

# Intro to Linux



## 3.1.1 - Shell Script Elements Part 1

# Shell Script Elements

In shell scripting, there are many loops, conditionals, variables, and functions



# Loops

- A “while” loop executes a set of commands if a specified condition is true

```
while [ condition ]; do  
    commands  
done
```

- A “for” loop iterates over a sequence, like a range of numbers, and performs commands

```
for variable in {start...end...step}; do  
    commands  
done
```

- An “until” loop executes a set of commands if the specified condition is false

```
until [ condition ]; do  
    commands  
done
```



# Conditionals

Control structures that allow one to make decisions and execute different sets of commands based on true/false conditions

- An if statement executes a set of commands based on the evaluation of a condition

```
if [ condition ]; then
    commands
fi
```

- Switch/Case provides multiple possible execution paths based on the value of an expression

```
case $variable in
    pattern1)
        commands
        ;;
    pattern2)
        commands
        ;;
    *)
        default commands
        ;;
esac
```



# Shell Parameters

A feature in shell scripting that allows one to manipulate and expand the values of variables

- Extract substrings, perform pattern matching, etc.
- Globbing matches filenames with patterns
- Brace expressions generate arbitrary strings using curly braces

```
files=*.txt
```

```
echo {1..5}
```



# Shell Comparisons

Essential for making decisions based on the values of variables or the success/failure of commands

- Using double parentheses, arithmetic operations can be performed

```
if (( num1 > num2 )); then  
    commands  
fi
```

- Strings can be compared with operators like == (the same) or != (not the same)

```
if [ "$str1" == "$str2" ]; then  
    commands  
fi
```



# Boolean Logic

- Shell scripting doesn't have a native boolean type but still uses boolean logic
- Users can combine conditions using logical operators, && for AND, || for OR

```
if [ condition1 ] && [ condition2 ]; then
    commands
fi
```



# Shell Variables and Search and Replace

- Used to store and manipulate data
- Act as placeholders that can be referenced or modified within a script

```
variable_name="value"
```

- If a piece of text needs to have a part replaced, "search and replace" does that

```
result=${string/old/new}
```





# Regular Expressions

- Regex or regexp are powerful patterns used for matching character combinations with strings
- Wildly used for tasks like string manipulation, text parsing, and pattern matching

