CSE 310: Operating System Lab
Lab 04

Bash Scripting (Part - 2)

Prepared By: Ridwan Mahbub, Lecturer, CSE, BUBT

Description:

Welcome to Lab-4! In this lab, we will dive deeper into Bash scripting and explore some fundamental concepts that are essential for creating robust and efficient scripts. We will focus on case statements, loops, and loop control mechanisms, which will allow you to tackle more complex tasks and automate repetitive processes effectively.

Lab Objectives:

1. Master the use of case statements for decision-making in Bash scripts.
2. Understand the concept of loops and their significance in scripting.
3. Learn about loop control mechanisms to manage loop behavior effectively.

Lab Content:

If else Statements in Bash:

- Introduction to decision statements and their role in decision-making.
- Syntax of if else statements.
- Differences from typical programming languages.

```bash
 #!/bin/bash

read -p "Enter a number: " num

if ((num > 0)); then
    echo "The number is positive."
elif ((num < 0)); then
    echo "The number is negative."
else
    echo "The number is zero."
fi
```

```bash
#!/bin/bash

read -p "Enter a number: " num

if [ $num -gt 0 ]
 then
    echo "The number is positive."
elif [ $num -lt 0 ]
 then
    echo "The number is negative."
else
    echo "The number is zero."
fi
```

Case Statements in Bash:

- Introduction to case statements and their role in decision-making.
- Syntax of the case statement and its components.
- Using patterns to match against a variable or expression.
- Executing code blocks based on matching patterns.
- Handling default cases using the *) pattern.
- Practical examples demonstrating the application of case statement.

```bash
#!/bin/bash

read -p "Enter a fruit name: " fruit

case $fruit in
    "apple")
        echo "It's a common fruit."
        ;;
    "orange")
        echo "It's a citrus fruit."
        ;;
    "banana")
        echo "It's a yellow fruit."
        ;;
    *)
        echo "Unknown fruit."
        ;;
esac
```

Loops in Bash:

- Understanding the concept of loops and their importance in automation.
- Introduction to different types of loops: while, for, and until.
- Syntax and usage of while loops to execute code based on a condition.
- Syntax and usage of for loops for iterating over lists of elements.
- Syntax and usage of until loops to execute code until a condition becomes true.
- Real-world scenarios demonstrating the use of different types of loops.

```
#!/bin/bash

counter=1

while [ $counter -le 5 ]
do
    echo "Count: $counter"
    ((counter++))
done
```

```
#!/bin/bash

fruits=("apple" "orange" "banana" "grape" "watermelon")

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

Loop Control Mechanisms:

- Controlling loop behavior with break and continue statements.
- Using break to exit a loop prematurely.
- Using continue to skip the current iteration and proceed to the next.
- Understanding nested loops and controlling them efficiently.
- Best practices for using loop control mechanisms.

```
 #!/bin/bash

for number in {1..10}
do
    if [ $number -eq 6 ]
 then
        echo "Reached the limit. Exiting the loop."
        break
    fi
    echo "Number: $number"
done
```

```bash
#!/bin/bash

for number in {1..5}
do
    if [ $number -eq 3 ]
 then
        echo "Skipping number 3 and continuing to the next iteration."
        continue
    fi
    echo "Number: $number"
done
```

Conclusion:

In this lab, you have learned about some powerful concepts in Bash scripting: case statements, loops, and loop control. You now have the knowledge and skills to make complex decisions in your scripts using case statements and efficiently automate repetitive tasks with loops. The ability to control loop behavior using break and continue statements will further enhance your scripting capabilities.

With this new knowledge, you are better equipped to tackle a wide range of scripting challenges and become a more proficient Bash scripter. Continue practicing and exploring these concepts to sharpen your scripting skills and become a more effective Linux enthusiast or administrator.