Green University of Bangladesh
Department of Computer Science and Engineering(CSE)

Faculty of Sciences and Engineering

Semester: (Summer, Year:2025), B.Sc. in CSE (Day)

Lab Report 02

Course Title: Operating System Lab

Course Code: CSE 310 Section: 223 D2

Student Details

Name	ID
Anisur Rahaman Maruf	222902078

Submission Date: 02/08/2025

Course Teacher's Name: Umme Habiba

[For Teachers use only: Don't Write Anything inside this box]

Lab Report Status

Marks: …………………………………	Signature:.....................
Comments:...	Date:..............................

1. Introduction

Shell scripting is a fundamental skill for automating repetitive tasks and controlling system processes in Unix-based environments. In this lab exercise, we practiced different scripting techniques using for, while, until loops, arrays, and functions to solve real-world problems using Bash.

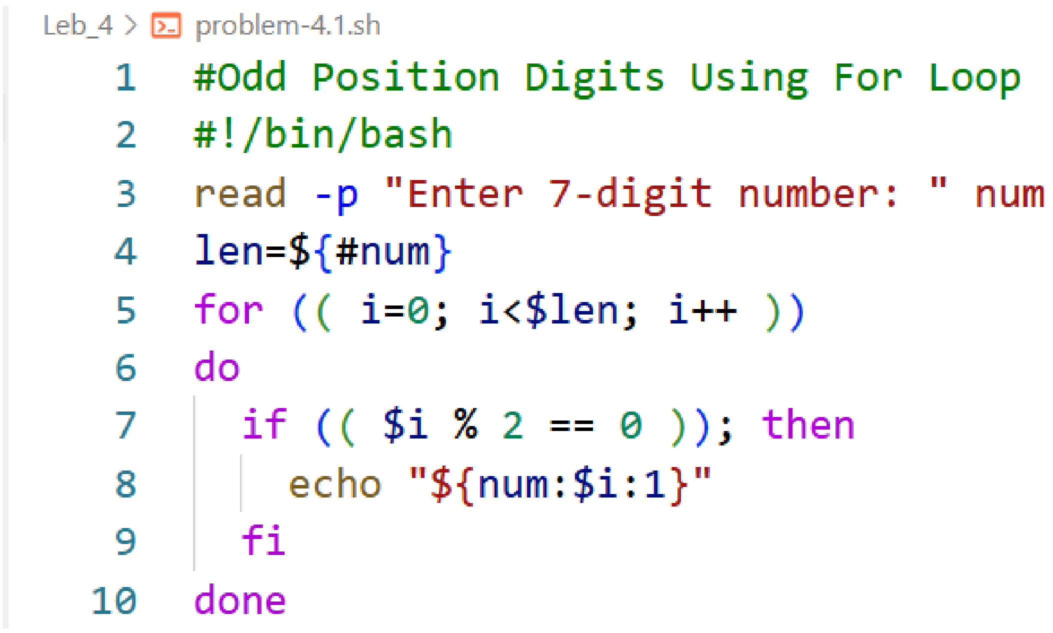
2. Objective

- To learn and implement shell scripts using different types of loops and functions.
- To solve specific problems through efficient scripting.
- To understand the usage of arrays and user-defined functions in Bash.

3. Implementation

3.1 Odd Position Digits Using For Loop

Problem: Display digits in odd positions from a 7-digit number.
Short Discussion:
Using for loop with index checking, we extracted characters at odd positions.

