# SHELL PROGRAMMING LAB ASSIGNMENT


## VAIBHAV KANSAL

*20214225*


# ASSIGNMENT -A

**1. What are the Functions of operating system?**

**Functions of an Operating System**

**Memory Management**

The operating system manages the Primary Memory or Main Memory. Main memory is made up of a large array of bytes or words where each byte or word is assigned a certain address. Main memory is fast storage and it can be accessed directly by the CPU. For a program to be executed, it should be first loaded in the main memory. An Operating System performs the following activities for Memory Management:

**Processor Management**

In a multi-programming environment, the OS decides the order in which processes have access to the processor, and how much processing time each process has. This function of OS is called Process Scheduling. An Operating System performs the following activities for Processor Management.

Keeps track of the status of processes. The program which performs this task is known as a traffic controller. Allocates the CPU that is a processor to a process. De-allocates processor when a process is no more required.

**Device Management**

An OS manages device communication via its respective drivers. It performs the following activities for device management. Keeps track of all devices connected to the system. designates a program responsible for every device known as the Input/Output controller. Decides which process gets access to a certain device and for how long. Allocates devices effectively and efficiently. Deallocates devices when they are no longer required.

**File Management**

A file system is organized into directories for efficient or easy navigation and usage. These directories may contain other directories and other files. An Operating System carries out the following file management activities. It keeps track of where information is stored, user access settings, the status of every file, and more… These facilities are collectively known as the file system.

**User Interface or Command Interpreter**

The user interacts with the computer system through the operating system.
Hence OS act as an interface between the user and the computer hardware. This user interface is offered through a set of commands or a graphical user interface (GUI). Through this interface, the user makes interaction with the applications and the machine hardware.

**Booting the Computer**

The process of starting or restarting the computer is known as booting. If the computer is switched off completely and if turned on then it is called cold booting. Warm booting is a process of using the operating system to restart the computer.

**Security**

The operating system uses password protection to protect user data and similar other techniques. it also prevents unauthorized access to programs and user data.

**2. What is kernel and it's types?**

Kernel is the main and central component of an OS. It has five types, namely, monolithic kernel, microkernel, hybrid kernel, nano kernel, and exo kernel. The functions of a kernel include accessing computer resources, memory management, device management, and resource management.

**3.What is deadlock, which are the necessary conditions of deadlock?**

A deadlock in OS is a situation in which more than one process is blocked because it is holding a resource and also requires some resource that is acquired by some other process. The four necessary conditions for a deadlock situation to occur are mutual exclusion, hold and wait, no preemption and circular set.

**4. What is semaphore?**

A semaphore is an integer variable, shared among multiple processes. The main aim of using a semaphore is process synchronization and access control for a common resource in a concurrent environment.

**5. What is a thread process and child process?**

The basic difference between a process and a thread is that a process takes place in different memory spaces, whereas a thread executes in the same memory space. Read through this article to find out how a process is different from a thread, in the context of operating systems.

**6. What are the File Locking systems using Semaphore?**

The file that we lock but never store anything in, we call a semaphore file. The way we actually use a semaphore file is by opening it and locking it before we access some other real resource (like a counter file), and then not closing the semaphore file until we're done with the real resource.

**7. Brief description**

**a.** **Function-** A function is a named set of statements that perform a certain task. Functions have unique names. A function can take a set of arguments on which to operate on and return a value that represents the result of the task it has performed.

**b.** **Library Function-** The Bash Shell Function Library (BSFL) is a small Bash script that acts as a library for bash scripts. It provides a couple of functions that makes the lives of most people using shell scripts a bit easier.

**c.** **System Call -** A system call is a request from computer software to an operating system's kernel. The Application Program Interface (API) connects the operating system's functions to user programs. It acts as a link between the operating system and a process, allowing user-level programs to request operating system services**.**

**d.** **Kernal Call-** A system call is implemented by a ``software interrupt'' that transfers control to kernel code; in Linux/i386 this is ``interrupt 0x80''. The specific system call being invoked is stored in the EAX register, abd its arguments are held in the other processor registers.

**e.** **Fork call-** The fork() system call is used in Unix-based OS to create a new process by duplicating the calling process. The new process is an exact copy of the parent process, with its own address space and memory.

**f.** **Procedural Call-** A procedure call is a simple statement made by stating the procedure name, listing actual parameter names or values within parentheses, and adding a final semi-colon.

**g.** **System Program-** System programs communicate and coordinate the activities and functions of hardware and software of a system and also controls the operations of the hardware. An operating system is one of the examples of system software.

