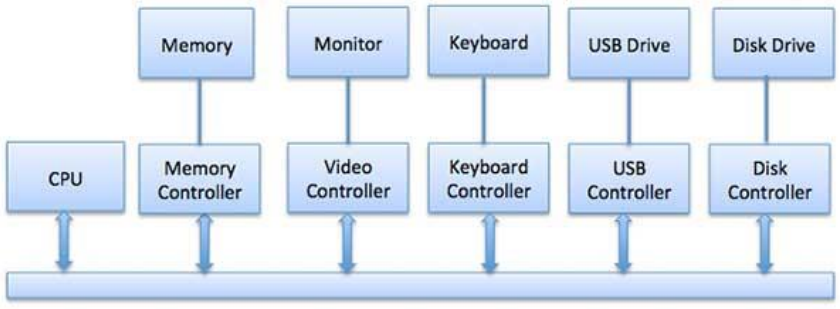
- Security − By means of password and similar other techniques, it prevents unauthorized access to programs and data.
- Control over system performance − Recording delays between request for a service and response from the system.
- Job accounting − Keeping track of time and resources used by various jobs and users.
- Error detecting aids − Production of dumps, traces, error messages, and other debugging and error detecting aids.
- Coordination between other softwares and users − Coordination and assignment of compilers, interpreters, assemblers and other software to the various users of the computer systems.

(d) Explain I/O Traffic Controller.

Ans: Device drivers are software modules that can be plugged into an OS to handle a particular device. Operating System takes help from device drivers to handle all I/O devices.

The Device Controller works like an interface between a device and a device driver. I/O units (Keyboard, mouse, printer, etc.) typically consist of a mechanical component and an electronic component where electronic component is called the device controller.

As an interface its main task is to convert serial bit stream to block of bytes, perform error correction as necessary.Any device connected to the computer is connected by a plug and socket, and the socket is connected to a device controller. Following is a model for connecting the CPU, memory, controllers, and I/O devices where CPU and device controllers all use a common bus for communication.



The CPU must have a way to pass information to and from an I/O device. There are three approaches available to communicate with the CPU and Device.

- Special Instruction I/O : This uses CPU instructions that are specifically made for controlling I/O devices. These instructions typically allow data to be sent to an I/O device or read from an I/O device.
- Memory-mapped I/O: Here the same address space is shared by memory and I/O devices. The device is connected directly to certain main memory locations so that I/O device can transfer block of data to/from memory without going through CPU.
  - Direct memory access (DMA) : Direct Memory Access (DMA) means CPU grants I/O module authority to read from or write to memory without involvement. DMA module itself controls exchange of data between main memory and the I/O device. CPU is only involved at the beginning and end of the transfer and interrupted only after entire block has been transferred.

(e)**Describe various file accessing methods**.
Ans: There are three ways to access a file into a computer system: Sequential-Access, Direct Access, Index sequential Method.

SequentialAccess–
It is the simplest access method. Information in the file is processed in order, one record after the other. for example, editor and compiler usually access the file in this fashion. Read and write make up the bulk of the operation on a file. A read operation -*read next*- read the next position of the file and automatically advance a file pointer, which keeps track I/O location. Similarly, for the write *write next* append to the end of the file and advance to the newly written material.

- Data is accessed one record right after another record in an order.
- When we use read command, it move ahead pointer by one
- When we use write command, it will allocate memory and move the pointer to the end of the file
- Such a method is reasonable for tape.

DirectAccess–
Another method is *direct access method* also known as *relative access method*. A filed-length logical record that allows the program to read and write record rapidly in no particular order. The direct access is based on the disk model of a file since disk allows random access to any file block. For direct access, the file is viewed as a numbered sequence of block or record. There is no restriction on the order of reading and writing for a direct access file.
A block number provided by the user to the operating system is normally a *relative block number*, the first relative block of the file is 0 and then 1 and so on.
Indexsequentialmethod–
It is the other method of accessing a file which is built on the top of the direct access method. These methods construct an index for the file. The index, like an index in the back of a book, contains the pointer to the various blocks. To find a record in the file, we first search the index and then by the help of pointer we access the file directly.
Key points:
- It is built on top of Sequential access.
- It control the pointer by using index.

