Introduction to Shell Scripting

- starts at 9:05 pm

Agenda

1. First script
2. Shebang
3. Script task
4. Environment Variables.
5. Comments
6. Wildcards
7. User Input
8. Control Structures

Bourne Shell → /bin/sh

Bourne Again Shell → /bin/bash.

Shebang (Hash Bang)

#! → specifying the interpreter.

#! /usr/bin/python3

#! /path to interpreter.

✓ /usr/local/bin/python3.

↓

problem if we don't know.

✓ #!/usr/bin/env python3

$PATH → : separated list of directories.

directory1: directory2 : directory3 : ...

first script

```
#!/bin/bash
echo "Hello World"
```

chmod +x script.sh 700

./ → to execute. 754

export PATH = $PATH: dir.

" " " : java/home

; → to run multiple commands.

Task.

1. create a user
2. create a group.
3. Assign the user to that group
4. Print a welcome message for that user when they login.

Break → 10:20 pm.

```
#!/bin/bash

useradd -m Vedant -s /bin/bash

groupadd developer

usermod -aG developer Vedant

echo ' echo "Welcome Vedant!"' >> /home/Vedant/.bashrc
```

Script.

→ Comments

→ single line comment

Multi line comments

→ :3,6s/^/#/ → comment
:3,6s/^#// → uncomment

→ here doc.

: << delimitter
.
.
.
delimitter

: → no-op command.

: << 'END_COMMENT'
.
.
.
END_COMMENT

Wild cards.

① *

↓

ls file *

matches all patterns.

② [ ] → ls file[1-3]

③ ? matches only 1 character

ls file?

Brace Expansion.

touch file{1..15}

touch {a, b, c, d}file

→ Rules important while creating a variable.

1. A variable name can have letters, numbers, and underscores and is case-sensitive.

2. There should be no whitespace on either side of the assignment operator(=).

3. The variable name cannot have special characters. $ @ !

4. The first character of the variable name cannot be a number

5. Variable names cannot be reserved words.(if,else,for)

6. Variable name cannot have whitespace in between.

→ User Input

read

read -p "Some text " variable

echo "$variable"

→ date

-s → silent (not printing the text)

echo -e "Hello \n Vedant "

-e → expressions.

Control structures if for while
case (switch)

if

if [condition]; then
commands
else
commands
fi

if [condition]; then
commands

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

for

```
for i in 1 2 3 4 ... ; do
    commands
done
```

```
for i in {1..5}; do
    commands
done
```

while

```
count = 1
while [condition] ; do
    commands
done
```

case

```
case $variable in
option1)
        commands
        ;;
option2)
        commands
        ;;
option3)
        command
        ;;
*)
        commands
        ;;
esac
```

```
#!/bin/bash

read -p "Enter your choice" number

case $number in

    1)
```

```
        echo "You chose 1"
        ;;
    2)
        echo "You chose 2"
        ;;
    *)
        echo "You didnt choose 1 or 2"
        ;;
esac
```

Case script

```
#!/bin/bash

for i in 1 2 3 4 5; do
    echo -e "$i\n"
done
```

For loop script