

Bash Scripting

- stands for Bourne Again Shell.
- easy commands
- most used shell language
- stands the test of time
- lightweight and always available
- is not mutually exclusive

Disadvantages

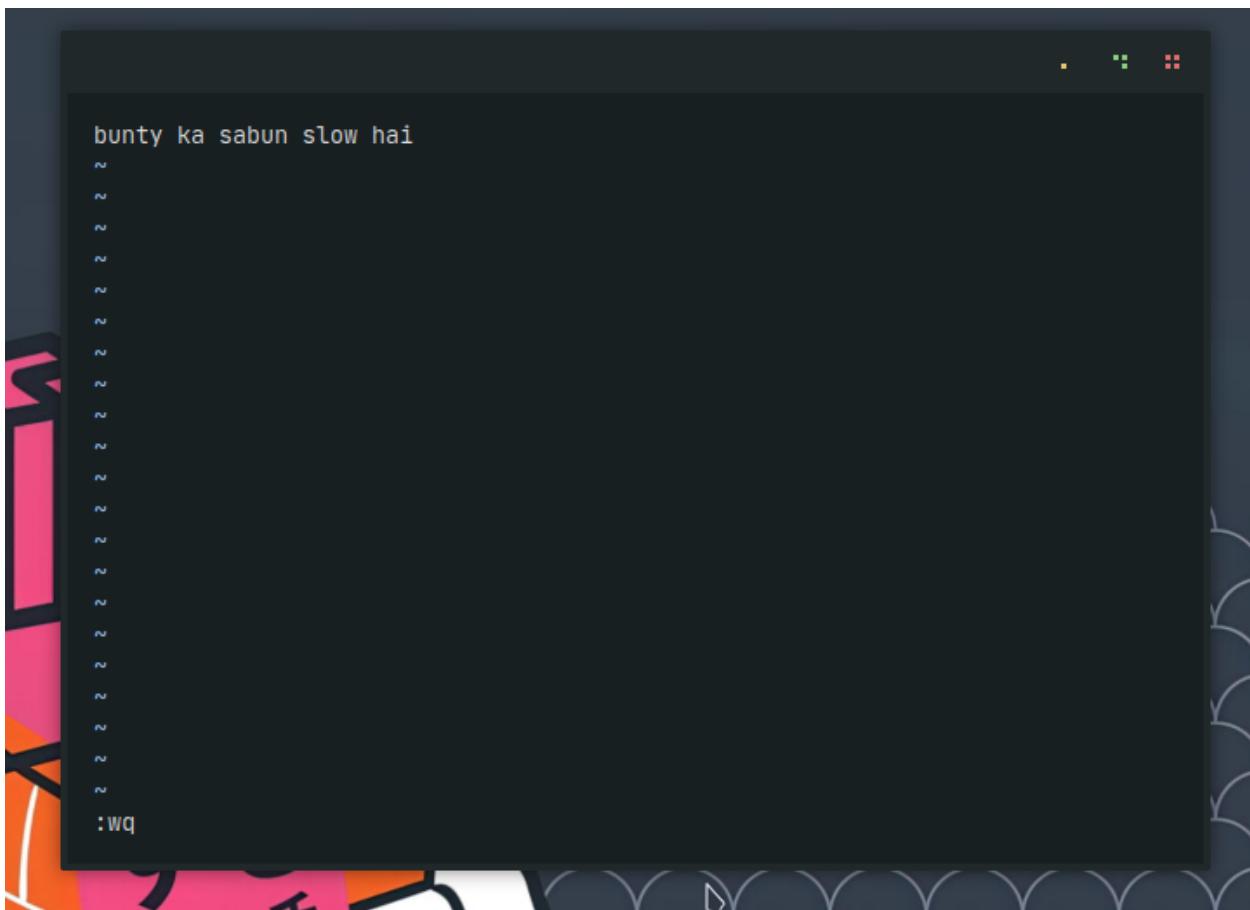
- lacks some features
- no OOP
- difficult syntax compared to python
- newer tools like Ansible

Basic Commands

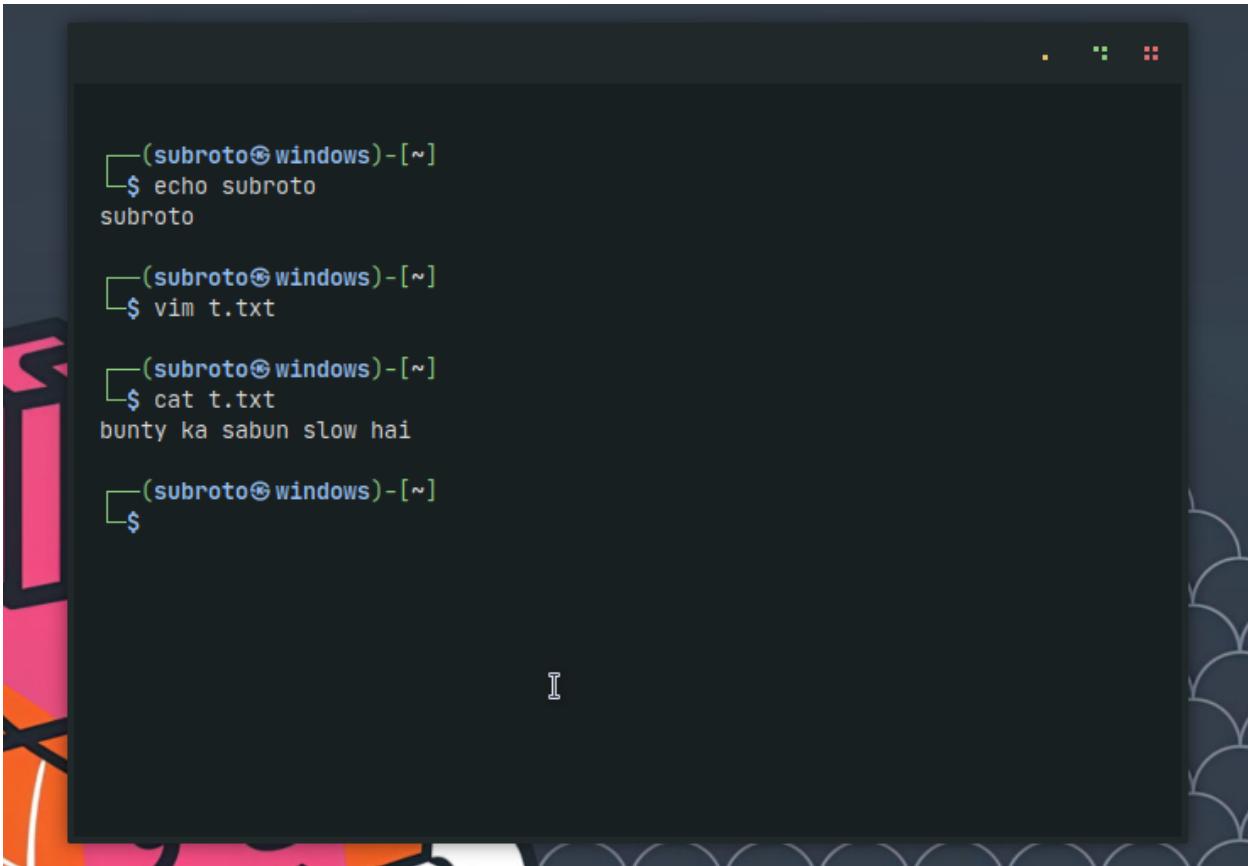
```
└─(subproto㉿windows)-[~]
$ echo subproto
subproto

└─(subproto㉿windows)-[~]
$ vim t.txt_
```

echo command prints its argument , here subroto is the argument so it is returned.
then we have created a text file in vim editor using vim t.txt



inside the vim editor press “ i ” to enter the writing mode, then press escape to come back to command mode, then type :wq to save and exit



```
(subroto@windows)~$ echo subroto
subroto

(subroto@windows)~$ vim t.txt

(subroto@windows)~$ cat t.txt
bunty ka sabun slow hai

(subroto@windows)~$
```

then after our textfile is saved , we use the cat command to print the contents of the file.

