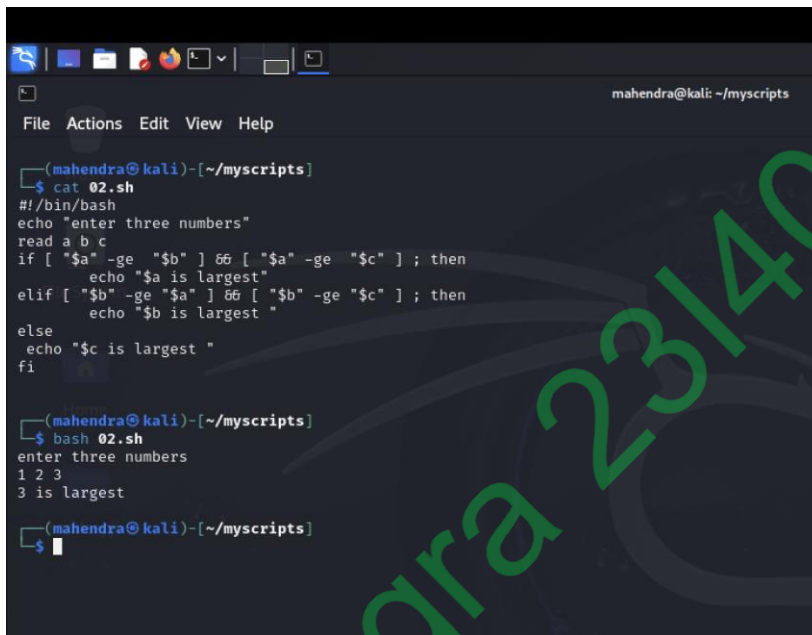


4ITRC2 Operating System Lab

Lab Assignment 3

Create shell scripts for the following questions

1. To find Largest of Three Numbers

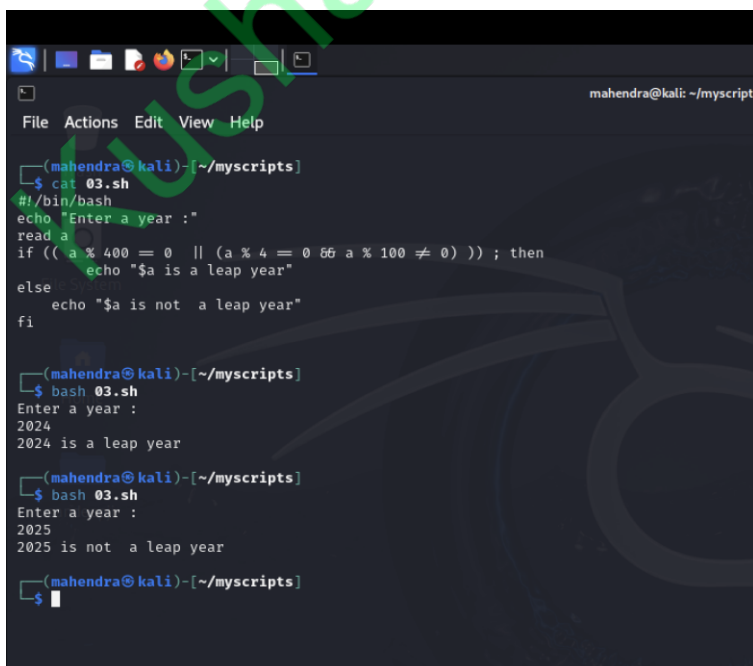


```
(mahendra@kali)-[~/myscripts]
$ cat 02.sh
#!/bin/bash
echo "enter three numbers"
read a b c
if [ "$a" -ge "$b" ] && [ "$a" -ge "$c" ]; then
    echo "$a is largest"
elif [ "$b" -ge "$a" ] && [ "$b" -ge "$c" ]; then
    echo "$b is largest"
else
    echo "$c is largest"
fi

(mahendra@kali)-[~/myscripts]
$ bash 02.sh
enter three numbers
1 2 3
3 is largest

(mahendra@kali)-[~/myscripts]
$
```

2. To find a year is leap year or not



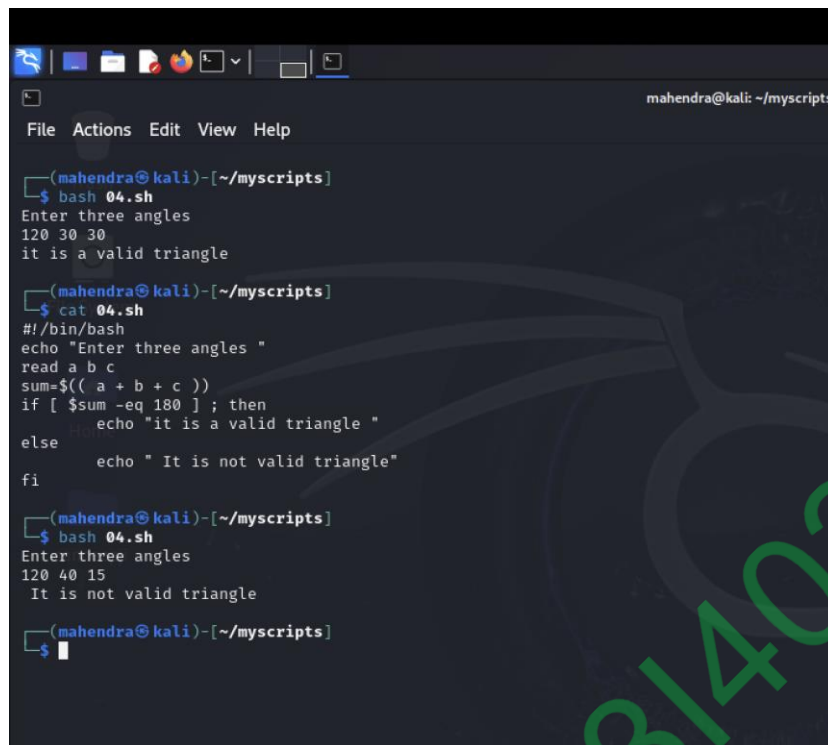
```
(mahendra@kali)-[~/myscripts]
$ cat 03.sh
#!/bin/bash
echo "Enter a year :"
read a
if (( a % 400 == 0 || (a % 4 == 0 && a % 100 != 0) )); then
    echo "$a is a leap year"
else
    echo "$a is not a leap year"
fi

(mahendra@kali)-[~/myscripts]
$ bash 03.sh
Enter a year :
2024
2024 is a leap year

(mahendra@kali)-[~/myscripts]
$ bash 03.sh
Enter a year :
2025
2025 is not a leap year

(mahendra@kali)-[~/myscripts]
$
```

