1. Getting Started with Shell Script

Essential Knowledge:

- **SheBang (#!)**: Must be first line in every script

  ```
  #!/bin/bash
  ```

- **Script Creation & Execution**:

  ```
   nano script.sh          # Create script
  chmod +x script.sh      # Make executable
  ./script.sh             # Execute script
  ```

---

2. Variables in Shell Scripting

Key Concepts:

- Declaration: `VAR_NAME=value`
- Access: `$VAR_NAME`
- User input: `read -p "Prompt" var`
- Command line args: `$1`, `$2`, etc.

Examples:

```
 # Basic variable usage
name="Alice"
age=25
echo "$name is $age years old"

# User input
read -p "Enter your name: " username
echo "Hello, $username"

# Command line arguments
echo "First argument: $1"
echo "Second argument: $2"
```

---

3. Operators in Shell Scripting

Arithmetic Operators:

- `+`, `-`, `*`, `/`, `%`, `++`, `--`

Comparison Operators:

- `-eq` (equal), `-ne` (not equal)
- `-gt` (greater), `-lt` (less)
- `-ge` (greater or equal), `-le` (less or equal)

Logical Operators:

- `&&` (AND), `||` (OR), `!` (NOT)

Examples:

```
 # Arithmetic operations
a=10
b=5
sum=$((a + b))
echo "Sum: $sum"

# Comparison
if [ $a -gt $b ]; then
    echo "$a is greater than $b"
fi
```

---

4. Conditionals in Shell Scripting

Essential Structures:

```
 # If-else
if [ condition ]; then
    # code
elif [ condition ]; then
    # code
else
    # code
fi

# Case statement
case $variable in
    pattern1)
        # code;;
    pattern2)
        # code;;
    *)
        # default code;;
esac
```

Examples:

```
 # Even/Odd check
read -p "Enter number: " num
if [ $((num % 2)) -eq 0 ]; then
    echo "Even"
else
    echo "Odd"
fi

# Grade calculator
read -p "Enter score: " score
if [ $score -ge 90 ]; then
    grade="A"
elif [ $score -ge 80 ]; then
    grade="B"
else
    grade="C"
fi
echo "Grade: $grade"
```

---

5. Loops in Shell Scripting

Loop Types:

```
# For loop
for (( i=1; i<=5; i++ )); do
    echo "Number: $i"
done

# While loop
count=1
while [ $count -le 5 ]; do
    echo "Count: $count"
    count=$((count + 1))
done

# Until loop
n=5
until [ $n -eq 0 ]; do
    echo $n
    n=$((n - 1))
done
```

Examples:

```
# Print multiplication table
read -p "Enter number: " num
for (( i=1; i<=10; i++ )); do
    echo "$num x $i = $((num * i))"
done

# Sum of digits
read -p "Enter number: " num
sum=0
while [ $num -gt 0 ]; do
    digit=$((num % 10))
    sum=$((sum + digit))
    num=$((num / 10))
done
echo "Sum of digits: $sum"
```

---

6. Arrays in Shell Scripting

Array Operations:

```
# Declaration
arr=("apple" "banana" "cherry")

# Access elements
echo "First element: ${arr[0]}"
echo "All elements: ${arr[@]}"

# Array length
echo "Array length: ${#arr[@]}"

# Iterate over array
for item in "${arr[@]}"; do
    echo "Item: $item"
done
```

Examples:

```
# Find smallest and largest elements
arr=(24 27 84 11 99)
min=${arr[0]}
max=${arr[0]}
for num in "${arr[@]}"; do
    if [ $num -lt $min ]; then
        min=$num
    fi
    if [ $num -gt $max ]; then
        max=$num
    fi
done
echo "Smallest: $min"
echo "Largest: $max"

# Sort array in ascending order
arr=(24 27 84 11 99)
IFS=$'\n' sorted=($(sort -n <<<"${arr[*]}"))
unset IFS
echo "Sorted array: ${sorted[@]}"
```

7. Functions in Shell Scripting

Function Syntax:

```
function_name() {
    # code
    # Access parameters: $1, $2, etc.
}

# Call function
function_name arg1 arg2
```