to give our script executable permission

```
chmod u+x filename.sh
```

```

[subrto@windows:~]
$ echo $SHELL
/usr/bin/zsh

[subrto@windows:~]
$ chsh
Password:
Changing the login shell for subrto
Enter the new value, or press ENTER for the default
Login Shell [/usr/bin/zsh]: 

[subrto@windows:~]
$ echo $SHELL
/usr/bin/zsh

[subrto@windows:~]
$ ls
521071.jpg          'Mid Semester.pdf'
Android             Music
android-studio      'new resume.pdf'
AndroidStudioProjects Pictures
'api notes.pdf'    platform-tools-android-dev
'auto gpt'          Postman
Blockchain          'Postman (copy 1)'
books               programmes
build               Public
'current resume.pdf' 'research papers'
Desktop             screenshots
Documents           shelltest.sh
Downloads           smurf
ebook               'spring freestyle'
'firefox developer edition' subrto.txt
'FLUTTER'           'Subrto vault'
'flutter apps'     sub.txt
gennmon-scripts    t.txt
git-for-shell      Videos
host.txt           備註

19:39 IITK-Routine-Autumn2022_Draftv11.pdf

[subrto@windows:~]
$ bash shelltest.sh
Hello world!

[subrto@windows:~]
$ vim shelltest.sh

[subrto@windows:~]
$ ./shelltest.sh
zsh: permission denied: ./shelltest.sh

[subrto@windows:~]
$ ls -l
total 8396
-rw-r--r-- 1 subrto subrto 639689 Apr 22 04:53 521071.jpg


```

```

zsh: permission denied: ./shelltest.sh

[subrto@windows:~]
$ ls -l
total 8396
-rw-r--r-- 1 subrto subrto 639689 Apr 22 04:53 521071.jpg
drwxr-xr-x  7 subrto subrto 4096 Apr  6 28:53 Android
drwxr-xr-x  4 subrto subrto 4096 Apr  5 21:12 AndroidStudioProjects
drwxr-xr-x  4 subrto subrto 4096 Apr  5 21:12 Blockchain
drwxr-xr-x  1 subrto subrto 4096 Apr 19 02:45 'api notes.pdf'
drwxr-xr-x  3 subrto subrto 4096 Apr 19 21:42 'auto gpt'
drwxr-xr-x  3 subrto subrto 4096 Apr 19 21:42 blockchain
drwxr-xr-x  2 subrto subrto 4096 Apr 21 22:42 books
drwxr-xr-x  2 subrto subrto 4096 Apr 21 18:53 'chrome'
drwxr-xr-x  1 subrto subrto 84770 Apr 15 01:52 'current resume.pdf'
drwxr-xr-x  2 subrto subrto 4096 Apr 22 12:08 Desktop
drwxr-xr-x  2 subrto subrto 4096 Mar 30 12:57 Documents
drwxr-xr-x 12 subrto subrto 4096 Apr 22 14:08 Downloads
drwxr-xr-x  8 subrto subrto 4096 Apr 22 12:08 ebook
drwxr-xr-x  2 subrto subrto 4096 Apr 22 12:08 'firefox developer edition'
drwxr-xr-x  3 subrto subrto 4096 Apr  4 19:56 'FLUTTER'
drwxr-xr-x  2 subrto subrto 4096 Apr 16 20:26 'flutter apps'
drwxr-xr-x  2 subrto subrto 4096 Apr 22 01:57 gennmon-scripts
drwxr-xr-x 12 subrto subrto 4096 Apr 22 13:23 'git-layer-shell'
drwxr-xr-x  2 subrto subrto 4096 Apr 21 18:53 host.txt
-rw-r--r--  1 subrto subrto 711280 Apr 13 14:44 'IITK-Routine-Autumn2022_Draftv11.pdf'
drwxr-xr-x  2 subrto subrto 77898 Mar 30 12:57 'Mid Semester.pdf'
drwxr-xr-x  1 subrto subrto 85477 Apr 18 21:46 'new resume.pdf'
drwxr-xr-x  2 subrto subrto 4096 Apr 22 19:11 Pictures
drwxr-xr-x  3 subrto subrto 4096 Apr  6 21:58 platform-tools-android-dev
drwxr-xr-x  2 subrto subrto 4096 Apr 22 19:11 Postman
lrwxrwxrwx  1 subrto subrto 39 Apr  4 18:43 'Postman (copy 1)' -> /home/subrto/Downloads/Postman/Postman
drwxr-xr-x 18 subrto subrto 4096 Apr 18 09:54 programmes
drwxr-xr-x  2 subrto subrto 4096 Mar 30 12:57 Public
drwxr-xr-x  2 subrto subrto 4096 Apr 28 15:56 'research papers'
drwxr-xr-x  2 subrto subrto 4096 Apr 17 23:23 screenshots
drwxr-xr-x  3 subrto subrto 4096 Apr 16 20:24 shelltest.sh
drwxr-xr-x  3 subrto subrto 4096 Apr 16 20:24 smurf
drwxr-xr-x  2 subrto subrto 4096 Apr 16 22:51 'spring freestyle'
-rw-r--r--  1 subrto subrto 12 Apr 22 19:11 subrto.txt
drwxr-xr-x  6 subrto subrto 4096 Apr 21 10:14 'Subrto vault'
-rw-r--r--  1 subrto subrto 10 Apr 22 19:11 sub.txt
-rw-r--r--  1 subrto subrto 34 Apr 22 19:11 sub1.txt
drwxr-xr-x  2 subrto subrto 4096 Mar 30 12:57 Videos
drwxr-xr-x  2 subrto subrto 4096 Apr 16 19:01 備註

19:39 [subrto@windows:~]
$ chmod u+x shelltest.sh

[subrto@windows:~]
$ ls -l
total 8396
-rw-r--r-- 1 subrto subrto 639689 Apr 22 04:53 521071.jpg

```

```

[sudo@subro@windows) ~]
$ chmod u+x shelltest.sh
[sudo@subro@windows) ~]
$ ls -l
total 8396
drwxr-xr-x  1 subro subro 639469 Apr 22 06:53 521071.jpg
drwxr-xr-x  3 subro subro 4896 Apr  6 20:04 Android
drwxrwxr-x  7 subro subro 4896 Jan  1 2010 android-studio
drwxr-xr-x  4 subro subro 4896 Apr  6 21:10 AndroidStudioProjects
-rw-r--r--  1 subro subro 6850942 Apr 19 02:45 api.notes.adf
drwxr-xr-x  3 subro subro 4896 Apr 19 06:45 gpt
drwxr-xr-x  3 subro subro 4896 Apr 19 11:42 blockchain
drwxr-xr-x  2 subro subro 4896 Apr 21 22:43 books
drwxr-xr-x  2 subro subro 4896 Apr 22 12:28 build
-rw-r--r--  1 subro subro 84378 Apr 13 01:32 'current resume.pdf'
drwxr-xr-x  3 subro subro 4896 Apr 22 12:04 Desktop
drwxr-xr-x  2 subro subro 4896 Apr 22 12:07 Documents
drwxr-xr-x 12 subro subro 4896 Apr 22 14:52 Downloads
drwxr-xr-x  8 subro subro 4896 Apr 22 12:58 ewm
drwxr-xr-x  7 subro subro 4896 Apr 13 10:20 'firefox developer edition'
drwxr-xr-x  6 subro subro 4896 Apr  6 19:34 FLUTTER
drwxr-xr-x  2 subro subro 4896 Apr 16 20:04 flutter_apps
drwxr-xr-x  2 subro subro 4896 Apr 16 20:57 flutter_Scripts
drwxr-xr-x 12 subro subro 4896 Apr 22 13:27 gtk-layer-shell
-rw-r--r--  1 subro subro  42 Apr 21 10:52 host.txt
-rw-r--r--  1 subro subro 711786 Apr  2 19:17 IIITK-Routine-Autumn2022_Draftv11.pdf
-rw-r--r--  1 subro subro 77098 Apr 13 14:44 Mid Semester.pdf
drwxr-xr-x  2 subro subro 4896 Apr 38 13:44 misc
drwxr-xr-x  1 subro subro 859469 Apr 22 14:44 'new resume.pdf'
drwxr-xr-x  2 subro subro 4896 Apr 22 19:19 Pictures
drwxr-xr-x  3 subro subro 4896 Apr  6 21:34 platform-tools-android-dev
drwxr-xr-x  3 subro subro 4896 Apr  4 18:41 Postman
drwxrwxr-x  1 subro subro 39 Apr  4 18:43 Postman (copy 1)' -> /home/subro/Downloads/Postman/Postman
drwxr-xr-x  3 subro subro 4896 Apr 18 09:54 programmes
drwxr-xr-x  1 subro subro 4896 Apr 22 12:04 screenshots
drwxr-xr-x  2 subro subro 4896 Apr 28 15:56 research_papers
drwxr-xr-x  2 subro subro 4896 Apr 17 23:35 screenshots
-rwxr--r--  1 subro subro 33 Apr 22 19:34 shelltest.sh
drwxr----  3 subro subro 4896 Apr 16 20:21 snap
drwxr-xr-x  2 subro subro 4896 Apr 22 22:11 'spring freestyle'
drwxr-xr-x  1 subro subro 4896 Apr 22 22:15 'spring boot'>
drwxr-xr-x  6 subro subro 4896 Apr 21 10:14 Subneto_vault
-rw-r--r--  1 subro subro 15 Apr 22 10:10 sub.txt
-rw-r--r--  1 subro subro 24 Apr 22 19:19 t.txt
drwxr-xr-x  2 subro subro 4896 Mar 38 12:57 Videos
drwxr-xr-x  2 subro subro 4896 Apr 14 19:01 標板

```

The terminal shows the execution of the command `chmod u+x shelltest.sh` followed by a `ls -l` command. The output lists numerous files and directories with their permissions, modification times, and sizes. The file `shelltest.sh` is explicitly mentioned in the listing.

