Bash Scripting: Step-by-Step (Basic to Advanced) with Real-World Examples

Absolutely! Here's a **comprehensive guide** to **Bash scripting from basic to advanced**, explained **step-by-step**, each topic with **5 real-world examples** complete with **code snippets**. This is everything you need in **one go** to master Bash scripting practically.

🧠 Bash Scripting: Step-by-Step (Basic to Advanced) with Real-World Examples

---

🔷 SECTION 1: Getting Started

1.1 Hello World & Execution

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

How to run:

```
chmod +x script.sh
./script.sh
```

✅ Real-World Examples:

1. **Print system time**

```
echo "Current Time: $(date)"
```

2. **Welcome message with username**

```
echo "Welcome, $USER!"
```

3. **Show current directory**

```
echo "You're in: $(pwd)"
```

4. **Show disk usage**

```
df -h
```

5. **Display logged-in users**

```
who
```

🔷 SECTION 2: Variables and Inputs

2.1 Variables

```
name="Alice"
echo "Name is $name"
```

✅ Real-World Examples:

1. **Set filename variable**

```
file="report.txt"
echo "Processing $file"
```

2. **Calculate sum**

```
a=5; b=3
sum=$((a + b))
echo "Sum: $sum"
```

3. **Store current date**

```
today=$(date +%F)
echo "Today: $today"
```

4. **Set home path**

```
home_dir=$HOME
echo "Home: $home_dir"
```

5. **Concatenate strings**

```
first="Dev"; last="Ops"
echo "$first$last"
```

2.2 User Input

```
read -p "Enter your name: " user
echo "Hello, $user"
```

✅ Real-World Examples:

1. **Enter filename to create**

```
read -p "File name: " fname
touch "$fname"
```

2. **Input age to determine category**

```
read -p "Enter age: " age
[ $age -gt 18 ] && echo "Adult" || echo "Minor"
```

3. **Choose from menu**

```
echo "1. Start 2. Stop"
read choice
```

4. **Get confirmation**

```
read -p "Continue (y/n)? " yn
[[ $yn == "y" ]] && echo "Go!" || echo "Abort"
```

5. **Read password**

```
read -s -p "Password: " pass
echo
```

🔷 SECTION 3: Conditional Statements

3.1 If-Else

```
if [ $age -ge 18 ]; then
  echo "Eligible"
else
  echo "Not eligible"
fi
```

✅ Real-World Examples:

1. **Check if file exists**

```
[ -f "data.txt" ] && echo "Exists" || echo "Missing"
```

2. **Directory check**

```
[ -d "/var/log" ] && echo "Log directory exists"
```

3. **Test service status**

```
systemctl is-active nginx && echo "Running" || echo "Stopped"
```

4. **Check string match**

```
input="yes"
[ "$input" == "yes" ] && echo "Proceed"
```

5. **Compare numbers**

```
[ 5 -gt 3 ] && echo "5 > 3"
```

3.2 Elif / Nested If

```
if [ $marks -ge 90 ]; then
  echo "A Grade"
elif [ $marks -ge 75 ]; then
  echo "B Grade"
else
  echo "C Grade"
fi
```

✅ Real-World Examples:

1. **Check CPU usage level**

```
cpu=$(top -bn1 | grep "Cpu(s)" | awk '{print $2}')
echo "CPU Usage: $cpu"
```

2. **Network status**

```
ping -c 1 google.com >/dev/null && echo "Online" || echo "Offline"
```

3. **Battery level check**

```
batt=$(cat /sys/class/power_supply/BAT0/capacity)
[ $batt -lt 20 ] && echo "Low Battery"
```

4. **Service response time check**

```
time=$(curl -o /dev/null -s -w "%{time_total}" https://example.com)
echo "Load time: $time"
```

5. **Login attempt retry logic**

```
[ $retry -gt 3 ] && echo "Too many tries"
```

🔷 SECTION 4: Loops

4.1 For Loop

```
for i in 1 2 3; do
  echo $i
done
```

✅ Real-World Examples:

1. **Loop through files**

```
for f in *.log; do
  echo "Processing $f"
done
```

