Some Commands with output

1. $passed

   OUTPUT

   old password:
   new password:
   re-enter new password:

2. $ uname

   OUTPUT

   MINGW64_NT-10.0-26100

3. $ ls

   OUTPUT

```
LEGION@ND MINGW64 /d/GEETANJALI/LINUX(NIKHIL)/shell scripts/shellpro
$ ls
28_3_break.sh          case1.shu    l1.txt.lnk*
28_3_continue.sh       change.sh    len.sh
28_3_for_loop.sh       cmd1.sh      linkexample.txt
28_3_if_elif_else.sh   cmd2.sh      ll2.txt
28_3_if_else.sh        cmd3.sh      lp1.sh
28_3_OPERATORS.SH      data.txt     lp2.sh
28_3_simple_if.sh      digsum.sh    output.txt
28_3_if_else.sh        cmd3.sh      lp1.sh
28_3_OPERATORS.SH      data.txt     lp2.sh
28_3_simple_if.sh      digsum.sh    output.txt
28_3_until_loop.sh     example.txt  palinum.sh
28_3_while_loop.sh     export1.sh   s1.sh
28_3_while_loop.txt    extra.sh     s2.sh
31_3_logical_op.sh     filelp.sh    tempCodeRunnerFile.SH
31_3_logical_op_2.txt  filelp1.sh   vowels.sh
31_3_logical_op_3.sh   if1.sh       vowels1.sh
case1.sh               l.txt        vowels2.sh
```

4. $ ls –l

   OUTPUT

```
LEGION@ND MINGW64 /d/GEETANJALI/LINUX(NIKHIL)/shell scripts/shellpro
$ ls -l
total 42
-rw-r--r-- 1 LEGION 197609 147 Mar 28 12:58 28_3_break.sh
-rw-r--r-- 1 LEGION 197609 137 Mar 28 13:00 28_3_continue.sh
-rw-r--r-- 1 LEGION 197609  99 Mar 28 12:50 28_3_for_loop.sh
-rw-r--r-- 1 LEGION 197609 126 Mar 28 12:34 28_3_if_elif_else.sh
-rw-r--r-- 1 LEGION 197609 134 Mar 28 12:28 28_3_if_else.sh
-rw-r--r-- 1 LEGION 197609 176 Mar 28 12:17 28_3_OPERATORS.SH
-rw-r--r-- 1 LEGION 197609  73 Mar 28 12:21 28_3_simple_if.sh
-rw-r--r-- 1 LEGION 197609  98 Mar 28 12:56 28_3_until_loop.sh
-rw-r--r-- 1 LEGION 197609 102 Mar 28 12:52 28_3_while_loop.sh
-rw-r--r-- 1 LEGION 197609 102 Mar 28 12:52 28_3_while_loop.txt
-rw-r--r-- 1 LEGION 197609 614 Mar 31 08:11 31_3_logical_op.sh
-rw-r--r-- 1 LEGION 197609 670 Mar 31 08:16 31_3_logical_op_2.txt
-rw-r--r-- 1 LEGION 197609 394 Mar 31 08:25 31_3_logical_op_3.sh
```

5. Cat > abc.txt

```
LEGION@ND MINGW64 /d/GEETANJALI/LINUX(NIKHIL)/shell scripts/shellpro
$ cat>abc.txt
testing
testing
abcd efg

[1]+  Stopped                 cat > abc.txt

LEGION@ND MINGW64 /d/GEETANJALI/LINUX(NIKHIL)/shell scripts/shellpro
$ cat abc.txt
testing
testing
abcd efg
```

6. Cat>>abc.txt

```
LEGION@ND MINGW64 /d/GEETANJALI/LINUX(NIKHIL)/shell scripts/shellpro
$ cat>>abc.txt
jnbvskdjbvs
vjsdvbsdf
vnjsddbnv
jkvsnfv

[2]+  Stopped                 cat >> abc.txt

LEGION@ND MINGW64 /d/GEETANJALI/LINUX(NIKHIL)/shell scripts/shellpro
$ cat abc.txt
testing
testing
abcd efg
jnbvskdjbvs
vjsdvbsdf
vnjsddbnv
jkvsnfv
```