**Application Programming Interface(API)-** API stands for Application Programming Interface. In the context of APIs, the word Application refers to any software with a distinct function. Interface can be thought of as a contract of service between two applications. This contract defines how the two communicate with each other using requests and responses.

## ASSIGNMENT -B

**1) What is Shell?**

**Answer:** Shell is a command interpreter, which interprets the command given by the user to the kernel. It can also be defined as an interface between a user and the operating system.

**2) What is Shell Scripting?**

**Answer:** Shell scripting is nothing but a series or sequence of UNIX commands written in a plain text file. Instead of specifying one job/command at a time, in shell scripting, we give a list of UNIX commands like a todo list in a file to execute it.

**3) Shell programs are stored in which file?**

**Answer:** Shell programs are stored in a file called **sh**.

**4) What are the different types of Shells available?**

**Answer:** There are mainly 4 important types of shells that are widely used.

**And they include:**

- Bourne Shell (sh)
- C Shell (csh)
- Korn Shell (ksh)
- Bourne Again Shell (bash)

**5) What are the advantages of C Shell over Bourne Shell**?

**Answer: The advantages of C Shell over Bourne Shell are:**

- C shell allows aliasing of commands i.e. a user can give any name of his choice to the command. This feature is mainly useful when a user has to type the lengthy command again and again. At that point of time, instead of typing a lengthy command a user can type the name that he has given.
    - Shell script takes input from the user, file and displays it on the screen.
    - Shell scripting is very useful in creating your own commands.
    - It is helpful in automating some tasks of the day to day life.
    - It is useful for automating system administration tasks.
    - Mainly it saves timeC shell provides a command history feature. It remembers the previously typed command. Thus, it avoids typing the command again and again.

**6)What is the importance of writing Shell Scripts?**

**Answer: Enlisted below points explain the importance of writing shell scripts. 7) In a typical UNIX environment how many kernels and shells are available?**

- **Answer:** In a typical UNIX environment, only one kernel and many shells are available.

**8) Is separate compiler required for executing a shell program?**

**Answer:** A separate compiler is not required to execute a shell program. The shell itself interprets the command in the shell program and executes them.

**9) How many shell scripts come with UNIX operating system?**

**Answer:** There are approximately 280 shell scripts that come with the UNIX operating system.

**10) When should shell programming/scripting not be used?**

**Answer: Generally, shell programming/scripting should not be used in the below instances.**

- When the task is very much complex like writing the entire payroll processing system.
- Where there is a high degree of productivity required.
- When it needs or involves different software tools.

**11) Basis of shell program relies on what fact?**

**Answer:** The basis of shell programming relies on the fact that the UNIX shell can accept commands not just only from the keyboard but also from a file.

**12) What are the default permissions of a file when it is created?**

**Answer:** 666 i.e. rw-rw-rw- is the default permission of a file, when it is created.

**13) What can be used to modify file permissions?**

**Answer:** File permissions can be modified using **umask**.

**14) How to accomplish any task via shell script?**

**Answer:** Any task can be accomplished via shell script at the dollar ($) prompt and vice versa.

**15) What are Shell Variables?**

**Answer:** Shell variables are the main part of shell programming or scripting. They mainly provide the ability to store and manipulate information within a shell program.

**16) What are the two types of Shell Variables? Explain in brief.**

**Answer:** The two types of shell variables are:

**#1) UNIX Defined Variables or System Variables** – These are standard or shell defined variables. Generally, they are defined in CAPITAL letters.

**Example:** SHELL – This is a Unix Defined or System Variable, which defines the name of the default working shell.

**#2) User Defined Variables** – These are defined by users. Generally, they are defined in lowercase letters

**Example:** $ a=10 –Here the user has defined a variable called ‘a’ and assigned value to it as 10.

**Q #18) How are shell variables stored? Explain with a simple example.**

**Answer:** Shell variables are stored as string variables.

**Example:** $ a=10

In the above statement a=10, the 10 stored in 'a' is not treated as a number, but as a string of characters 1 and 0.

**17) What is the lifespan of a variable inside a shell script**?

**Answer:** The lifespan of a variable inside shell script is only until the end of execution.

**18) How to make variables as unchangeable?**

**Answer:** Variables can be made unchangeable using **readonly**. For instance, if we want variable '**a'** value to remain as **10** and not change, then we can achieve this using **readonly**.