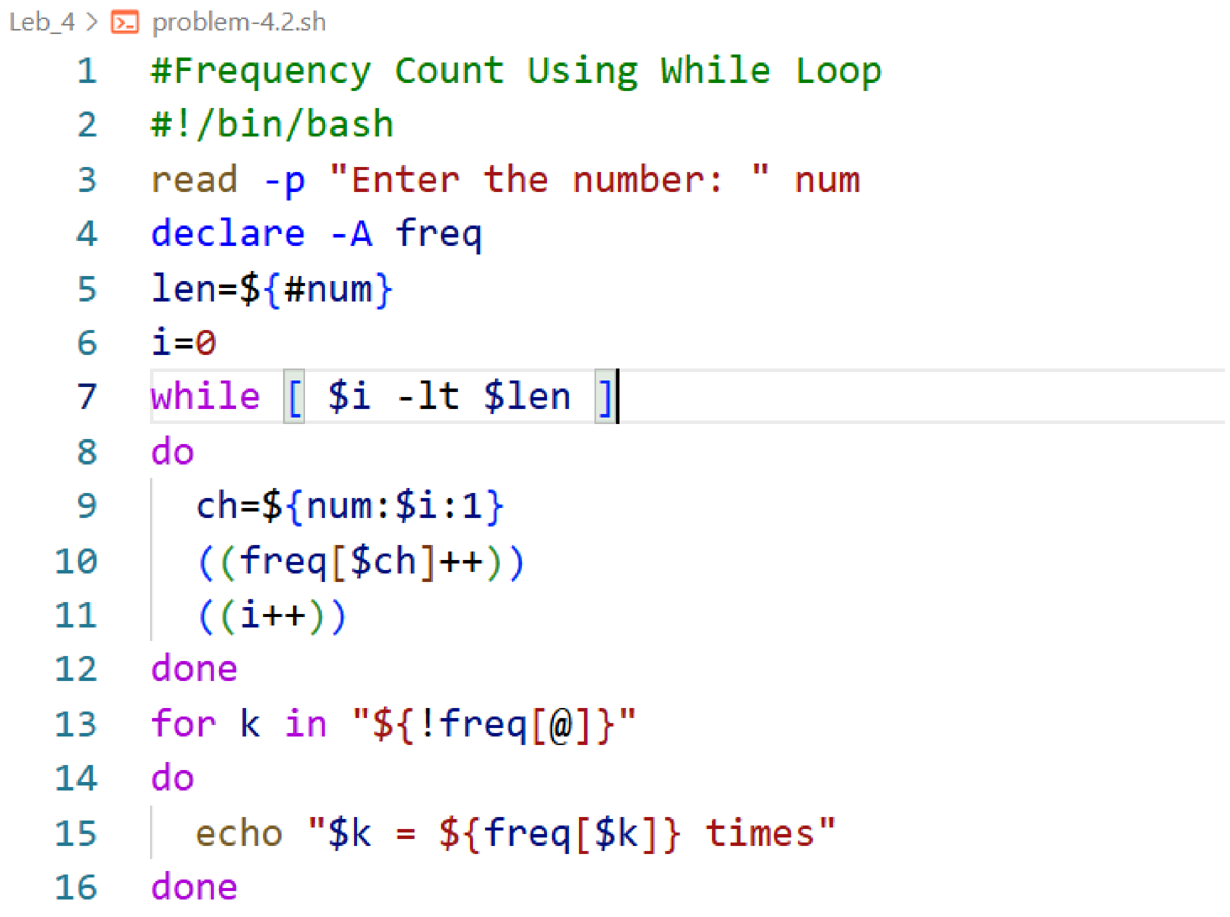


Leb_4 > problem-4.1.sh

```bash
#Odd Position Digits Using For Loop
#!/bin/bash
read -p "Enter 7-digit number: " num
len=${#num}
for (( i=0; i<$len; i++ ))
do
    if (( $i % 2 == 0 )); then
        echo "${num:$i:1}"
    fi
done
```

3.2 Frequency Count Using While Loop

Problem: Count frequency of each digit in a number.
Short Discussion:
A while loop reads character by character and uses an array to track frequency.