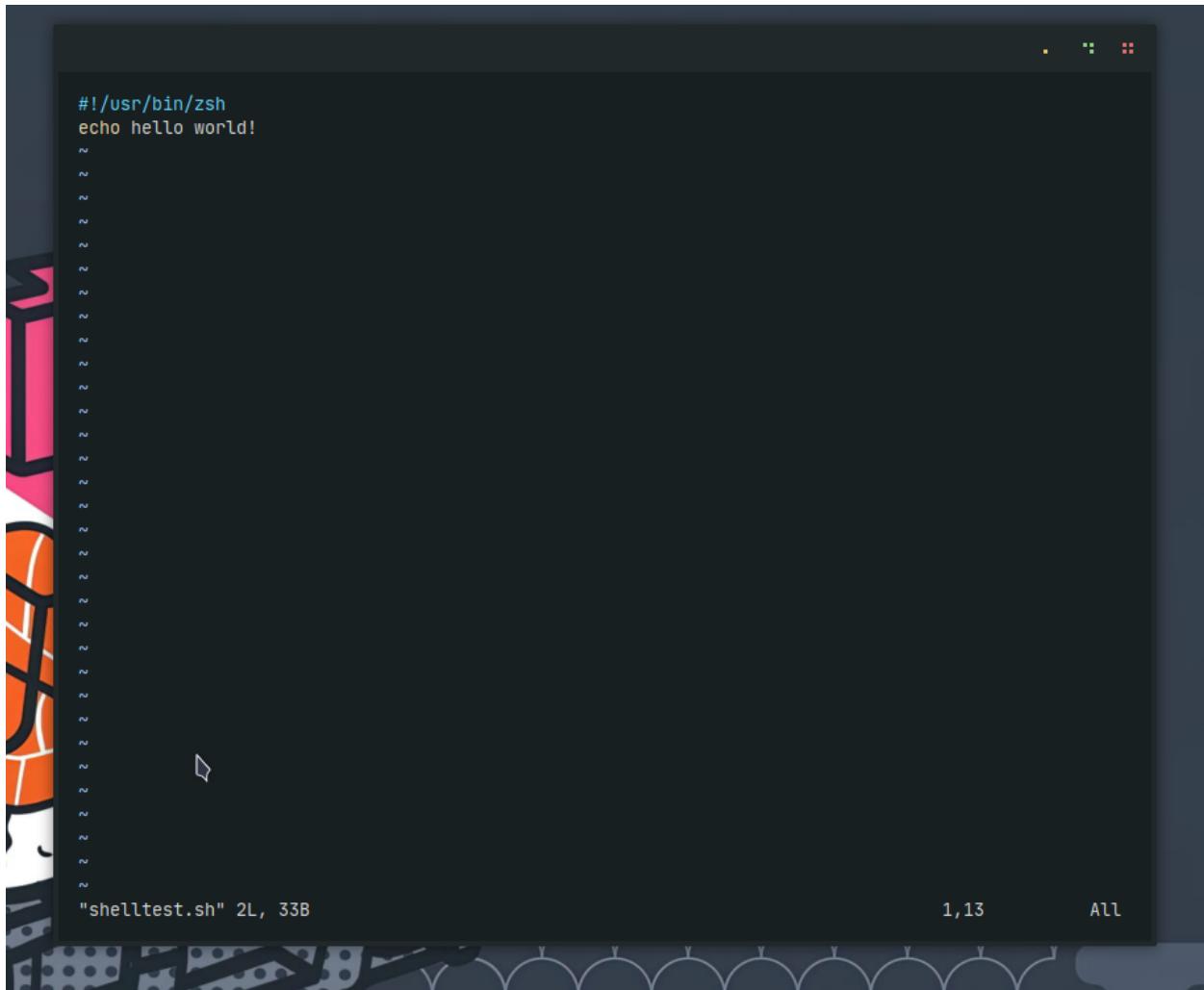
the command `ls -l`

lists the files with the permissions it is granted

after performing all this we finally can run the `./filename.sh` command to run our scripts

also we have to put the location obtained from the command `echo $SHELL`

in the file.



A screenshot of a terminal window with a dark background. The window title bar is visible at the top. Inside the terminal, there is a single line of code:

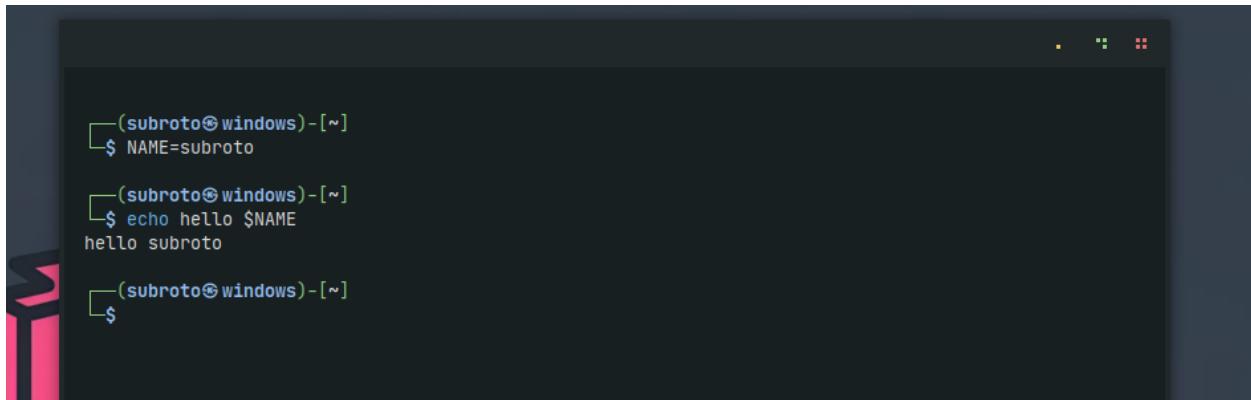
```
#!/usr/bin/zsh
echo hello world!
```

The cursor is located on the second line of the code. In the bottom right corner of the terminal window, there are status indicators: "1,13" and "All".

the topmost command after # is the one we're talking about. It shows from where the file has to be run.