2. **Create multiple users**

   ```
   for user in john jane jack; do
     useradd $user
   done
   ```

3. **Ping multiple hosts**

   ```
   for ip in 192.168.1.{1..5}; do
     ping -c 1 $ip
   done
   ```

4. **Loop through numbers**

   ```
   for i in {1..10}; do
     echo $i
   done
   ```

5. **Loop through services**

   ```
   for svc in nginx sshd; do
     systemctl is-active $svc
   done
   ```

4.2 While Loop

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

✅ Real-World Examples:

1. **Read log file line-by-line**

```
while IFS= read -r line; do
  echo $line
done < access.log
```

2. **Wait until service is up**

```
while ! nc -z localhost 3306; do
  sleep 1
done
```

3. **Keep checking disk space**

```
while true; do
  df -h /
  sleep 60
done
```

4. **Retry on failure**

```
while ! curl -s https://site.com; do
  sleep 5
done
```

5. **Countdown timer**

```
i=10; while [ $i -gt 0 ]; do echo $i; ((i--)); sleep 1; done
```

🔷 SECTION 5: Functions

5.1 Define & Use Functions

```
greet() {
  echo "Hello $1"
}
greet "Alice"
```

✅ Real-World Examples:

1. **Disk check function**

```
check_disk() {
  df -h | grep "$1"
}
check_disk "/"
```

2. **Send email**

```
send_alert() {
  mail -s "Alert" user@example.com <<< "$1"
}
```

3. **Backup a directory**

```
backup() {
  tar -czf "$1".tar.gz "$1"
}
backup /etc
```

4. **Convert to uppercase**

```
upper() {
  echo "$1" | tr a-z A-Z
}
```

5. **Get system uptime**

```
get_uptime() {
  uptime -p
}
```

🔷 SECTION 6: Arrays

6.1 Indexed Arrays

```
arr=(apple banana cherry)
echo ${arr[1]}
```

✅ Real-World Examples:

1. **Loop through services**

   ```
   svcs=(nginx mysql sshd)
   for s in "${svcs[@]}"; do systemctl status $s; done
   ```

2. **Store filenames**

   ```
   files=(log1.txt log2.txt)
   ```

3. **Store and access user roles**

   ```
   roles=(admin editor viewer)
   echo ${roles[0]}
   ```

4. **Command arguments in array**

   ```
   args=("$@")
   ```

5. **Dynamic array from ls**

```
files=($(ls *.sh))
```

🔷 SECTION 7: File Handling

7.1 Read and Write

✅ Real-World Examples:

1

. Read lines from file

```
while IFS= read -r line; do
  echo $line
done < file.txt
```

1. **Write logs**

   ```
   echo "$(date): Backup done" >> backup.log
   ```

2. **Clear file content**

   ```
   > error.log
   ```

3. **Append command output**

   ```
   ls -l >> files.log
   ```

4. **Count lines in file**

   ```
   wc -l file.txt
   ```

🔷 SECTION 8: Command Line Args

8.1 Positional Parameters

```
echo "File to process: $1"
```

✅ Real-World Examples:

1. **Take input filename**

   ```
   echo "Working on: $1"
   ```

2. **Add 2 numbers**

   ```
   echo $(( $1 + $2 ))
   ```

3. **Backup input directory**

   ```
   tar czf "$1".tar.gz "$1"
   ```

4. **Create a user**

   ```
   useradd $1
   ```

5. **Log passed argument**

   ```
   echo "Argument: $1" >> run.log
   ```

🔷 SECTION 9: Advanced Topics

9.1 Error Handling

```
cp file.txt /target/ || { echo "Copy failed"; exit 1; }
```

9.2 Traps

```
trap "echo 'Ctrl+C pressed! Exiting'; exit" SIGINT
```

9.3 Background Jobs

```
sleep 60 &
echo "Job PID: $!"
```

9.4 Case Statement

```
case "$1" in
  start) echo "Starting...";;
  stop) echo "Stopping...";;
  *) echo "Unknown";;
esac
```

9.5 Scheduling with Cron

```
crontab -e
# Add:
# 0 2 * * * /path/to/backup.sh
```

🧪 Want all this as a PDF, zip project, or GitHub repo?

Let me know, and I'll generate and share it with you!