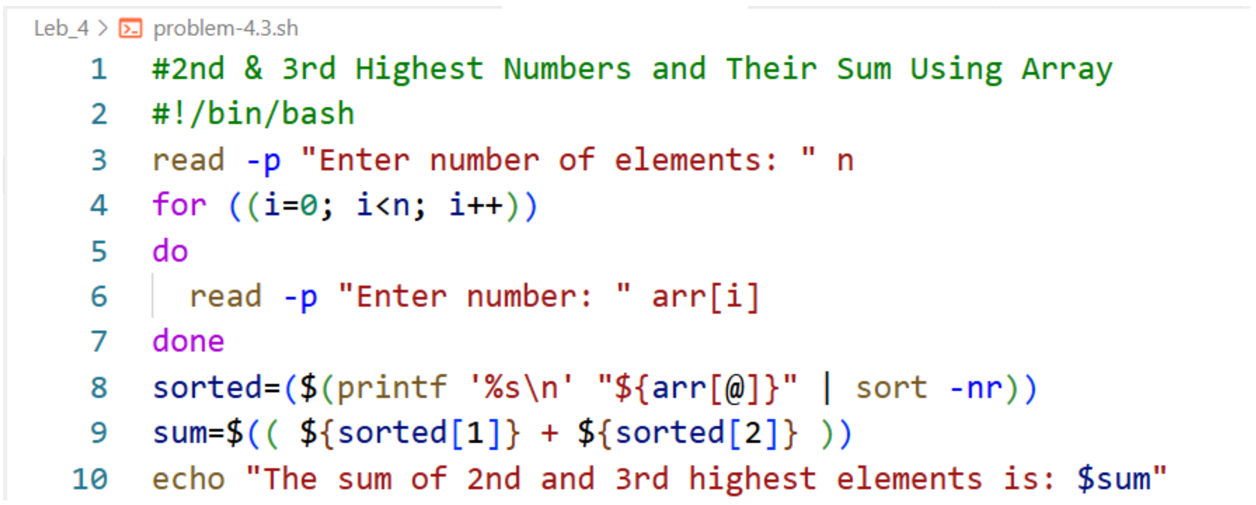


Leb_4 > problem-4.2.sh

```
#Frequency Count Using While Loop
#!/bin/bash
read -p "Enter the number: " num
declare -A freq
len=${#num}
i=0
while [ $i -lt $len ]
do
  ch=${num:$i:1}
  ((freq[$ch]++))
  ((i++))
done
for k in "${!freq[@]}"
do
  echo "$k = ${freq[$k]} times"
done
```

3.3 2nd & 3rd Highest Numbers and Their Sum Using Array

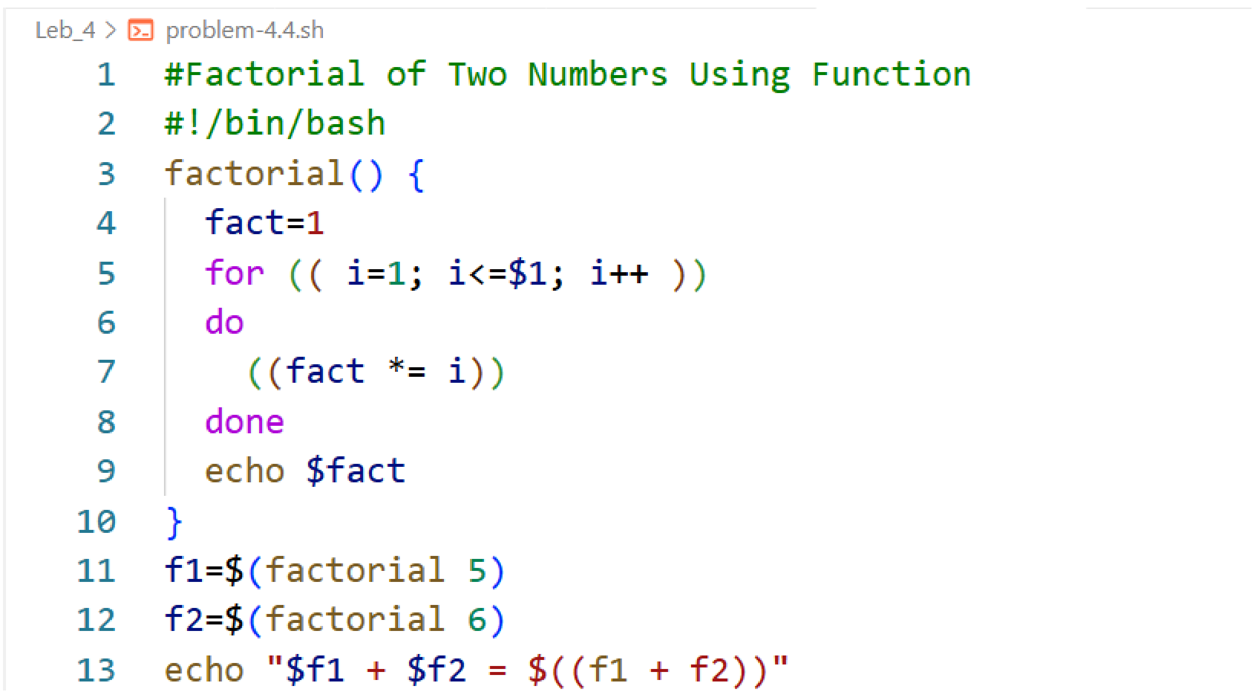
Problem: Find 2nd and 3rd highest numbers and sum them.
Short Discussion:
Used arrays, sorting, and indexing to extract and sum desired elements.


