Bansilal Ramnath Agarwal Charitable Trust's

Vishwakarma Institute of Information Technology

Department of

Artificial Intelligence and Data Science

Student Name: Siddhesh Dilip Khairnar

Class: T.Y.

Division: B

Roll No: 372028

Semester: V

Academic Year: 2023 - 24

Subject Name & Code: Cloud Computing and DevOps: ADUA31203

Title of Assignment: Write The shell scripting for demonstrate the following logic.
1) Decision Logic
2) Looping Logic
3) Decision and Looping Logic.

Date of Performance: 30/08/2023

Date of Submission: 6/09/2023

Assignment: 2 (B)

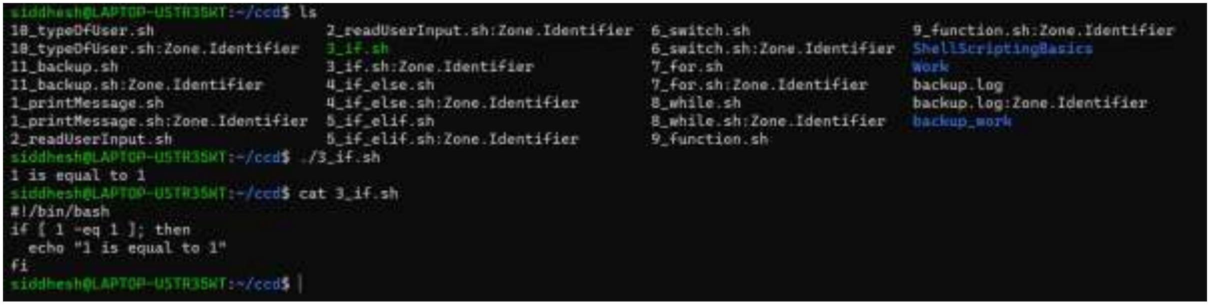
Aim: write shell scripting for looping and decision logic.

Problem Statement: Write The shell scripting for demonstrate the following logic.

1. Decision Logic
2. Looping Logic
3. Decision and Looping Logic.

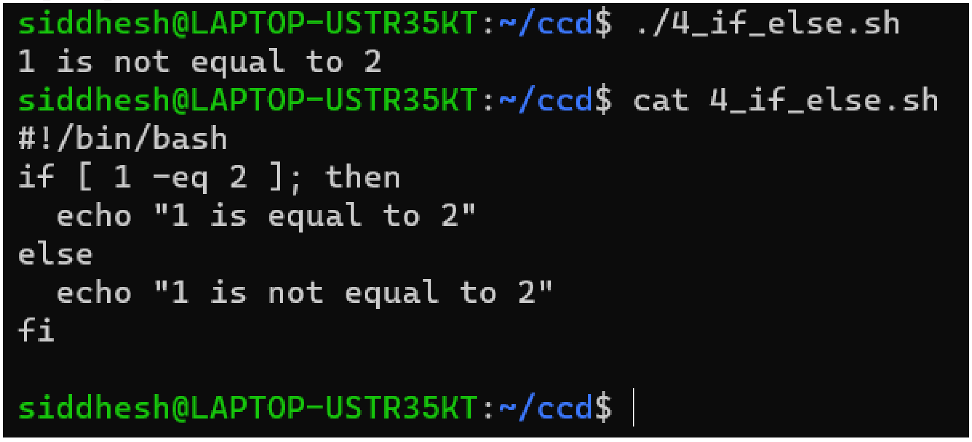
Theory:

1. **Decision Logic:** Decision logic in shell scripting involves making choices or decisions based on certain conditions. The primary construct for decision-making in shell scripts is the if statement, which allows you to execute different blocks of code depending on whether a specified condition is true or false. Let's break down the key components of decision logic in the context of shell scripting:
   - ❖ **The if statement:** The if keyword marks the beginning of the decision block. It is followed by a condition that is enclosed in square brackets [].

