LAB 2 .md

Objective: Understand how existing scripts in repo work.

📄 Table of Contents

Script 1: print_numbers.sh Script 2: array_loop.sh

Script 1: print_numbers.sh

📝 Purpose This script prints numbers from 1 to 10 using a simple for loop.

Script Code

```
#!/bin/bash
# Script to print numbers from 1 to 10

for i in {1..10}
do
 echo "Number: $i"
done
```

· Line-by-Line Explanation #!/bin/bash → Shebang, tells the system to use Bash to run the script. #Script to print numbers from 1 to 10 → A comment explaining what the script does. for i in {1..10} → Loop starts, i takes values from 1 to 10. do → Marks the beginning of loop commands. echo “Number: $i” → Prints the current number with a message. done → Ends the loop.

▶️ Example Run Command: bash print_numbers.sh Output: Number: 1 Number: 2 Number: 3 Number: 4 Number: 5 Number: 6 Number: 7 Number: 8 Number: 9 Number: 10

Script 2: array_loop.sh

📝 Purpose This script loops through an array of fruit names and prints each one.

Script Code

```
#!/bin/bash
#Script to print all items in an array

fruits=("Apple" "Banana" "Cherry" "Mango")

for fruit in "${fruits[@]}"
do
 echo "Fruit: $fruit"
done
```

Line-by-Line Explanation #!/bin/bash → Tells the system to run the script with Bash. #Script to print all items in an array → A comment. fruits=(“Apple” “Banana” “Cherry” “Mango”) → Declares an array with four fruit names. for fruit in ”${fruits[@]}” → Loop goes through each item in the array. do → Starts the commands for each loop iteration. echo “Fruit: $fruit” → Prints the current fruit. done → Ends the loop.

▶️ Example Run Command: bash array_loop.sh Output: Fruit: Apple Fruit: Banana Fruit: Cherry Fruit: Mango

Extra Questions

1. What is the purpose of #!/bin/bash at the top of a script?

- This line is called a shebang (or hashbang). It tells the system which interpreter should be used to run the script.

#!/bin/bash means the script should be executed using the Bash shell (located at /bin/bash). Without it, the system might use a different shell (like sh or dash), and some commands may not work as expected. 👉 Example: #!/bin/bash echo “Hello, World!” If you run this script, the OS knows to use Bash to interpret it.

2. How to make a script executable

- Suppose your script file is named myscript.sh. Give it execute permission: chmod +x

myscript.sh chmod = change file permissions +x = add execute permission Run the script: ./myscript.sh (./ means “run from current directory”). ⚡ Quick summary:

#!/bin/bash → ensures the script runs with Bash. chmod +x script.sh → makes it executable.