Leb_4 > problem-4.3.sh

```bash
#2nd & 3rd Highest Numbers and Their Sum Using Array
#!/bin/bash
read -p "Enter number of elements: " n
for ((i=0; i<n; i++))
do
  read -p "Enter number: " arr[i]
done
sorted=($(printf '%s\n' "${arr[@]}" | sort -nr))
sum=$(( ${sorted[1]} + ${sorted[2]} ))
echo "The sum of 2nd and 3rd highest elements is: $sum"
```

3.4 Factorial of Two Numbers Using Function

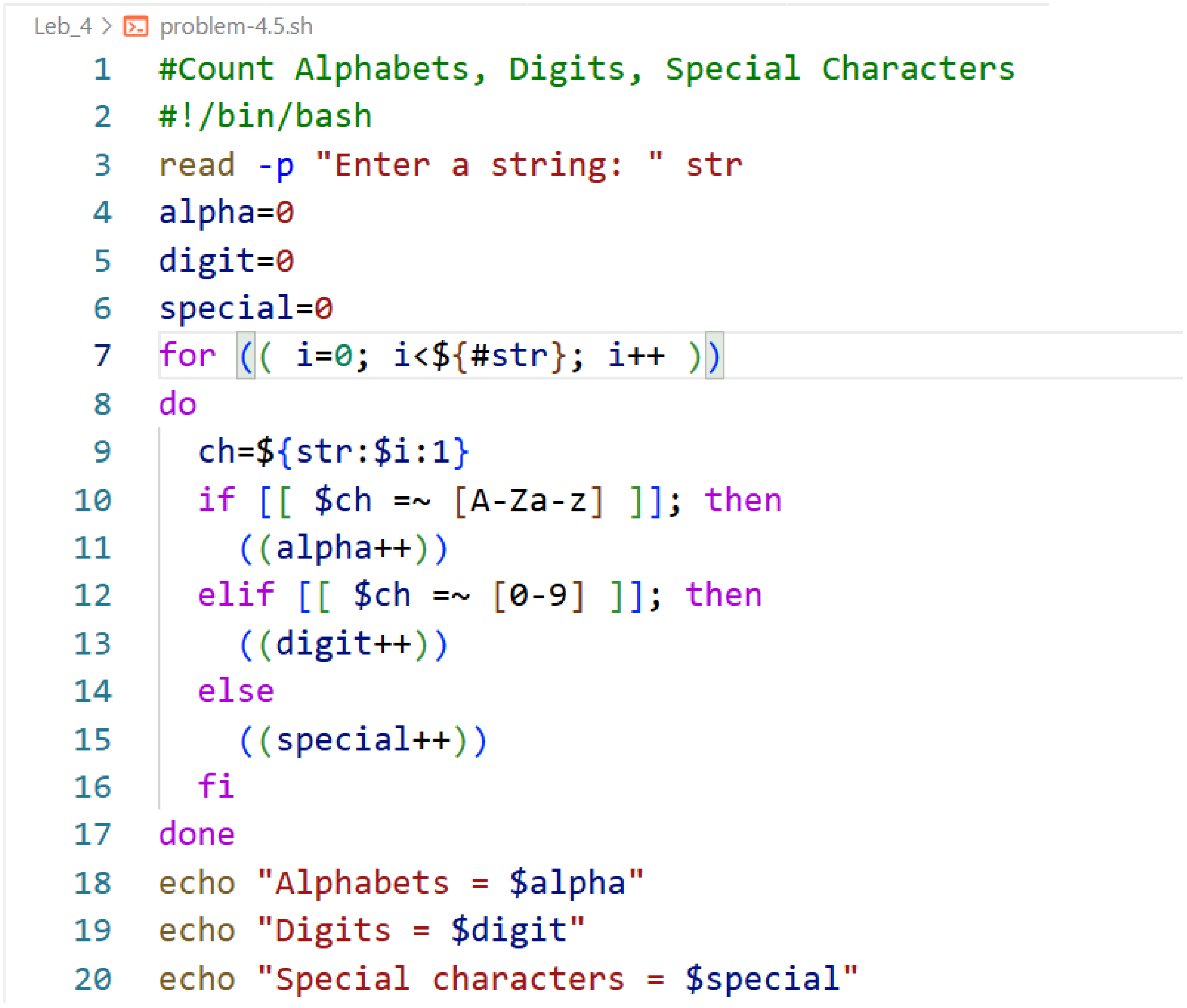
Problem: Calculate factorials of two numbers and sum them.
Short Discussion:
Created a reusable function factorial and invoked it twice.