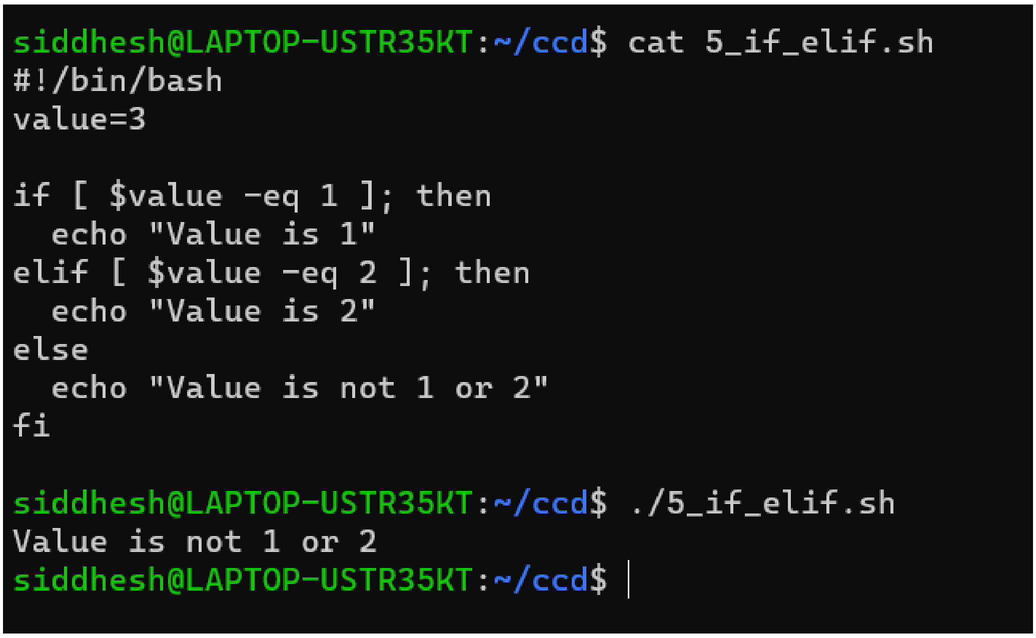


```
siddhesh@LAPTOP-USTR35KT:~/ccd$ ls
10_typeOfUser.sh                    2_readUserInput.sh:Zone.Identifier  6_switch.sh                    9_function.sh:Zone.Identifier
10_typeOfUser.sh:Zone.Identifier    3_if.sh                             6_switch.sh:Zone.Identifier    ShellScriptingBasics
11_backup.sh                        3_if.sh:Zone.Identifier             7_for.sh                       Work
11_backup.sh:Zone.Identifier        4_if_else.sh                        7_for.sh:Zone.Identifier       backup.log
1_printMessage.sh                   4_if_else.sh:Zone.Identifier        8_while.sh                     backup.log:Zone.Identifier
1_printMessage.sh:Zone.Identifier   5_if_elif.sh                        8_while.sh:Zone.Identifier     backup_work
2_readUserInput.sh                  5_if_elif.sh:Zone.Identifier        9_function.sh
siddhesh@LAPTOP-USTR35KT:~/ccd$ ./3_if.sh
1 is equal to 1
siddhesh@LAPTOP-USTR35KT:~/ccd$ cat 3_if.sh
#!/bin/bash
if [ 1 -eq 1 ]; then
  echo "1 is equal to 1"
fi
siddhesh@LAPTOP-USTR35KT:~/ccd$ |
```

   - ❖ **Condition:** The condition is an expression that evaluates to either true or false. Common operators used in conditions include:
     - -eq: Equal to



```
siddhesh@LAPTOP-USTR35KT:~/ccd$ ./4_if_else.sh
1 is not equal to 2
siddhesh@LAPTOP-USTR35KT:~/ccd$ cat 4_if_else.sh
#!/bin/bash
if [ 1 -eq 2 ]; then
  echo "1 is equal to 2"
else
  echo "1 is not equal to 2"
fi

siddhesh@LAPTOP-USTR35KT:~/ccd$ |
```



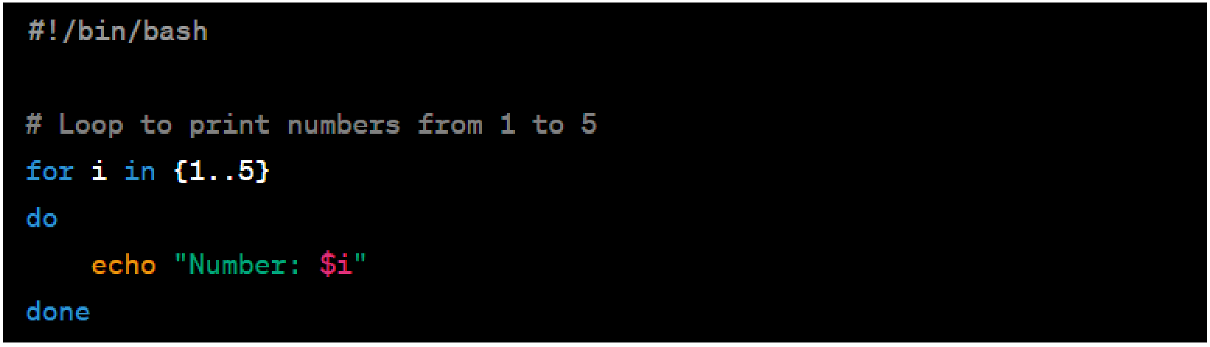
```
siddhesh@LAPTOP-USTR35KT:~/ccd$ cat 5_if_elif.sh
#!/bin/bash
value=3

if [ $value -eq 1 ]; then
  echo "Value is 1"
elif [ $value -eq 2 ]; then
  echo "Value is 2"
else
  echo "Value is not 1 or 2"
fi

siddhesh@LAPTOP-USTR35KT:~/ccd$ ./5_if_elif.sh
Value is not 1 or 2
siddhesh@LAPTOP-USTR35KT:~/ccd$ 
```

