USN □□□□□□□□□□

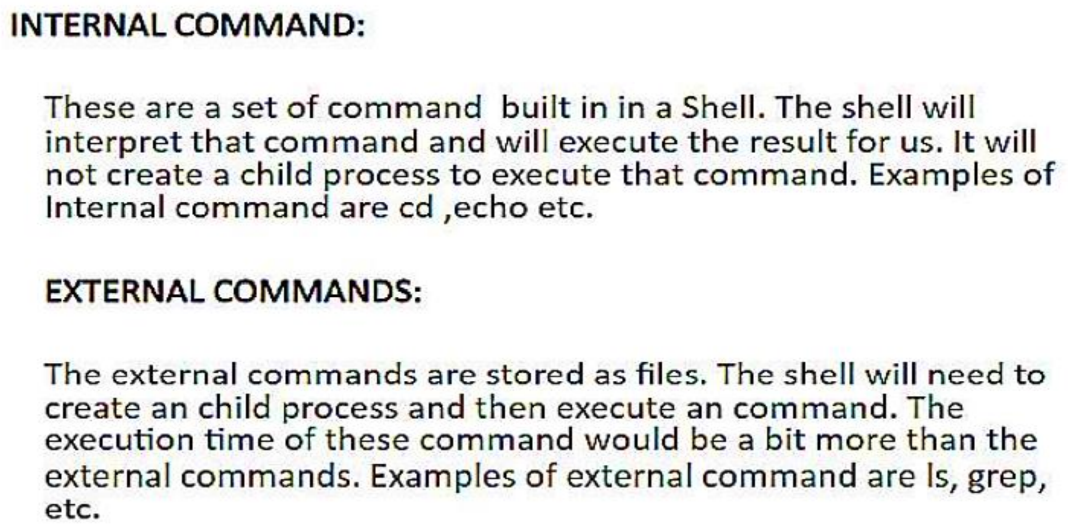
Internal Assessment Test 1 – NOV 2022
Scheme of Evaluation

Sub:	**UNIX Programming**				Sub Code:	18CS56	Branch:	ISE
Date:	**05/11/2022**	Duration:	90 min	Max Marks:	50	Sem/Sec:	V / A, B & C	**OBE**

<table>
<tr><th></th><th>Answer any FIVE FULL Questions</th><th>MARKS</th><th>CO</th><th>RBT</th></tr>
<tr><td><b>1 (a)</b></td><td>
<b>Explain Internal and External Commands with an example.</b>

<b>Scheme:-</b> Definition + explanation+examples= 2+2+1= 5M

<b>Solution:-</b>



INTERNAL COMMAND:

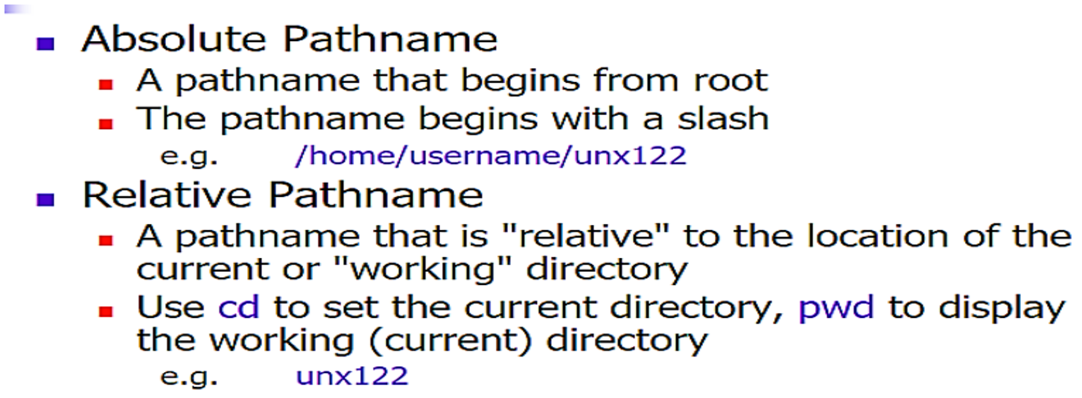
These are a set of command built in in a Shell. The shell will interpret that command and will execute the result for us. It will not create a child process to execute that command. Examples of Internal command are cd ,echo etc.

EXTERNAL COMMANDS:

The external commands are stored as files. The shell will need to create an child process and then execute an command. The execution time of these command would be a bit more than the external commands. Examples of external command are ls, grep, etc.
</td><td>[05]</td><td>CO1</td><td>L2</td></tr>
<tr><td><b>(b)</b></td><td>
<b>Define Absolute and Relative Pathnames with an example.</b>

<b>Scheme:-</b> Definition + explanation+examples= 2+2+1= 5M

<b>Solution:-</b>



- Absolute Pathname
  - A pathname that begins from root
  - The pathname begins with a slash

    e.g. /home/username/unx122
- Relative Pathname
  - A pathname that is "relative" to the location of the current or "working" directory
  - Use cd to set the current directory, pwd to display the working (current) directory

    e.g. unx122
</td><td>[05]</td><td>CO1</td><td>L1</td></tr>
<tr><td><b>2</b></td><td>
<b>Describe the salient features of UNIX.</b>

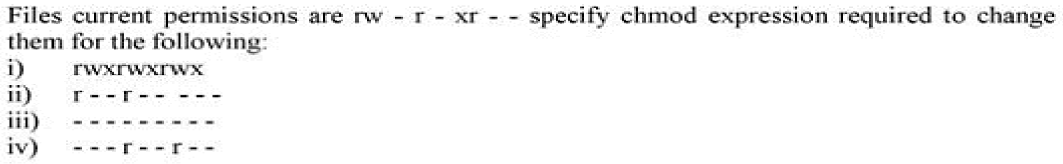
<b>Scheme:-</b> Listing of feature + explanation with examples= 3+7=10 M

<b>Solution:-</b>

➢ Multiuser system
➢ Multitasking System
➢ The Building Block Approach
➢ UNIX Tool Kit
➢ Pattern Matching
➢ Programming facility
➢ Documentation
</td><td>[10]</td><td>CO1</td><td>L1</td></tr>
</table>

3

Define any 2 shells wild-cards and Apply chmod expression for the following permissions using both absolute and relative methods.

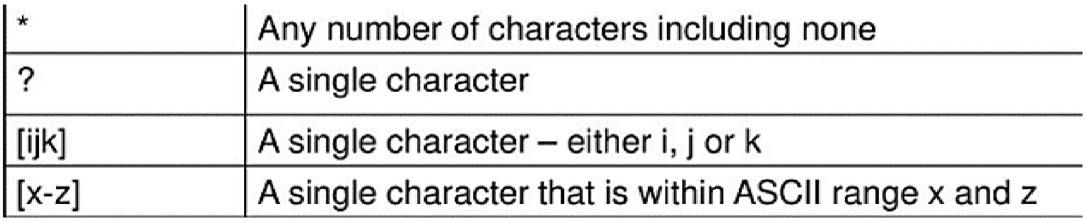


Files current permissions are rw - r - xr - - specify chmod expression required to change them for the following:

i) rwxrwxrwx
ii) r - - r - - - - -
iii) - - - - - - - - -
iv) - - - r - - r - -

Scheme:- Defining shell wildcards + permissions using relative/absolute = 2+8 = 10 M

Solution:-



*	Any number of characters including none
?	A single character
[ijk]	A single character – either i, j or k
[x-z]	A single character that is within ASCII range x and z

i) relative: $chmod u+x,g+w,o+wx file.txt
absolute: $chmod 777 file.txt
ii) relative: $chmod u-w,g-x,o-r file.txt
absolute: $chmod 440 file.txt
iii) relative: $chmod u-rw,g-rx,o-r file.txt
absolute: $chmod 000 file.txt
iv) relative: $chmod u-rw,g-x file.txt
absolute: $chmod 044 file.txt

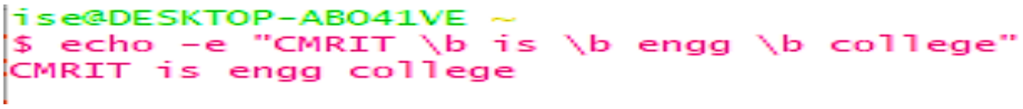
[10] CO2 L3

4

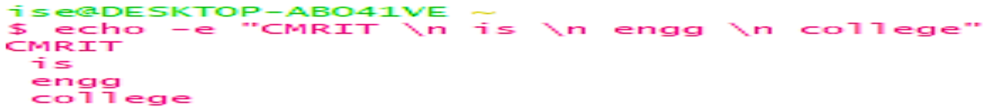
Explain the use of following commands with examples: echo, printf, mv, and cp.

Scheme:- explanation of commands + examples= 5+5=10 M

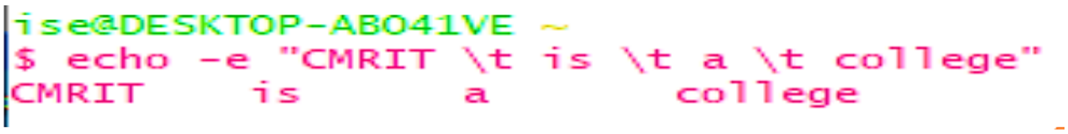
Solution:-



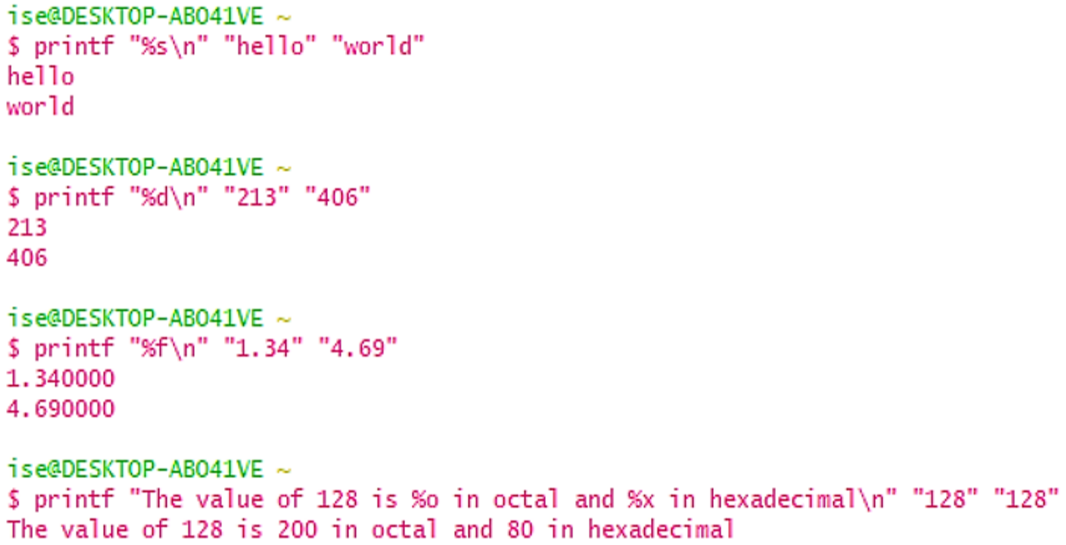
```
ise@DESKTOP-ABO41VE ~
$ echo -e "CMRIT \b is \b engg \b college"
CMRIT is engg college
```



```
ise@DESKTOP-ABO41VE ~
$ echo -e "CMRIT \n is \n engg \n college"
CMRIT
 is
 engg
 college
```



```
ise@DESKTOP-ABO41VE ~
$ echo -e "CMRIT \t is \t a \t college"
CMRIT    is      a       college
```



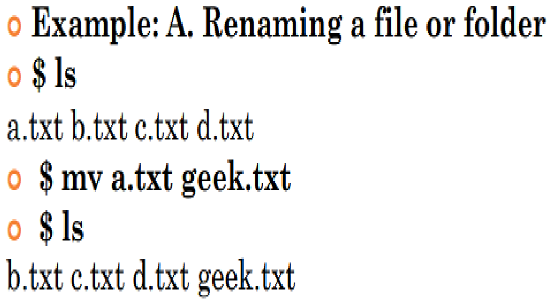
```
ise@DESKTOP-ABO41VE ~
$ printf "%s\n" "hello" "world"
hello
world

ise@DESKTOP-ABO41VE ~
$ printf "%d\n" "213" "406"
213
406

ise@DESKTOP-ABO41VE ~
$ printf "%f\n" "1.34" "4.69"
1.340000
4.690000

ise@DESKTOP-ABO41VE ~
$ printf "The value of 128 is %o in octal and %x in hexadecimal\n" "128" "128"
The value of 128 is 200 in octal and 80 in hexadecimal
```

[10] CO1 L2

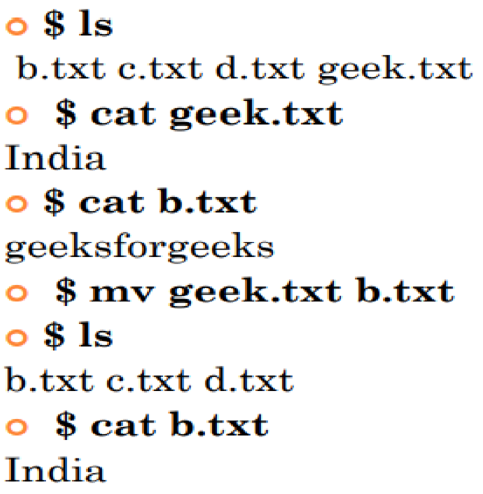


- Example: A. Renaming a file or folder
- $ ls

a.txt b.txt c.txt d.txt

- $ mv a.txt geek.txt
- $ ls

b.txt c.txt d.txt geek.txt



- $ ls

b.txt c.txt d.txt geek.txt

- $ cat geek.txt

India

- $ cat b.txt

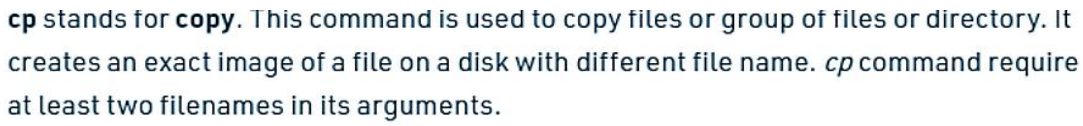
geeksforgeeks

- $ mv geek.txt b.txt
- $ ls

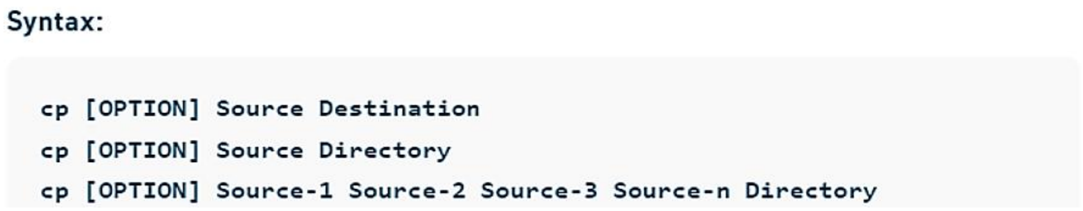
b.txt c.txt d.txt

- $ cat b.txt

India

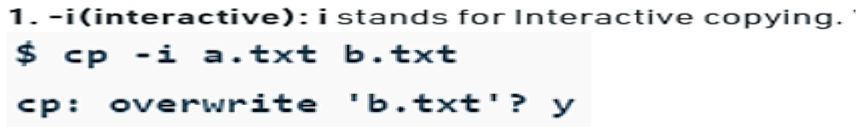


cp stands for **copy**. This command is used to copy files or group of files or directory. It creates an exact image of a file on a disk with different file name. *cp* command require at least two filenames in its arguments.



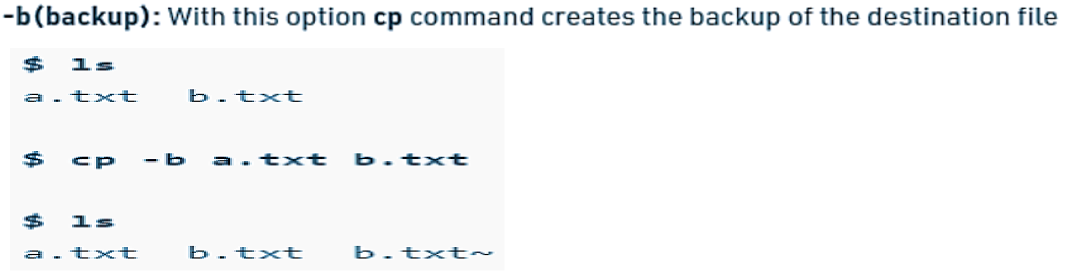
Syntax:

```
cp [OPTION] Source Destination
cp [OPTION] Source Directory
cp [OPTION] Source-1 Source-2 Source-3 Source-n Directory
```



1. **-i(interactive):** **i** stands for Interactive copying.

```
$ cp -i a.txt b.txt
cp: overwrite 'b.txt'? y
```



-b(backup): With this option **cp** command creates the backup of the destination file

```
$ ls
a.txt   b.txt

$ cp -b a.txt b.txt

$ ls
a.txt   b.txt   b.txt~
```

5 (a) **What is the Output of the following?**

a. date + %D   b. date +"%H %M %S" c. cal 09 1961 d. echo $SHELL

Scheme:- output of each command = 5M

Solution:-

a. Displays current date **11/07/22**

b. Displays time with hrs,min and seconds **14 39 44**

c. Displays calendar of September **1961**

```
September 1961
Su Mo Tu We Th Fr Sa
                1  2
 3  4  5  6  7  8  9
10 11 12 13 14 15 16
17 18 19 20 21 22 23
24 25 26 27 28 29 30
```