3. To input angles of a triangle and find out whether it is valid triangle or not



```
mahendra@kali: ~/myscripts
File Actions Edit View Help

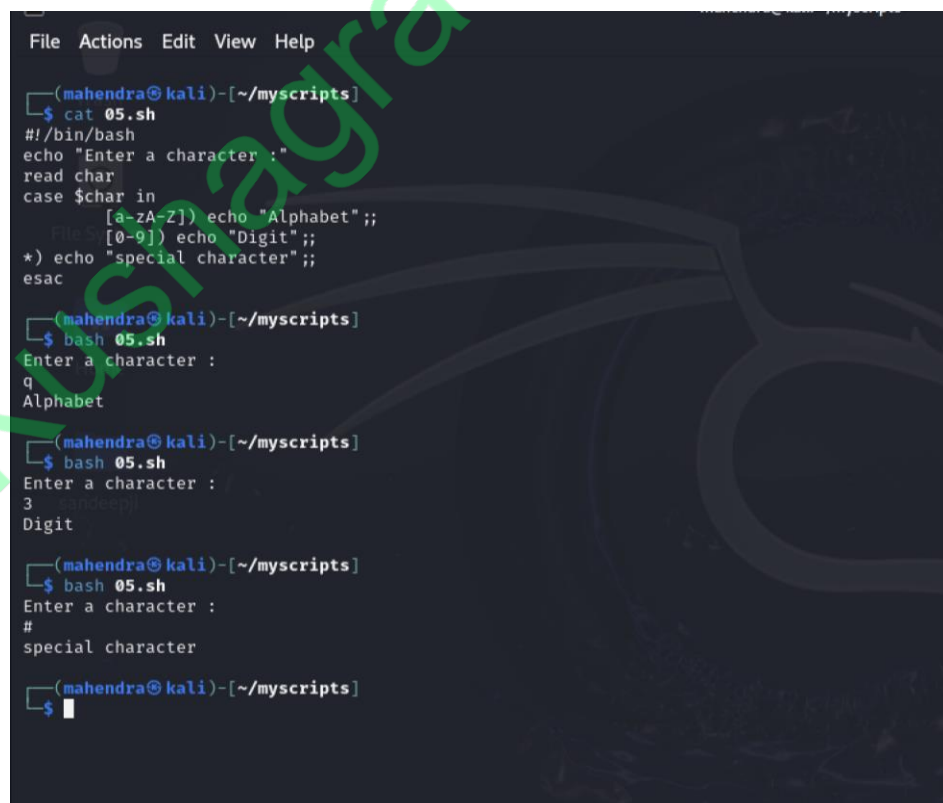
(mahendra@kali)-[~/myscripts]
$ bash 04.sh
Enter three angles
120 30 30
it is a valid triangle

(mahendra@kali)-[~/myscripts]
$ cat 04.sh
#!/bin/bash
echo "Enter three angles "
read a b c
sum=$(( a + b + c ))
if [ $sum -eq 180 ] ; then
    echo "it is a valid triangle "
else
    echo " It is not valid triangle"
fi

(mahendra@kali)-[~/myscripts]
$ bash 04.sh
Enter three angles
120 40 15
It is not valid triangle

(mahendra@kali)-[~/myscripts]
$
```

