RN SHETTY TRUST®

RNS INSTITUTE OF TECHNOLOGY

Autonomous Institution Affiliated to Visvesvaraya Technological University, Belagavi
Approved by AICTE, New Delhi, Accredited by NAAC with 'A+' Grade
Channasandra, Dr. Vishnuvardhan Road, Bengaluru - 560 098
Ph: (080) 28611880, 28611881 URL: www.rnsit.ac.in

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING (AI & ML)

1	R	N			C	I			

CIE – Test 1

Sem: V **Date:17/10/2025**
Subject: Unix System Programming **Time: 9:30am-11:00am**
Subject Code: BCS515C **Max. Marks: 50**

Instruction: Answer any 5 full questions, selecting ONE question from each part.

Q.No.		Question	Marks	RBT*	COs
1		Explain with a Figure, the kernel and shell relationship in Unix operating system.	10	L2	CO1
		OR			
2		Describe the salient features of the Unix operating system	10	L2	CO1
3		Demonstrate the use of the following Unix commands by providing their syntax, options, and suitable examples **i) echo ii) cal iii) date iv) cat**	10	L3	CO1
		OR			
4		Explain the following commands with example. **i) cp ii) rm iii) mv iv) cat v) wc**	10	L3	CO1
5	a	Define File and discuss valid filenames in Unix?	4	L2	CO2
	b	Explain the different types of files supported in UNIX.	6	L2	
		OR			
6		Which command is used for listing file attributes? Explain the significance of each field in the attributes.	10	L2	CO2
7	a	Explain shell's interpretive life cycle.	4	L2	CO2
	b	What are wild cards characters? Explain each of them with suitable examples.	6	L2	CO2
		OR			
8		Describe the standard files used in UNIX with suitable examples. .	10	L2	CO2
9		What is the purpose of grep? Explain grep command with all options.	10	L2	CO2
		OR			
10		Demonstrate the different ways of setting file permissions in UNIX with suitable examples."	10	L3	CO2

*** Revised Bloom's Taxonomy: L1-Remmber, L2-Understand, L3-Apply, L4-Analyze, L5-Evaluate, L6-Create**

Vision: Empowering AI & ML Engineers to seamlessly integrate society and technology

Course Coordinator **Module Coordinator** **Program Coordinator/HOD**

RN SHETTY TRUST®

RNS INSTITUTE OF TECHNOLOGY

Autonomous Institution Affiliated to Visvesvaraya Technological University, Belagavi
Approved by AICTE, New Delhi, Accredited by NAAC with 'A+' Grade
Channasandra, Dr. Vishnuvardhan Road, Bengaluru - 560 098
Ph: (080) 28611880, 28611881 URL: www.rnsit.ac.in

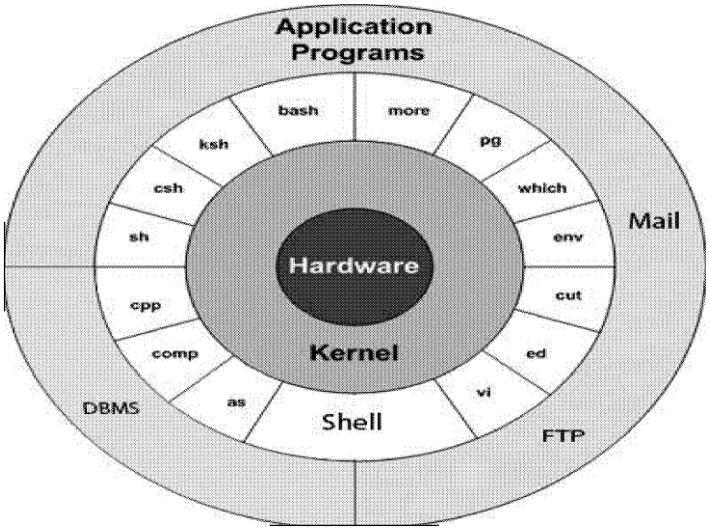
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING (AI & ML)

CIE – TEST I

Course: Unix System Programming	Course Code: BCS515C	Semester : 5
Max. Marks : 50	Date: 17/09/2025	Faculty Name: Dr.Sharvani G S Dr.Usha M Ms.Pooja

Scheme and Solution

1.	**Explain with a Figure, the kernel and shell relationship in Unix operating system.**	**Marks** **10M**



Kernel: is the core of operating system. A collection of routines mostly written in C. It is loaded into memory when the system is booted and communicates directly with the hardware. User programs that need to access the hardware use the services of the kernel which performs the job on the user's behalf. These programs access the kernel through a set of functions called system calls. The kernel manages system memory, processes, decides priorities.