**Example:**
$ a=10
$ readonly a

**19) How variables can be wiped out?**

**Ans:** Variables can be wiped out or erased using the **unset** command.

**Example:**
$ a =20
$ unset a

Upon using the above command the variable '**a**' and its value **20** get erased from shell's memory.

***CAUTION:*** Be careful while using this **unset** command.

**20) What are positional parameters? Explain with an example.**

**Answer:** Positional parameters are the variables defined by a shell. And they are used whenever we need to convey information to the program. And this can be done by specifying arguments at the command line.

There is a total of 9 positional parameters present i.e. from $1 to $9.

**Example:** $ Test Indian IT Industry has grown very much faster In the
above statement, positional parameters are assigned like this.
$0 -> Test (Name of a shell program/script) $1 -
>Indian
$2 -> IT and so on.

**21) What does the. (dot) indicate at the beginning of a file name and how should it be listed?**

**Answer:** A file name that begins with a. (dot) is called as a hidden file. Whenever we try to list the files it will list all the files except hidden files.

But, it will be present in the directory. And to list the hidden file we need to use –a option of ls. i.e. $ ls –a.

**22) Generally, each block in UNIX is how many bytes?**

**Answer:** Each block in UNIX is 1024 bytes.

**23) By default, a new file and a new directory that is being created will have how many links?**

**Answer:** New file contains one link. And a new directory contains two links.

**24) Explain about file permissions.**

**Answer: There are 3 types of file permissions as shown below:**

| Permissions | Weight |
|---|---|
| **r – read** | 4 |
| **w – write** | 2 |
| **x - execute** | 1 |

**25) What is a file system?**

**Answer:** The file system is a collection of files that contain related information of the files.

**26) What are the different blocks of a file system? Explain in brief.**

**Answer: Given below are the main 4 different blocks available on a file system.**

| File System | |
|---|---|
| **Block No.** | Name of the Block |
| **1st Block** | Boot Block |
| **2nd Block** | Super Block |
| **3rd Block** | Inode Table |
| **4th Block** | Data Block |

- **Super Block**: This block mainly tells about a state of the file system like how big it is, maximum how many files can be accommodated, etc.
- **Boot Block**: This represents the beginning of a file system. It contains the bootstrap loader program, which gets executed when we boot the host machine.
- **Inode Table**: As we know all the entities in a UNIX are treated as files. So, the information related to these files is stored in an Inode table.
- **Data Block**: This block contains the actual file contents.

**27) What are the three different security provisions provided by UNIX for a file or data?**

**Answer: Three different security provisions provided by UNIX for a file or data are:**

- It provides a unique user id and password to the user, so that unknown or unauthorized person should not be able to access it.
- At the file level, it provides security by providing read, write & execute permissions for accessing the files.
- Lastly, it provides security using file encryption. This method allows encoding a file in an unreadable format. Even if someone succeeds in opening a file, but they cannot read its contents until and unless it is decrypted

**28) What are the three editors available in almost all the versions of UNIX?**

**Answer:** The three editors are ed, ex & vi.

**29) What are the three modes of operation of vi editor? Explain in brief.**

**Answer:** The three modes of operation of **vi editors** are,

1. **Command Mode**: In this mode, all the keys pressed by a user are interpreted as editor commands.
2. **Insert Mode**: This mode allows for the insertion of a new text and editing of an existing text etc.
3. **The ex-command Mode**: This mode allows a user to enter the commands at a command line.

**30) What is the alternative command available to echo and what does it do?**

**Answer: tput** is an alternative command to **echo**.

Using this, we can control the way in which the output is displayed on the screen.

**31) How to find out the number of arguments passed to the script?**

**Answer:** The number of arguments passed to the script can be found by the below command.

**echo $ #**

**32) What are control instructions and how many types of control instructions are available in ashell? Explain in brief.**

**Answer:** Control Instructions are the ones, which enable us to specify the order in which the various instructions in a program/script are to be executed by the computer. Basically, they determine a flow of control in a program.

**There are 4 types of control instructions that are available in a shell.**

- **Sequence Control Instruction**: This ensures that the instructions are executed in the same order in which they appear in the program.
- **Selection or Decision Control Instruction**: It allows the computer to take the decision as to which instruction is to be executed next.
- **Repetition or Loop Control Instruction**: It helps a computer to execute a group of statements repeatedly.
- **Case-Control Instruction**: This is used when we need to select from several alternatives.

**33) What are Loops and explain three different methods of loops in brief?**

**Answer:** Loops are the ones, which involve repeating some portion of the program/script either a specified number of times or until a particular condition is being satisfied.