4. To check whether a character is alphabet, digit or special character.



```
File Actions Edit View Help

(mahendra@kali)-[~/myscripts]
$ cat 05.sh
#!/bin/bash
echo "Enter a character :"
read char
case $char in
    [a-zA-Z]) echo "Alphabet";;
    [0-9]) echo "Digit";;
    *) echo "special character";;
esac

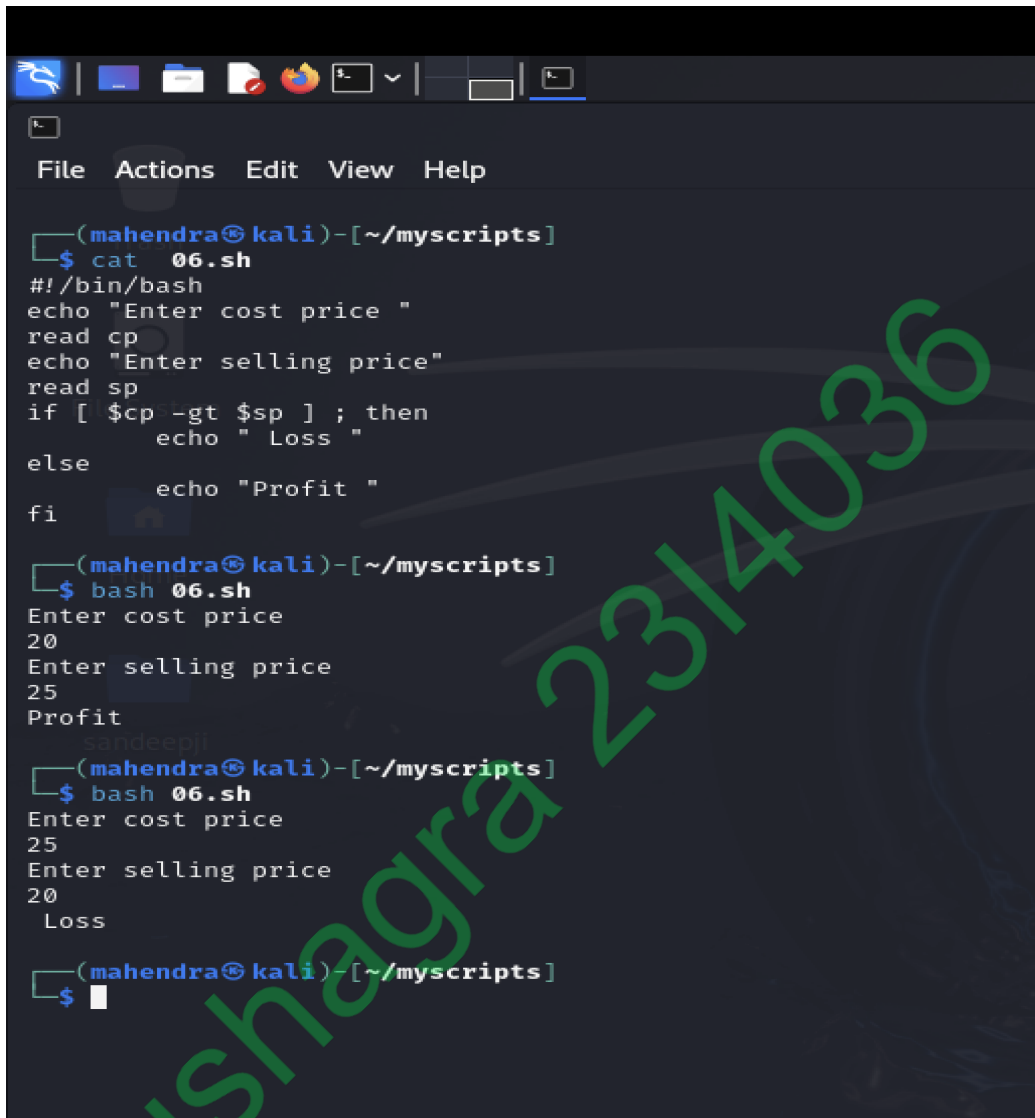
(mahendra@kali)-[~/myscripts]
$ bash 05.sh
Enter a character :
q
Alphabet

(mahendra@kali)-[~/myscripts]
$ bash 05.sh
Enter a character :
3
Digit

(mahendra@kali)-[~/myscripts]
$ bash 05.sh
Enter a character :
#
special character

(mahendra@kali)-[~/myscripts]
$
```

5. To calculate profit or loss



```
(mahendra@kali)-[~/myscripts]
$ cat 06.sh
#!/bin/bash
echo "Enter cost price "
read cp
echo "Enter selling price"
read sp
if [ $cp -gt $sp ] ; then
    echo " Loss "
else
    echo "Profit "
fi


(mahendra@kali)-[~/myscripts]
$ bash 06.sh
Enter cost price
20
Enter selling price
25
Profit

(mahendra@kali)-[~/myscripts]
$ bash 06.sh
Enter cost price
25
Enter selling price
20
Loss

(mahendra@kali)-[~/myscripts]
$
```

The image shows a terminal window with a dark background. At the top, there is a menu bar with 'File', 'Actions', 'Edit', 'View', and 'Help'. Below the menu bar, the terminal displays the command prompt for a user named 'mahendra' on a machine named 'kali', in the directory '~/myscripts'. The user enters the command 'cat 06.sh', which displays the contents of a shell script. The script prompts the user to enter a cost price and a selling price, then checks if the cost price is greater than the selling price. If it is, it prints ' Loss '; otherwise, it prints 'Profit '. The user then runs the script twice. In the first run, the cost price is 20 and the selling price is 25, resulting in 'Profit'. In the second run, the cost price is 25 and the selling price is 20, resulting in 'Loss'. A large green watermark 'Kushagra 23/4036' is overlaid diagonally across the terminal output.