(f) Distinguish between Compiler and Interpreter.
Ans:

Interpreter	Compiler
Translates program one statement at a time.	Scans the entire program and translates it as a whole into machine code.

It takes less amount of time to analyze the source code but the overall execution time is slower.	It takes a large amount of time to analyze the source code but the overall execution time is comparatively faster.
No intermediate object code is generated, hence are memory efficient.	Generates intermediate object code which further requires linking, hence requires more memory.
Continues translating the program until the first error is met, in which case it stops. Hence debugging is easy.	It generates the error message only after scanning the whole program. Hence debugging is comparatively hard.
Programming languages like Python, Ruby use interpreters.	Programming languages like C, C++ use compilers.

(g) Explain types of scheduling.

Ans: Types of Scheduling

Long-term Scheduling

Long term scheduling is performed when a new process is created. If the number of ready processes in the ready queue becomes very high, then there is a overhead on the processor for maintaining long lists, context switching and dispatching increases. Therefore, allow only limited number of processes in to the ready queue. The "long-term scheduler" managers this. Long-term scheduler determines which programs are admitted into the system for processing.

Medium-term Scheduling

Medium-term scheduling is a part of the swapping function. When part of the main memory gets freed, the operating system looks at the list of suspend ready processes, decides which one is to be swapped in. This scheduler works in close conjunction with the long-term scheduler. It will perform the swapping-in function among the swapped-out processes. Medium-term scheduler executes some what more frequently.

Short-term Scheduling

Short-term scheduler is also called as dispatcher. Short-term scheduler is invoked whenever an event occurs, that may lead to the interruption of the current running process. Short-term scheduler executes most frequently. It selects from among the processes that are ready to execute and allocates the CPU to one of them. It must select a new process for the CPU frequently. It must be very fast.

3. Explain Banker Algorithm of multiple instances.

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.

Banker’s algorithm consists of Safety algorithm and Resource request algorithm

Safety Algorithm: The algorithm for finding out whether or not a system is in a safe state can be described as follows:

1) Let Work and Finish be vectors of length ‘m’ and ‘n’ respectively.
Initialize: Work = Available
Finish[i] = false; for i=1, 2, 3, 4….n

2) Find an i such that both
a) Finish[i] = false
b) $Need_i$ <= Work
if no such i exists goto step (4)
3) Work = Work + Allocation[i]
Finish[i] = true
goto step (2)

4) if Finish [i] = true for all i
then the system is in a safe state

Resource-Request Algorithm

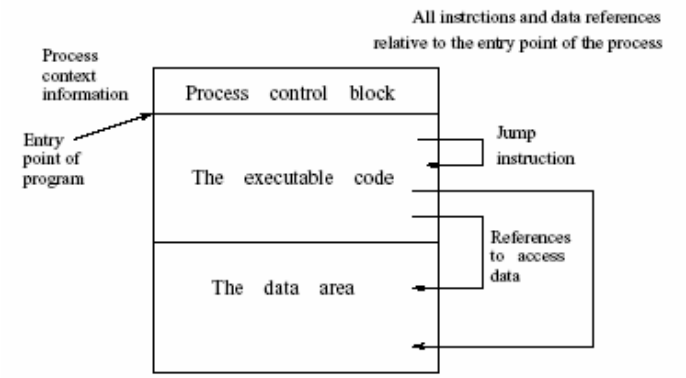
Let $Request_i$ be the request array for process P_i. $Request_i$ [j] = k means process P_i wants k instances of resource type R_j. When a request for resources is made by process P_i, the following actions are taken:

1) If $Request_i$ <= $Need_i$
Goto step (2) ; otherwise, raise an error condition, since the process has exceeded its maximum claim.
2) If $Request_i$ <= Available
Goto step (3); otherwise, P_i must wait, since the resources are not available.
3) Have the system pretend to have allocated the requested resources to process Pi by modifying the state as
follows:
Available = Available – Requesti
$Allocation_i$ = $Allocation_i$ + $Request_i$
$Need_i$ = $Need_i$– $Request_i$

4. Explain various memory allocation techniques.

Ans: Allocation: First of all the processes that are scheduled to run must be resident in the memory. These processes must be allocated space in main memory.

