

Linux (Continue)

Shell Scripting II: Conditions & Loops

Overview & Learning Outcomes

This lab you will:

- Review **if-statements** and **test** conditions
- Write **for-loops** to iterate over lists and files
- Write **while-loops** for repeated tasks and menus
- Use **case** for simple multi-choice logic

Review: "if" & "test"

Basic structure:

```
if [ CONDITION ]; then
    commands
elif [ OTHER_CONDITION ]; then
    other commands
else
    default commands
fi
```

Remember:

- Spaces around [and]
- Use quotes around variables: "\$VAR"

Loops **"for"**

Loop over numbers:

```
for N in {1..5}; do  
    echo "N = $N"  
done
```

Loop over a fixed list:

```
for ITEM in one two three; do  
    echo "$ITEM"  
done
```

Loop over files:

```
for FILE in *.txt; do  
    echo "Processing $FILE"  
done
```

Loop over command line output:

```
for FILE in $(ls); do  
    echo "File: '$FILE'"  
done
```

Loops **"while"**

Basic structure:

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

Example (counting):

```
COUNT=1  
while [ "$COUNT" -le 5 ]; do  
    echo "COUNT=$COUNT"  
    COUNT=$((COUNT + 1))  
done
```

"case" Statement

Useful for multi-choice logic:

```
case "$CHOICE" in
  1)      echo "Option 1" ;;
  2)      echo "Option 2" ;;
  q|Q)    echo "Quit" ;;
  *)      echo "Invalid option" ;;
esac
```

NOTE: Patterns can match multiple values (**q|Q**)


Combining Conditions & Loops

Example: check many files:

```
for FILE in *.log; do  
    if [ -f "$FILE" ]; then  
        echo "Found log: $FILE"  
    fi  
done
```

Example: simple menu loop:

```
while true; do  
    echo "1) Show date"  
    echo "2) Show uptime"  
    echo "q) Quit"  
  
    read -p "Choice: " CHOICE  
    # TODO  
  
done  
  
# TODO: Use case to handle $CHOICE here
```





Thank You