Variables

simple variables

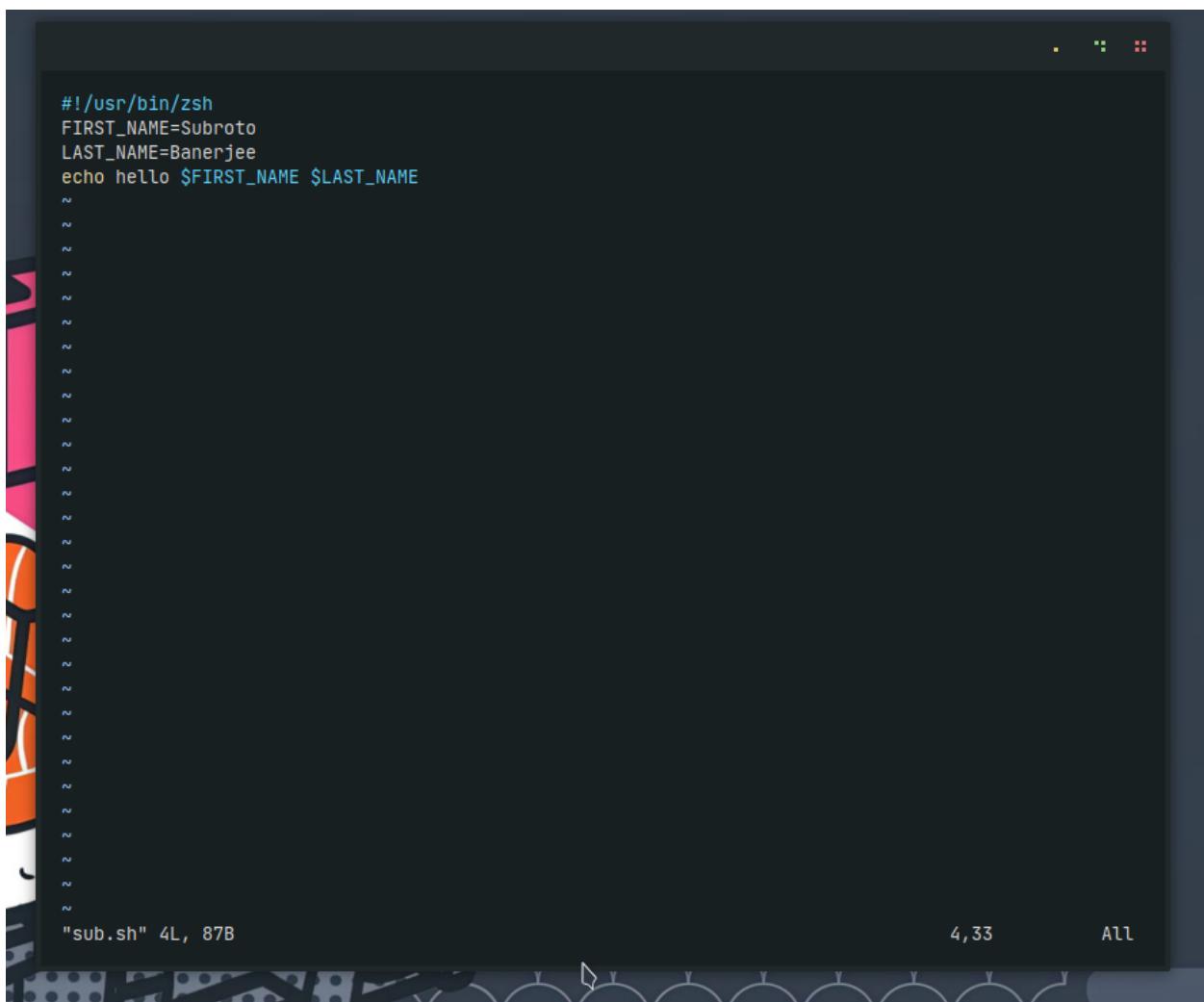


```
(subroto@windows) - [~]
$ NAME=subroto

(subroto@windows) - [~]
$ echo hello $NAME
hello subroto

(subroto@windows) - [~]
$
```

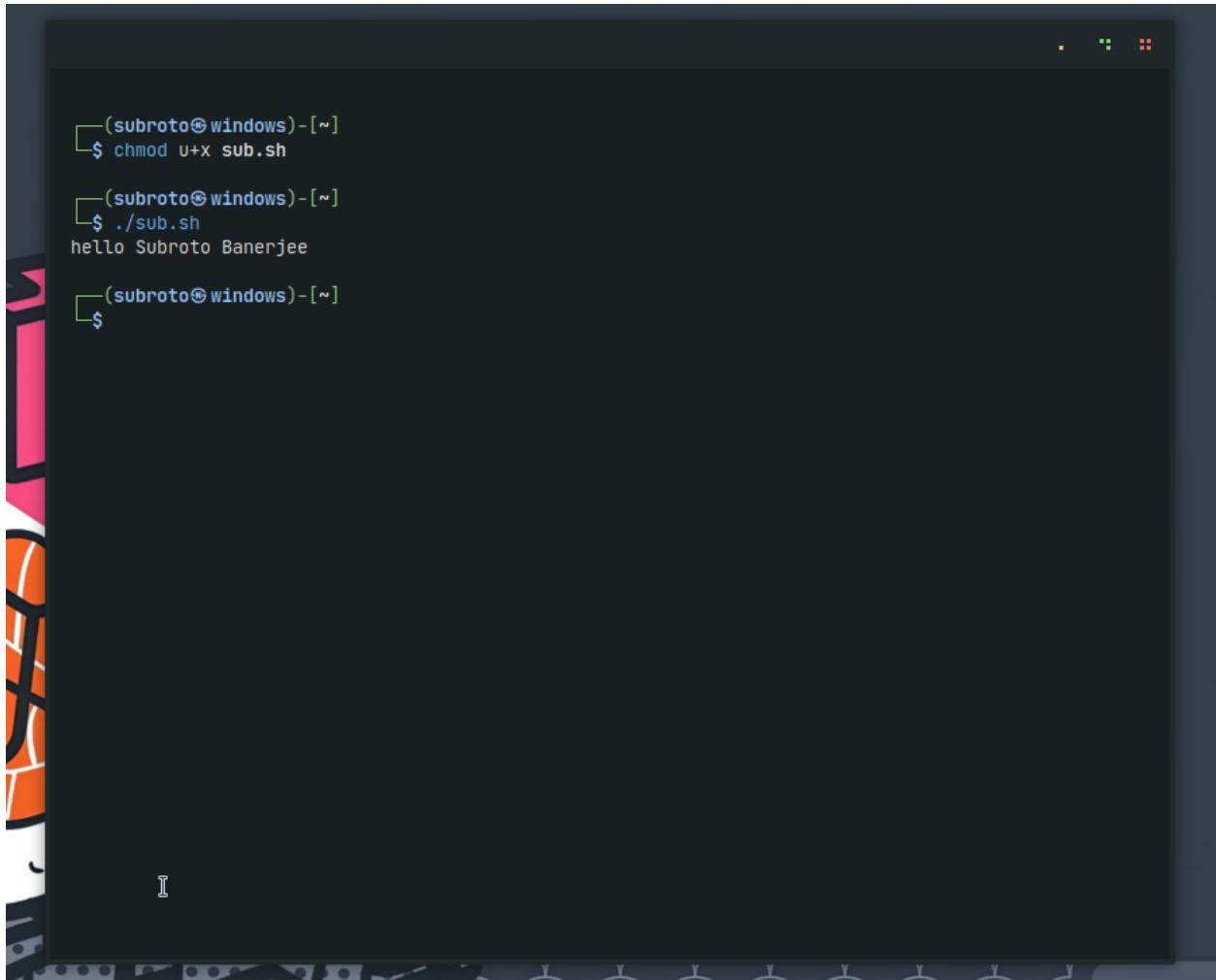
using variables in a script and printing them



```
#!/usr/bin/zsh
FIRST_NAME=Subroto
LAST_NAME=Banerjee
echo hello $FIRST_NAME $LAST_NAME
```

The terminal shows the script being run. The output is visible at the top of the terminal window, followed by several blank lines indicating the script's execution.