Let us assume the main memory is a linear map or one-dimensional array. If the address is known then its content can be fetched. So, a process residing in the main memory, set the program counter to an absolute address of its first instruction and can initiate its run. Also, if the locations of data is known then it can be fetched. This means that a process can be load with only absolute addresses for instructions and data, only when those specific addresses are free in main memory. But This will loose flexibility with regard to loading a process. For instance, we cannot load a process, if some other process is currently occupying that area which is needed by this process. This may happen even though we may have enough space in the memory. To avoid this processes are generated to be relocatable.



Initially, all the addresses in the process are relative to the start address. With this flexibility the OS can allocate any area in the memory to load this process. Its instruction, data, process context (process control block) and any other data structure required by the process can be accessed easily if the addresses are relative. Suppose a process created a hole on moving out. If non-relocatable addresses are to be used then severe problem can occurred.

When the process moves back in, that particular hole (or area) may not be available any longer. In case we can relocate, moving a process back in creates no problem. This is so because the process can be relocated in some other free area.

Contiguous memory allocation:

Contiguous memory allocation is a classical memory allocation model that assigns a process consecutive memory blocks (that is, memory blocks having consecutive addresses). When a process needs to execute, memory is requested by the process. The size of the process is compared with the amount of contiguous main memory available to execute the process. If sufficient contiguous memory is found, the process is allocated memory to start its execution. Otherwise, it is added to a queue of waiting processes until sufficient free contiguous memory is available.

The contiguous memory allocation scheme can be implemented in operating systems with the help of two registers, known as the base and limit registers. When a process is executing in main memory, its base register contains the starting address of the memory location where the process is executing, while the amount of bytes consumed by the process is stored in the limit register. A process does not directly refer to the actual address for a corresponding memory location. Instead, it uses a relative address with respect to its base register. All addresses referred by a program are considered as virtual addresses. The CPU generates the logical or virtual address, which is converted into an actual address with the help of the memory management unit (MMU). The base address register is used for address translation by the MMU. Thus, a physical address is calculated as follows:

Physical Address = Base register address + Logical address/Virtual address

The address of any memory location referenced by a process is checked to ensure that it does not refer to an address of a neighboring process. This processing security is handled by the underlying operating system.

One disadvantage of contiguous memory allocation is that the degree of multiprogramming is reduced due to processes waiting for free memory.

Non-contiguous memory allocation:

Noncontiguous memory allocation assigns the separate memory blocks at a different location in memory space in a nonconsecutive manner to a process requesting for memory. The noncontiguous memory allocation also reduces the memory wastage caused due to internal and external fragmentation. As it utilizes the memory holes, created during internal and external fragmentation.

Types of memory allocation:

1. Best fit memory allocation: In this method, the operating system first searches the whole of the memory according to the size of the given job and allocates it to the closest-fitting free partition in the memory, making it able to use memory efficiently. Here the jobs are in the order from smallest job to largest job.
2. Worst fit memory allocation: Worst Fit allocates a process to the partition which is largest sufficient among the freely available partitions available in the main memory. If a large process comes at a later stage, then memory will not have space to accommodate it.
3. first fit memory allocation: This method keeps the free/busy list of jobs organized by memory location, low-ordered to high-ordered memory. In this method, first job claims the first available memory with space more than or equal to it's size. The operating system doesn't search for appropriate partition but just allocate the job to the nearest memory partition available with sufficient size.

5. Define Operating System. Explain various types of Operating System.

Ans: An Operating System (OS) is an interface between a computer user and computer hardware. An operating system is a software which performs all the basic tasks like file management, memory management, process management, handling input and output, and controlling peripheral devices such as disk drives and printers.