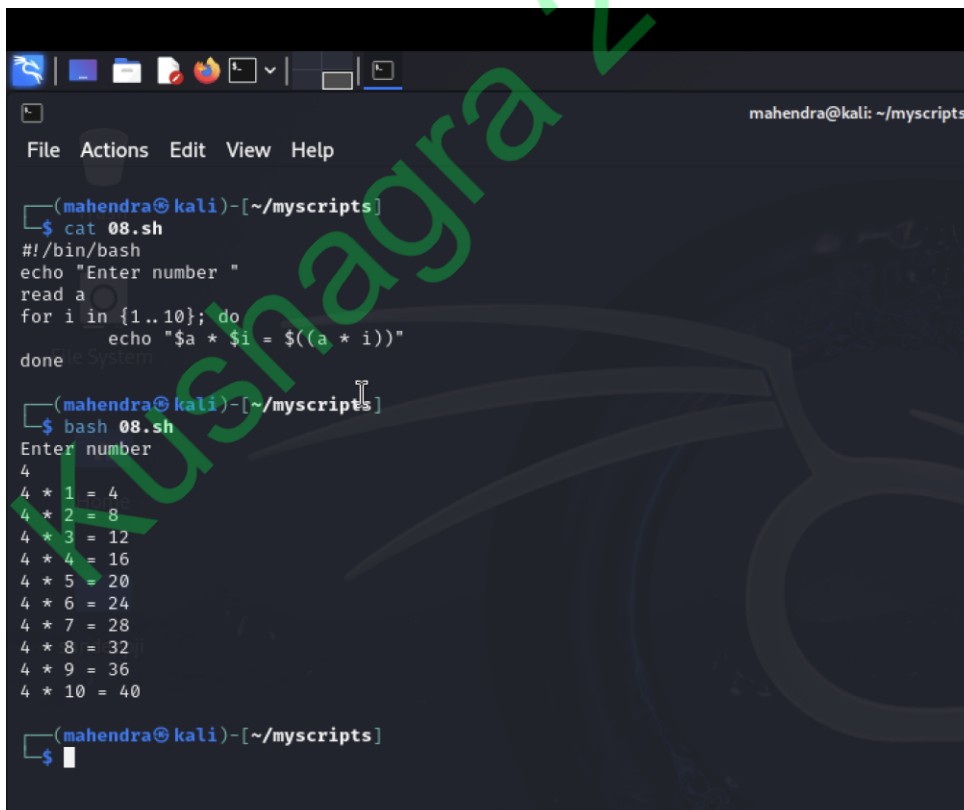
6. To print all even and odd number from 1 to 10



```
(mahendra@kali)~[/myscripts]
$ cat 07.sh
#!/bin/bash
echo "Even Numbers "
for i in {1..10}; do
    if (( i % 2 == 0 )); then echo $i ; fi
done
echo "Odd Numbers"
for i in {1..10}; do
    if (( i % 2 != 0 )); then echo $i ; fi
done

(mahendra@kali)~[/myscripts]
$ bash 07.sh
Even Numbers
2
4
6
8
10
Odd Numbers
1
3
5
7
9
(mahendra@kali)~[/myscripts]
$
```

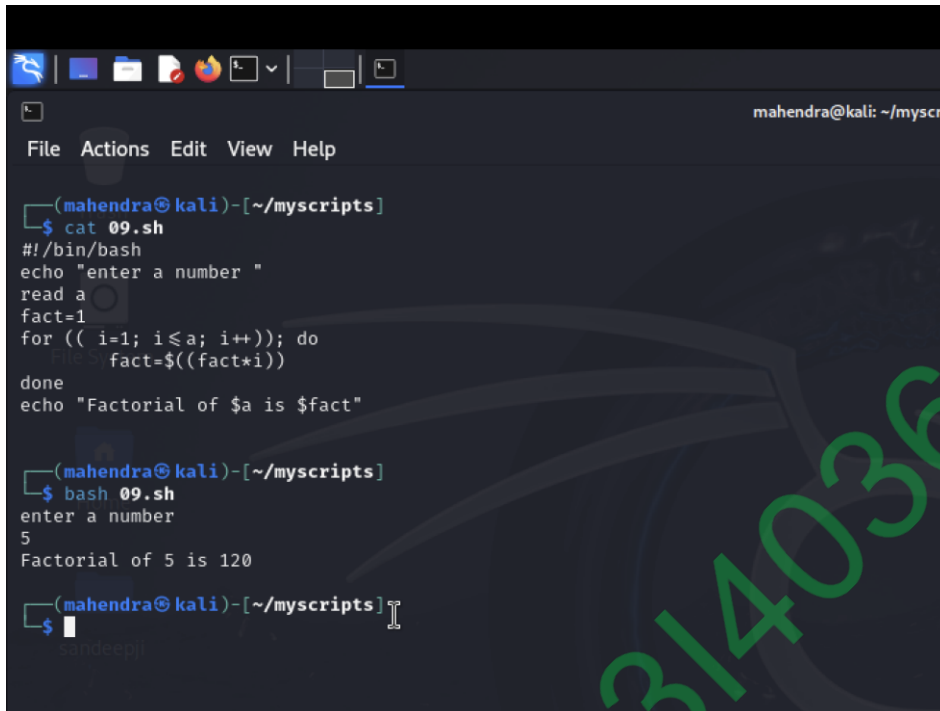
7. To print table of a given number



```
(mahendra@kali)~[/myscripts]
$ cat 08.sh
#!/bin/bash
echo "Enter number "
read a
for i in {1..10}; do
    echo "$a * $i = $((a * i))"
done

(mahendra@kali)~[/myscripts]
$ bash 08.sh
Enter number
4
4 * 1 = 4
4 * 2 = 8
4 * 3 = 12
4 * 4 = 16
4 * 5 = 20
4 * 6 = 24
4 * 7 = 28
4 * 8 = 32
4 * 9 = 36
4 * 10 = 40
(mahendra@kali)~[/myscripts]
$
```