7. pwd

```
LEGION@ND MINGW64 /d/GEETANJALI/LINUX(NIKHIL)/shell scripts/shellpro
$ pwd
/d/GEETANJALI/LINUX(NIKHIL)/shell scripts/shellpro
```

8. stat abc.txt

OUTPUT

```
LEGION@ND MINGW64 /d/GEETANJALI/LINUX(NIKHIL)/shell scripts/shellpro
$ stat abc.txt
  File: abc.txt
  Size: 67              Blocks: 1          IO Block: 65536  regular file
Device: e6dfc574h/3873424756d   Inode: 1125899907065834  Links: 1
Access: (0644/-rw-r--r--)  Uid: (197609/  LEGION)   Gid: (197609/ UNKNOWN)
Access: 2025-04-02 06:38:24.653052300 +0530
Modify: 2025-04-02 06:38:17.142849700 +0530
Change: 2025-04-02 06:38:17.142849700 +0530
 Birth: 2025-04-02 06:33:59.955254500 +0530
```

9. **grep command**
   a. **welcome.txt**

      Welcome to Linux !

      Linux is a free and opensource Operating system that is mostly used by developers and in production servers for hosting crucial components such as web and database servers. Linux has also made a name for itself in PCs.

      Beginners looking to experiment with Linux can get started with friendlier linux distributions such as Ubuntu, Mint, Fedora and Elementary OS.

   b. **grep "Linux" welcome.txt**

      OUTPUT

   c.
```
LEGION@ND MINGW64 /d/GEETANJALI/LINUX(NIKHIL)/shell scripts/shellpro
$ grep "Linux" welcome.txt
Welcome to Linux !
Linux is a free and opensource Operating system that is mostly used by
and database servers. Linux has also made a name for itself in PCs.
Beginners looking to experiment with Linux can get started with friendlier linux
```

      grep --color "free and opensource" welcome.txt

      OUTPUT

```
LEGION@ND MINGW64 /d/GEETANJALI/LINUX(NIKHIL)/shell scripts/shellpro
$ grep --color "free and opensource" welcome.txt
Linux is a free and opensource Operating system that is mostly used by
```

   d. **count the total number of lines**

      grep -c "Linux" welcome.txt

```
LEGION@ND MINGW64 /d/GEETANJALI/LINUX(NIKHIL)/shell scripts/shellpro
$ grep -c "Linux" welcome.txt
4
```

   e. **Search for exact matching**

      grep -w "opensource" welcome.txt

```
LEGION@ND MINGW64 /d/GEETANJALI/LINUX(NIKHIL)/shell scripts/shellpro
$ grep -w "opensource" welcome.txt
Linux is a free and opensource Operating system that is mostly used by
```

10. Join command

a. File1:

cat << EOF > device_names.log
10.0.1.10 WINSHARE01
10.0.1.13 WEBSERVER03
10.0.1.15 FINSERVER02
143.204.74.129 WEBSERVER02
152.120.106.21 HRSERVER01
192.168.8.28 MYWORKSTATION
192.168.10.4 PRINTER02
EOF

b. File2

cat << EOF > device_policies.log
10.0.1.10 RED
10.0.1.13 YELLOW
10.0.1.15 RED
143.204.74.129 YELLOW
152.120.106.21 BLUE
192.168.8.28 GREEN 1
92.168.10.4 RED
EOF

c. join device_names.log device_policies.log

OUTPUT

```
LEGION@ND MINGW64 /d/GEETANJALI/LINUX(NIKHIL)/shell scripts/shellpro
$ join device_names.log device_policies.log
10.0.1.10 WINSHARE01 RED
10.0.1.13 WEBSERVER03 YELLOW
10.0.1.15 FINSERVER02 RED
143.204.74.129 WEBSERVER02 YELLOW
152.120.106.21 HRSERVER01 BLUE
192.168.8.28 MYWORKSTATION GREEN
192.168.10.4 PRINTER02 RED
```

11.sort command

cat<< EOF > cities.txt
New York City
Paris
Beijing
Hamburg
Los Angeles
Amsterdam
EOF

Sort cities.txt

OUTPUT

```
LEGION@ND MINGW64 /d/GEETANJALI/LINUX(NIKHIL)/shell scripts/shellpro
$ sort cities.txt
Amsterdam
Beijing
Hamburg
Los Angeles
New York City
Paris
```