"sub.sh" 4L, 87B 4,33 All

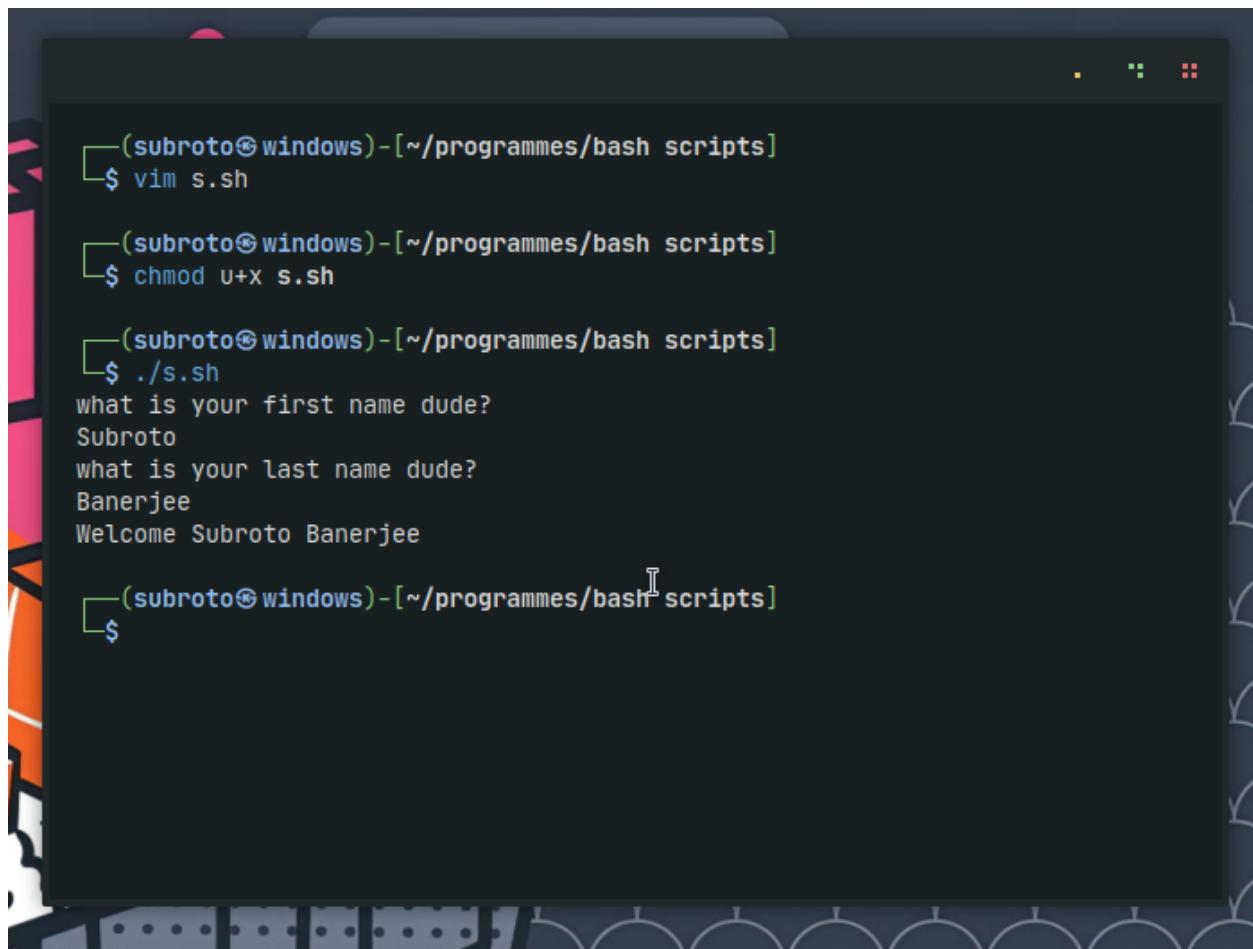


The image shows a terminal window with a dark background and light-colored text. The terminal session starts with the prompt '(subproto@windows) - [~]'. The user runs the command '\$ chmod u+x sub.sh', followed by '\$./sub.sh'. The script outputs 'hello Subroto Banerjee' and ends with another prompt '(subproto@windows) - [~]'. The terminal is set against a background featuring a colorful abstract graphic on the left.

```
(subproto@windows) - [~]
$ chmod u+x sub.sh
(subproto@windows) - [~]
$ ./sub.sh
hello Subroto Banerjee
(subproto@windows) - [~]
$
```

Getting input from user

first create a script and :



The image shows a terminal window with a dark background and light-colored text. The terminal is running on a system named 'subproto@windows' in the directory '/programmes/bash scripts'. The user has run a script named 's.sh' using Vim, given it execute permissions with 'chmod u+x s.sh', and then executed it with './s.sh'. The script prompts for the user's first name ('what is your first name dude?'), receives the input 'Subroto', prompts for the last name ('what is your last name dude?'), receives the input 'Banerjee', and then prints a welcome message ('Welcome Subroto Banerjee').

```
(subproto@windows)-[~/programmes/bash scripts]
$ vim s.sh

(subproto@windows)-[~/programmes/bash scripts]
$ chmod u+x s.sh

(subproto@windows)-[~/programmes/bash scripts]
$ ./s.sh
what is your first name dude?
Subroto
what is your last name dude?
Banerjee
Welcome Subroto Banerjee

(subproto@windows)-[~/programmes/bash scripts]
$
```

Positional Arguments

A screenshot of a terminal window with a dark background. The window title bar at the top right shows three small icons. The main area of the terminal contains the following text:

```
#!/usr/bin/bash
echo hello $1 $2
~
```

The terminal window has a vertical scroll bar on the right side. At the bottom of the window, there is status information: "sh.sh" 2L, 33B on the left, 2,16 in the center, and All on the right.

The terminal window shows the following session:

```
(subroto@windows)-[~/programmes/bash scripts]
$ vim sh.sh

(subroto@windows)-[~/programmes/bash scripts]
$ chmod u+x sh.sh

(subroto@windows)-[~/programmes/bash scripts]
$ ./sh.sh Subroto Banerjee
hello Subroto Banerjee

(subroto@windows)-[~/programmes/bash scripts]
$ vim sh.sh

(subroto@windows)-[~/programmes/bash scripts]
$
```

Piping

doing something with the output

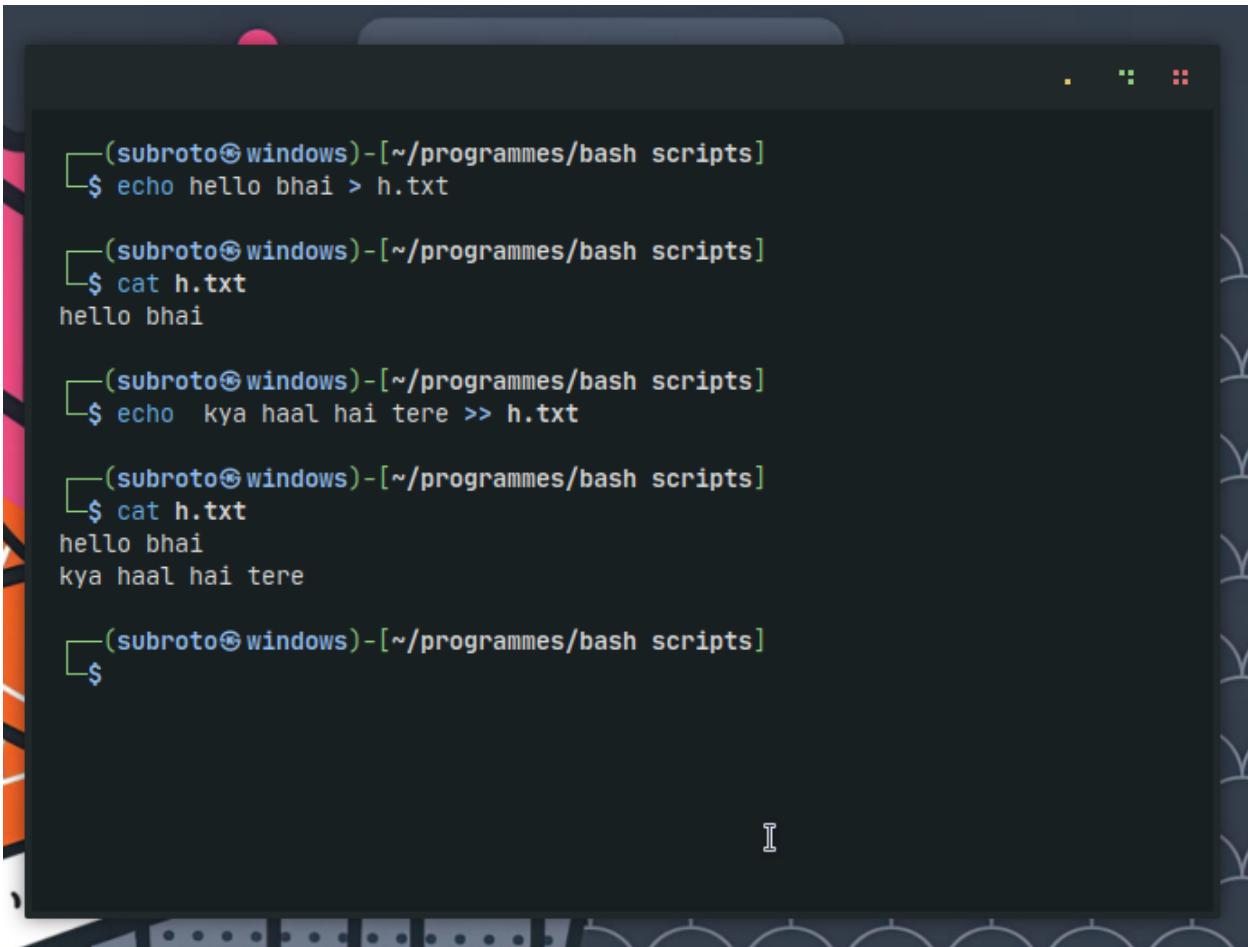
eg: filtering the output

```
(subproto@windows)~]$ ls -l /usr/bin | grep bash
-rwxr-xr-x 1 root      root      1265648 Feb 13 01:35 bash
-rwxr-xr-x 1 root      root       6865 Feb 13 01:35 bashbug
-rwxr-xr-x 1 root      root      4414 Aug 28 2021 dh_bash-completion
lrwxrwxrwx 1 root      root        4 Feb 13 01:35 rbash -> bash

(subproto@windows)~]$ _
```

this filters out only the files having bash term in them in the directory /usr/bin

Writing to a file



The screenshot shows a terminal window with a dark background and light-colored text. It displays several commands related to file operations:

```
(subroto@windows)-[~/programmes/bash scripts]
$ echo hello bhai > h.txt

(subroto@windows)-[~/programmes/bash scripts]
$ cat h.txt
hello bhai

(subroto@windows)-[~/programmes/bash scripts]
$ echo kya haal hai tere >> h.txt

(subroto@windows)-[~/programmes/bash scripts]
$ cat h.txt
hello bhai
kya haal hai tere

(subroto@windows)-[~/programmes/bash scripts]
$
```

so > command overwrites whereas >> appends

Getting input from text files

< command

The screenshot shows a terminal window with a dark background and light-colored text. It displays three separate command-line sessions. Each session starts with the prompt '(subproto@windows)'. The first session shows the command '\$ wc -w h.txt' followed by the output '6 h.txt'. The second session shows '\$ wc -w < h.txt' followed by the output '6'. The third session shows '\$ _' at the prompt.

