

LAB2 – Shell Scripting Basics

This lab demonstrates running and analyzing two simple shell scripts from the `Scripts/` folder:

- `print_numbers.sh`
- `array_loop.sh`

We will explore their purpose, line-by-line explanations, and example outputs.

♦ Script 1: `print_numbers.sh`

Purpose

To print numbers from **1 to 5** using a simple loop.

Code Snippet (with output)

```
sneha@sneha-HP-Laptop-15s-fq5xxx:~/LINUX_LAB/scripts$ cat array1_scripts.sh
#!/bin/bash
fruits=("apple" "banana" "cherry")
length=${#fruits[@]}
echo "Array length:$length"
i=0
while [ $i -lt $length ] ; do
    echo "Fruit: ${fruits[$i]}"
    ((i++))
done
sneha@sneha-HP-Laptop-15s-fq5xxx:~/LINUX_LAB/scripts$ ./array1_scripts.sh
Array length:3
```

Line-by-Line Explanation

Line	Code	Explanation
1	<code>#!/bin/bash</code>	Shebang – tells the system to use the Bash shell.
2	<code>fruits=("apple" "banana" "cherry")</code>	Creates an array named <code>fruits</code> with 3 elements: <code>apple</code> , <code>banana</code> , and <code>cherry</code> .
3	<code>length=\${#fruits[@]}</code>	Finds the length of the array using <code>\${#array[@]}</code> . Here, <code>length = 3</code> .
4	<code>echo "Array length:\$length"</code>	Prints the length of the array.
5	<code>i=0</code>	Initializes the loop counter <code>i</code> to 0 (first index).
6	<code>while [\$i -lt \$length] ; do</code>	Starts a <code>while</code> loop that runs while <code>i</code> is less than the array length .
7	<code>echo "Fruit: \${fruits[\$i]}"</code>	Prints the element of the array at index <code>i</code> .

Line	Code	Explanation
8	<code>((i++))</code>	Increments <code>i</code> by 1 after each iteration.
9	<code>done</code>	Marks the end of the <code>while</code> loop.

◆ Script 2: `array_loop.sh`

🎯 Purpose

To demonstrate a for loop with a numeric range in Bash.

📄 Code Snippet (with output)

```
sneha@sneha-HP-Laptop-15s-fq5xxx:~/LINUX_LAB/scripts$ cat range.sh
#!/bin/bash

for i in {0..7}
do echo "Number: $i"
done
sneha@sneha-HP-Laptop-15s-fq5xxx:~/LINUX_LAB/scripts$ ./range.sh
Number: 0
Number: 1
Number: 2
Number: 3
Number: 4
Number: 5
Number: 6
Number: 7
```

📖 Line-by-Line Explanation

Line	Code	Explanation
1	<code>#!/bin/bash</code>	Shebang → ensures the script runs with the Bash shell.
2	<code>(blank)</code>	Just an empty line for readability.
3	<code>for i in {0..7}</code>	Starts a <code>for</code> loop. The loop variable <code>i</code> will take values from <code>0</code> to <code>7</code> (inclusive).
4	<code>do</code>	Marks the beginning of the loop body.
5	<code>echo "Number: \$i"</code>	Prints the current value of <code>i</code> prefixed with <code>"Number: "</code> .
6	<code>done</code>	Ends the <code>for</code> loop.

? Extra Questions

- Q1. What is the purpose of `#!/bin/bash`?
- It is called the shebang. It tells the system to run the script using the Bash shell, ensuring correct execution.
- Q2. How do you make a script executable?

- `chmod +x scriptname.sh`
- Example:

```
chmod +x print_numbers.sh  
./print_numbers.sh
```

✨ Summary

- `print_numbers.sh` → Demonstrates a basic for loop.
- `array_loop.sh` → Demonstrates using arrays in Bash.
- `#!/bin/bash` → Ensures the script runs in the Bash shell.
- `chmod +x` → Makes a script executable.