12.uniq command

cat << EOF > countries.txt
Germany
South Africa
Japan
USA
England
Spain
Italy
Cameroon
Japan
EOF

a. Printing Duplicate Lines

sort countries.txt | uniq –d

```
LEGION@ND MINGW64 /d/GEETANJALI/LINUX(NIKHIL)/shell scripts/shellpro
$ sort countries.txt | uniq -d
Japan
```

b. Counting Duplicate Lines

sort countries.txt | uniq –c

```
LEGION@ND MINGW64 /d/GEETANJALI/LINUX(NIKHIL)/shell scripts/shellpro
$ sort countries.txt | uniq -c
      1 Cameroon
      1 England
      1 Germany
      1 Italy
      2 Japan
      1 South Africa
      1 Spain
      1 USA
```

c. Removing Duplicate Lines

sort countries.txt | uniq –u

```
LEGION@ND MINGW64 /d/GEETANJALI/LINUX(NIKHIL)/shell scripts/shellpro
$ sort countries.txt | uniq -u
Cameroon
England
Germany
Italy
South Africa
Spain
USA
```

13.Printing to Standard Output (stdout)

echo "Hello, LabEx!"
printf "The value of x is: %d\n" 42

14.Saving Output to a File

echo "This output will be saved to a file" > output.txt

If you want to append data to an existing file, you can use the >> operator.
echo "Appending to the file" >> output.txt

15.Capturing Command Output

You can capture the output of a command and store it in a variable using the $(command) or `command` syntax.

current_date=$(date)
echo "The current date is: $current_date"

OUTPUT

```
LEGION@ND MINGW64 /d/GEETANJALI/LINUX(NIKHIL)/shell scripts/shellpro
$ current_date=$(date)
echo "The current date is: $current_date"
The current date is: Fri, Mar 28, 2025 12:03:52 PM
```

16.Formatting Output with ANSI Escape Codes

echo -e "\033[1;32mThis text is green and bold.\033[0m"

OUTPUT

```
LEGION@ND MINGW64 /d/GEETANJALI/LINUX(NIKHIL)/shell scripts/shellpro
$ echo -e "\033[1;32mThis text is green and bold.\033[0m"
This text is green and bold.
```

17.Multiline Output

```
cat << EOF
This is the first line.
This is the second line.
This is the third line.
EOF
```

OUTPUT

```
This is the first line.
This is the second line.
This is the third line.
```

Scripts

1. **Script to take 2 values form user and perform arithmetic operations**

```
echo "Enter the first number: "
read num1
echo "Enter the second number: "
read num2
result1=$((num1 + num2))
result2=$((num1 - num2))
result3=$((num1 * num2))
result4=$((num1 / num2))
echo "The sum of $num1 and $num2 is $result1"
echo "The sum of $num1 and $num2 is $result2"
echo "The sum of $num1 and $num2 is $result3"
echo "The sum of $num1 and $num2 is $result4"
```

OUTPUT

```
LEGION@ND MINGW64 /d/GEETANJALI/LINUX(NIKHIL)/shell scripts/shellpro
$ bash "d:\GEETANJALI\LINUX(NIKHIL)\shell scripts\shellpro\03_04_demo.sh"
Enter the first number:
12
Enter the second number:
12
The sum of 12 and 12 is 24
The sum of 12 and 12 is 0
The sum of 12 and 12 is 144
The sum of 12 and 12 is 1
```

2. **Conditional Output and Logging**

```
echo $?
if [ $? -eq 0 ]; then
  echo "Command executed successfully."
```

```
else
  echo "Command failed." >&2
fi
```

OUTPUT

```
Command executed successfully.

LEGION@ND MINGW64 /d/GEETANJALI/LINUX(NIKHIL)/shell scripts/shellpro
$ echo $?
0
```

3. **Example of OPERATORS**

```
COST_PINEAPPLE=50
COST_BANANA=4
COST_WATERMELON=23
COST_BASKET=1

TOTAL=$((COST_PINEAPPLE + (COST_BANANA * 2) +
(COST_WATERMELON * 3) + COST_BASKET))

echo "Total Cost is $TOTAL"
```

