OSSP-S-2017

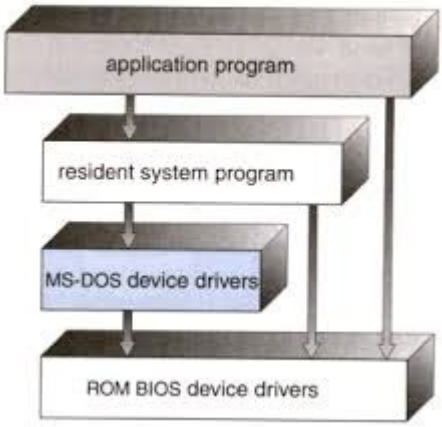
1. Define operating system

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.

a. Explain structure of os with diagram

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.

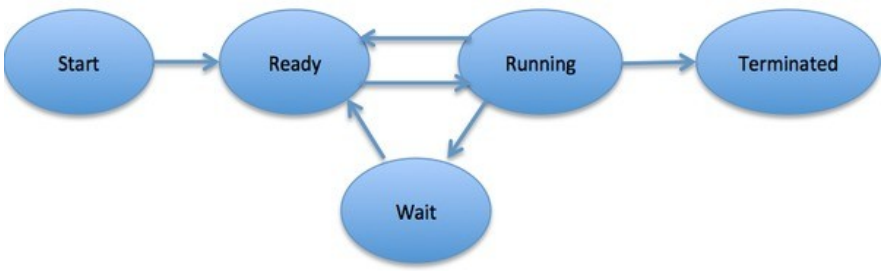
If an error is found during the debugging of a particular layer, the error must be on that layer because the layers below it are already debugged. Thus the design & implementation of the system are simplified when the system is broken down into layers.

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.

b. What is process? Explain process state diagram

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.

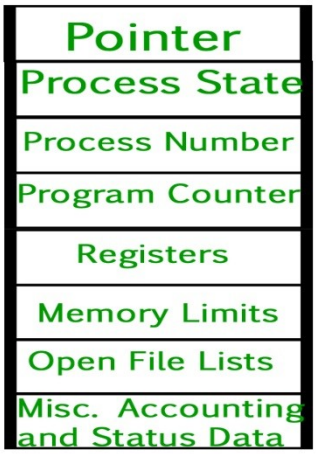
2. Define semaphore

a. 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. Define PCB. Explain different field of PCB.

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

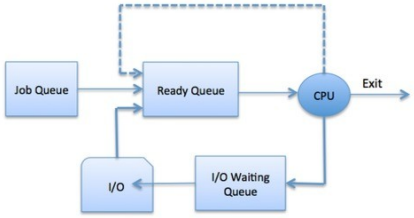
- Pointer – It is a stack pointer which is required to be saved when the process is switched from one state to another to retain the current position of the process.
- Process state – It stores the respective state of the process.
- Process number – Every process is assigned with a unique id known as 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. State process scheduling. Explain types of scheduler.

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.

3. State context switching

Ans: Context Switching involves storing the context or state of a process so that it can be reloaded when required and execution can be resumed from the same point as earlier

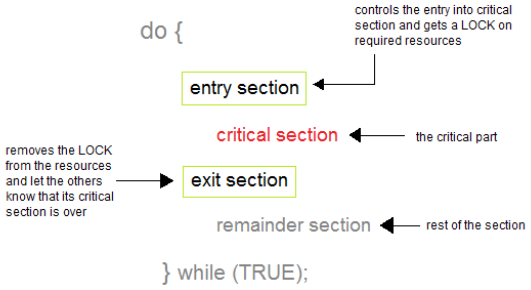
a. What is process synchronization? Explain

Ans:

Process Synchronization is the task of coordinating the execution of processes in a way that no two processes can have access to the same shared data and resources.

It is specially needed in a multi-process system when multiple processes are running together, and more than one processes try to gain access to the same shared resource or data at the same time.

This can lead to the inconsistency of shared data. So the change made by one process not necessarily reflected when other processes accessed the same shared data. To avoid this type of inconsistency of data, the processes need to be synchronized with each other.



