1. Introduction

Q.3.3.1.1 State different components of an operating system (OS).
Answer: OS is a system software, closely work with the hardware of the machine. It has important components, such as process management, file management, I/O device management, memory management, protection, network management, network services (distributed computing) and user interface, so that a user can work efficiently and conveniently.

Q.3.3.1.2 List common services provided by all operating systems.
Answer: A partial list of common services provided by an operating system are user interfacing, program development, program execution, controlled access to files, input / output operations, communications, resource allocation, error detection, accounting, security and protection. An OS creates a working environment for the users.

Q.3.3.1.3 Describe the nature of the following operating systems: batch operating system, distributed operating system and real time operating system.
Answer: *Batch operating system*: It generally requires the program, data and appropriate system commands to be submitted together in the form of a job. It usually allows little or no modification between user and the executing programs. It has a greater potential for resource utilization than the simple serial processing in computer system, serving multiple user programs that do not require interaction. Program with long execution time fits well in a batch system. Some applications suach as payroll and bank transaction processing can fit well under a batch operating system.
Distributed operating system: Network operating system is a good example of a distributed operating system. Networking provides communication paths between two or more systems. It depends on networking for the functions which have to be performed in a distributed mode. This is also used to share communication task, and provides a rich feature to the user network that varies upon the protocols used. The most commonly used protocol is TCP.
Real-time operating system: These are used in the environment where large number of events, mostly external to the computer system, must be accepted and processed in a short time or within certain deadlines.

Such applications include industrial controls telephone switching equipments, light control and real-time simulations. These systems are also frequently used in military applications and air crafts.

Q.3.3.1.4 What is multi-programming? How does it differ from multitasking?
Answer: A modern operating system allows to run multiple processes concurrently. When a process performs I/O operations, the next process gets an opportunity to be executed by the CPU. The ability to execute multiple processes simultaneouly is called multi-programming. The operating system manages all the processes effectively and efficiently. These processes are also known as jobs, and are kept in a job pool. The job pool consists of all those processes awaiting for allocation of main memory and CPU. CPU is given one job out of all these waiting jobs, and the job is brought to the main memory for execution. The processor executes a job until it is interrupted by some external factor or it goes for an I/O task.
Multi-tasking refers to execution of multiple tasks, such as playing MP3 music, editing documents in Microsoft Word and surfing internet using Google Chrome, simultaneously. Each task is given a CPU time slice using a logical procedure, such as round robin algorithm. Multitasking is a logical extension of multi programming. The way in which multitasking differs from multi programming is that multiprogramming works solely on the concept of context switching, whereas multitasking is based on time sharing alongwith the concept of context switching.

Q.3.3.1.5 State the advantages of multiprogramming.
Answer: Multiprogramming has a number of advantages as stated in the following paragraphs:
Increased CPU utilization: Multiprogramming improves CPU utilization as it organizes a number of jobs, where CPU always has one to execute.
Increased throughput: Throughput means total number of programs executed over a fixed period of time. In multiprogramming, CPU can execute anothe job, when the current job is engaged in I/O operations. It results in an increased throughput.
Shorter turn around time: Turnaround time for short jobs is improved greatly in multiprogramming.
Improved memory utilization: In multiprogramming, more than one program resides in main memory. Thus, memory is utilized optimally.
Increased resources utilization: Multiple programs are actively compet-

ing for resources. Thus, higher degree of resource utilization ac be achieved.
Multiple users: Multiprogramming supports multiple users at a time.

Q.3.3.1.6 Operating system works as a resource manager - Comment on it.
Answer: A computer system has hardware resources such as processor, memory, I/O controllers, timers, disk devices, mice, network interfaces, and printers; and software resources such as files and databases. An important job of the operating system is to provide for an orderly and controlled allocation of theses resources. When a computer (or network) has multiple users, the need for managing and protecting these resources become an important issue. Often, two or more users need to access the same resource simultaneously. It is the job of the operating system to manage such requests efficiently.
Resource management includes sharing resources in two different ways: giving access to the resource by a small amount of time and sharing a part of the resource. For example, single CPU could be allocated to each precess for a small quantum of time, where as entire memory could be shared among various processes.

Q.3.3.1.7 Discuss the advantages of a multiprocessor system.
Answer: A multiprocessor system offers many advantages:
i. *Increased throughput*: By increasing the number of processors, we hope to get more work done in less time.
ii. *Economy of scale*: Multiprocessor systems can save more money than multiple signal processor systems, since they share peripherals, mass storage and power suppliers.
iii. *Increased reliability*: Work can be distributed among several processors. The failure of one processor does not halt the system.

2. Preliminary Concepts

Q.3.3.2.1 Elaborate the following services offered by an OS: error detection, protection and security, accounting.
Answer: An OS is always vigilant of possible errors. It takes an appropriate action to ensure the correctness and consistency of current computation. An error may occur from any part of the system. It could be a hardware error or a software error. Examples of hardware error are keyboard problem, mouse problem, memory problem and power cord problem. Software errors include communication error, calculation error, control flow error and functionality error. Debugging facilities help user to rectify certain types of error.
In a multiuser or networked computer system, there are several pieces of information created by different people. Multiple processes may get executed concurrently. All the system resources are needed to have access in a controlled manner. The protecttion ensures such controlled accces. The security feature allows each user to authenticate himself or herself to the system by means of user identification and password. It may involve several levels of security for critical applications.
System keeps track of different resources allocated to different users. In an application, a user may be billed based on the duration and usage of different resources. OS maintains various information of a user, when the user enters into the system. Accounting information is helpful to reconfigure the system to improve overall performance.