OUTPUT

```
LEGION@ND MINGW64 /d/GEETANJALI/LINUX(NIKHIL)/shell scripts/shellpro
$ bash "d:\GEETANJALI\LINUX(NIKHIL)\shell scripts\shellpro\tempCodeRunnerFile.SH"
Total Cost is 128
```

4. **Example of SIMPLE IF**

```
NAME="ND"
if [ "$NAME" = "ND" ]; then
  echo "True - my name is $NAME"
fi
```

OUTPUT

```
LEGION@ND MINGW64 /d/GEETANJALI/LINUX(NIKHIL)/shell scripts/shellpro
$ bash "d:\GEETANJALI\LINUX(NIKHIL)\shell scripts\shellpro\28_3_simple_if.sh"
True - my name is ND
```

5. **Example of IF ELSE STATEMENT**

```
NAME="ND"
if [ "$NAME" = "ABC" ]; then
  echo "True - my name is $NAME"
else
  echo "False"
```

```
  echo "You must mistake me for $NAME"
fi
```

OUTPUT

```
LEGION@ND MINGW64 /d/GEETANJALI/LINUX(NIKHIL)/shell scripts/shellpro
$ bash "d:\GEETANJALI\LINUX(NIKHIL)\shell scripts\shellpro\28_3_if_else.sh"
False
You must mistaken me for ND
```

6. **Example of IF ELIF ELSE STATEMENT**

```
NAME="ND"
if [ "$NAME" = "ABC" ]; then
  echo "ABCD"
elif [ "$NAME" = "ND" ]; then
  echo "NIKHIL DAVE"
else
  echo "ERROR"
fi
```

OUTPUT

```
LEGION@ND MINGW64 /d/GEETANJALI/LINUX(NIKHIL)/shell scripts/shellpro
$ bash "d:\GEETANJALI\LINUX(NIKHIL)\shell scripts\shellpro\28_3_if_elif_else.sh"
NIKHIL DAVE
```

EXTRA

Numeric Comparisons
-lt: less than
-gt: greater than
-le: less than or equal to
-ge: greater than or equal to
-eq: equal to
-ne: not equal to
"=": equal to
"==": equal to
"!=": not equal to
"-z": empty string

7. **PROG.-1**

```
NUMBER=10
APPLES=10
KING=GEORGE

if [ $NUMBER -gt 15 ]; then
  echo 1
fi
```

```
if [ $NUMBER -eq $APPLES ]; then
  echo 2
fi
```

8. **PROG.-2**

```
NUMBER=10
APPLES=12
KING=LUIS

if [[ ($APPLES -eq 12) || ("$KING" = "LUIS") ]]; then
  echo 3
fi
```

9. **Example of FOR LOOP**

```
NAMES=("ND" "NIKHIL" "TEST" "TESTING")
for name in "${NAMES[@]}"; do
  echo "My name is $name"
done
```

OUTPUT

```
LEGION@ND MINGW64 /d/GEETANJALI/LINUX(NIKHIL)/shell scripts/shellpro
$ bash "d:\GEETANJALI\LINUX(NIKHIL)\shell scripts\shellpro\28_3_for_loop.sh"
My name is ND
My name is NIKHIL
My name is TEST
My name is TESTING
```

10. **Example of WHILE LOOP**

```
count=4
while [ $count -gt 0 ]; do
  echo "Value of count is: $count"
  count=$(($count - 1))
done
```

OUTPUT

```
LEGION@ND MINGW64 /d/GEETANJALI/LINUX(NIKHIL)/shell scripts/shellpro
$ bash "d:\GEETANJALI\LINUX(NIKHIL)\shell scripts\shellpro\28_3_while_loop.sh"
Value of count is: 4
Value of count is: 3
Value of count is: 2
Value of count is: 1
```

11. **Example of UNTIL LOOP**

```
count=1
until [ $count -gt 5 ]; do
  echo "Value of count is: $count"
  count=$(($count + 1))
done
```

OUTPUT

```
LEGION@ND MINGW64 /d/GEETANJALI/LINUX(NIKHIL)/shell scripts/shellpro
$ bash "d:\GEETANJALI\LINUX(NIKHIL)\shell scripts\shellpro\28_3_until_loop.sh"
Value of count is: 1
Value of count is: 2
Value of count is: 3
Value of count is: 4
Value of count is: 5
```