To synchronize processes, the process is logically devided into four segments: Entry section, critical section, exit section and remainder section. A critical section is a segment of code which can be accessed by a signal process at a specific point of time. The section consists of shared data resources that required to be accessed by other processes.

- The entry to the critical section is handled by the wait() function, and it is represented as P().
- The exit from a critical section is controlled by the signal() function, represented as V().

In the critical section, only a single process can be executed. Other processes, waiting to execute their critical section, need to wait until the current process completes its execution.

The critical section need to must enforce all three rules:

- Mutual Exclusion: Mutual Exclusion is a special type of binary semaphore which is used for controlling access to the shared resource. It includes a priority inheritance mechanism to avoid extended priority inversion problems. Not more than one process can execute in its critical section at one time.
- Progress: This solution is used when no one is in the critical section, and someone wants in. Then those processes not in their reminder section should decide who should go in, in a finite time.
- Bound Waiting: When a process makes a request for getting into critical section, there is a specific limit about number of processes can get into their critical section. So, when the limit is reached, the system must allow request to the process to get into its critical section.

So binary semaphore , mutex , monitor are used to achive process synchronization among different process.

b. State and explain job scheduling.

Ans: Job scheduling is the process of allocating system resources to many different tasks by an operating system. 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.

Some job scheduling algorithms are: First Come First Serve (FCFS), Shortest-Job-First (SJF) Scheduling, Priority Scheduling, Round Robin Scheduling.

First Come First Serve (FCFS):

- The process which requests the CPU first gets the CPU allocation first.
- It offers non-preemptive and pre-emptive scheduling algorithm.
- Jobs are always executed on a first-come, first-serve basis
- It is easy to implement and use.
- However, this method is poor in performance, and the general wait time is quite high.

Shortest-Job-First (SJF) Scheduling

- Here he process with the shortest execution time should be selected for execution next
- It is associated with each job as a unit of time to complete.
- In this method, when the CPU is available, the next process or job with the shortest completion time will be executed first.
- It is Implemented with non-preemptive policy.
- This algorithm method is useful for batch-type processing, where waiting for jobs to complete is not critical.
- It improves job output by offering shorter jobs, which should be executed first, which mostly have a shorter turnaround time.

Priority Scheduling:

- Priority scheduling is a method of scheduling processes based on priority.
- In this method, the scheduler selects the tasks to work as per the priority.
- Priority scheduling also helps OS to involve priority assignments.
- The processes with higher priority should be carried out first, whereas jobs with equal priorities are carried out on a round-robin or FCFS basis.
- Priority can be decided based on memory requirements, time requirements, etc.

Round Robin Scheduling:

- Here each person gets an equal share of something in turn.
- It is mostly used for scheduling algorithms in multitasking.
- This algorithm method helps for starvation free execution of processes.
- Round robin is a hybrid model which is clock-driven
- Time slice should be minimum, which is assigned for a specific task to be processed. However, it may vary for different processes.
- It is a real time system which responds to the event within a specific time limit.

4. State principle of concurrency.

Ans: Concurrency is the execution of several instruction sequences at the same time (Many threads running in Parallel). In an operating system, this happens when there are several process threads or functions running in parallel.

a. What are interacting process? Explain process synchronization.

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.

Process Synchronization is the task of coordinating the execution of processes in a way that no two processes can have access to the same shared data and resources.

It is specially needed in a multi-process system when multiple processes are running together, and more than one processes try to gain access to the same shared resource or data at the same time.

This can lead to the inconsistency of shared data. So the change made by one process not necessarily reflected when other processes accessed the same shared data. To avoid this type of inconsistency of data, the processes need to be synchronized with each other.

To synchronize processes, the process is logically devided into four segments: Entry section, critical section, exit section and remainder section. A critical section is a segment of code which can be accessed by a signal process at a specific point of time. The section consists of shared data resources that required to be accessed by other processes.

- The entry to the critical section is handled by the wait() function, and it is represented as P().
- The exit from a critical section is controlled by the signal() function, represented as V().