8. To find factorial of a given integer



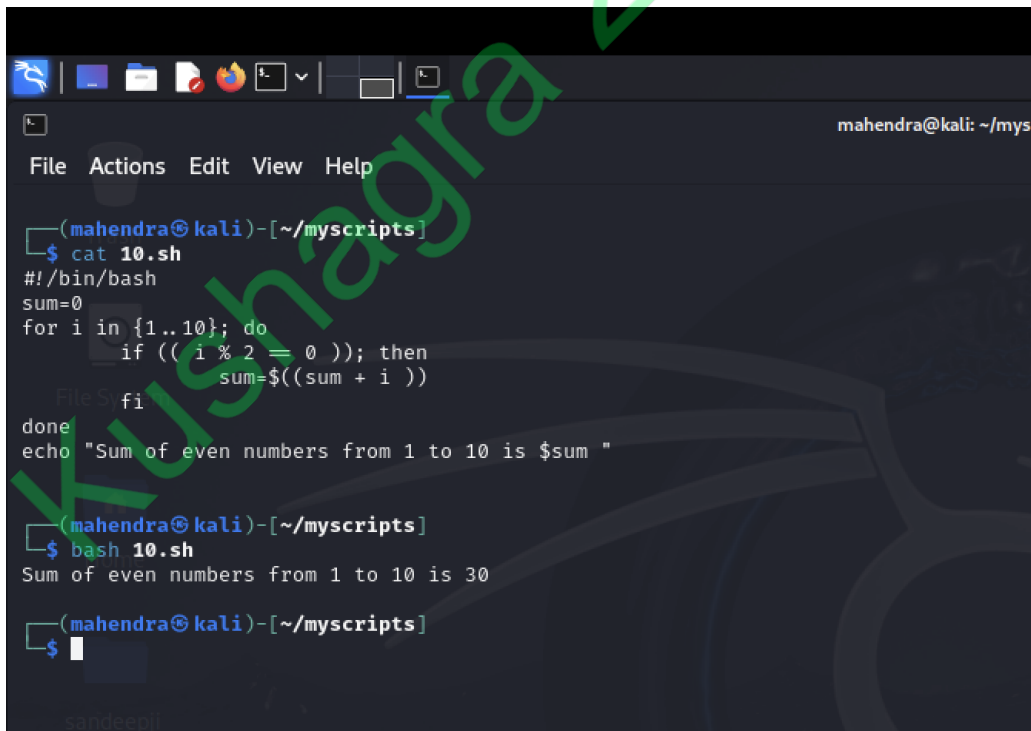
The screenshot shows a terminal window with a dark background. At the top, there is a window title bar with icons for file manager, applications, and terminal. Below the title bar, the terminal shows the user 'mahendra@kali' in the directory '~/myscripts'. The user enters the command 'cat 09.sh', which displays the contents of a shell script. The script is a bash script that prompts the user to enter a number, reads the input, and calculates its factorial using a for loop. The user then enters 'bash 09.sh', which prompts for a number. The user enters '5', and the script outputs 'Factorial of 5 is 120'. The terminal window has a menu bar with 'File', 'Actions', 'Edit', 'View', and 'Help'.

```
(mahendra@kali)-[~/myscripts]
$ cat 09.sh
#!/bin/bash
echo "enter a number "
read a
fact=1
for (( i=1; i<=a; i++ )); do
    fact=$((fact*i))
done
echo "Factorial of $a is $fact"

(mahendra@kali)-[~/myscripts]
$ bash 09.sh
enter a number
5
Factorial of 5 is 120

(mahendra@kali)-[~/myscripts]
$
```

