

Experiment [10]: [Shell programming]

Name: Vaishnavi Kumari , Roll No: 590029048 Date: 2025-11-21

AIM:

- [To learn Basics of Shell programming]
- [To understand how to check if file Exists]

Requirement:

- [Any linux distro, any kind of text editor (vs code, vim, notepad, nano,etc)]

Theory:

- [learning the command line ,looping ,function and conditional statements.]

procedure & observations

Exercise:

1.

- [length of string]

task statement:

- [write script in shell programming]

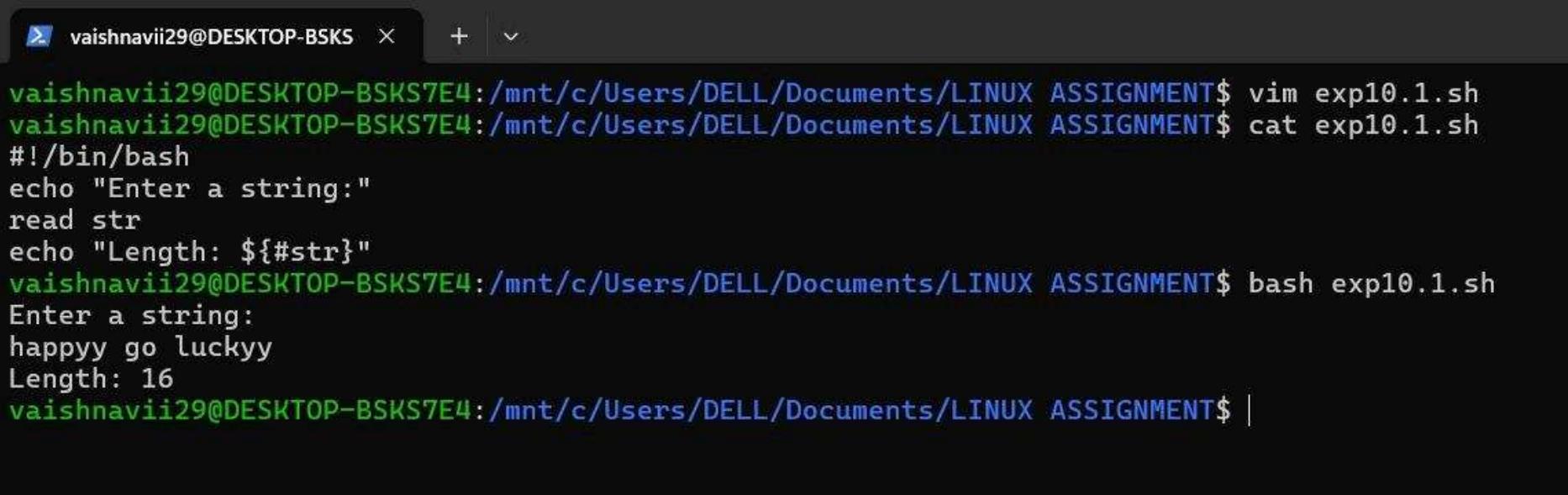
Explanation:

* \${#str} is a bash parameter expansion that returns the length of the variable Much faster than echo \$str | wc -c (which creates subshells and pipes).

command(s):

```
#!/bin/bash
echo "Enter a string:"
read str
echo "Length: ${#str}"
```

output:



The screenshot shows a terminal window with a dark background and light-colored text. The title bar indicates the user is 'vaishnavii29' on 'DESKTOP-BSKS'. The terminal displays the following session:

```
vaishnavii29@DESKTOP-BSKS7E4:/mnt/c/Users/DELL/Documents/LINUX ASSIGNMENT$ vim exp10.1.sh
vaishnavii29@DESKTOP-BSKS7E4:/mnt/c/Users/DELL/Documents/LINUX ASSIGNMENT$ cat exp10.1.sh
#!/bin/bash
echo "Enter a string:"
read str
echo "Length: ${#str}"
vaishnavii29@DESKTOP-BSKS7E4:/mnt/c/Users/DELL/Documents/LINUX ASSIGNMENT$ bash exp10.1.sh
Enter a string:
happy go luckyy
Length: 16
vaishnavii29@DESKTOP-BSKS7E4:/mnt/c/Users/DELL/Documents/LINUX ASSIGNMENT$ |
```

2.