- -ne: Not equal to
- -lt: Less than
- -le: Less than or equal to
- -gt: Greater than
- -ge: Greater than or equal to

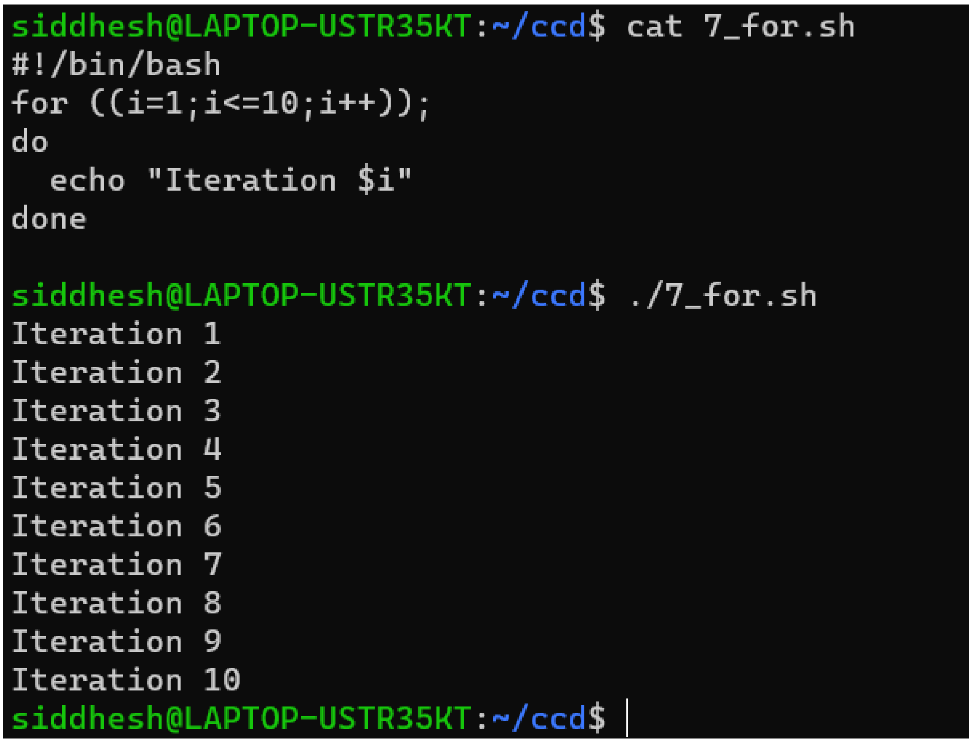
- **then keyword:** The then keyword follows the condition and marks the beginning of the code block that will be executed if the condition is true.
- **Code to execute if the condition is true:** This is the code that gets executed if the condition specified in the if statement evaluates to true. It should be indented for readability, although shell scripting is not as strict about indentation as some other programming languages.
- **The else keyword (optional):** The else keyword is used to define an alternative code block that is executed if the condition in the if statement is false. It's not required; you can have an if statement without an else block.
- **Code to execute if the condition is false:** This is the code that gets executed if the condition specified in the if statement evaluates too false. Like the code block for the true condition, it should also be indented.
- **The fi keyword:** The fi keyword marks the end of the if statement. It is used to close the decision block.

2. **Looping logic:** looping logic in shell scripting allows you to repeat a set of commands or actions multiple times. There are several types of loops in shell scripting, but the most used ones are the for loop and the while loop. Let's explore these loops in more detail:

- for Loop: The for loop is used to iterate over a range of values, a list of items, or a sequence. It repeats a block of code for each value in the specified range or list.

