🚀 EXPERIMENTING WITH SHELL SCRIPT

📝 Assignment 3 – Modify an Existing Script

Objective: Enhance and customize a script by adding user input and validation.

📌 Task 1: Original Script (`print_numbers.sh`)

🎯 Purpose

The original script simply prints numbers from **1 to 5** in sequence.

📜 Original Script Code

```
#!/bin/bash
# print_numbers.sh

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

▶️ Original Behavior

- Always prints numbers from **1 to 5**.
- No user input, no flexibility.

Example Run:

```
./print_numbers.sh
```

Output:

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

```
  GNU nano 7.2                    print_numbers.sh
#!/bin/bash
#print_numbers.sh


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

```
┌─[✗]─[mukund@parrot]─[~/linux]
└──╼ $chmod  7777 print_numbers.sh
┌─[mukund@parrot]─[~/linux]
└──╼ $./print_numbers.sh
Number : 1
Number : 2
Number : 3
Number : 4
Number : 5
```

📌 Task 2: Enhanced Script (`enhanced_numbers.sh`)

🎯 Purpose

Enhance the script so the user provides `start`, `end`, and `step values`. The script validates inputs (step must be positive).

📜 Enhanced Script Code

```
#!/bin/bash
# enhanced_numbers.sh

# Check if 3 arguments are provided
if [ $# -ne 3 ]; then
  echo "Usage: $0 <start> <end> <step>"
  exit 1
fi

start=$1
end=$2
step=$3

# Validate step
if [ $step -le 0 ]; then
  echo "Error: Step must be a positive number."
  exit 1
fi

# Loop through the range
for ((i=start; i<=end; i+=step))
do
  echo "Number: $i"
done
```

```
#!/bin/bash
# enhanced_numbers.sh

# Check if 3 arguments are provided
if [ $# -ne 3 ]; then
  echo "Usage: $0 <start> <end> <step>"
  exit 1
fi

start=$1
end=$2
step=$3

# Validate step
if [ $step -le 0 ]; then
  echo "Error: Step must be a positive number."
  exit 1
fi

# Loop through the range
for ((i=start; i<=end; i+=step))
do
  echo "Number: $i"
done
```

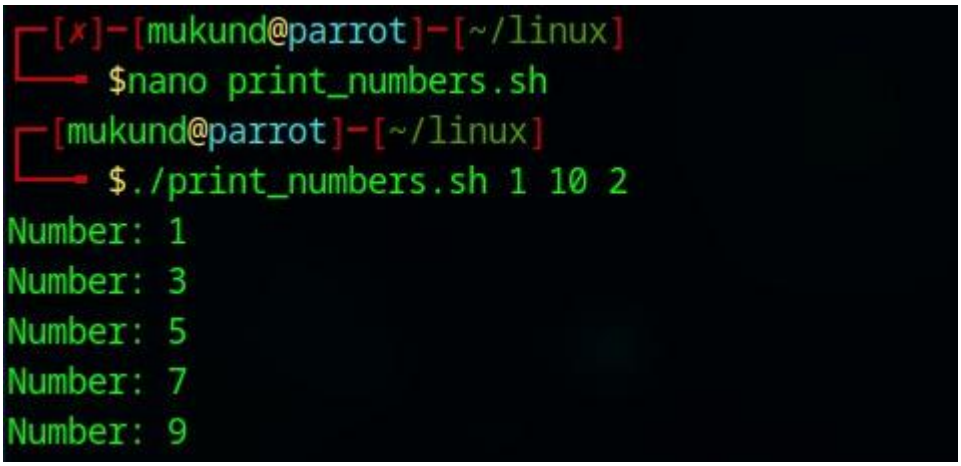
Example Run:

Run 1: Normal case

```
./enhanced_numbers.sh 1 10 2
```

Output:

```
Number: 1
Number: 3
Number: 5
Number: 7
Number: 9
```

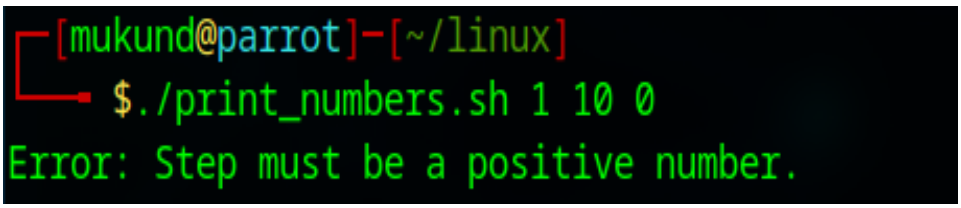


Run 2: Invalid step

```
./enhanced_numbers.sh 1 10 0
```

Output:

```
Error: Step must be a positive number.
```



❓ Extra Questions

Q1: Difference between `$1`, `$@`, and `$#` in Bash?

- ***`$1`*** → the first argument passed to the script.
- ***`$@`*** → all arguments as a list (`"$1" "$2" "$3"...`).
- ***`$#`*** → the number of arguments passed to the script.

Q2: What does `exit 1` mean in a script?

- `exit` ends the script immediately.
- `exit 0` means **successful execution**.
- `exit 1` means the script **failed** (an error occurred).
- Other numbers can also be used for different **error codes**.