Name: Veeransh Shah
Reg Id: 221070063

Assignment - IV

Aim:
To develop a comprehensive shell script that demonstrates the execution of various programming constructs, including loops, arithmetic operations, conditional statements, and the creation of a resume in a text file format.

Theory:

1. Execution of different loops.
    - The for loop iterates over a fixed range {1..5}, executing a block of code five times, printing each iteration number.
    - The `while` loop executes as long as the condition `count -le 5` is true, incrementing a counter until the condition is no longer met.
    - Both loops terminate based on their respective conditions—either the completion of a range or a dynamic check—ensuring controlled and predictable execution

```
function loops_example() {

   echo "For loop example: "

   for i in {1..5}

   do

       echo "Iteration: $i"

   done

   echo -e "\nWhile loop example: "

   count=1

   while [ $count -le 5 ]

   do

       echo "Iteration: $count"
```

```
        count=$((count + 1))

    done

}
```

2. Execution of arithmetic operations by taking values from the user.
    - The script prompts the user to enter two numbers, storing the input values in variables num1 and num2.
    - It performs basic arithmetic operations (addition, subtraction, multiplication, and division) on the input values using shell arithmetic.
    - The script outputs the results of each arithmetic operation, providing the user with the sum, difference, product, and quotient of the entered numbers.

```
function arithmetic_operations() {

    echo -e "\nEnter two numbers:"

    read -p "First number: " num1

    read -p "Second number: " num2



    sum=$((num1 + num2))

    difference=$((num1 - num2))

    product=$((num1 * num2))

    quotient=$((num1 / num2))



    echo -e "\nResults:"

    echo "Sum: $sum"

    echo "Difference: $difference"

    echo "Product: $product"

    echo "Quotient: $quotient"

}
```

3. Execution of conditional statements.
    - The script prompts the user to enter a number, storing the input in the variable number.
    - It uses an if-elif-else structure to evaluate whether the entered number is positive, negative, or zero.
    - Based on the condition met, the script outputs a message indicating whether the number is positive, negative, or zero.

```
function conditional_statements() {

   read -p "Enter a number: " number



   if [ $number -gt 0 ]; then

       echo "$number is positive."

   elif [ $number -lt 0 ]; then

       echo "$number is negative."

   else

       echo "$number is zero."

   fi

}
```

4. Write a shell script to create a text file (.txt) which will store your personal data including academic details in resume format.
    - The script prompts the user to input personal and academic details, such as name, email, qualifications, and skills.
    - It formats the entered information into a structured resume layout and saves it in a .txt file.
    - The script generates a text file named after the user, containing the resume details in a well-organized format.

```
function create_resume() {

   read -p "Enter your full name: " name

   read -p "Enter your email: " email

   read -p "Enter your phone number: " phone
```

```
    read -p "Enter your academic qualification: " qualification
    read -p "Enter your skills (comma-separated): " skills

    filename="${name// /_}_Resume.txt"
    echo -e "\nCreating resume file: $filename"

    cat <<EOF > $filename
Personal Information
--------------------
Name: $name
Email: $email
Phone: $phone

Academic Details
----------------
Qualification: $qualification

Skills
------
$skills
EOF

    echo -e "\nResume created successfully!"
```