Shell: interface between Kernel and User. It functions as **command interpreter** i,e it receives and interprets the command from user and interacts with the hardware. There is only one kernel running on the system, there could be several shells in action - one for each user who is logged in. When user enters a command, shell thoroughly examines the keyboard input and simplifies to a command line, and communicates with kernel to see that the command is executed.

Files and Process: File is an array of bytes and it contain virtually anything. Unix considers even the directories and devices as members of file system. The dominant file type is text and behavior of system is mainly controlled by text

files.The second entity is the process, which is the name given to a file when it is executed as a program. Process is simply a "time image" of an executable file.

System Calls: Though there are thousands of commands in the Unix system, they all use a handful of functions called system calls. User programs that need to access the hardware use the services of the kernel, which performs the job on user's behalf. These programs access the kernel through a set of functions called system calls.

Ex: open()-- system call to access both file and device. Write()-system call to write a file.

Diagram -2M Explanation 8M

2 **Describe the salient features of the Unix operating system.**

FEATURES OF UNIX

- **Portable:** UNIX can be installed on many hardware platforms. Its widespread use can be traced to the decision to develop it using the C language. Because C programs are easily moved from one hardware environment to another, it is relatively simple to port it to different environments.
- **A Multiuser & Multitasking system:** Multiple users can run separate job. In unix, the resources are actually shared between all users. The computer breaks up a unit n terminate them time expires, the previous job is kept in waiting, & the next user's job. A single user can run multiple job. It's usual for a user to edit a file, print another file, send email etc. without leaving any applications. The kernel is designed to handle a user's multiple needs.
- **The Building-block approach:** Unix comes with hundreds of simple commands which perform one simple job. It's through pipes & filters unix implement small-is-beautiful philosophy. Using this feature, the output of one tool can be used as input of another tool.
- **Unix toolkit:** Unix supplied with a set of applications such as compilers, interpreters, text- manipulation utilities, general purpose tools, & system administration tools.
- **Pattern Matching:** Unix has very strong pattern matching features. The * (known as metacharacter) is a special character used by the system to indicate that it can match a number of filenames.
- **Programming facility:** The Unix shell is a programming language. It has all the necessary features of a language like control structures, loops & variables. This makes it a powerful programming language.
- **Documentation:** The principal online help facility is available is the man command in unix, which remains the most important reference for commands and their configuration files.

Mention and Explaination – 10M

10M

3 **Demonstrate the use of the following Unix commands by providing their syntax, options, and suitable examples** **10M**

i) echo ii) cal iii) date iv) cat

i) echo

- The **echo** command used to display a message.
- To display the diagnostic messages on the terminal or to issue prompts for taking user input (like echo Sun Solaris).To evaluate shell variable (like echo $SHELL)

Syntax: **$echo [Short-Option]... [String]...**

- Orginally, echo was an external command, but nowadays all shells have echo built-in.
- Most of the echo's behavior differences relates to the way echo's interprets certain strings known as escape sequences.

ii) cal

It displays the calendar of any specific month, or a complete year.If a user wants a quick view of calendar in Unix terminal, cal is the command. By default, cal command shows current month calendar as output.

General Syntax: cal [[month] year]

cal : Shows current month calendar on the terminal.

cal 011990 : Shows calendar of selected month and year.

cal 2020 : Shows the whole calendar of the year.

cal -3 : Shows calendar of previous, current and next month

iii) date

It is a command that instructs the machine to display the current date & time. This command does not allow the user to change the date or the time. Only administrator can do that.

date %a	Displays Weekday name in short (like Mon, Tue, Wed)
date +%A	Displays Weekday name in full short (like Monday, Tuesday)
date +%b	Displays Month name in short (like Jan, Feb, Mar)
date +%B	Displays Month name in full short (like January, February)
date +%d	Displays Day of month (e.g., 01)
date +%D	Displays Current Date; shown in MM/DD/YY
date +%F	Displays Date; shown in YYYY-MM-DD
date +%H	Displays hour in (00..23) format
date +%m	Displays month (01..12)

iv) cat

The cat command in UNIX/Linux stands for concatenate. It is one of the most commonly used commands for viewing, creating, and combining files.

Syntax: **cat [options] [file...]**

To create a file **$cat >newfile** This is a new file which contains some text, just to add some contents to the file new [ctrl-d] $

Each Command Explanation - 2.5M

2.5X4 = 10M

4 **Explain the following commands with example.**

i) cp ii) rm iii) mv iv) cat v) wc

10M

i)cp- Copying A File

The cp command copies a file or a group of files. It creates an exact image of the file on the disk with a different name. The syntax takes two filename to be specified in the command line.

