Memory Management

Memory management refers to management of Primary Memory or Main Memory. Main memory is a large array of words or bytes where each word or byte has its own address.

Main memory provides a fast storage that can be accessed directly by the CPU. For a program to be executed, it must in the main memory. 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.

Process
A process is basically a program in execution. The execution of a process must progress in a sequential fashion.

A process is defined as an entity which represents the basic unit of work to be implemented in the system.
To put it in simple terms, we write our computer programs in a text file and when we execute this program, it becomes a process which performs all the tasks mentioned in the program.

When a program is loaded into the memory and it becomes a process, it can be divided into four sections ─ stack, heap, text and data. The following image shows a simplified layout of a process inside main memory −

Process Components
S.N. Component & Description
1
Stack

The process Stack contains the temporary data such as method/function parameters, return address and local variables.

2
Heap

This is dynamically allocated memory to a process during its run time.

3
Text

This includes the current activity represented by the value of Program Counter and the contents of the processor's registers.

4
Data

This section contains the global and static variables.

Program
A program is a piece of code which may be a single line or millions of lines. A computer program is usually written by a computer programmer in a programming language. For example, here is a simple program written in C programming language –

```
#include <stdio.h>

int main() {
   printf("Hello, World! \n");
   return 0;
}
```

A computer program is a collection of instructions that performs a specific task when executed by a computer. When we compare a program with a process, we can conclude that a process is a dynamic instance of a computer program.

A part of a computer program that performs a well-defined task is known as an algorithm. A collection of computer programs, libraries and related data are referred to as a software.

Process Life Cycle
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.

S.N. State & Description
1
Start

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

2
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.

3
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.

4
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.

5
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.

File
A file is a named collection of related information that is recorded on secondary storage such as magnetic disks, magnetic tapes and optical disks. In general, a file is a sequence of bits, bytes, lines or records whose meaning is defined by the files creator and user.

File Structure
A File Structure should be according to a required format that the operating system can understand.

A file has a certain defined structure according to its type.

A text file is a sequence of characters organized into lines.

A source file is a sequence of procedures and functions.

An object file is a sequence of bytes organized into blocks that are understandable by the machine.

When operating system defines different file structures, it also contains the code to support these file structure. Unix, MS-DOS support minimum number of file structure.

File Type
File type refers to the ability of the operating system to distinguish different types of file such as text files source files and binary files etc. Many operating systems support many types of files. Operating system like MS-DOS and UNIX have the following types of files −

Ordinary files
These are the files that contain user information.
These may have text, databases or executable program.

The user can apply various operations on such files like add, modify, delete or even remove the entire file.
Directory files
These files contain list of file names and other information related to these files.
Special files
These files are also known as device files.
These files represent physical device like disks, terminals, printers, networks, tape drive etc.
These files are of two types –

Character special files – data is handled character by character as in case of terminals or printers.

Block special files – data is handled in blocks as in the case of disks and tapes.

File Access Mechanisms
File access mechanism refers to the manner in which the records of a file may be accessed. There are several ways to access files –

Sequential access
Direct/Random access
Indexed sequential access
Sequential access
A sequential access is that in which the records are accessed in some sequence, i.e., the information in the file is processed in order, one record after the other. This access method is the most primitive one. Example: Compilers usually access files in this fashion.

Direct/Random access
Random access file organization provides, accessing the records directly.

Each record has its own address on the file with by the help of which it can be directly accessed for reading or writing.

The records need not be in any sequence within the file and they need not be in adjacent locations on the storage medium.

Indexed sequential access
This mechanism is built up on base of sequential access.
An index is created for each file which contains pointers to various blocks.
Index is searched sequentially and its pointer is used to access the file directly.
Space Allocation
Files are allocated disk spaces by operating system. Operating systems deploy following three main ways to allocate disk space to files.

Contiguous Allocation
Linked Allocation
Indexed Allocation
Contiguous Allocation
Each file occupies a contiguous address space on disk.

Assigned disk address is in linear order.
Easy to implement.
External fragmentation is a major issue with this type of allocation technique.
Linked Allocation
Each file carries a list of links to disk blocks.
Directory contains link / pointer to first block of a file.
No external fragmentation
Effectively used in sequential access file.
Inefficient in case of direct access file.
Indexed Allocation
Provides solutions to problems of contiguous and linked allocation.
A index block is created having all pointers to files.
Each file has its own index block which stores the addresses of disk space occupied by the file.
Directory contains the addresses of index blocks of files.