DO YOU KNOW?

BASIC SHELL SCRIPTS

FOLLOW

OSCAR MULO
LINUX EXPERT

01 HELLO WORLD SCRIPT

This is the simplest example of a shell script that prints "Hello, World!" to the terminal.

Script:

bash

```
#!/bin/bash
# This is a comment. It describes the script's purpose.
echo "Hello, World!"
```

Explanation:

- #!/bin/bash: The shebang tells the system that this script should be run in the Bash shell.
- echo: The echo command prints text to the terminal.

02 VARIABLES IN SHELL SCRIPTS

You can use variables to store and manipulate data.

Script:

```
bash
#!/bin/bash
# Define a variable
name="John"
# Use the variable
echo "Hello, $name!"
```

Explanation:

- name="John": Assigns the string "John" to the variable name.
- $name: The $ symbol is used to reference the variable's value.

03 SIMPLE CONDITIONAL STATEMENT

Shell scripts can make decisions using conditional statements (if statements).

Script:

```
bash
#!/bin/bash
# Conditional statement example
num=10
if ( $num -gt 5 ); then
echo “The number is greater than 5."
else
echo "The number is 5 or less."
fi
```

Explanation:

- if ($num -gt 5): Checks if the variable num is greater than 5 (-gt stands for "greater than").
- then and else: Executes different blocks of code depending on the result of the condition.

04 LOOPS IN SHELL SCRIPTS

Loops are used to repeat a block of code.

Script:

bash

```
#!/bin/bash
# Loop example
for i in {1..5}
do
echo "Iteration $i"
done
```

Explanation:

- for i in {1..5}: Loops through the numbers 1 to 5.
- do and done: Marks the beginning and end of the loop block.

05 USER INPUT IN SHELL SCRIPTS

You can take input from the user using the read command.

Script:

```
bash
#!/bin/bash
# Read user input
echo "Enter your name:"
read user_name
echo "Hello, $user_name!"
```

Explanation:

- read user_name: Prompts the user for input and stores it in the variable user_name.

06 BASIC ARITHMETIC IN SHELL SCRIPTS

You can perform arithmetic operations using the expr command or double parentheses.

Script:

bash

```
#!/bin/bash
# Arithmetic operations
num1=5
num2=3
sum=$((num1 + num2))
echo "The sum of $num1 and $num2 is $sum"
```

Explanation:

- $((num1 + num2)): Performs arithmetic between two variables.
- sum: Stores the result of the addition.

07 FUNCTIONS IN SHELL SCRIPTS

You can define functions in shell scripts to reuse code.

Script:

```
bash
#!/bin/bash
# Define a functiongreet() {
  echo "Hello, $1!"
}

# Call the function
greet "Alice"
greet "Bob"
```

Explanation:

- greet(): Defines a function named greet.
- $1: Refers to the first argument passed to the function.

08 FILE EXISTENCE CHECK

You can check if a file exists before performing operations.

Script:

```
bash
#!/bin/bash
# Check if a file exists
file="testfile.txt"
if ( -e "$file" ); thenecho "$file exists."
else
echo "$file does not exist."
fi
```

Explanation:

- -e: Checks if the file exists.

09 COMMAND LINE ARGUMENTS

You can pass arguments to a shell script when you run it.

Script:

bash

```
#!/bin/bash
# Using command-line arguments
echo "The script name is $0"
echo "First argument: $1"
echo "Second argument: $2"
```

Explanation:

- $0: The name of the script.
- $1, $2: The first and second arguments passed to the script.

10 BASIC FILE OPERATIONS

You can perform operations on files like creating, copying, and moving them.

Script:

```
bash
#!/bin/bash
# File operations example
touch newfile.txt  # Creates a new file
cp newfile.txt backupfile.txt  # Copies the file
mv newfile.txt renamedfile.txt  # Renames (or moves) the file
rm renamedfile.txt  # Deletes the file
```

Explanation:

- touch: Creates a new file.
- cp: Copies a file.
- mv: Renames or moves a file.
- rm: Deletes a file.

11 CRON JOB EXAMPLE (AUTOMATING A TASK)

You can schedule the execution of shell scripts using cron. Here's an example of a shell script that could be scheduled with cron.

Script:

```
bash
#!/bin/bash
# Backup example
tar -czf /backup/backup.tar.gz
/home/user/documents
echo "Backup completed on $(date)" >>
/var/log/backup.log
```

Explanation:

- tar: Archives and compresses files.
- $(date): Inserts the current date and time.