Syntax:

cp [option] source
destination cp
[option] source
directory
cp [option] source-I source-2 source-3 source-n directory

1.Two file names: If the command contains two file names, then it copies the contents of 1st file to the 2nd file. If the 2nd file doesn't exist, then first it creates one and content is copied to it. But if it existed then it is simply overwritten without any warning. So be careful when you choose destination file name.

2.One or more arguments: If the command has one or more arguments, specifying file names and following those arguments, an argument specifying directory name then this command copies each source file to the destination directory with the same name, created if not existed but if already existed then it will be overwritten.

ii) rm -Deleting Files

rm stands for **remove** here. rm command is used to remove objects such as files, directories, symbolic links and so on from the file system like UNIX. To be more precise, rm removes references to objects from the filesystem, where those objects might have had multiple references (for example, a file with two different names). By default, it does not remove directories,once you delete the files then you are not able to recover the contents of files and directories.

Syntax: rm [option] [filename]

Options:

- **-i (Interactive Deletion):** Like in the -i option makes the command ask the user for confirmation before removing each file, you have to press **y** for confirm deletion, any other key leaves the file un-deleted.
- **-f (Force Deletion): rm** prompts for confirmation removal if a file is **write protected.** The **-f** option overrides this minor protection and removes the file forcefully. **-f** option of rm command will not work for write-protect directories.
- **-r (Recursive Deletion):** With **-r(or -R)** option rm command performs a tree-

walk and will delete all the files and sub-directories recursively of the parent directory. At each stage it deletes everything it finds. Normally, **rm** wouldn't delete the directories but when used with this option, it will delete.

- **-version:** This option is used to display the version of **rm** which is currently running on your system.

iii) mv -Renaming Files

The mv command renames (moves) files. The main two functions are:

- It renames a file(or directory)
- It moves a group of files to different directory

It doesn't create a copy of the file; it merely renames it. No additional space is consumed on disk during renaming.

Syntax: mv [option] source destination

Ex: To rename the file a.txt as movefile.txt we can use the following command

$ mv a.txt movefile.txt

If the destination file doesn't exist in the current directory, it will be created. Or else it will just rename the specified file in mv command.

iv) cat

The cat command in UNIX/Linux stands for concatenate. It is one of the most commonly used commands for viewing, creating, and combining files.

Syntax: **cat [options] [file...]**

To create a file **$cat >newfile** This is a new file which contains some text, just to add some contents to the file new [ctrl-d] $

v) wc - Counting Lines, Words and Characters

wc command performs Word counting including counting of lines and characters in a specified file. It takes one or more filename as arguments and displays a four columnar output.

Syntax: $ wc filename

- Line: Any group of characters not containing a newline
- Word: group of characters not containing a space, tab or newline
- Character: smallest unit of information, and includes a space, tab and newline

wc offers 3 options to make a specific count. -I option counts only number of lines, - w and-c options count words and characters, respectively.

Each Command Explanation - 2M

2X4 = 10M

5a **Define File and discuss valid filenames in Unix?** **4M**

- A **file** is a collection of related information stored on a disk.
- It can contain text, data, programs, or even device information.

A filename can consist of up to 255 characters. Files may or may not have extensions and can consist of practically any ASCII character except the / and the Null character. Users are permitted to use control characters or other nonprintable characters in a filename. However, user should avoid using these characters while naming a file. It is recommended that only the following characters be used in filenames:

- Alphabets and numerals.
- The period(.), hyphen(-) and underscore(_).

UNIX imposes no restrictions on the extension. In all cases, it is the application that imposes that restriction. Eg. A C-Compiler expects C program filenames to end with .c, Oracle requires SQL scripts to have .sq! extension.

A file can have as many dots embedded in its name. A filename can also begin with or end with a dot. ex: .sample, .one

The file name may consist of ordinary characters, digits and special tokens like the underscore, except the forward slash (/).

It is permitted to use special tokens like the ampersand(&) or spaces in a filename.

UNIX is case sensitive; chap01, Chap01 and CHAP01 are three different filenames that can coexist in the same directory.

File Definition – 1M
Rules for Files Naming – 3 M
Total 4M

5b) **Explain the different types of files supported in UNIX.** **6M**

Ordinary (Regular) File

This is the most common file type. An ordinary file can be either a text file or a binary file. A text file contains only printable characters and user can view and edit them. All C and Java program sources, shell scripts are text files. Every line of a text file is terminated with the newline character.

A binary file, on the other hand, contains both printable and nonprintable characters that cover the entire ASCII range. The object code and executables that you produce by compiling C programs are binary files. Sound and video files are also binary files.

