

LAB2 – Script Execution & Explanation

📝 Script 1: print_numbers.sh

◊ Purpose:

This script prints numbers from 1 to 5 using a loop.

🔍 Script Code:

```
#!/bin/bash

for i in {1..5}
do
    echo "Number: $i"
done
```

Line	Explanation
`#!/bin/bash`	Shebang line - tells the OS to run the script using Bash shell.
`for i in {1..5}`	Loops through numbers 1 to 5 and assigns each to variable `i`.
`do`	Begins the body of the loop.
`echo "Number: \$i"`	Prints the current value of `i`.
`done`	Ends the loop.

▶ Example Run:

Command:

```
bash print_numbers.sh
```

Output:

```
Number: 1
Number: 2
Number: 3
Number: 4
Number: 5
```

📝 Script 2: array_loop.sh

◊ Purpose:

This script demonstrates how to loop through an array **in** Bash.

🔍 Script Code:

```
#!/bin/bash

fruits=("apple" "banana" "cherry")

for fruit in "${fruits[@]}"
do
    echo "I like $fruit"
done
```

🧠 Line-by-Line Explanation:

Line	Explanation
#!/bin/bash	Specifies that the script should be run with Bash shell.
fruits=("apple" "banana" "cherry")	Declares an array with 3 elements.
for fruit in "\${fruits[@]}"	Loops through all elements in the array.
do	Starts the loop block.
echo "I like \$fruit"	Prints each fruit name with a message.
done	Ends the loop block.

▶ Example Run:

Command:

```
bash array_loop.sh
```

Output:

```
I like apple
I like banana
I like cherry
```

❓ Extra Questions

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

The line `#!/bin/bash` is called a shebang. It tells the operating system to use the Bash shell to interpret and run the script. Without it, the system may default to another shell like sh, which may behave differently.

How **do** you make a script executable?

You can make a script executable using the `chmod` command:

```
chmod +x script_name.sh
```

After that, you can run it directly like this:

```
./script_name.sh
```

📄 Final Deliverables

LAB2.md (this file)

Convert to LAB2.pdf