In the critical section, only a single process can be executed. Other processes, waiting to execute their critical section, need to wait until the current process completes its execution.

The critical section need to must enforce all three rules:

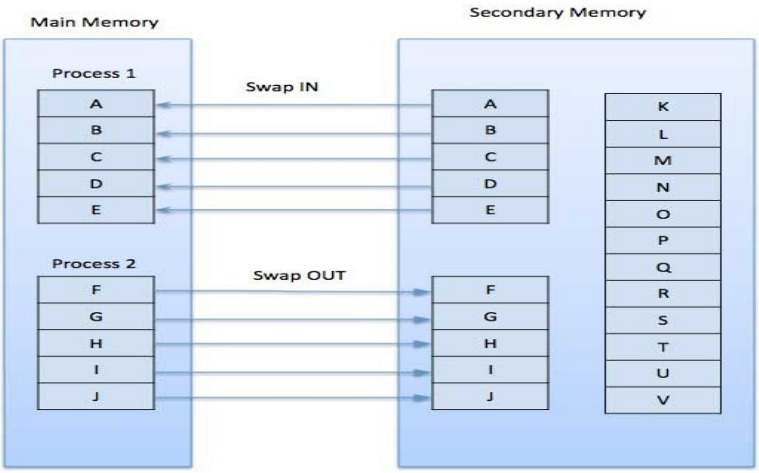
- Mutual Exclusion: Mutual Exclusion is a special type of binary semaphore which is used for controlling access to the shared resource. It includes a priority inheritance mechanism to avoid extended priority inversion problems. Not more than one process can execute in its critical section at one time.
- Progress: This solution is used when no one is in the critical section, and someone wants in. Then those processes not in their reminder section should decide who should go in, in a finite time.
- Bound Waiting: When a process makes a request for getting into critical section, there is a specific limit about number of processes can get into their critical section. So, when the limit is reached, the system must allow request to the process to get into its critical section.

So binary semaphore , mutex , monitor are used to achive process synchronization among different process.

b. Define page? Explain demand paging.

Ans: A page, memory page, or virtual page is a fixed-length contiguous block of virtual memory, described by a single entry in the page table. It is the smallest unit of data for memory management in a virtual memory operating system.

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.

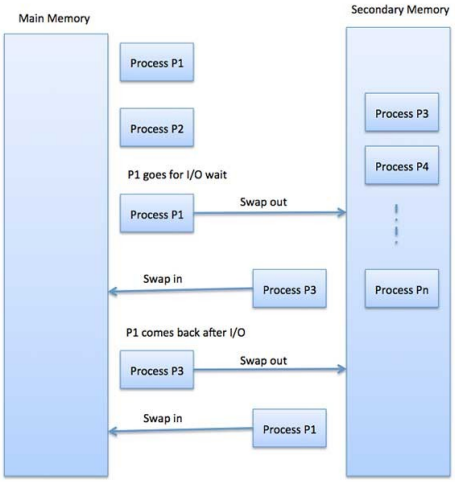
5. Name two os.

Ans: windows, linux

a. What is swapping? Explain swapping with 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

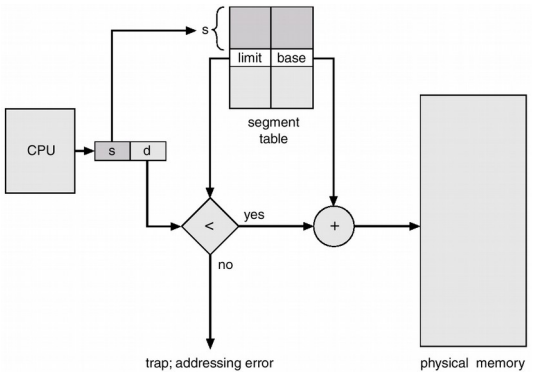
Here, are major benefits/pros of swapping:

- 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.