Leb_4 > problem-4.4.sh

```bash
#Factorial of Two Numbers Using Function
#!/bin/bash
factorial() {
  fact=1
  for (( i=1; i<=$1; i++ ))
  do
    ((fact *= i))
  done
  echo $fact
}
f1=$(factorial 5)
f2=$(factorial 6)
echo "$f1 + $f2 = $((f1 + f2))"
```

3.5 Count Alphabets, Digits, Special Characters

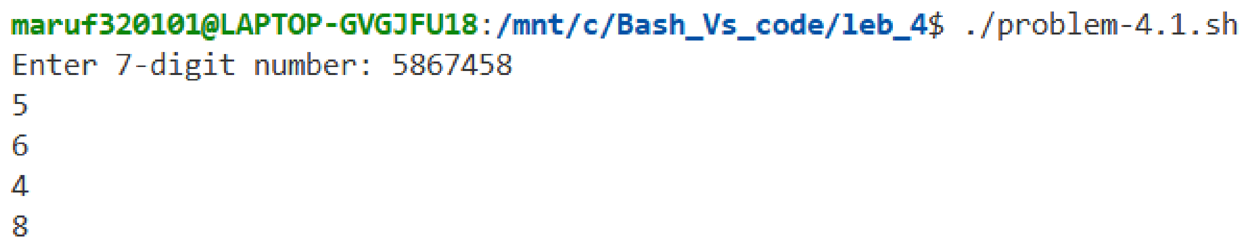
Problem: Analyze string content.
Short Discussion:
Looped through characters and used regex pattern matching.


Leb_4 > problem-4.5.sh

```bash
#Count Alphabets, Digits, Special Characters
#!/bin/bash
read -p "Enter a string: " str
alpha=0
digit=0
special=0
for (( i=0; i<${#str}; i++ ))
do
  ch=${str:$i:1}
  if [[ $ch =~ [A-Za-z] ]]; then
    ((alpha++))
  elif [[ $ch =~ [0-9] ]]; then
    ((digit++))
  else
    ((special++))
  fi
done
echo "Alphabets = $alpha"
echo "Digits = $digit"
echo "Special characters = $special"
```

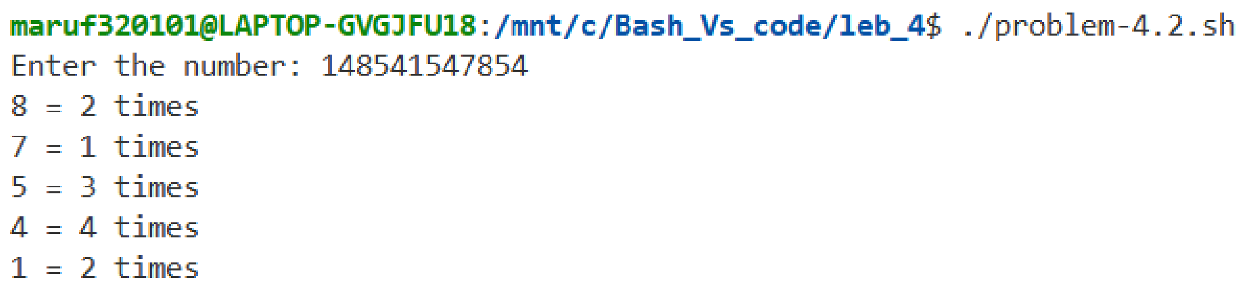
4. Output

Odd Position Digits Output:



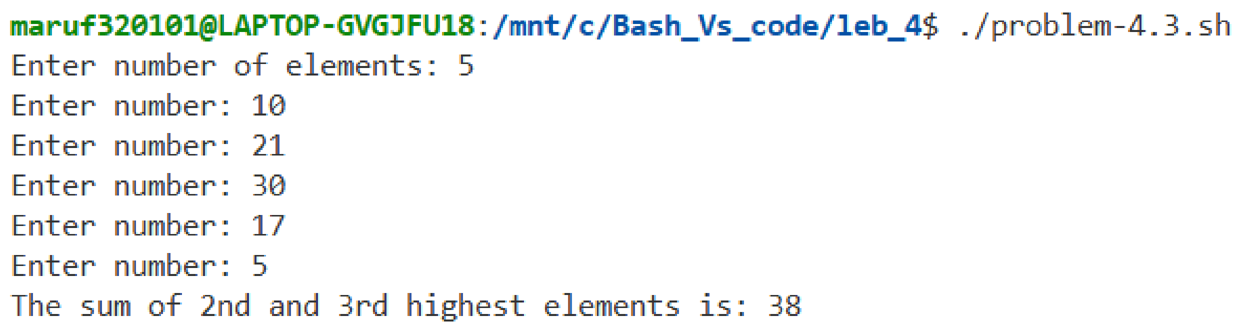
```
maruf320101@LAPTOP-GVGJFU18:/mnt/c/Bash_Vs_code/leb_4$ ./problem-4.1.sh
Enter 7-digit number: 5867458
5
6
4
8
```

Digit Frequency Output:



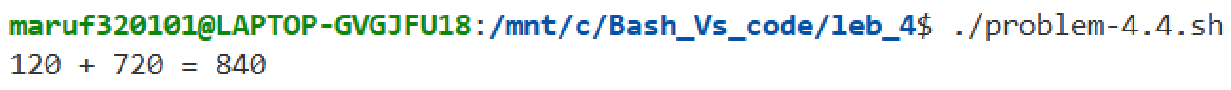
```
maruf320101@LAPTOP-GVGJFU18:/mnt/c/Bash_Vs_code/leb_4$ ./problem-4.2.sh
Enter the number: 148541547854
8 = 2 times
7 = 1 times
5 = 3 times
4 = 4 times
1 = 2 times
```

Array-Based Output:



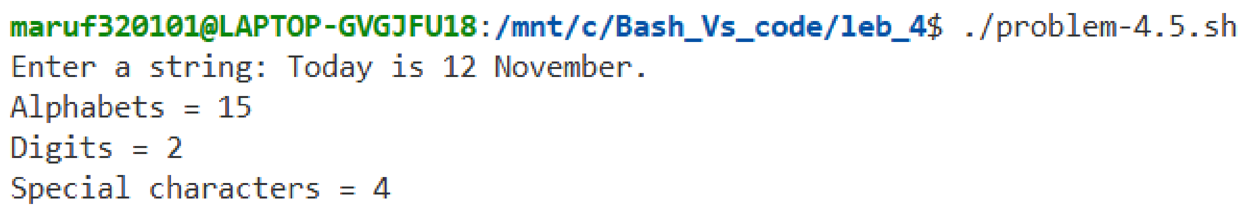
```
maruf320101@LAPTOP-GVGJFU18:/mnt/c/Bash_Vs_code/leb_4$ ./problem-4.3.sh
Enter number of elements: 5
Enter number: 10
Enter number: 21
Enter number: 30
Enter number: 17
Enter number: 5
The sum of 2nd and 3rd highest elements is: 38
```

Factorial Function Output:



```
maruf320101@LAPTOP-GVGJFU18:/mnt/c/Bash_Vs_code/leb_4$ ./problem-4.4.sh
120 + 720 = 840
```

Character Count Output:



```
maruf320101@LAPTOP-GVGJFU18:/mnt/c/Bash_Vs_code/leb_4$ ./problem-4.5.sh
Enter a string: Today is 12 November.
Alphabets = 15
Digits = 2
Special characters = 4
```

5. Discussion

Through these exercises, I gained hands-on experience in Bash scripting. Each task highlighted different aspects of control flow and data handling. Writing scripts for number analysis, factorial calculation, and array operations gave me confidence in shell programming logic. The integration of loops, conditions, and user-defined functions is particularly useful for automation.