Directory File

A directory contains no data, but keeps details of the files and subdirectories that it contains. A directory file contains one entry for every file and subdirectory that it houses. Each entry has two components namely, the filename and a unique identification number of the file or directory (called the inode number). ls -i is the command to be used to know inode number.

Device File

All the operations on the devices are performed by reading or writing the file representing the device. Device filenames are found in a single directory structure, /dev. A device file is not really a stream of characters. It is the attributes of the file that entirely govern the operation of the device. The kernel identifies a device from its attributes and uses them to operate the device.

Explaining each type – 2M

3X2M=6M

6. **Which command is used for listing file attributes? Explain the significance of each field in the attributes.** **10M**

ls command is used to obtain a list of all filenames in the current directory.

It lists seven attributes of all files in the current directory and they are:

File type and Permissions: The first column shows the type and permissions associated with each file.
Links: The second column indicates the number of links associated with the file.
Ownership: When you create a file, you automatically become its owner. The third column shows the owner of the files. The owner has full authority to tamper with a file's contents and permissions.
Group ownership: When opening a user account, the system administrator also assigns the user to same group. The fourth column represents the group owner of the file.
File size: The fifth column shows the size of the file in bytes i.e the amount of data it contains.
Last modification date and time: The sixth, seventh and eight columns indicate the last modification time of the file, which is stored to the nearest second.
File name: The last column displays the filenames arranged in ASCII collating sequence.

Mention about ls command – 1M
Explaining all 9 columns – 9M
Total 10M

7a) **Explain shell's interpretive life cycle.**

- Shell is the interface between user and the UNIX system.
- Shell is a combination of command interpreter and a programming language.
- When user key in a command, it will be considered as input to the shell. Shell first scans the command line for **metacharacters**(>, |, * etc).
- The shell meta characters include: \ / < > ! $ % ^ & * | { } [ ] " ' ` ~ ;
- Shell performs all actions represented by the symbol before the command can be executed.

4M

$ rm -R *

* metacharacter is not part of rm command, but shell replaces * with all file names in the pwd. rm finally runs with these file names as arguments.

7b) **What are wild cards characters? Explain each of them with suitable examples.** **6M**

A pattern is framed using ordinary characters and a metacharacters (like *,?) using well- defined rules. The pattern can then be used as an argument to the command, and the shell will expand it suitably before the command is executed.

The metacharacters that are used to construct the generalized pattern for matching filenames belong to a category called wild-cards. The following table lists the shell's wild-cards:

Wild-Card	Matches
*	Any number of characters including none
?	A single character
[ijk]	A single character - either an i, j or k
[x-z]	A single character that is within the ASCII range of characters x and x
[!ijkJ	A single character that is not an i,j or k
[!x-z]	A single character that is not within the ASCII range of the characters x and x (Not in C Shell)
{pat! ..1 J	Patl, pat2, etc. (Not in Bourne shell)

Wild Cards Definition – 1M
Explaining different wild cards- 5M
Total 6M

8 **Describe the standard files used in UNIX with suitable examples.** **10M**

The shell associates three files with the terminal - two for display and one for the keyboard. Each stream is associated with a default device:

1) **Standard input**
2) **Standard output**
3) **Standard error**

Normal input for commands cat and wc are disk files.
When no disk file names are passed as an argument to these commands, they read the file representing **standard input.**
Standard input file represents three input sources

- ➢ Keyboard, the default source
- ➢ A file using *redirection* with < symbol (a metacharacter)
- ➢ Another program using a pipeline

i) Keyboard, the default source
$ wc
Standard input
Can be redirected *[Ctrl-d]*
ii) A file using redirection with the < symbol
Shell can reassign the standard input file to a disk file. It can redirect the standard input to

originate from a file on disk. This reassignment or redirection requires the < symbol

```
$ wc < x.c
3  14  71
```

iii) Another program using a pipeline

```
ls *.txt | cat > txtFile
```

Commands displaying output on the terminal, writes to the **standard output** file as a stream of characters, and **not directly** to the terminal.

There are 3 possible destinations of this stream.

- ➢ The terminal, the default destination
- ➢ A file using the redirection symbols > and >>
- ➢ As input to another program using a pipeline

$ ls -l displays file names with long listing option on the output terminal screen

$ wc sample.txt > newfile

Standard errors:

Each of the three standard files is represented by a number called file descriptor. *A* file is opened by referring to its pathname, but subsequent read and write operations identify the file by a unique number called a file descriptor. The kernel maintains a table of file descriptors for every process running in the system. The first three slots are generally allocated to the three standard streams as,

0 - Standard input

1 - Standard output

2 - Standard error

These descriptors are implicitly prefixed to the redirection symbols.