Task statement:

- [reverse the given string]

Explanation:

```
* ${str:$i:1} extracts 1 character from position $i (string slicing)
```

```
Loop runs from last character to first
```

```
Alternative: echo $str | rev (if rev command is available)
```

command(s):

```
#!/bin/bash
echo "Enter a string:"
read str
rev=""
len=${#str}
for (( i=$len-1; i>=0; i-- ))
do
    rev="$rev${str:$i:1}"
done
echo "Reversed: $rev"
```

output:

```
vaishnavii29@DESKTOP-BSKS7E4:/mnt/c/Users/DELL/Documents/LINUX ASSIGNMENT$ vim exp10.2.sh
vaishnavii29@DESKTOP-BSKS7E4:/mnt/c/Users/DELL/Documents/LINUX ASSIGNMENT$ cat exp10.2.sh
#!/bin/bash
echo "Enter a string:"
read str
rev=""
len=${#str}
for (( i=$len-1; i>=0; i-- ))
do
    rev="$rev${str:$i:1}"
done
echo "Reversed: $rev"

vaishnavii29@DESKTOP-BSKS7E4:/mnt/c/Users/DELL/Documents/LINUX ASSIGNMENT$ bash exp10.2.sh
Enter a string:
happy go lucky
Reversed: ykul og yppah
vaishnavii29@DESKTOP-BSKS7E4:/mnt/c/Users/DELL/Documents/LINUX ASSIGNMENT$ |
```

3.

Task Statement:

- Concatenate Strings.

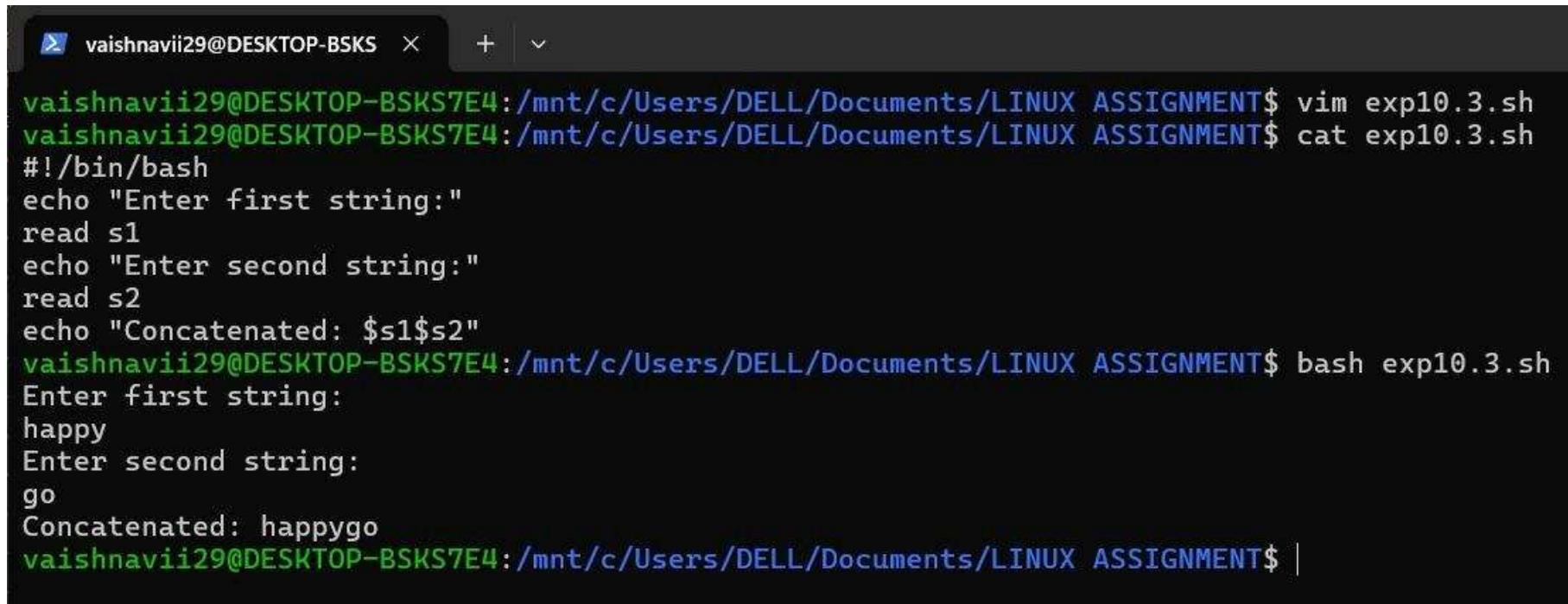
Explanation:

- In bash, simple variable juxtaposition concatenates strings No need for special operators or functions

command(s):

```
#!/bin/bash
echo "Enter first string:"
read s1
echo "Enter second string:"
read s2
echo "Concatenated: $s1$s2"
```

output:



The screenshot shows a terminal window with a dark background and light-colored text. The title bar indicates the user is on a desktop named 'DESKTOP-BSKS7E4'. The command history shows the user opening a file 'exp10.3.sh' with vim, then viewing its contents with cat. The script itself is displayed. The user then runs the script with bash, prompting for two strings ('Enter first string:' and 'Enter second string:'). The user inputs 'happy' and 'go', respectively, and the script outputs the concatenated result 'happygo'.

```
vaishnavii29@DESKTOP-BSKS7E4:/mnt/c/Users/DELL/Documents/LINUX ASSIGNMENT$ vim exp10.3.sh
vaishnavii29@DESKTOP-BSKS7E4:/mnt/c/Users/DELL/Documents/LINUX ASSIGNMENT$ cat exp10.3.sh
#!/bin/bash
echo "Enter first string:"
read s1
echo "Enter second string:"
read s2
echo "Concatenated: $s1$s2"
vaishnavii29@DESKTOP-BSKS7E4:/mnt/c/Users/DELL/Documents/LINUX ASSIGNMENT$ bash exp10.3.sh
Enter first string:
happy
Enter second string:
go
Concatenated: happygo
vaishnavii29@DESKTOP-BSKS7E4:/mnt/c/Users/DELL/Documents/LINUX ASSIGNMENT$ |
```

Assignment

1.

Task Statement:

- Factorial Function.

Explanation:

- using function with math.sh and main_script.sh

command(s):

```
#!/bin/bash
factorial() {
    local n=$1
    local result=1

    if [ $n -eq 0 ] || [ $n -eq 1 ]; then
        echo 1
        return
    fi

    for (( i=2; i<=n; i++ ))
    do
        result=$((result * i))
    done

    echo $result
}

echo "Enter a number:"
read num
```

```
result=$(factorial $num)
echo "Factorial of $num is: $result"
```

output:

```
vaishnavii29@DESKTOP-BSKS7E4:/mnt/c/Users/DELL/Documents/LINUX ASSIGNMENT$ vim exp10.4.sh
vaishnavii29@DESKTOP-BSKS7E4:/mnt/c/Users/DELL/Documents/LINUX ASSIGNMENT$ cat exp10.4.sh
#!/bin/bash
factorial() {
    local n=$1
    local result=1

    if [ $n -eq 0 ] || [ $n -eq 1 ]; then
        echo 1
        return
    fi

    for (( i=2; i<=n; i++ ))
    do
        result=$((result * i))
    done

    echo $result
}

echo "Enter a number:"
read num

result=$(factorial $num)
echo "Factorial of $num is: $result"

vaishnavii29@DESKTOP-BSKS7E4:/mnt/c/Users/DELL/Documents/LINUX ASSIGNMENT$ bash exp10.4.sh
Enter a number:
55
Factorial of 55 is: 6711489344688881664
vaishnavii29@DESKTOP-BSKS7E4:/mnt/c/Users/DELL/Documents/LINUX ASSIGNMENT$ |
```

2.

Task Statement:

- Fibonacci script.