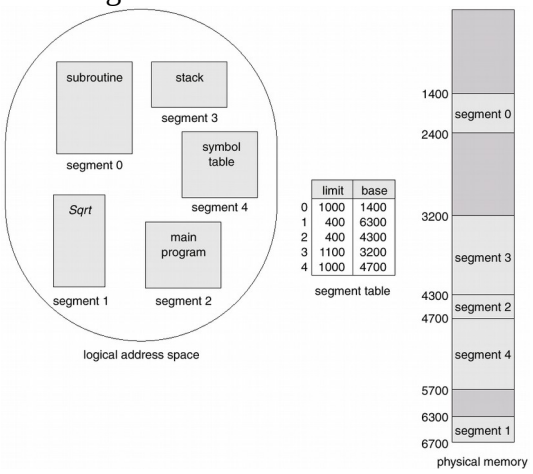
b. Define virtual memory. Explain how segmentation uses the segmentation.

Ans: Virtual memory is a memory management capability of an operating system (OS) that 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.
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.



6. Define 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.

a. Define device management. Explain different techniques of device management.

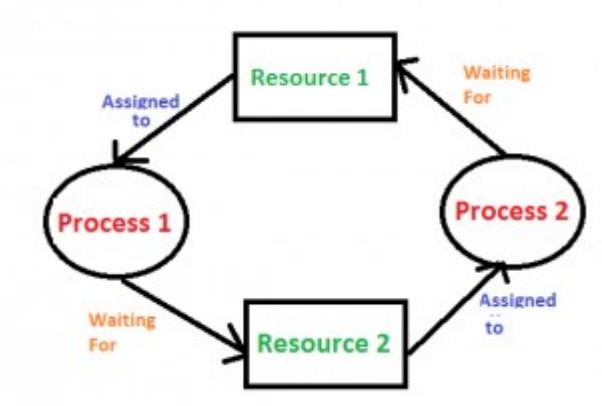
Ans: Device management is the process of managing the implementation, operation and maintenance of a physical and/or virtual device.

b. Explain how deadlock can be 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 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.

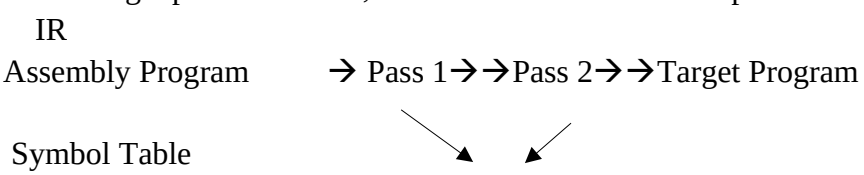
Circular Wait: Let R = {R1, R2, …….Rn}be the set of resource types. We assign to each resource type a unique integer number, which allows us to compare two resources and to determine whether one precedes another in our ordering. This can be ensure that this condition never holds by ordering of all resource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

7. Define system programming.

Ans: System programming involves designing and writing computer programs that allow the computer hardware to interface with the programmer and the user, leading to the effective execution of application software on the computer system.

a. What is assembler? Explain function carried out by assembler.

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 the function of the compiler? Explain the seven phase of the compiler.

Ans: Different phases of a compiler are

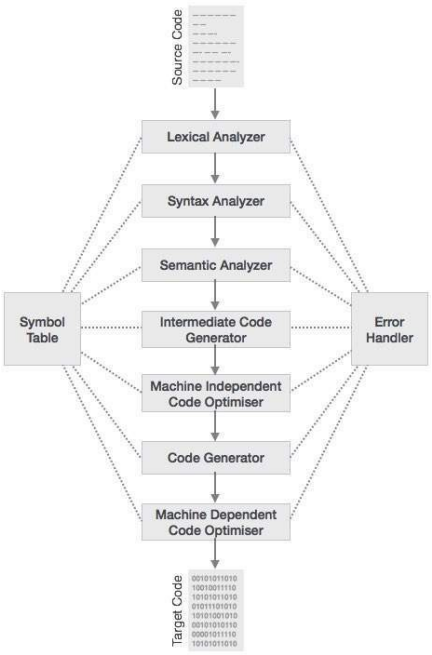
Lexical Analysis

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.