> and l>

< and 0< mean the same

When we enter an incorrect command or when the input file to a command doesn't exist, error messages show up on the screen.

Trying to "cat " a non-existent file produces the error stream.

Ex: $ cat foo

cat: cannot open foo

Cat files to open the file and writes to standard error. If you are not using the c shell you can redirect this stream to a file.

Each type 4M+3M=3M = 10M

9 **What is the purpose of grep? Explain grep command with all options.**

The grep Command: The name "grep" derives from the ed (a UNIX line editor) command g/re/p which means "globally search for a regular expression and print all lines containing it."

It scans the file / input for a pattern and displays lines containing the pattern, the

10M

line numbers or filenames where the pattern occurs. It's a command from a special family in UNIX for handling search requirements.

grep *options patternfilename(s)*

A pattern or regular expression is either some plain text (a word, for example) and/or special characters used for pattern matching.

$grep "sales" empJst

This will display lines containing sales from the file emp.lst. Patterns with and without quotes is possible. It's generally safe to quote the pattern. Quote is mandatory when pattern involves more than one word. It returns the prompt in case the pattern can't be located.

When grep is used with multiple filenames, it displays the filenames along with the output.

$grep "director" empt.1st emp2.lst

Where it shows filename followed by the contents

The simplest use of grep is to look for a pattern consisting of a single word. It can be used in a pipe so that only those lines of the input files containing a given string are sent to the standard output. If you don't give grep a filename to read, it reads its standard input; that's the way all filter programs work:

S ls -1 grep "Aug"

-rw-rw-rw-1 john doc

-rw-rw-rw-1 john doc

-rw-rw-r-- 1 john doc

-rw-rw-r-- I Amrit doc

Option	Significance
-i	Ignores case for matching
-v	Does not display lines matching expression
-n	Displays line numbers along with lines
-c	Displays count of number of occurrences
-l	Displays list of filenames only
-e exp	specifies expression with this option
-E	treats pattern as an extended RE
-x	matches pattern with entire line
-F	matches multiple fixed strings

Examples:

$grep -i 'agarwal' emp.lst

$grep -v 'director' emp.lst > otherlist

Purpose of grep =3M

Explanation of different options – 7M

Total 10M

10 **Demonstrate the different ways of setting file permissions in UNIX with suitable examples.** **10M**

A file or a directory is created with a default set of permissions, which can be determined by umask. Let us assume that the file permission for the created file is -rw-r--r--. Using chmod command, we can change the file permissions and allow the owner to execute his file. The command can be used in two ways:

- In a relative manner by specifying the changes to the current permissions.
- In an absolute manner by specifying the final permissions.

Relative Permissions

Changing permissions in a relative manner, chmod only *changes the permissions specified in the command line* and *leaves the other permissions unchanged*.

Syntax: chmod *category operation permission filenames(s)*

chmod takes as its argument an expression comprising letters and symbols that completely describe the user category and the type of permission begin assigned or removed.

User *category* (user, group, others)

The *operation* to be performed (assign or remove a permission)

The type of *permission* (read, write, execute)

Category	Operation	Permission
u - User g - Group o - Others a - All	+ Assigns permission - Removes permission = Assigns absolute permission	r - Read permission w - Write permission x - Execute permission

$ chmod u+x test

$ ls -l test

-rwxr--r-- 1 kumar …. test

If test is an executable file it can be executed by the owner.

To enable others to execute the file

$ chmod ugo+x test ; ls -l test

String ugo combines all three categories - user, group and others. Synonym for 'ugo' is 'a'.

Other way round if no string is specified then permission is applied for all categories.

$ chmod +x test ; ls -l test

Absolute Permissions

In absolute mode, permission is set through octal numbers.

Octal numbers use the base 8, and octal digits have the values 0 to 7, which means that a set of 3 bits can represent one octal digit.

Binary	Octal	Permission	Significance
000	0	---	No permission
001	1	--x	Executable only
010	2	-w-	Writable only
011	3	-wx	Writable and executable
100	4	r--	Readable only
101	5	r-x	Readable and executable
110	6	rw-	Readable and writable
111	7	rwx	Readable, writable and executable

There are 3 categories and 3 permissions for each category, hence 3 octal digits can describe a file's permissions completely.

Most Significant Digit represents user and the least one represents others.

$ chmod a+rw test

$ chmod 666 test ; ls -l test

-rw-rw-rw- …… test

$ chmod 761 test

Relative Permissions – 5M

Absolute Permissions – 5M

Total 10M