**3 methods of loops are:**

- **For Loop:** This is the most commonly used loop. For loop allows specifying a list of values that the control variable in the loop can take. The loop is then executed for each value mentioned in the list.
- **While Loop:** This is used in a program when we want to do something for a fixed number of times. While loop gets executed until it returns a zero value.
- **Until Loop:** This is similar to while loop except that the loop executes until the condition is true. Until the loop gets executed at least once, it returns a non-zero value.

**34) What is IFS?**

**Answer:** IFS stands for Internal Field Separator. And it is one of the system variables. By default, its value is space, tab, and a new line. It signifies that in a line where one field or word ends and another begins. **35) What is a Break statement and what is it used for?**

**Answer:** The break is a keyword and is used whenever we want to jump out of a loop instantly without waiting to get back to the control command.

When the keyword break is encountered inside any loop in the program, control will get passed automatically to the first statement after a loop. A break is generally associated with an if.

**36) What is Continue statement and what is it used for?**

**Answer:** Continue is a keyword and is used whenever we want to take the control to the beginning of the loop, by passing the statements inside the loop which have not yet been executed.

When the keyword Continue is encountered inside any loop in the program, control automatically passes to the beginning of the loop. Continue is generally associated with an if.

**37) What are Metacharacters in a shell? Explain with some examples.**

**Answer:** Metacharacters are special characters in a program or data field which provides information about other characters. They are also called, regular expressions in a shell.

**38) How to execute multiple scripts? Explain with an example.**

**Answer:** In a shell, we can easily execute multiple scripts i.e. one script can be called from the other. We need to mention the name of a script to be called when we want to invoke it.

**Example:** In the below program/script upon executing the first two echo statements of script1, shell script executes script2. Once after executing script2, the control comes back to script1 which executes a **pwd** command and then terminates.

**39) Which command needs to be used to know how long the system has been running?**

**Answer: uptime** command needs to be used to know how long the system has been running.

**Example:** $ uptime

On entering the above command at shell prompt i.e. $ uptime, the output should look like this.

9:21am up 86 day(s), 11:46, 3 users, load average: 2.24, 2.18, 2.16 **40) How**

**to find the current shell which you are using?**

**Answer:** We can find the current shell that we are using with echo $SHELL.

**Example:** $ echo $SHELL

**41) How to find all the available shells in your system?**

**Answer:** We can find all the available shells in our system with $ cat /etc/shells.

**Example:** $ cat /etc/shells

**42) How to read keyboard inputs in shell scripts?**

**Answer:** Keyboard inputs can be read in shell scripts as shown below,

**43) How many fields are present in a crontab file and what does each field specify?**

**Answer:** The **crontab** file has six fields. The first five fields tell **cron** when to execute the command: minute(0-59), hour(0-23), day(1-31), month(1-12), and day of the week(0-6, Sunday = 0).

And the sixth field contains the command to be executed.

**44) What are the two files of crontab command?**

**Answer: Two files of crontab command are**:

- **cron.allow** – It decides which users need to be permitted from using crontab command.
- **cron.deny** – It decides which users need to be prevented from using crontab command.

**45) What command needs to be used to take the backup?**

**Answer: tar** is the command which needs to be used to take the backup. It stands for tape archive. The **tar** command is mainly used to save and restore files to and from an archive medium like tape.

**46) What are the different commands available to check the disk usage?**

**Answer:** There are three different commands available to check the disk usage.

**They are:**

- **df** – This command is used to check the free disk space.
- **du** – This command is used to check the directory wise disk usage.
- **dfspace** – This command is used to check the free disk space in terms of MB.

**47) What are the different communication commands available in Unix/Shell?**

**Answer:** Basically, there are 4 different communication commands available in Unix/Shell. And they are mail, news, wall & motd.

**48) How to find out the total disk space used by a specific user, say for example username is John?**

**Answer:** The total disk space used by John can be found out as:

**du –s/home/John**

**49) What is Shebang in a shell script?**

**Answer:** Shebang is a # sign followed by an exclamation i.e. !. Generally, this can be seen at the beginning or top of the script/program. Usually, a developer uses this to avoid repetitive work. Shebang mainly determines the location of the engine which is to be used in order to execute the script.

Here '#' symbol is called hash and '!' is called a bang.

**Example:** #!/bin/bash

The above line also tells which shell to use.

**50) What is the command to be used to display the shell's environment variables?**

**Answer:** Command to be used to display the shell's environment variables is **env** or **printenv**.