12.Example of BREAK

```
count=0
while [ $count -ge 0 ]; do
  echo "Value of count is: $count"
  count=$((count + 1))
  if [ $count -ge 5 ]; then
    break
  fi
done
```

OUTPUT

```
LEGION@ND MINGW64 /d/GEETANJALI/LINUX(NIKHIL)/shell scripts/shellpro
$ bash "d:\GEETANJALI\LINUX(NIKHIL)\shell scripts\shellpro\28_3_break.sh"
Value of count is: 0
Value of count is: 1
Value of count is: 2
Value of count is: 3
Value of count is: 4
```

13.Example of CONTINUE

```
count=0
while [ $count -lt 10 ]; do
  count=$((count + 1))
  if [ $(($count % 2)) = 0 ]; then
    continue
  fi
  echo $count
done
```

OUTPUT

```
LEGION@ND MINGW64 /d/GEETANJALI/LINUX(NIKHIL)/shell scripts/shellpro
$ bash "d:\GEETANJALI\LINUX(NIKHIL)\shell scripts\shellpro\28_3_continue.sh"
1
3
5
7
9
```

14.Example of LOGICAL OPERATORS 1

```
SAVED_USERNAME="ND"
SAVED_PASSWORD="ND"
read -p "Enter your username: " USERNAME
if [[ -z $USERNAME ]]
```

```
then
  echo "Error: Username cannot be empty"
  exit 1
fi
read -sp "Enter password: " PASSWORD
echo
if [[ -z $PASSWORD ]]
then
  echo "Error: Password cannot be empty"
  exit 1
fi
if [[ $USERNAME == $SAVED_USERNAME &&
$PASSWORD == $SAVED_PASSWORD ]]
then
  echo "Logged in successfully!"
elif [[ $USERNAME == $SAVED_USERNAME &&
$PASSWORD != $SAVED_PASSWORD || $PASSWORD ==
$SAVED_PASSWORD && $USERNAME !=
$SAVED_USERNAME ]]
then
  echo "Invalid credentials, Username and Password did not match"
fi
exit 0
```

OUTPUT

```
LEGION@ND MINGW64 /d/GEETANJALI/LINUX(NIKHIL)/shell scripts/shellpro
$ bash "d:\GEETANJALI\LINUX(NIKHIL)\shell scripts\shellpro\31_3_logical_op.sh"
Enter your username: ND
Enter password:
Logged in successfully!
```

15. **Example of LOGICAL OPERATORS 2**

```
read -p 'Enter a : ' a
read -p 'Enter b : ' b
if(( $a==$b ))
then
   echo a is equal to b.
else
   echo a is not equal to b.
fi
if(( $a!=$b ))
then
   echo a is not equal to b.
else
```

```
    echo a is equal to b.
fi
if(( $a<$b ))
then
    echo a is less than b.
else
    echo a is not less than b.
fi
if(( $a<=$b ))
then
    echo a is less than or equal to b.
else
    echo a is not less than or equal to b.
fi
if(( $a>$b ))
then
    echo a is greater than b.
else
    echo a is not greater than b.
fi
if(( $a>=$b ))
then
    echo a is greater than or equal to b.
else
    echo a is not greater than or equal to b.
fi
```

OUTPUT

```
LEGION@ND MINGW64 /d/GEETANJALI/LINUX(NIKHIL)/shell scripts/shellpro
$ bash "d:\GEETANJALI\LINUX(NIKHIL)\shell scripts\shellpro\31_3_logical_op_2.txt"
Enter a : 12
Enter b : 32
a is not equal to b.
a is not equal to b.
a is less than b.
a is less than or equal to b.
a is not greater than b.
a is not greater than or equal to b.
```

16.Example of CASE statement

```
echo "Which color do you like best?"
echo "1 - Blue"
echo "2 - Red"
echo "3 - Yellow"
echo "4 - Green"
```

```
echo "5 - Orange"
read color;
case $color in
  1) echo "Blue is a primary color.";;
  2) echo "Red is a primary color.";;
  3) echo "Yellow is a primary color.";;
  4) echo "Green is a secondary color.";;
  5) echo "Orange is a secondary color.";;
  *) echo "This color is not available. Please choose a different
one.";;
esac
```