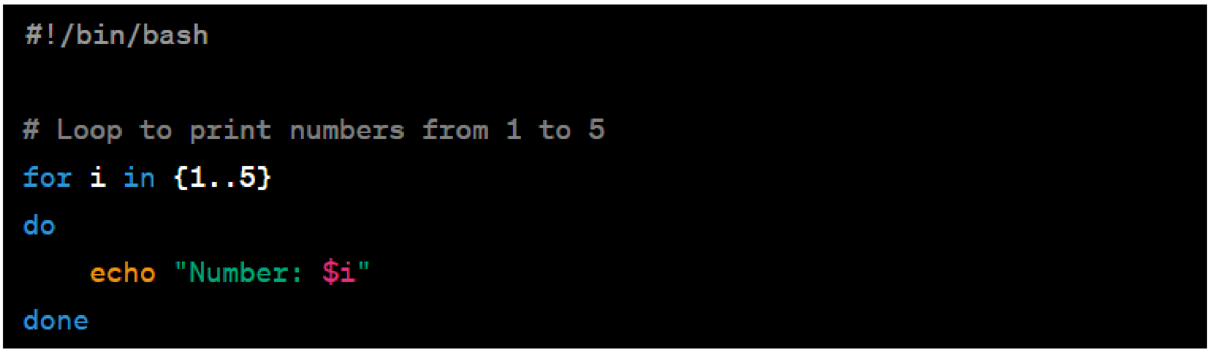


```
#!/bin/bash

# Loop to print numbers from 1 to 5
for i in {1..5}
do
    echo "Number: $i"
done
```

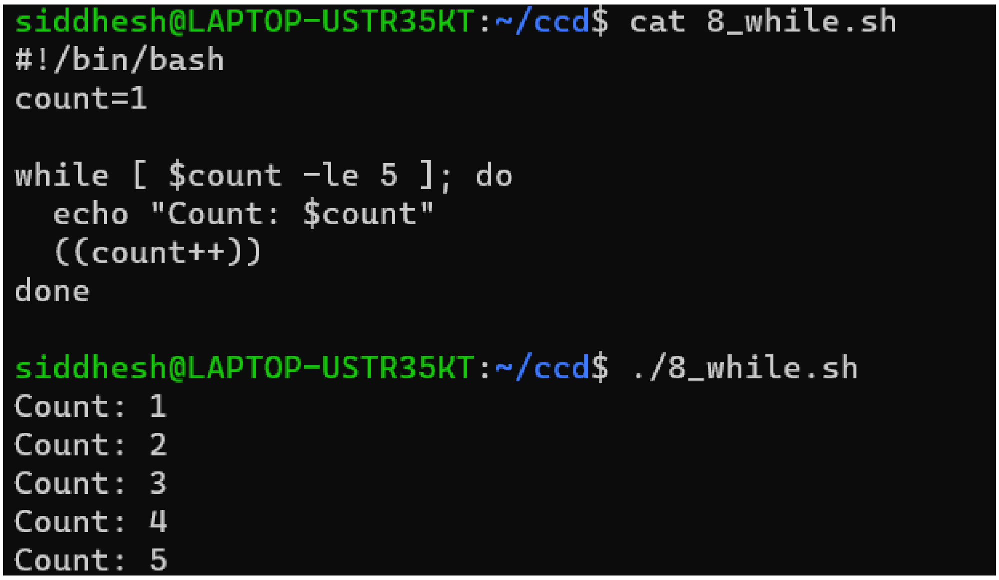


```
siddhesh@LAPTOP-USTR35KT:~/ccd$ cat 7_for.sh
#!/bin/bash
for ((i=1;i<=10;i++));
do
  echo "Iteration $i"
done

siddhesh@LAPTOP-USTR35KT:~/ccd$ ./7_for.sh
Iteration 1
Iteration 2
Iteration 3
Iteration 4
Iteration 5
Iteration 6
Iteration 7
Iteration 8
Iteration 9
Iteration 10
siddhesh@LAPTOP-USTR35KT:~/ccd$ 
```

- while Loop: The while loop is used to repeat a block of code as long as a specified condition is true. It continues executing as long as the condition remains true.