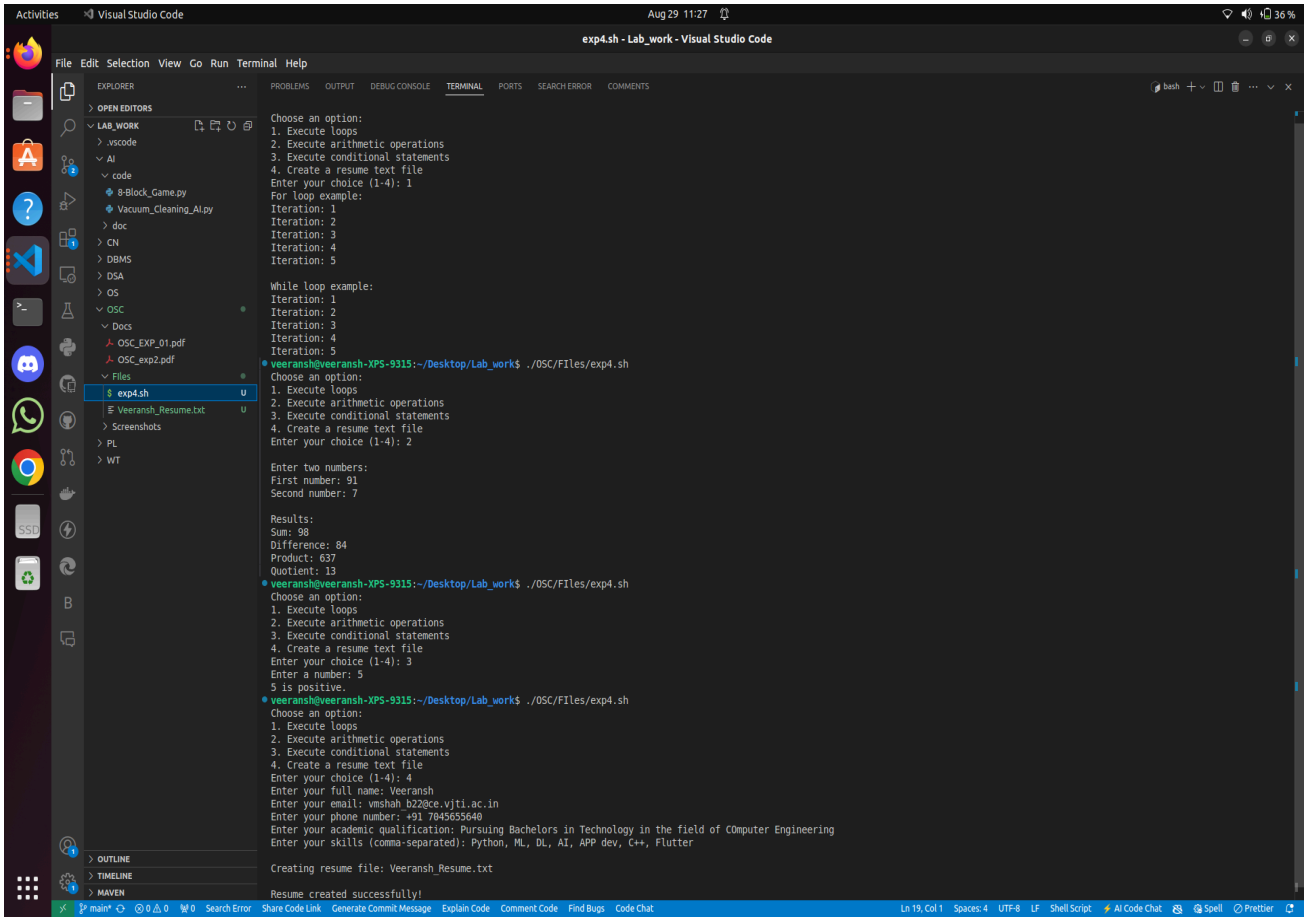
```
}

echo "Choose an option:"

echo "1. Execute loops"

echo "2. Execute arithmetic operations"

echo "3. Execute conditional statements"

echo "4. Create a resume text file"

read -p "Enter your choice (1-4): " choice



case $choice in

   1) loops_example ;;

   2) arithmetic_operations ;;

   3) conditional_statements ;;

   4) create_resume ;;

   *) echo "Invalid choice!" ;;

esac
```

Shell scripting is a powerful tool in UNIX-like operating systems that allows users to automate tasks, manage files, and perform a wide range of operations through a series of commands and logic. Shell scripts are essential for system administrators and developers alike, as they enable the efficient management of repetitive tasks and the execution of complex sequences of commands.

This script is designed to showcase the fundamental concepts of shell scripting, including loops, arithmetic operations, and conditional statements. Loops are critical in programming as they allow repetitive execution of a block of code, enabling the automation of tasks that would otherwise require manual repetition. Arithmetic operations, essential in any programming language, are used to perform calculations and data manipulation. Conditional statements allow the script to make decisions based on user input or other conditions, adding flexibility and control to the script's execution flow.

Additionally, the script includes a practical feature to create a resume in a text file format. This demonstrates the ability to interact with users, take input, and format data within a script. The resume creation portion of the script highlights how shell scripts can be used to gather, format, and store information in a structured manner, demonstrating their versatility in handling various tasks beyond basic system operations.

Output:



Conclusion:

The shell script developed in this exercise effectively demonstrates the execution of basic programming constructs such as loops, arithmetic operations, and conditional statements within the shell scripting environment. Furthermore, it illustrates the practical application of these concepts through the creation of a resume text file, showcasing the script's ability to interact with users, process input, and generate structured output. This exercise underscores the power and flexibility of shell scripting in automating tasks and managing data, making it an invaluable tool for both developers and system administrators.