Examples:

```
# Check if number is prime
is_prime() {
    num=$1
    if [ $num -le 1 ]; then
        echo "Not prime"
        return
    fi
    for (( i=2; i*i<=num; i++ )); do
        if [ $((num % i)) -eq 0 ]; then
            echo "Not prime"
            return
        fi
    done
    echo "Prime"
}

read -p "Enter number: " num
is_prime $num

# Convert Fahrenheit to Celsius
fahrenheit_to_celsius() {
    f=$1
    c=$(( (f - 32) * 5 / 9 ))
    echo "$f°F = $c°C"
}

read -p "Enter temperature in Fahrenheit: " f
fahrenheit_to_celsius $f
```

8. String Manipulation

Common Operations:

```
# String length
str="Hello World"
echo "Length: ${#str}"

# Substring
echo "First 5 chars: ${str:0:5}"

# Replace text
echo "${str/World/Everyone}"

# Reverse string
echo $str | rev

# Convert to lowercase
echo $str | tr '[:upper:]' '[:lower:]'
```

Examples:

```
# Palindrome check
read -p "Enter string: " str
reverse=$(echo $str | rev)
if [ "$str" == "$reverse" ]; then
    echo "Palindrome"
else
    echo "Not palindrome"
fi

# Capitalize first letter
str="hello"
capitalized=$(echo ${str:0:1} | tr '[:lower:]' '[:upper:]')${str:1}
echo "Capitalized: $capitalized"
```

9. File Operations

Essential Commands:

```
# Check file existence
if [ -f "file.txt" ]; then
    echo "File exists"
fi

# Check directory existence
if [ -d "mydir" ]; then
    echo "Directory exists"
fi

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

# Write to file
echo "Hello" > file.txt      # Overwrite
echo "World" >> file.txt     # Append

# Copy file
cp source.txt destination.txt

# File permissions
chmod 755 file.txt
```

Examples:

```
# Count lines in file
echo "Number of lines: $(wc -l < file.txt)"

# Find specific pattern in file
read -p "Enter pattern: " pattern
grep -n "$pattern" file.txt

# Create backup with timestamp
backup="backup_$(date +%Y%m%d_%H%M%S).txt"
cp file.txt $backup
echo "Backup created: $backup"
```

10. System Information & Process Management

Key Commands:

```
# System information
uname -a          # OS info
df -h             # Disk usage
free -h           # Memory usage
uptime            # System uptime

# Process management
ps aux            # List processes
top               # Monitor processes
kill PID          # Kill process
```

Examples:

```
# Check disk usage
df -h | grep -vE '^Filesystem|tmpfs|cdrom'

# Find top 5 memory-consuming processes
ps aux --sort=-%mem | head -6

# Check if specific process is running
read -p "Enter process name: " process
if pgrep -x "$process" > /dev/null; then
    echo "$process is running"
else
    echo "$process is not running"
fi
```

Practice Problems

1. **Find Smallest and Largest Elements in an Array**

```
#!/bin/bash
arr=(24 27 84 11 99)
min=${arr[0]}
max=${arr[0]}
for num in "${arr[@]}"; do
    if [ $num -lt $min ]; then
        min=$num
    fi
    if [ $num -gt $max ]; then
        max=$num
    fi
done
echo "Smallest: $min"
echo "Largest: $max"
```

2. **Sort an Array of Integers in Ascending Order**

```
#!/bin/bash
arr=(24 27 84 11 99)
IFS=$'\n' sorted=($(sort -n <<<"${arr[*]}"))
unset IFS
echo "Sorted array: ${sorted[@]}"
```

3. **Check if a Number is Prime**

```bash
#!/bin/bash
is_prime() {
    num=$1
    if [ $num -le 1 ]; then
        echo "Not prime"
        return
    fi
    for (( i=2; i*i<=num; i++ )); do
        if [ $((num % i)) -eq 0 ]; then
            echo "Not prime"
            return
        fi
    done
    echo "Prime"
}
read -p "Enter number: " num
is_prime $num
```

4. **Convert Fahrenheit to Celsius**

```bash
#!/bin/bash
fahrenheit_to_celsius() {
    f=$1
    c=$(( (f - 32) * 5 / 9 ))
    echo "$f°F = $c°C"
}
read -p "Enter temperature in Fahrenheit: " f
fahrenheit_to_celsius $f
```