9. To print sum of all even numbers from 1 to 10.



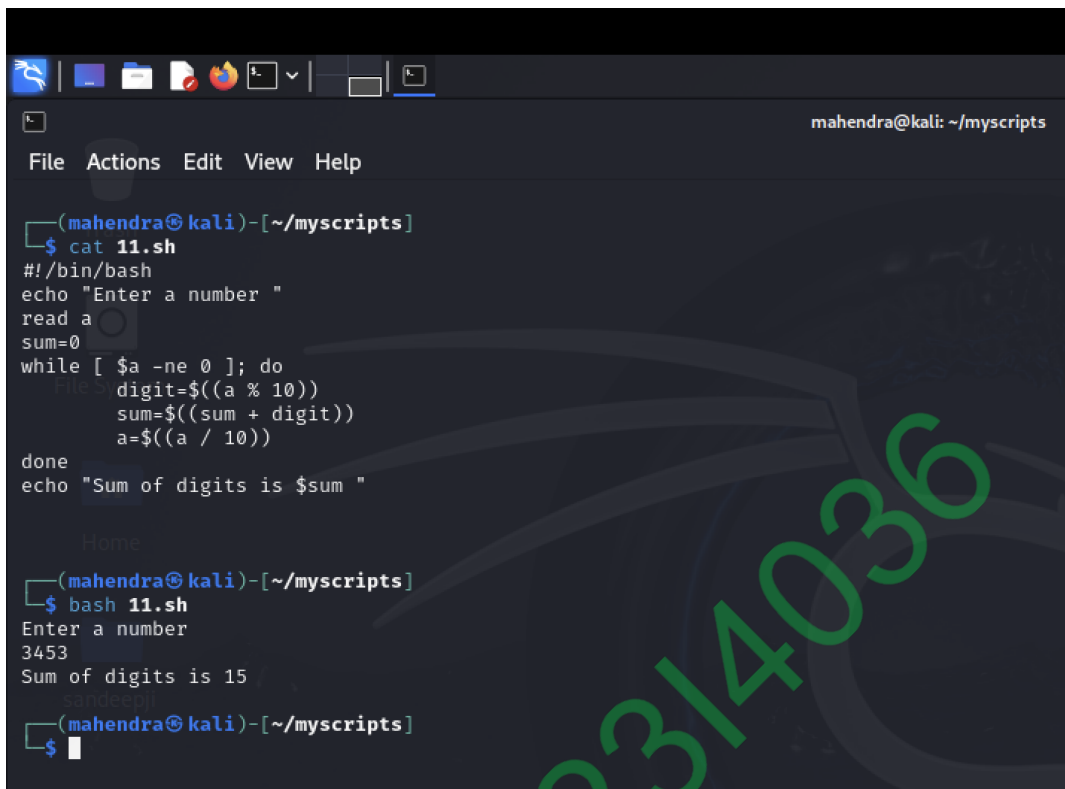
The screenshot shows a terminal window with a dark background. At the top, there is a window title bar with icons for file manager, applications, and terminal. Below the title bar, the terminal shows the user 'mahendra@kali' in the directory '~/myscripts'. The user enters the command 'cat 10.sh', which displays the contents of a shell script. The script is a bash script that initializes a sum variable to 0, then uses a for loop to iterate over numbers from 1 to 10. For each number, it checks if it is even (i % 2 == 0) and if so, adds it to the sum. The user then enters 'bash 10.sh', which outputs 'Sum of even numbers from 1 to 10 is 30'. The terminal window has a menu bar with 'File', 'Actions', 'Edit', 'View', and 'Help'.

```
(mahendra@kali)-[~/myscripts]
$ cat 10.sh
#!/bin/bash
sum=0
for i in {1..10}; do
    if (( i % 2 == 0 )); then
        sum=$((sum + i))
    fi
done
echo "Sum of even numbers from 1 to 10 is $sum "

(mahendra@kali)-[~/myscripts]
$ bash 10.sh
Sum of even numbers from 1 to 10 is 30

(mahendra@kali)-[~/myscripts]
$
```

10. To print sum of digit of any number.



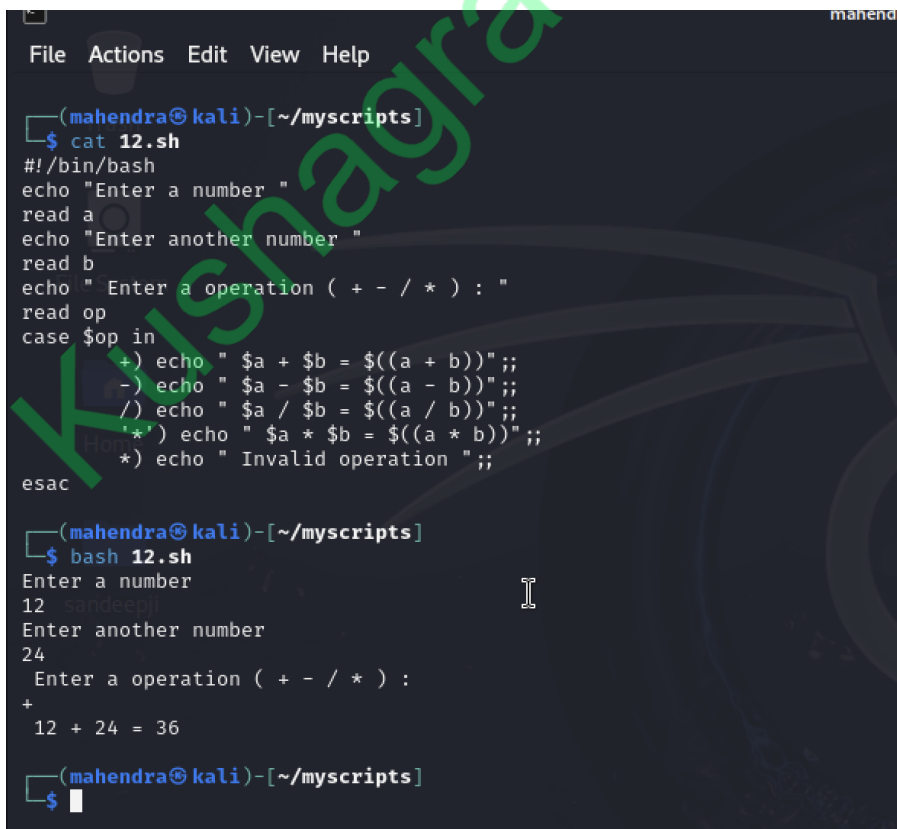
```
mahendra@kali: ~/myscripts
File Actions Edit View Help

(mahendra@kali)-[~/myscripts]
$ cat 11.sh
#!/bin/bash
echo "Enter a number "
read a
sum=0
while [ $a -ne 0 ]; do
    digit=$((a % 10))
    sum=$((sum + digit))
    a=$((a / 10))
done
echo "Sum of digits is $sum "

(mahendra@kali)-[~/myscripts]
$ bash 11.sh
Enter a number
3453
Sum of digits is 15

(mahendra@kali)-[~/myscripts]
$
```

11. To make a basic calculator which performs addition, subtraction, Multiplication, division



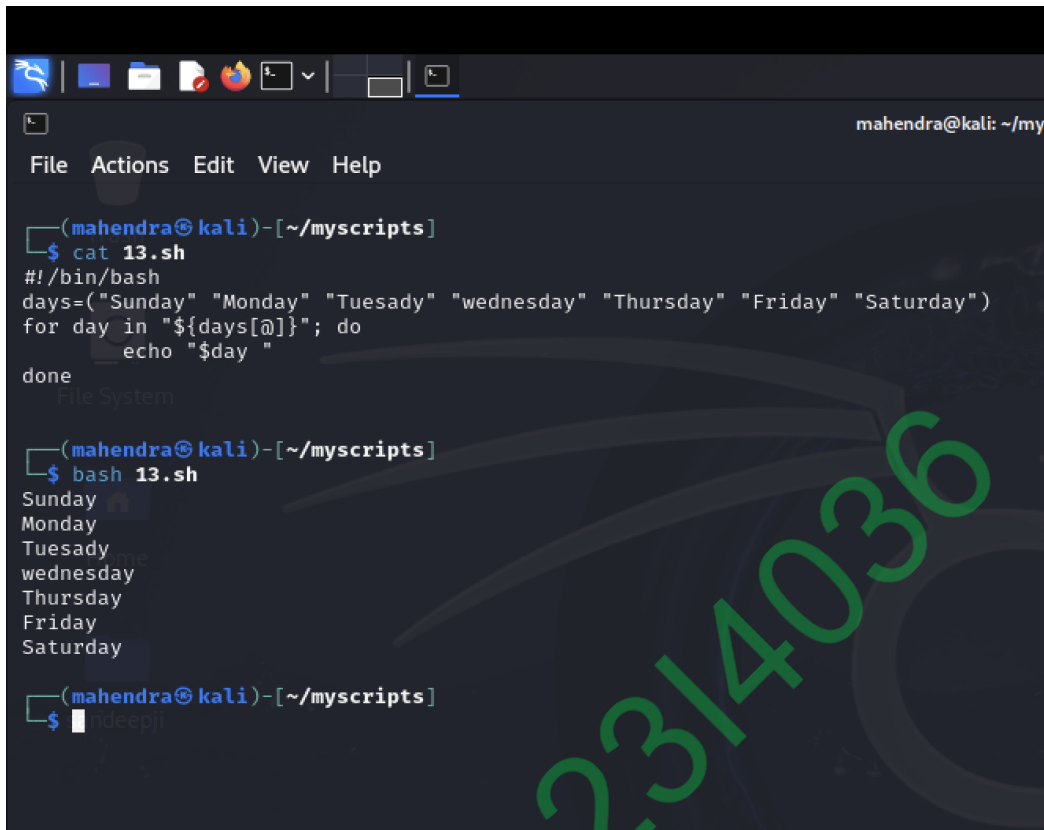
```
mahendra@kali: ~/myscripts
File Actions Edit View Help

(mahendra@kali)-[~/myscripts]
$ cat 12.sh
#!/bin/bash
echo "Enter a number "
read a
echo "Enter another number "
read b
echo " Enter a operation ( + - / * ) : "
read op
case $op in
    +) echo " $a + $b = $((a + b))" ;;
    -) echo " $a - $b = $((a - b))" ;;
    /) echo " $a / $b = $((a / b))" ;;
    *) echo " $a * $b = $((a * b))" ;;
    *) echo " Invalid operation " ;;
esac

(mahendra@kali)-[~/myscripts]
$ bash 12.sh
Enter a number
12
Enter another number
24
Enter a operation ( + - / * ) :
+
12 + 24 = 36

(mahendra@kali)-[~/myscripts]
$
```

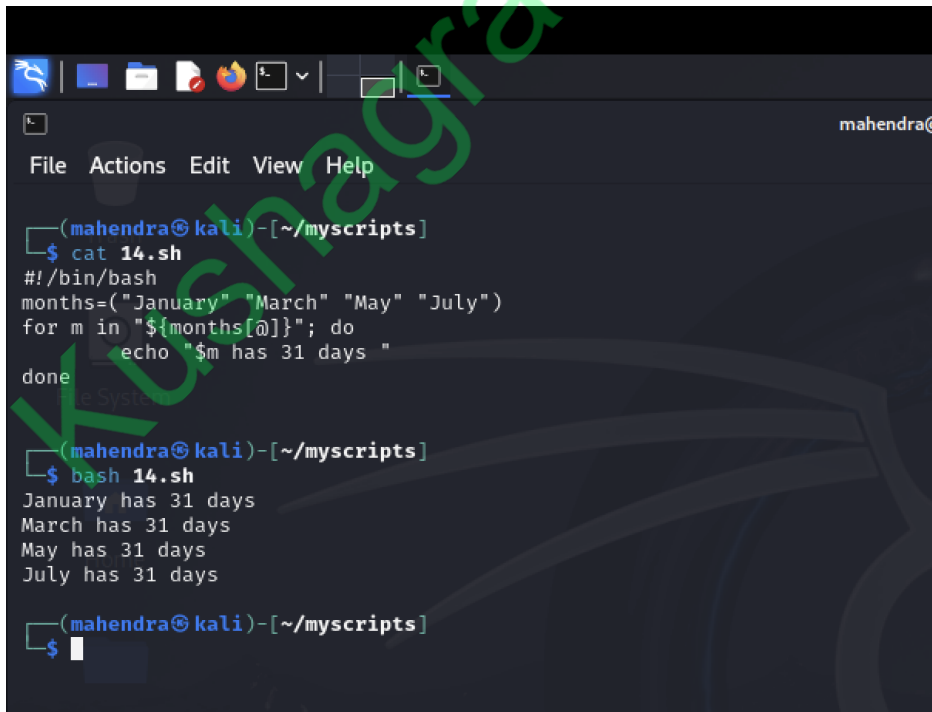
12. To print days of a week.



The screenshot shows a terminal window with a dark background. At the top, there is a menu bar with 'File', 'Actions', 'Edit', 'View', and 'Help'. The terminal prompt is '(mahendra@kali) - [~/myscripts]'. The user enters '\$ cat 13.sh', and the script content is displayed:
#!/bin/bash
days=("Sunday" "Monday" "Tuesady" "wednesday" "Thursday" "Friday" "Saturday")
for day in "\${days[@]}; do
 echo "\$day"
done
The user then enters '\$ bash 13.sh', and the output is:
Sunday
Monday
Tuesady
wednesday
Thursday
Friday
Saturday
The terminal prompt returns to '(mahendra@kali) - [~/myscripts]'. A large green watermark '23/4036' is visible diagonally across the terminal output.

```
(mahendra@kali) - [~/myscripts]
$ cat 13.sh
#!/bin/bash
days=("Sunday" "Monday" "Tuesady" "wednesday" "Thursday" "Friday" "Saturday")
for day in "${days[@]}; do
    echo "$day"
done
(mahendra@kali) - [~/myscripts]
$ bash 13.sh
Sunday
Monday
Tuesady
wednesday
Thursday
Friday
Saturday
(mahendra@kali) - [~/myscripts]
$
```

13. To print starting 4 months having 31 days.



The screenshot shows a terminal window with a dark background. At the top, there is a menu bar with 'File', 'Actions', 'Edit', 'View', and 'Help'. The terminal prompt is '(mahendra@kali) - [~/myscripts]'. The user enters '\$ cat 14.sh', and the script content is displayed:
#!/bin/bash
months=("January" "March" "May" "July")
for m in "\${months[@]}; do
 echo "\$m has 31 days"
done
The user then enters '\$ bash 14.sh', and the output is:
January has 31 days
March has 31 days
May has 31 days
July has 31 days
The terminal prompt returns to '(mahendra@kali) - [~/myscripts]'. A large green watermark '23/4036' is visible diagonally across the terminal output.

```
(mahendra@kali) - [~/myscripts]
$ cat 14.sh
#!/bin/bash
months=("January" "March" "May" "July")
for m in "${months[@]}; do
    echo "$m has 31 days"
done
(mahendra@kali) - [~/myscripts]
$ bash 14.sh
January has 31 days
March has 31 days
May has 31 days
July has 31 days
(mahendra@kali) - [~/myscripts]
$
```

14. Using functions,

- To find given number is Armstrong number or not
- To find whether a number is palindrome or not

- c. To print Fibonacci series upto n terms
- d. To find given number is prime or composite
- e. To convert a given decimal number to binary equivalent

```
mahendra@kali: ~/my
File Actions Edit View Help
(mahendra@kali)-[~/myscripts]
$ cat 15.sh
#!/bin/bash
is_armstrong() {
    num=$1
    sum=0
    temp=$num
    while [ $temp -gt 0 ]; do
        digit=$((temp % 10))
        sum=$((sum + digit * digit * digit))
        temp=$((temp / 10))
    done
    if [ $sum -eq $num ]; then
        echo "$num is an armstrong number "
    else
        echo "$num is not an armstrong number "
    fi
}
is_palindrome() {
    num=$1
    reverse=0
    temp=$num
    while [ $temp -gt 0 ]; do
        digit=$((temp % 10))
        reverse=$((reverse * 10 + digit))
        temp=$((temp / 10))
    done
    if [ $reverse -eq $num ]; then
        echo "Given number is palindrome "
    else
        echo "Given number is not an palindrome"
    fi
}
fibonacci() {
    $n=1
    a=0
    b=1
    echo "Fibonacci series upto $n terms:"
    for (( i=0 ; i<$n ; i++ )); do
        echo "$a"
        fn=$((a+b))
        a=$b
        b=$fn
    done
}
```



```

        b=$fn
    done
    echo
}
is_prime() {
    num=$1
    if [ $num -le 1 ]; then
        echo " $num is neither prime nor composite "
        return
    fi
    for (( i=2; i*i<=$num; i++ )); do
        if [ $((num % i)) -eq 0 ]; then
            echo "Given number is composite number "
            return
        fi
    done
    echo "Given number is prime number "
}
decimal_to_binary() {
    num=$1
    binary=""
    while [ $num -gt 0 ]; do
        rem=$((num % 2))
        binary="$rem$binary"
        num=$((num / 2))
    done
    echo "Binary equivalent : ${binary:-0} "
}
echo "Enter a number "
read num1
echo "Enter another number for fobonacci"
read n

echo
is_armstrong $num1
is_palindrome $num1
is_prime $num1
decimal_to_binary $num1
fibonacci $n

```

```

(mahendra@kali)-[~/myscripts]
$ bash 15.sh

```

```

(mahendra@kali)-[~/myscripts]
$ bash 15.sh
Enter a number
121
Enter another number for fobonacci
5

121 is not an armstrong number
Given number is palindrome
15.sh: line 47: [: syntax error: '-' unexpected
Given number is composite number
Binary equivalent : 1111001
15.sh: line 33: 5=1: command not found
Fibonacci series upto 5 terms:
0
1
1
2
3

```

```

(mahendra@kali)-[~/myscripts]
$

```