Explanation:

- using the functions with main script and input validation.

command(s):

```
#!/bin/bash
fibonacci() {
    local n=$1
    local a=0
    local b=1
    local temp

    echo "Fibonacci series up to $n terms:"

    for (( i=0; i<n; i++ ))
    do
        echo -n "$a "
        temp=$((a + b))
        a=$b
        b=$temp
    done
    echo
}

echo "Enter number of terms:"
read terms
```

```
if [[ ! $terms =~ ^[0-9]+$ ]] || [ $terms -lt 1 ]; then
    echo "Error: Please enter a positive integer"
    exit 1
fi

fibonacci $terms
```

output:

```
vaishnavii29@DESKTOP-BSKS ~ + ×
vaishnavii29@DESKTOP-BSKS7E4:/mnt/c/Users/DELL/Documents/LINUX ASSIGNMENT$ vim exp10.5.sh
vaishnavii29@DESKTOP-BSKS7E4:/mnt/c/Users/DELL/Documents/LINUX ASSIGNMENT$ cat exp10.5.sh
#!/bin/bash
fibonacci() {
    local n=$1
    local a=0
    local b=1
    local temp

    echo "Fibonacci series up to $n terms:"

    for (( i=0; i<n; i++ ))
    do
        echo -n "$a "
        temp=$((a + b))
        a=$b
        b=$temp
    done
    echo
}

echo "Enter number of terms:"
read terms

if [[ ! $terms =~ ^[0-9]+\$ ]] || [ $terms -lt 1 ]; then
    echo "Error: Please enter a positive integer"
    exit 1
fi

fibonacci $terms
vaishnavii29@DESKTOP-BSKS7E4:/mnt/c/Users/DELL/Documents/LINUX ASSIGNMENT$ ./exp10.5.sh
Enter number of terms:
3
```

```
Fibonacci series up to 3 terms:  
0 1 1  
vaishnavii29@DESKTOP-BSKS7E4:/mnt/c/Users/DELL/Documents/LINUX ASSIGNMENT$ |
```

3.

Task Statement:

- length of filename.

Explanation:

- using loops and conditional statement.

command(s):

```
#!/bin/bash

echo "Enter directory path (press enter for current directory):"
read dirpath

if [ -z "$dirpath" ]; then
    dirpath=".."
fi

if [ ! -d "$dirpath" ];then
echo "Error: Directory '$dirpath' does not exist"
    exit 1
fi

echo "Filename lengths in '$dirpath':"
echo "-----"
```

```
for file in "$dirpath"/*
do
    if [ -e "$file" ]; then
        filename=$(basename "$file")
        length=${#filename}
        printf "%-30s : %2d characters\n" "$filename" "$length"
    fi
done
```

output:

```
vaishnavii29@DESKTOP-BSKS7E4:~/mnt/c/Users/DELL/Documents/LINUX ASSIGNMENT$ vim exp10.6.sh
vaishnavii29@DESKTOP-BSKS7E4:~/mnt/c/Users/DELL/Documents/LINUX ASSIGNMENT$ cat exp10.6.sh
echo "Enter directory path (press enter for current directory):"
read dirpath

if [ -z "$dirpath" ]; then
    dirpath=".
fi

if [ ! -d "$dirpath" ];then
echo "Error: Directory '$dirpath' does not exist"
    exit 1
fi

echo "Filename lengths in '$dirpath':"
echo "-----"

for file in "$dirpath"/*
do
    if [ -e "$file" ]; then
        filename=$(basename "$file")
        length=${#filename}
        printf "%-30s : %2d characters\n" "$filename" "$length"
    fi
done
vaishnavii29@DESKTOP-BSKS7E4:~/mnt/c/Users/DELL/Documents/LINUX ASSIGNMENT$ ./exp10.6.sh
Enter directory path (press enter for current directory):

Filename lengths in '.':
-----
Documents - Shortcut.lnk      : 24 characters
Exp10.md                      : 8 characters
WhatsApp Image 2025-09-24 at 10.23.27_364bfa62.jpg : 50 characters
```

```
WhatsApp Image 2025-09-24 at 10.26.15_3006fc04.jpg : 50 characters
WhatsApp Image 2025-09-24 at 10.26.23_7603836c.jpg : 50 characters
WhatsApp Image 2025-09-24 at 10.53.18_1540e88b.jpg : 50 characters
WhatsApp Image 2025-09-24 at 10.54.25_d7bc8449.jpg : 50 characters
WhatsApp Image 2025-11-30 at 21.10.45_e437640d.jpg : 50 characters
```

Result:

- The exercise were successfully completed for funtions and looping,conditional statements in shell scripting.