Q.3.3.2.2 Draw a neat diagram of client-server model of an operating system.
Answer: There are two modes of operation in an operating system: user mode and kernel mode. In the user mode, an application or a service runs.It concerns with the actual interface between the user and the system. In the kernel mode, functions such as accessing system resources, controlling hardware functions and processing program instructions are done at the background. The kernel forms the core of an operating system. It controls everything that happens in the computer. In the client-server model of an operating system, the user mode is considered as a client. The user mode accesses resources provided by the kernel mode. Thus, kernel mode acts as the server. Figure 2.1 shows the details of an operating system using the ideas of user mode and kernel

mode.

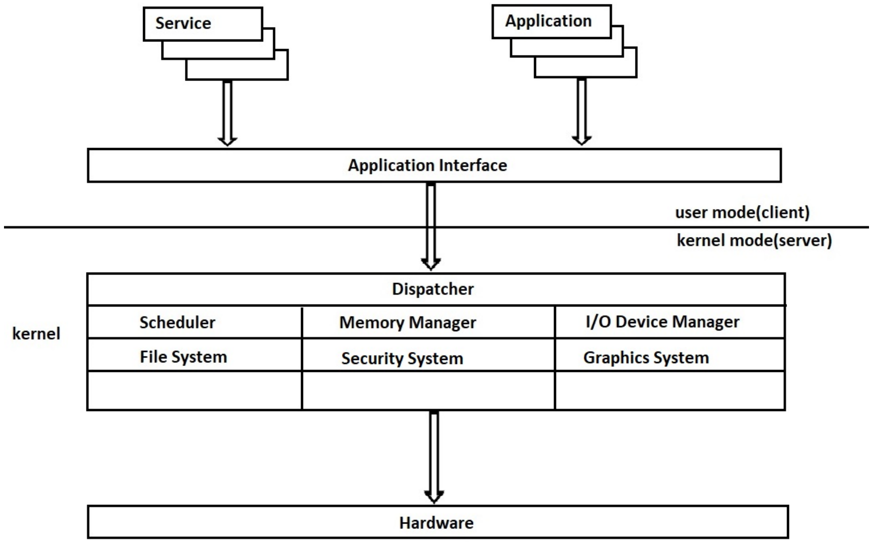

Figure 2.1: Structure of an operating system using client-server model

Q.3.3.2.3 State the functions of a command interpreter. Give examples of command interpreter. State the approaches for implementing a command interpreter.
Answer: The functions of a command interpreter are to read a text command from the user and execute the command.
Some examples of command interpreter are given below:
i. The Bourne shell was the default shell for Version 7 Unix.
ii. Bash is a Unix shell and command language written by Brian Fox for the GNU Project.
iii. The C shell is a Unix shell created by Bill Joy.
iv. CMD.EXE is the default command-line interpreter of the Windows NT-family.
v. Windows PowerShell is a command processor based on .NET Framework.
vi. COMMAND.COM is the default command-line interpreter for DOS.
viii. CMD.EXE is the default command-line interpreter for OS/2.
ix. DROS is a Java ME platform based DOS-like shell for smartphones.
x. Macintosh Programmer's Workshop (MPW) is a software development environment for the Classic Mac OS operating system, written by

Apple Computer.
There are two major approaches for implementing a command interpreter:
i. The command interpreter itself contains the code to execute the command. For example, a command to delete a file may cause the command interpreter to jump to a section of its code that sets up the parameters and makes the appropriate system call.
ii. The command interpreter does not understand the command in any way. It merely uses the command to identify a file to be loaded into memory and to be executed. In Unix, to copy the file "file1.txt" from the currently selected directory to the "newdir" directory, we type: *cp file1.txt newdir*. The command interpreter would search for a file called *cp*, load the file into memory, and execute it with the parameters $file1.txt$ and $newdir$.

Q.3.3.2.4 What is kernel?
Answer: The kernel is a computer program that forms the core of an operating system. It has complete control over everything in the system. During start-up of the computer, it is one of the first programs loaded. It then handles the rest of start-ups. Kernel handles memory devices and peripheral devices such as keyboard, monitor, printer, and speaker. Kernel connects the application software to the hardware of a computer.

Q.3.3.2.5 If you are designing an operating system (OS) using objected oriented programming technique, then state the design structure of your OS.
Answer: For designing an operating system using objected oriented programming technique, we create a modular kernel. Kernel is considered as a set of core components and links to additional services either during boot time or during run time. It allows dynamically loadable modules. It is common in modern implementation of operating system.
Cetral module: Core kernel
Connecting modules: device and bus drivers, executable formats, file systems, loadable system calls, scheduling classes, stream modules, etc.
Each of these connecting modules has a direct link with the core kernel. Overall, it represents a star-like structure with core kernel being at the center.