Types of Operating Systems: Some of the widely used operating systems are as follows-

Batch Operating System –

This type of operating system does not interact with the computer directly. There is an operator which takes similar jobs having same requirement and group them into batches. It is the responsibility of operator to sort the jobs with similar needs.

Disadvantages of Batch Operating System:

- The computer operators should be well known with batch systems
- Batch systems are hard to debug
- It is sometime costly
- The other jobs will have to wait for an unknown time if any job fails

Examples of Batch based Operating System: Payroll System, Bank Statements etc.

Time-Sharing Operating Systems –

Each task is given some time to execute, so that all the tasks work smoothly. Each user gets time of CPU as they use single system. These systems are also known as Multitasking Systems.

Disadvantages of Time-Sharing OS:

- Reliability problem
- One must have to take care of security and integrity of user programs and data
- Data communication problem

Examples of Time-Sharing OSs are: Multics, Unix etc.

Distributed Operating System –
Various autonomous interconnected computers communicate each other using a shared communication network. Independent systems possess their own memory unit and CPU. These are referred as distributed systems. These system's processors differ in size and function. The major benefit of working with these types of operating system is that it is always possible that one user can access the files or software which are not actually present on his system but on some other system connected within this network.
Disadvantages of Distributed Operating System:

- Failure of the main network will stop the entire communication
- To establish distributed systems the language which are used are not well defined yet
- These types of systems are not readily available as they are very expensive.

Examples of Distributed Operating System are- LOCUS etc.
Network Operating System –
These systems run on a server and provide the capability to manage data, users, groups, security, applications, and other networking functions. These type of operating systems allow shared access of files, printers, security, applications, and other networking functions over a small private network..
Disadvantages of Network Operating System:

- Servers are costly
- User has to depend on central location for most operations
- Maintenance and updates are required regularly

Examples of Network Operating System are: Microsoft Windows Server 2003, Microsoft Windows Server 2008, UNIX, Linux, Mac OS X, Novell NetWare, and BSD etc.
Real-Time Operating System –
These types of OSs serves the real-time systems. The time interval required to process and respond to inputs is very small. This time interval is called response time.
Disadvantages of RTOS:

- Limited Tasks: Very few tasks run at the same time and their concentration is very less on few applications to avoid errors.
- Use heavy system resources: Sometimes the system resources are not so good and they are expensive as well.
- Complex Algorithms: The algorithms are very complex and difficult for the designer to write on.
- Device driver and interrupt signals: It needs specific device drivers and interrupt signals to response earliest to interrupts.
- Thread Priority: It is not good to set thread priority as these systems are very less prone to switching tasks.

Examples of Network Operating System are missile systems, air traffic control systems, robots etc.

6. Describe different phases of Compiler.

Ans: The compilation process is a sequence of various phases. Each phase takes input from its previous stage, has its own representation of source program, and feeds its output to the next phase of the compiler. Different phases of a compiler are

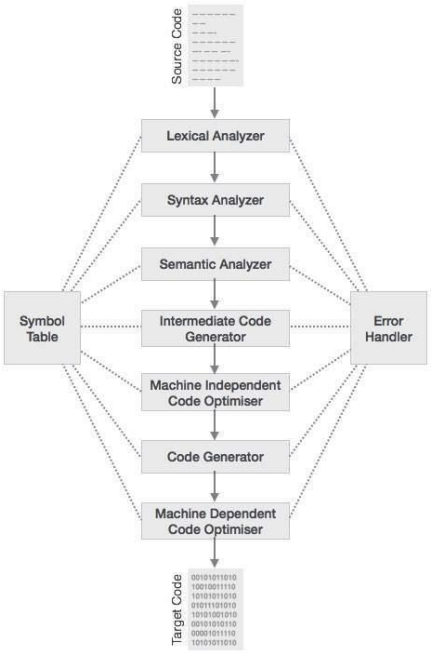
Lexical Analysis

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.

7. Describe the device management techniques.

Ans: