NAME : PARITOSH LAHRE   ID : 12341550   COURSE : CSP203

1.GREP

email : grep -E "[a-zA-Z0-9._%+-]+@[a-zA-Z0-9.-]+\.[a-zA-Z]{2,}" filename

phone no. : grep -E "(\([0-9]{3}\)|[0-9]{3})[-.]?[0-9]{3}[-.]?[0-9]{4}" filename

dd/mm/yyyy : grep -E "([0-9]{2})/([0-9]{2})/([0-9]{4})" filename

ip address : grep -E "([0-9]{1,3}\.){3}[0-9]{1,3}" filename

URL : grep -E "https?://[a-zA-Z0-9./?=_-]+" filename

hexa : grep -E "0[xX][0-9a-fA-F]+|[0-9a-fA-F]+" filename

credit card : grep -E "[0-9]{4}(-[0-9]{4}){3}" filename

MAC address : grep -E "([0-9A-Fa-f]{2}:){5}[0-9A-Fa-f]{2}" filename

-q: Quiet mode. Suppresses output, useful for just checking if a pattern exists.
-i: Ignore case. Matches patterns regardless of case.
-r or -R: Recursive search through directories.
-E: Use extended regular expressions.

^ for starting pos , $ for end pos

2.SED COMMAND

sed [options] 'command' filename

find-replace : sed 's/pattern/replacement/' filename ||&|| sed 's/\(pattern\)/replacement\1/' filename || sed 's|pattern|replacement|' filename

replace all occ in a line : sed 's/pattern/replacement/g' filename

to ignore cases : sed 's/pattern/replacement/I' filename

delete line or range : sed 'nd' filename or sed 'm,nd' filename

delete all line matching pattern : sed '/pattern/d' filename

inserting line before a line no. : sed 'n i\new_line_text' filename

inserting line after a line no. : sed 'n a\new_line_text' filename

to replace on particular line : sed 'n s/pattern/replacement/' filename

to replace on ranges of line : sed 'm,n s/pattern/replacement/' filename

to remove empty lines : sed '/^$/d' filename

to remove trailing whitespaces : sed 's/[\t]*$//' filename

to add line number to file : sed = filename | sed 'N;s/\n/\t/'

to append a string at the end of each line : sed 's/$/string_to_append/' filename

to print specific line : sed -n 'np' filename ||&&|| to print ranges of line : sed -n 'm,np' filename

to only print matching pattern : sed -n '/pattern/p' filename

to replace last occurr of a pattern on a line : sed 's/\(.*\)pattern/\1replacement/' filename

to replace nth character in a line : sed 's/^\(.\{n\}\)(to be replaced..)/\1replacement/' filename

to stop after a certain pattern is matched : sed '/pattern/q' filename

-i: Edit files in place.
-e: Add the script to the commands to be executed.
-f: Take the script from a file.
-n: Suppress automatic printing of pattern space.

3.SSH

copy files to from : scp /path/to/localfile user@remote_host:/path/to/remotefile

4.SHELL SCRIPTING

bash : #!/bin/bash

variable : var_name="value"

to access it : echo $var_name || echo to print

IF-ELSE :

```
if [ condition ]; then
   # commands
elif [ condition ]; then
   # commands
else
   # commands
fi
```

FOR LOOP:

```
for i in {1..n}; do
   echo "Welcome $i"
done
```

WHILE LOOP:

```
while [ condition ];
do
   # commands
done
```

Comparision :
-eq, -ne, -lt, -le, -gt, -ge

=, !=, -z (empty), -n (not empty)

INPUT OUTPUT

```
echo "This is a message"

read -r var
echo "You entered: $var"
```

COMMAND ASSIGNMENT :

```
result=$(command)
echo $result
```

5.FILE OPERATIONS

If file exists:

```
if [ -f filename ]; then
   echo "File exists"
fi
```

If dir exists :

```
if [ -d dirname ]; then
   echo "Directory exists"
fi
```

Reading file line by line :

```
while IFS= read -r line; do
   echo "$line"
done < filename
```

SWITCH CASE :

```
case "$variable" in
    pattern1)
        # commands to execute if pattern1
matches
        ;;
    pattern2)
        # commands to execute if pattern2
matches
        ;;
    *)
        # commands to execute if no patterns
match (optional)
        ;;
esac
```

MISCELLANEOUS:

[ ] or [[ ]]: Test conditions (single brackets is POSIX-compliant, double brackets are Bash-specific).
(( )): Arithmetic evaluations.
-f: Check if file exists.
-d: Check if directory exists.

```
sed 's/old/new/' file.txt          # Replace 'old' with 'new' in file.txt
sed -i 's/old/new/g' file.txt        # In-place replacement, global (all occurrences)
sed -n 's/old/new/p' file.txt         # Replace 'old' with 'new' and print the result
sed -f script.sed file.txt           # Use commands from script.sed to edit file.txt
```

awk 'pattern { action }' [file...]
Pattern Matching:

awk processes lines that match a specified pattern.
Patterns can be regular expressions, relational expressions, or logical expressions.
Field Separator:

By default, awk uses whitespace (spaces or tabs) as the field separator.
You can change this with the -F option.
Fields:

awk treats each line of input as a record and splits it into fields.
Fields are referenced using $1, $2, ..., $n for the first, second, ..., nth field