3. Processes

Q.3.3.3.1 What are the tasks involved in process management?
Answer: The task of process management is performed by the OS. In this regard, it performs
i. scheduling of processes and threads on processors;
ii. creating / deleting processes;
iii. starting, suspending and resuming processes;
iv. providing mechanisms for process synchronization and communication.

Q.3.3.3.2 State the difference between a program and a process.
Answer: A few differentiating points are given below:
i. A program is a sets of instructions to complete a task. A program in execution is known as a process.
ii. A program is a passive entity. But, a process is an active entity.
iii. A program is stored in the secondary storage as a file. A process is kept in the main memory. It also holds other resources such as CPU, I/O device and disk.

Q.3.3.3.3 How do you describe a process?
Answer: A process is described by many elements, including the following:
Accounting information: Different accounting information such as processor time and clock time are maintained.
Context data: They refer to data that are present in registers with in the processor when the process is in running state.
ID: Each process is given a unique identifier to distinguish it from other processes.
I/O status: A process may have many outstanding I/O requests associated with files and I/O devices.
Memory pointers: There may be many pointers associated with a process. It includes pointer to the program code, data associated with this process, any memory block to be shared with other processes.
Priority: All the processes may not have the same pririty. High priority processes are executed early by the CPU.
Program counter: The address of the next instruction to be executed.
State: A process has a number of states. For example, an executing

process is in the *running* state.

Q.3.3.3.4 With the help of a diagram, explain the states of a process.
Answer: The state diagram of a process is depicted in Fig. 3.1.

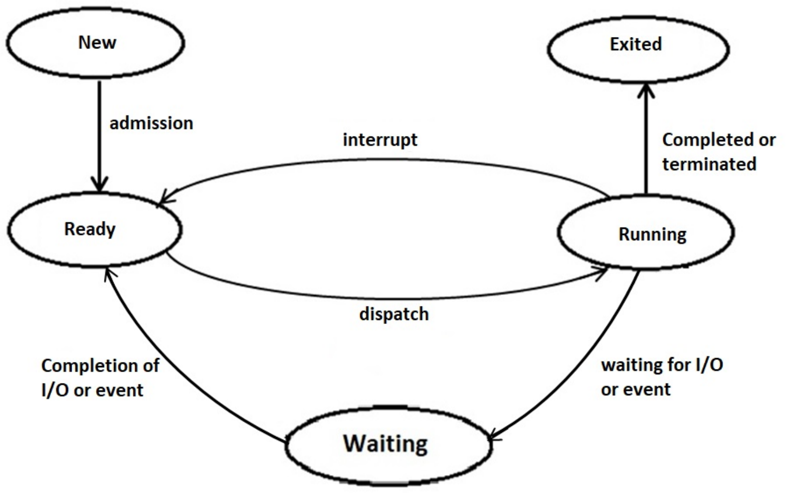

Figure 3.1: Process state diagram with five states

When a process executes, it changes the state. The state of a process is defined by the current activity of the process and each process goes through the following states.
New: The process has been created.
Running: Instructions are being executed.
Waiting: In this state process is waiting for some event, such as I/O completion, to occur.
Ready: The process is waiting to be assigned to a processor.
Exited: The process has finished execution.

Q.3.3.3.5 State whether the statement true or false:
Each process is represented in the operating system by a process control block.
Answer: True

Q.3.3.3.6 What is context switching?
Answer: Context switching involves storing the context or state of executing process, so that it can be reloaded when required. Then the

execution can be resumed from the same point as earlier. This is an important feature of a multi-tasking OS. It allows a single CPU to be shared by multiple processes. The main purpose of this switching is to allocate the old process into the memory, and then switch to a new process for execution.

Q.3.3.3.7 State a few reasons of a process termination.
Answer: A process may get terminated due to various reasons. It includes the following reasons.
∗ Attempting to execute a privileged instruction
∗ Completion of the process
∗ Exceeding the allocated time slice
∗ Failure to perform an I/O operation
∗ ∗ Termination of the parent
∗ Unavailability of required memory demanded at a later stage

Q.3.3.3.8 If we have to design a two-state process model instead of a five-state process model as shown in Fig. 3.1, propose the states and present the model.
Answer: We consider two main states of the process stated as follows: process is running on CPU, and process is not running on CPU. Process state diagram using the above two states is given below:

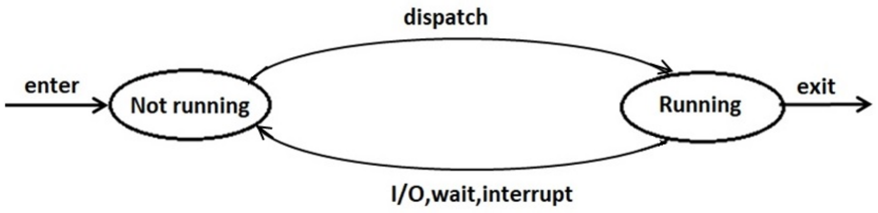

Figure 3.2: Two-state process diagram