

# Experiment 8: Shell Programming

---

## Experiment 8: Shell Programming

---

Name: Rishika Purushotham Roll No.: 590029145 Date: 2025-09-23

---

### Aim:

To extend shell programming concepts by using conditional statements, advanced scripting constructs, and command-line arguments.

To practice writing scripts that perform decision-making and parameter handling.

---

### Requirements

A Linux system with bash shell.

Text editor and permission to create/execute shell scripts.

---

### Theory

Conditional execution in shell scripts allows branching logic using `if`, `elif`, `else`, and `case` statements. Scripts can accept command-line arguments using `$1`, `$2`, ... and `$@` for all arguments. Control flow constructs combined with user input and arguments allow dynamic and reusable scripts.

---

### Procedure & Observations

#### Exercise 1: Using if-else

##### Task Statement:

Write a script to check whether a given number is positive, negative, or zero.

##### Explanation:

We used an if-elif-else construct to compare the number against 0.

##### Command(s):

```
#!/bin/bash
num=$1
if [ num -gt 0 ]; then echo "num is positive"
elif [ num -lt 0 ]; then echo "num is negative"
else
echo "$num is zero"
fi
```

Output:

```
$ cat > check_num.sh << 'EOF'
> #!/bin/bash
> num=$1
> if [ $num -gt 0 ]; then
>     echo "$num is positive"
> elif [ $num -lt 0 ]; then
>     echo "$num is negative"
> else
>     echo "$num is zero"
> fi
> EOF

$ chmod +x check_num.sh
$ ./check_num.sh 5
5 is positive
$ ./check_num.sh -3
-3 is negative
$ ./check_num.sh 0
0 is zero
```

Exercise 2: Using case

Task Statement:

Write a script that takes a character as input and classifies it as vowel, consonant, digit, or special character.

Explanation:

The case statement provides pattern matching for multiple options.

Command(s):

```
#!/bin/bash
ch=$1
case $ch in [aeiouAEIOU]) echo "ch is a vowel" ;
echo "$ch is a consonant" ; [0 - 9]) echo "ch is a digit" ;
*) echo "$ch is a special character" ;
esac
```

Output:

```
$ cat > char_type.sh << 'EOF'
> #!/bin/bash
> ch=$1
> case $ch in
>   [aeiouAEIOU]) echo "$ch is a vowel" ;;
>   [bcdfghjklmnpqrstvwxyzBCDFGHJKLMNPQRSTVWXYZ]) echo "$ch is a consonant" ;;
>   [0-9]) echo "$ch is a digit" ;;
>   *) echo "$ch is a special character" ;;
> esac
> EOF

$ chmod +x char_type.sh
$ ./char_type.sh a
a is a vowel
$ ./char_type.sh Z
Z is a consonant
$ ./char_type.sh 7
7 is a digit
$ ./char_type.sh @
@ is a special character
```

### Exercise 3: Command-line arguments

#### Task Statement:

Write a script that accepts filename(s) as arguments and prints the number of lines in each file.

#### Explanation:

Command-line arguments are accessed using `$@`. Looping through each argument allows file-wise operations.

#### Command(s):

```
#!/bin/bash
for file in "$@"; do if [ -f "$file" ]; then
echo "$file: $(wc -l < "$file") lines"
else
echo "$file not found"
fi
done
```

Output:

```
$ cat > line_count.sh << 'EOF'
> #!/bin/bash
> for file in "$@"; do
>   if [ -f "$file" ]; then
>     echo "$file: $(wc -l < "$file") lines"
>   else
>     echo "$file not found"
>   fi
> done
> EOF

$ chmod +x line_count.sh
$ echo -e "line1\nline2\nline3" > file1.txt
$ echo -e "a\nb" > file2.txt
$ ./line_count.sh file1.txt file2.txt missing.txt
file1.txt: 3 lines
file2.txt: 2 lines
missing.txt not found
```

#### Exercise 4: Nested conditionals

Task Statement:

Write a script to check if a year is a leap year.

Explanation:

A leap year is divisible by 4, but if divisible by 100 it must also be divisible by 400.

Command(s):

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

```
year=1if((year%4==0))then echo "year is a leap year"
```

```
elif ((year % 100 == 0)); then
```

```
echo "year is not a leap year" elif((year%4==0))then echo "year is a leap year"
```

```
else
```

```
echo "$year is not a leap year"
```

```
fi
```

Output:

```
$ cat > leap_year.sh << 'EOF'
> #!/bin/bash
> year=$1
> if (( year % 400 == 0 )); then
>   echo "$year is a leap year"
> elif (( year % 100 == 0 )); then
>   echo "$year is not a leap year"
> elif (( year % 4 == 0 )); then
>   echo "$year is a leap year"
> else
>   echo "$year is not a leap year"
> fi
> EOF

$ chmod +x leap_year.sh
$ ./leap_year.sh 2020
2020 is a leap year
$ ./leap_year.sh 1900
1900 is not a leap year
$ ./leap_year.sh 2000
2000 is a leap year
$ ./leap_year.sh 2023
2023 is not a leap year
```

---

## Result

Implemented conditional statements ( if-else , case ) in shell scripts.

Practiced handling command-line arguments and nested conditions.

Wrote reusable and flexible shell scripts.

## Challenges Faced & Learning Outcomes

Challenge 1: Forgetting to quote variables in conditions — resolved by using "\$var" to avoid word splitting.

Challenge 2: Pattern matching in case — practiced with multiple examples

## Learning:

Learned practical use of branching and decision-making in shell scripting.

Understood command-line argument handling for automation.

## Conclusion

This experiment extended shell programming by introducing decision-making and parameter handling.

The scripts demonstrate the flexibility of shell programming for different use cases.

---