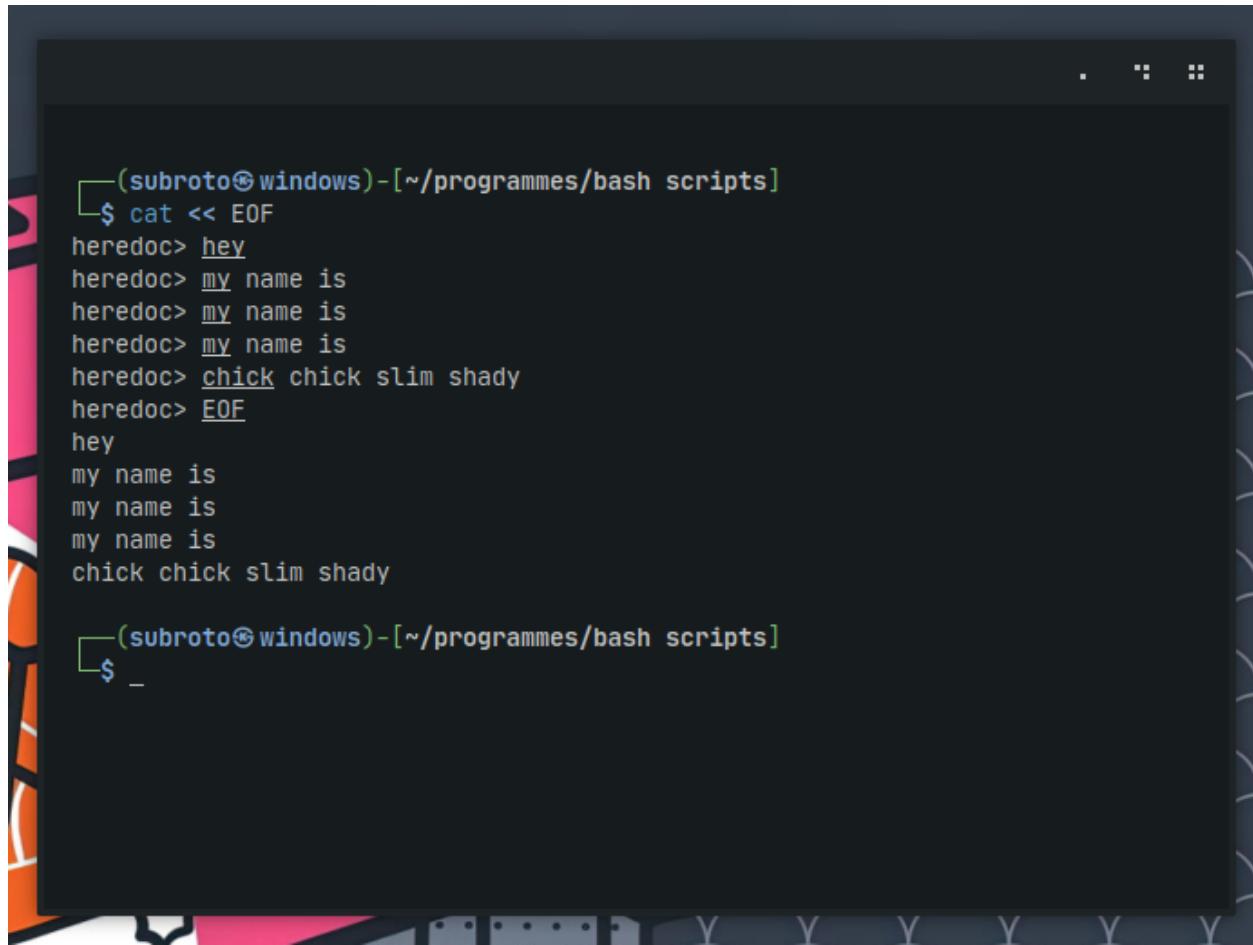
```
(subproto@windows) -[~/programmes/bash scripts]
$ wc -w h.txt
6 h.txt

(subproto@windows) -[~/programmes/bash scripts]
$ wc -w < h.txt
6

(subproto@windows) -[~/programmes/bash scripts]
$ _
```

<< command :

gives the text between first occurrence and last occurrence of the entered word



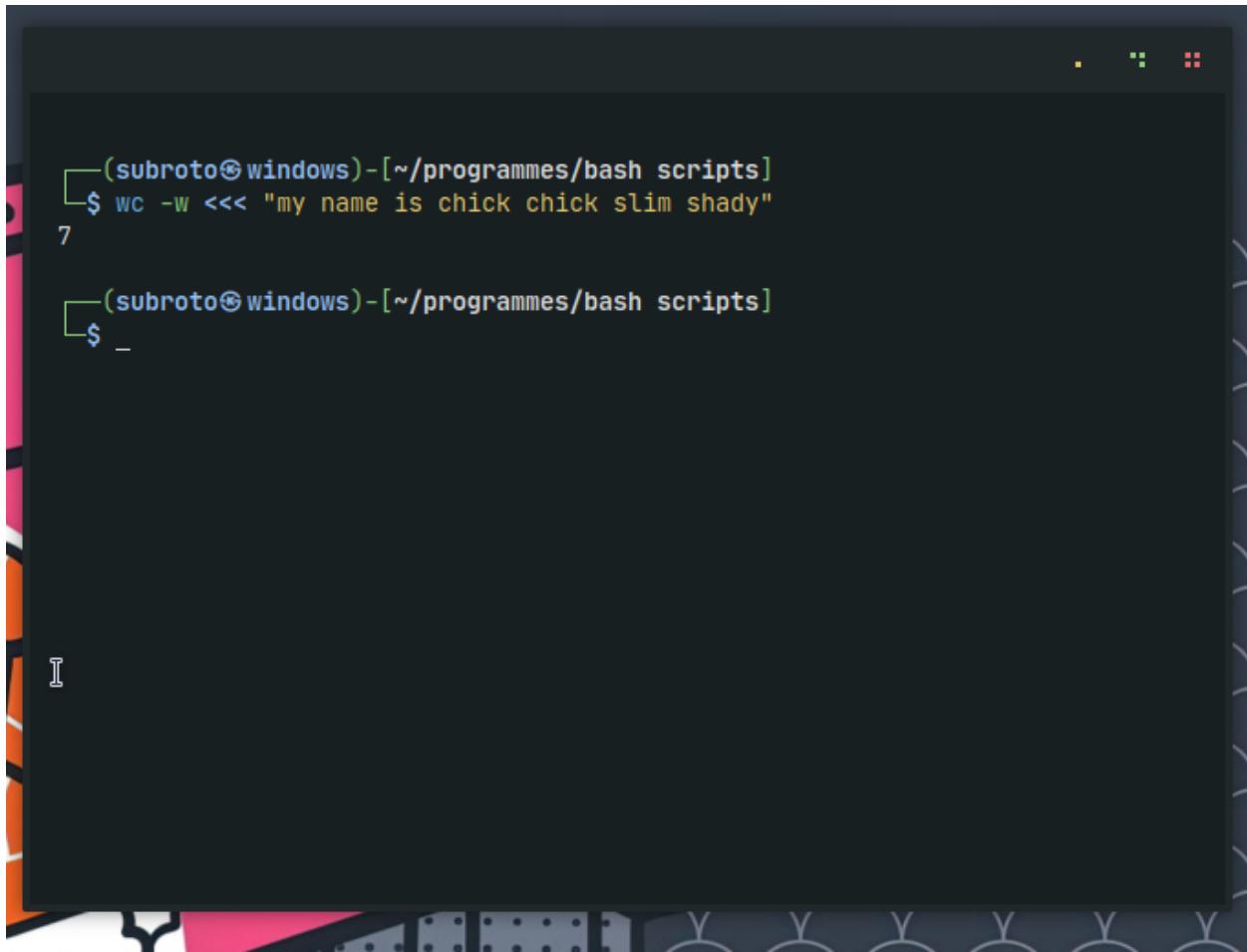
The screenshot shows a terminal window with a dark background and light-colored text. The terminal prompt is '(subproto@windows) - [~/programmes/bash scripts]'. The user enters the command '\$ cat << EOF' followed by a series of lines starting with 'heredoc>'. These lines contain the text 'hey', 'my name is', 'my name is', 'my name is', 'chick chick slim shady', and 'EOF'. After the 'EOF' marker, the output begins with 'hey' and then repeats 'my name is' three times, followed by 'chick chick slim shady'. Finally, the command '\$ _' is entered at the prompt.

```
(subproto@windows) - [~/programmes/bash scripts]
$ cat << EOF
heredoc> hey
heredoc> my name is
heredoc> my name is
heredoc> my name is
heredoc> chick chick slim shady
heredoc> EOF
hey
my name is
my name is
my name is
chick chick slim shady

(subproto@windows) - [~/programmes/bash scripts]
$ _
```

<<< command:

feeds a string into a command and the string should be between double quotes only “ ”.

