Operating Systems Laboratory (CSE 4510)

Week-3: Bash Shell Scripting

---

1. What is a Shell Script?
   - A shell script is a text file containing Linux commands that run automatically.

Concept	Meaning
Shell	Program that executes commands (Bash, Zsh etc.)
Script	File containing commands
Extension	.sh
Interpreter	#!/bin/bash

2. Creating & Running a Script

Step	Command	Explanation
1	nano hello.sh or touch hello.sh	Create file
2	Add inside →	#!/bin/bash echo "Hello UIU Students!"
3	chmod +x hello.sh	Make script executable
4	./hello.sh or bash file_name.ext	Run script

3. Variables in Bash

3.1 String Variable Initialization

Type	Example	Output
Set string	name="UIU Student"	—
Print string	echo $name	UIU Student
String + variable	echo "Welcome, $name"	Welcome, UIU Student
Concatenation	full="$name Dept"	—

3.2 Integer Variable Initialization

Type	Example
Integer	x=10
Print	echo $x
With text	echo "Value: $x"

4. Mathematical Expressions (All Methods)

Method	Example	Output
Double Parentheses	echo $((5+3))	8
expr command	expr 5 + 3	8
let command	let x=5+3 ; echo $x	8
Using variables	a=5; b=3; echo $((a+b))	8
Increment/Decrement	((x++)), ((x--))	—

5. Echo Commands (All Forms)

Purpose	Example	Output
Print text	echo "Hello"	Hello
Print variable	echo $name	Value of name
Text + variable	echo "Hello $name"	Hello UIU
Variable inside text	echo "Name: ${name}"	Name: UIU
Escape characters	echo -e "A\nB"	A (newline) B
No newline	echo -n "Loading..."	Loading...

6. Command Line Arguments

Bash	Run	Output
#!/bin/bash **echo "Hello $1 $2"**	bash file.sh Enamul Haque	Hello Enamul Haque

Special Argument Variables

Variable	Meaning
$0	Script name
$1, $2	First, second argument
$@	All arguments
$#	Total number of arguments
$?	Exit status of last command

7. Special Variable: $? (Exit Status):
Code 0 (mean success), Code non-zero number (mean error)
Example:

ls file.txt echo $?	If ls file.txt execute correctly, it echo will provide 0 else other non-zero number.

8. Comparison Operators

8.1 Numeric

Operator	Meaning
-lt	Less than
-gt	Greater than
-le	Less or equal
-ge	Greater or equal
-eq	Equal
-ne	Not equal
Example	if [$x -gt 10]

8.2 Logical Operators

Operator	Meaning
-a	AND
-o	OR
Example	if [$x -gt 10 -a $y -lt 5]

8.3 Logical Expression Writing process

Using square brackets with $	x=12 y=2 if [$x -gt 10 -a $y -lt 5] then echo “$x is greater than 10, $y is less than 5” fi
Using double parenthesis with $	x=12 y=2 if (($x > 10 && $y < 5)) then echo "$x is greater than 10, $y is less than 5" fi
Using double parenthesis without $	x=10 y=2 if ((x > 10 && y < 5)) then echo "$x is greater than 10, $y is less than 5" fi

8.3 User input from prompt

Way 1	read -p "Enter number: " n
Way 2	echo -n "Enter number: " read n

9. File Test Operators

Flag	Meaning	Example	True When
-f	Regular file	[-f sample.txt]	File exists
-d	Directory	[-d myfolder]	Directory exists
-r	Readable	[-r file]	Read permission exists
-w	Writable	[-w file]	Write allowed
-x	Executable	[-x script.sh]	Execute allowed

10. If-Else Statements with Math Expressions

Syntax

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

Example

```
read -p "Enter number: " n

if [ $n -lt 10 ]
then
   echo "Small"
elif [ $n -le 50 ]
then
   echo "Medium"
else
   echo "Large"
fi
```

11. Example

1) Even Odd

```
read -p "Enter number: " n

if (( n % 2 == 0 ))
then
   echo "Even"
else
   echo "Odd"
fi
```

2) File Checker Using -f, -d, -x, -r

```
read -p "Enter path: " p

if [ -f $p ]
then
   echo "It is a file"
elif [ -d $p ]
then
   echo "It is a directory"
else
   echo "Not found"
fi
```

Practice Problems

1) Grade Checker
Input marks:
- ≥ 80 → A
- ≥ 60 → B
- ≥ 40 → C
- Else → Fail

2) Age Category
Input age:
- < 13 → Child
- < 20 → Teenager
- < 60 → Adult
- Else → Senior