Lesson Plan

Introduction to Bash Scripting

PW SKILLS

Introduction to Bash Scripting

- Bash is a Unix shell and command language. It is widely available on various operating systems, and it is also the default command interpreter on most Linux systems.

- Bash stands for Bourne-Again SHell. As with other shells, you can use Bash interactively directly in your terminal, and also, you can use Bash like any other programming language to write scripts. This course will help you learn the basics of Bash scripting including Bash Variables, User Input, Comments, Arguments, Arrays, Conditional Expressions, Conditionals, Loops, Functions, Debugging, and testing.

Why Bash Scripting:

1. **Automation:**
   - **Repetitive Tasks:** Bash scripts allow you to automate repetitive tasks, saving time and reducing the risk of human error.
   - **System Administration:** Automate routine maintenance, backups, and other system administration tasks.
2. **Cross-Platform Compatibility:**
   - **Unix-Like Systems:** Bash is available on most Unix-like systems, including Linux and macOS, making scripts portable across different environments.
3. **Integration with the Command Line:**
   - **Command Line Interface (CLI):** Bash scripts can easily interact with the command line, run other scripts or programs, and manage files and processes.
   - **Interactive Scripts:** Bash scripts can prompt users for input, making them interactive and flexible.
4. **Cost-Effective:**
   - **Free and Open Source:** Bash is freely available and open-source, making it accessible for anyone to use and modify.

Steps to Create and Run a Simple Bash Script

1. Create a Script File:

- Open a text editor.
- Type your commands, starting with #!/bin/bash at the top to specify the script should be run with Bash.

```
#!/bin/bash
echo "Hello, World!"
```

- Save the file with a '.sh' extension, like 'my_script.sh'.

2. Make the Script Executable:

- In the terminal, navigate to the directory where your script is saved.
- Make the script executable by typing:

```
chmod +x my_script.sh
```

3. Run the Script:

- Execute the script by typing:

```
./my_script.sh
```

- You should see the output: 'Hello, World!'

Basic Bash Script Components

- **Variables: Store data like this:**

```
NAME="Alice"
echo "Hello, $NAME"
```

- **User Input: Ask the user for input:**

```
echo "Enter your name:"
read NAME
echo "Hello, $NAME"
```

- **If Statements: Make decisions in your script:**

```
if [ $NAME == "Alice" ]; then
    echo "Hi, Alice!"
else
    echo "You are not Alice."
fi
```

- **Loops: Repeat tasks:**

```
for i in 1 2 3
do
    echo "Number $i"
done
```

Running and Debugging

- Run your script: ‘./my_script.sh’
- Debug your script: Add ‘-x’ when running: bash ‘-x my_script.sh’

PW SKILLS

THANK YOU !