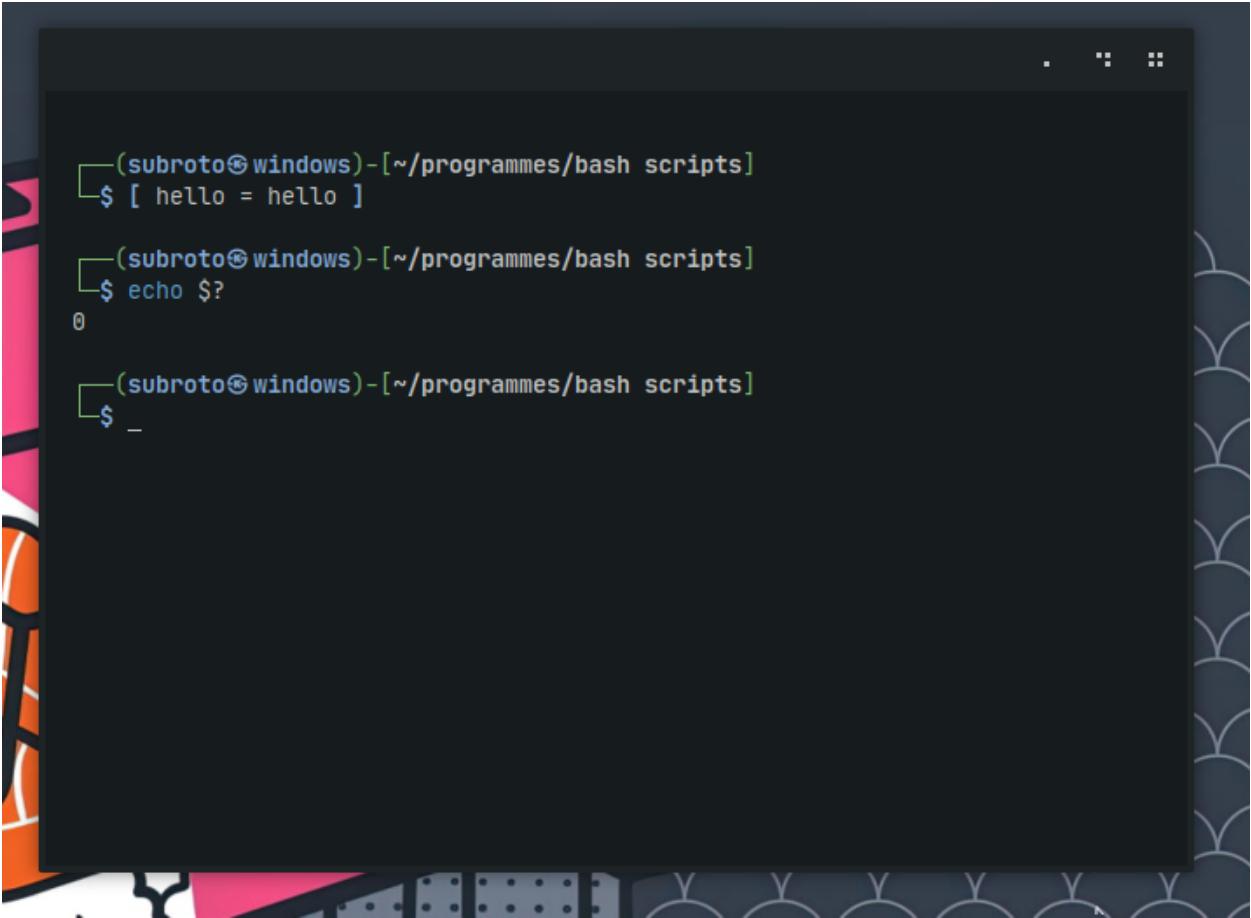
A screenshot of a terminal window titled '(subproto@windows) [~/programmes/bash scripts]'. The user has entered the command '\$ wc -w <<< "my name is chick chick slim shady"' followed by a carriage return. The terminal then displays the number '7' on a new line, indicating the word count of the input string.

here it returned the word count of the entered string as we used the command `wc -w`.

Test Operators

helps finding whether something is equal or not, comparing different values etc etc.

eg:



The screenshot shows a terminal window with a dark background and light-colored text. It displays three lines of bash script code:

```
(subroto@windows)-[~/programmes/bash scripts]
$ [ hello = hello ]
(subroto@windows)-[~/programmes/bash scripts]
$ echo $?
0
(subroto@windows)-[~/programmes/bash scripts]
$ _
```

here it returned zero because both strings are equal

```
(subproto@windows)-[~/programmes/bash scripts]
$ [ hello = hello ]

(subproto@windows)-[~/programmes/bash scripts]
$ echo $?
0

(subproto@windows)-[~/programmes/bash scripts]
$ [ 1 = 0 ]

(subproto@windows)-[~/programmes/bash scripts]
$ echo $?
1

(subproto@windows)-[~/programmes/bash scripts]
$ [ 1 -eq 1 ]

(subproto@windows)-[~/programmes/bash scripts]
$ echo $?
0

(subproto@windows)-[~/programmes/bash scripts]
$
```

If/Else/Elif


```
(subroto@windows)-[~/programmes/bash scripts]
$ vim login.sh

(subroto@windows)-[~/programmes/bash scripts]
$ ./login.sh subroto
oh you are the boss here welcome !

(subroto@windows)-[~/programmes/bash scripts]
$ ./login.sh help
just enter username, duh

(subroto@windows)-[~/programmes/bash scripts]
$ ./login.sh dipak
I dont know you, you are not the boss of me!

(subroto@windows)-[~/programmes/bash scripts]
$
```

Case Statements

```
#!/usr/bin/bash
case ${1,,} in
    subroto | administrator)
        echo "Welcome back boss!!"
        ;;
    help)
        echo "Just enter your username"
        ;;
    *)
        echo "You are not the boss of me :("
esac
```

here first case is for either subroto or administrator

second case for help

and * is for default/any other word

esac is to end the cases.

```
(subroto@windows)-[~/programmes/bash scripts]
$ vim login.sh

(subroto@windows)-[~/programmes/bash scripts]
$ chmod u+x login.sh

(subroto@windows)-[~/programmes/bash scripts]
$ ./login.sh subroto
Welcome back boss!!

(subroto@windows)-[~/programmes/bash scripts]
$ ./login.sh administrator
Welcome back boss!!

(subroto@windows)-[~/programmes/bash scripts]
$ ./login.sh help
Just enter your username

(subroto@windows)-[~/programmes/bash scripts]
$ ./login.sh dipak
You are not the boss of me :(
```

Arrays

```
(subroto@windows)-[~/programmes/bash scripts]
$ arr=(one two three)

(subroto@windows)-[~/programmes/bash scripts]
$ echo $arr
one

(subroto@windows)-[~/programmes/bash scripts]
$ echo ${arr[@]}
one two three

(subroto@windows)-[~/programmes/bash scripts]
$ echo ${arr[0]}
one

(subroto@windows)-[~/programmes/bash scripts]
$ echo ${arr[1]}
two

(subroto@windows)-[~/programmes/bash scripts]
$ echo ${arr[2]}
three

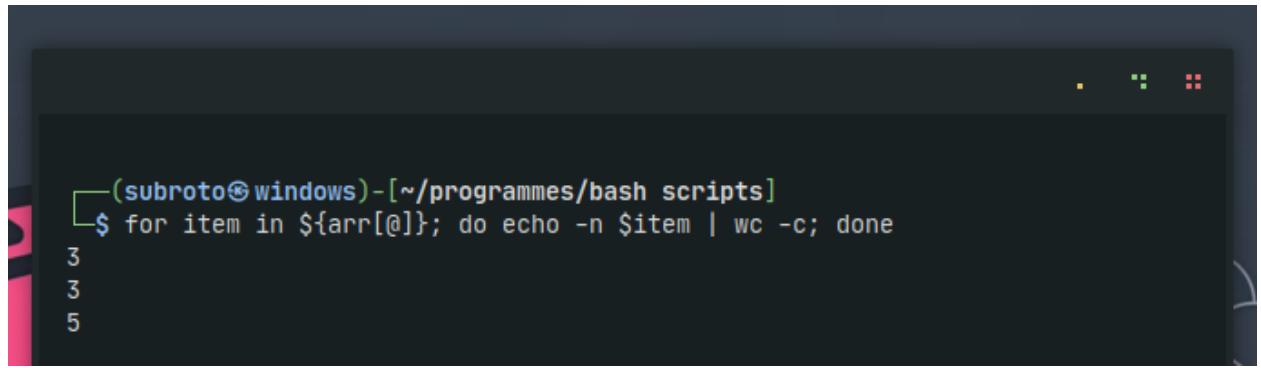
(subroto@windows)-[~/programmes/bash scripts]
$ echo ${arr[3]}

(subroto@windows)-[~/programmes/bash scripts]
$
```

first line shows how to create array

then next shows how to print it and its elements.