d. Displays Shell Terminal details **/bin/bash**

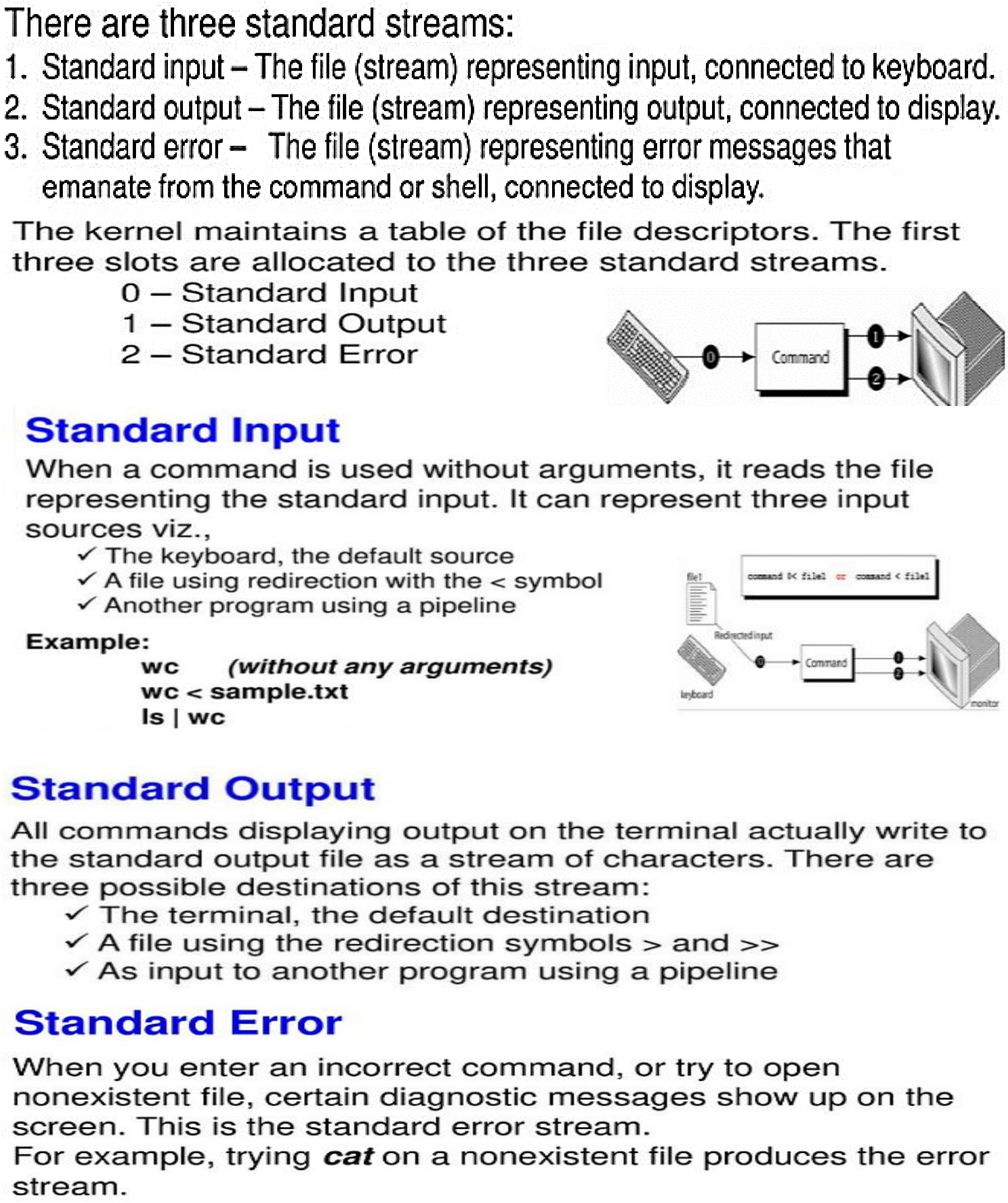
[05] CO1 L3

(b) **Apply cat command to perform the following and write the output:**

i) Create and display the file contents of file file1.txt, file2.txt.

ii) To display content of file2.txt with line numbers.

[05] CO1 L3

	iii) To display content of file1.txt in reverse order. **Scheme:-** Creating/displaying+ displaying Contents with line numbers + content in reverse order = **5M** **Solution:- cat > file1.txt cat > file2.txt** **// creating the contents of files** **cat file1.txt cat file2.txt** **// displaying the contents of files** **cat file2.txt** **// displaying the contents with line numbers** **cat file1.txt** **// displaying the contents in reverse order**			
6	**Define Redirection. Explain three standard Files for redirection.** **Scheme:-** Definition + explanation+examples = 2+5+3= 10 M **Solution:-**  There are three standard streams: 1. Standard input – The file (stream) representing input, connected to keyboard. 2. Standard output – The file (stream) representing output, connected to display. 3. Standard error – The file (stream) representing error messages that emanate from the command or shell, connected to display. The kernel maintains a table of the file descriptors. The first three slots are allocated to the three standard streams. 0 – Standard Input 1 – Standard Output 2 – Standard Error **Standard Input** When a command is used without arguments, it reads the file representing the standard input. It can represent three input sources viz., ✓ The keyboard, the default source ✓ A file using redirection with the < symbol ✓ Another program using a pipeline **Example:** **wc** ***(without any arguments)*** **wc < sample.txt** **ls | wc** file1 command 0< file1 or command < file1 Redirected input Command keyboard monitor **Standard Output** All commands displaying output on the terminal actually write to the standard output file as a stream of characters. There are three 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 **Standard Error** When you enter an incorrect command, or try to open nonexistent file, certain diagnostic messages show up on the screen. This is the standard error stream. For example, trying ***cat*** on a nonexistent file produces the error stream.	[10]	CO2	L2

Faculty Signature CCI Signature HOD Signature