OUTPUT

```
LEGION@ND MINGW64 /d/GEETANJALI/LINUX(NIKHIL)/shell scripts/shellpro
$ bash "d:\GEETANJALI\LINUX(NIKHIL)\shell scripts\shellpro\case1.sh"
Which color do you like best?
1 - Blue
Which color do you like best?
1 - Blue
1 - Blue
2 - Red
3 - Yellow
3 - Yellow
4 - Green
5 - Orange
8
This color is not available. Please choose a different one.
```

17.Example to print pyramid

12345
1234
123
12
1

```
for((i=5;i>=1;i--));
do
   for((j=1;j<=$i;j++));
   do
      printf "%d" $j;
   done;
   printf "\n";
done
```

OUTPUT

```
LEGION@ND MINGW64 /d/GEETANJALI/LINUX(NIKHIL)/shell scripts/shellpro
$ bash "d:\GEETANJALI\LINUX(NIKHIL)\shell scripts\shellpro\03_04_demo_1.sh"
12345
1234
123
12
1
```

18.Example to count no. of characters from input string

```
echo "Enter Your Name : "
read name
echo "Length : "
echo $name | wc -c
```

OUTPUT

```
LEGION@ND MINGW64 /d/GEETANJALI/LINUX(NIKHIL)/shell scripts/shellpro
$ bash "d:\GEETANJALI\LINUX(NIKHIL)\shell scripts\shellpro\len.sh"
Enter Your Name :
NIKHIL
Length :
7
```

19.Example to find out number is Palindrome or not

```
echo "Enter Number : "
read num
extra=$num
revnum=0
while [ $num -ne 0 ]
do
   ((d=$num%10))
   ((revnum=($revnum*10)+$d))
   ((num=$num/10))
done
if [ $revnum -eq $extra ]
then
   echo "Palindrom Number"
else
   echo "Not Palindrom Number"
fi
```

OUTPUT

```
LEGION@ND MINGW64 /d/GEETANJALI/LINUX(NIKHIL)/shell scripts/shellpro
$ bash "d:\GEETANJALI\LINUX(NIKHIL)\shell scripts\shellpro\tempCodeRunnerFile.sh"
Enter Number :
121
Palindrom Number
```

```
LEGION@ND MINGW64 /d/GEETANJALI/LINUX(NIKHIL)/shell scripts/shellpro
$ bash "d:\GEETANJALI\LINUX(NIKHIL)\shell scripts\shellpro\palinum.sh"
Enter Number :
123
Not Palindrom Number
```

20.Example of Function

```
# Calling one function from another
number_one ()
{
  echo "FUNCTION 1_____"
  number_two
}
number_two ()
{
  echo "FUNCTION 2_____"
}
# Calling function one.
number_one
```

OUTPUT

```
LEGION@ND MINGW64 /d/GEETANJALI/LINUX(NIKHIL)/shell scripts/shellpro
$ bash "d:\GEETANJALI\LINUX(NIKHIL)\shell scripts\shellpro\udf_1.sh"
FUNCTION 1_____
FUNCTION 2_____
```

21.Script to creates a simple digital clock on terminal

```
#T   Displays the 24-hour clock (00-23) in the format equivalent to
HH:MM:SS
function clock_1
{
        clear
        while [ 1 ]
        do
            date +'%T'
            sleep 1
            clear
        done
 }
clock_1
```

OUTPUT

```
20:29:25
```

22.Example of ARRAY

```
NAME[0]="ND"
NAME[1]="NIKHIL"
NAME[2]="ABC"
NAME[3]="XYZ"
NAME[4]="PQR"
echo "First Index: ${NAME[0]}"
echo "Second Index: ${NAME[1]}"

#You can access all the items in an array
NAME[0]="ND"
NAME[1]="NIKHIL"
NAME[2]="ABC"
NAME[3]="XYZ"
NAME[4]="PQR"
echo "First Method: ${NAME[*]}"
echo "Second Method: ${NAME[@]}"
```

OUTPUT

```
LEGION@ND MINGW64 /d/GEETANJALI/LINUX(NIKHIL)/shell scripts/shellpro
$ bash "d:\GEETANJALI\MORNING\array.sh"
First Index: ND
Second Index: NIKHIL
First Method: ND NIKHIL ABC XYZ PQR
Second Method: ND NIKHIL ABC XYZ PQR
```