For LOOP



A screenshot of a terminal window on a Mac OS X desktop. The title bar says 'Terminal'. The command line shows a script being run:

```
(subroto@windows)-[~/programmes/bash scripts]
$ for item in ${arr[@]}; do echo -n $item | wc -c; done
3
3
5
```

here , do is for instructing what to do

-n means ignore the newline characters

wc -c is to return word counts

FUNCTIONS

```
#!/usr/bin/bash
showuptime(){
    up=$(uptime -p | cut -c4-)
    since=$(uptime -s)
    cat << EOF
-----
This machine has been up for ${up}
It has been running since ${since}
-----
EOF
}
showuptime
~
```

▷

```
"fun.sh" 12L, 222B          12,10      All
```

```
(subproto@windows)-[~/programmes/bash scripts]
$ vim fun.sh

(subproto@windows)-[~/programmes/bash scripts]
$ chmod u+x fun.sh

(subproto@windows)-[~/programmes/bash scripts]
$ ./fun.sh
-----
This machine has been up for 3 hours, 0 minutes
It has been running since 2023-04-22 18:23:32
-----
```

Creating local variables:

if we do not specify the variables to be local in the function, then they would be overwritten by the function

so use local to make them local to the function block only

```
#!/usr/bin/bash
up="subroto"
since="banerjee"
echo $up
echo $since
showuptime(){
    local up=$(uptime -p | cut -c4-)
    local since=$(uptime -s)
    cat << EOF
-----
This machine has been up for ${up}
It has been running since ${since}
-----
EOF
}
showuptime
echo $up
echo $since
~
```

18, 11

All

```
(subroto@windows)-[~/programmes/bash scripts]
$ vim fun.sh

(subroto@windows)-[~/programmes/bash scripts]
$ ./fun.sh
subroto
banerjee
-----
This machine has been up for 3 hours, 8 minutes
It has been running since 2023-04-22 18:23:32
-----
subroto
banerjee

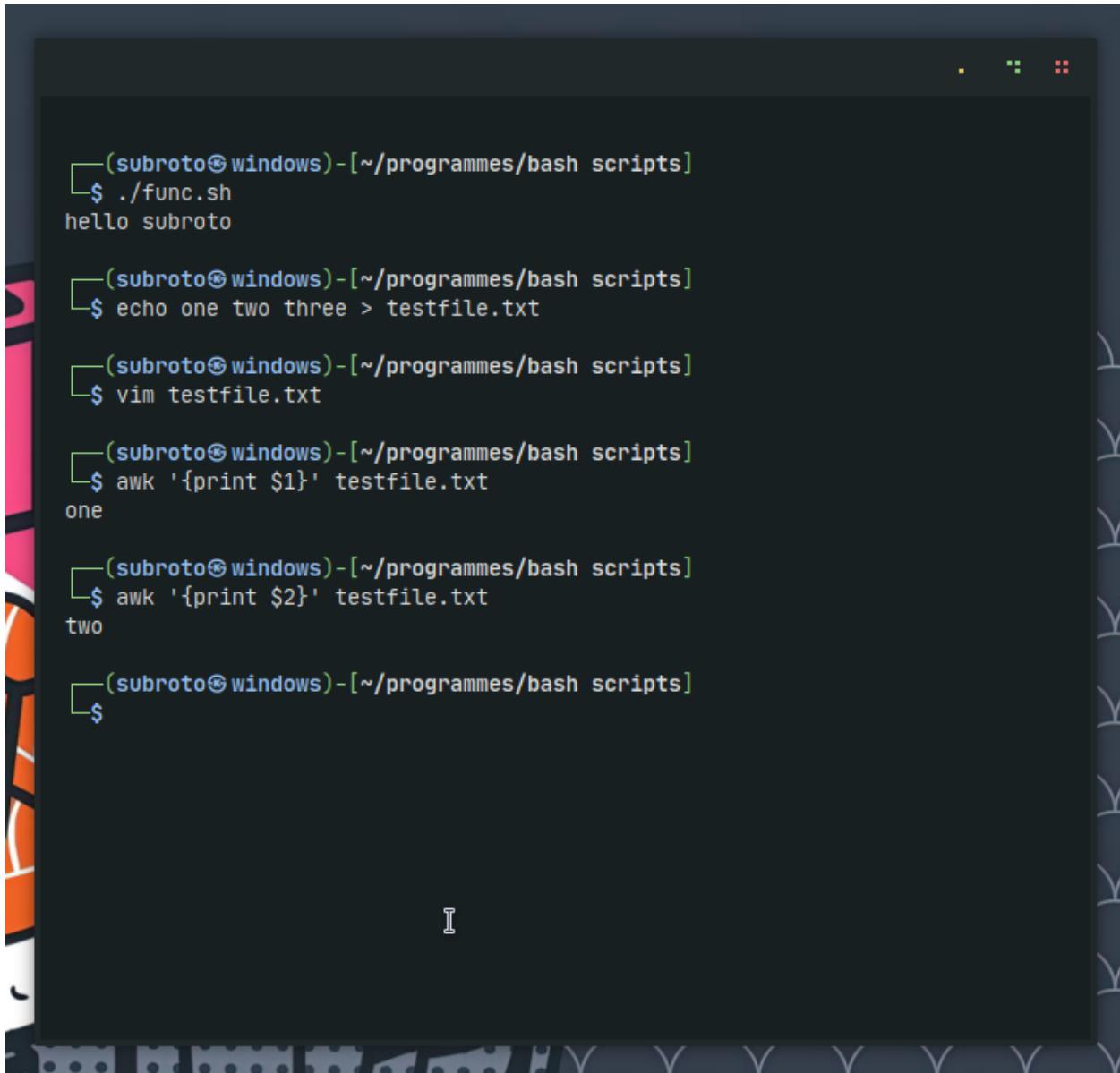
(subroto@windows)-[~/programmes/bash scripts]
$
```

Arguements in function

```
[subroto@windows] - [~/programmes/bash scripts]
$ ./func.sh
hello subroto
```

AWK

can be used to filter output of a file or a command output.



```
(subroto@windows)-[~/programmes/bash scripts]
$ ./func.sh
hello subroto

(subroto@windows)-[~/programmes/bash scripts]
$ echo one two three > testfile.txt

(subroto@windows)-[~/programmes/bash scripts]
$ vim testfile.txt

(subroto@windows)-[~/programmes/bash scripts]
$ awk '{print $1}' testfile.txt
one

(subroto@windows)-[~/programmes/bash scripts]
$ awk '{print $2}' testfile.txt
two

(subroto@windows)-[~/programmes/bash scripts]
$
```

SED

it is a command line tool that allows you to modify textfiles using simple commands

The fly flies like no fly flies.
A fly is an insect that has wings and a fly likes to eat leftovers.

"set.txt" 2L, 101B

2,67

All

```
(subroto@windows)-[~/programmes/bash scripts]
$ vim set.txt

(subroto@windows)-[~/programmes/bash scripts]
$ sed 's/fly/dick/g' set.txt
The dick flies like no dick flies.
A dick is an insect that has wings and a dick likes to eat leftovers.

(subroto@windows)-[~/programmes/bash scripts]
$ vim set.txt

(subroto@windows)-[~/programmes/bash scripts]
$
```

s means substitute

then we provide the word for which we wanna substitute, then the word which is to be substituted then g means globally , we want this to change globally in the whole file.