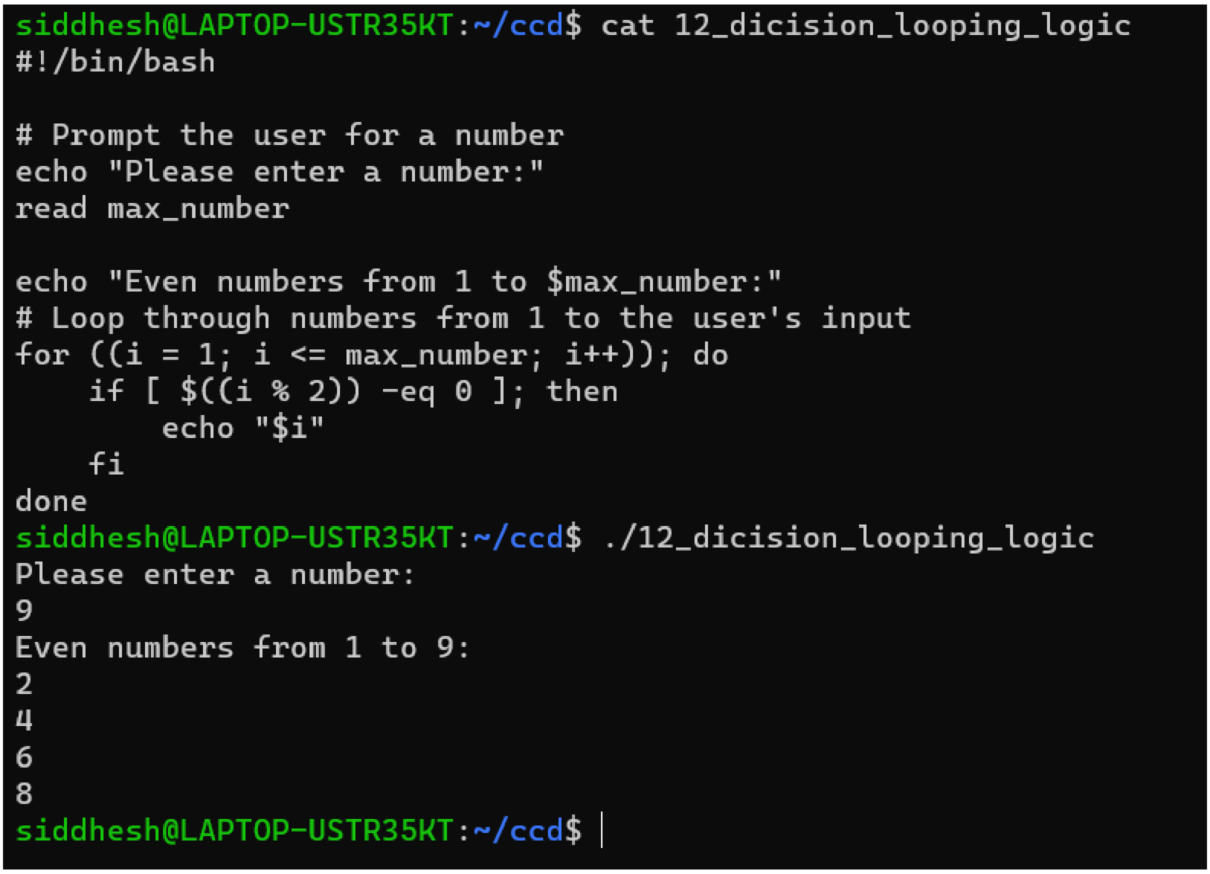


```
#!/bin/bash

# Loop to print numbers from 1 to 5
for i in {1..5}
do
    echo "Number: $i"
done
```



```
siddhesh@LAPTOP-USTR35KT:~/ccd$ cat 8_while.sh
#!/bin/bash
count=1

while [ $count -le 5 ]; do
  echo "Count: $count"
  ((count++))
done

siddhesh@LAPTOP-USTR35KT:~/ccd$ ./8_while.sh
Count: 1
Count: 2
Count: 3
Count: 4
Count: 5
```

3. **Decision and Looping Logic:** It’s referring to combining both decision-making (using if statements) and looping (using for or while loops) within a shell script. This combination allows you to create more complex scripts that perform iterative tasks based on certain conditions.



```
siddhesh@LAPTOP-USTR35KT:~/ccd$ cat 12_dicision_looping_logic
#!/bin/bash

# Prompt the user for a number
echo "Please enter a number:"
read max_number

echo "Even numbers from 1 to $max_number:"
# Loop through numbers from 1 to the user's input
for ((i = 1; i <= max_number; i++)); do
    if [ $((i % 2)) -eq 0 ]; then
        echo "$i"
    fi
done
siddhesh@LAPTOP-USTR35KT:~/ccd$ ./12_dicision_looping_logic
Please enter a number:
9
Even numbers from 1 to 9:
2
4
6
8
siddhesh@LAPTOP-USTR35KT:~/ccd$
```

Conclusion: By understanding and effectively using decision and looping logic, we can create shell scripts that automate tasks, perform data processing, and solve a wide range of problems in a systematic and efficient manner. Shell scripting is a valuable skill for system administrators, developers, and anyone working in a Unix/Linux environment.