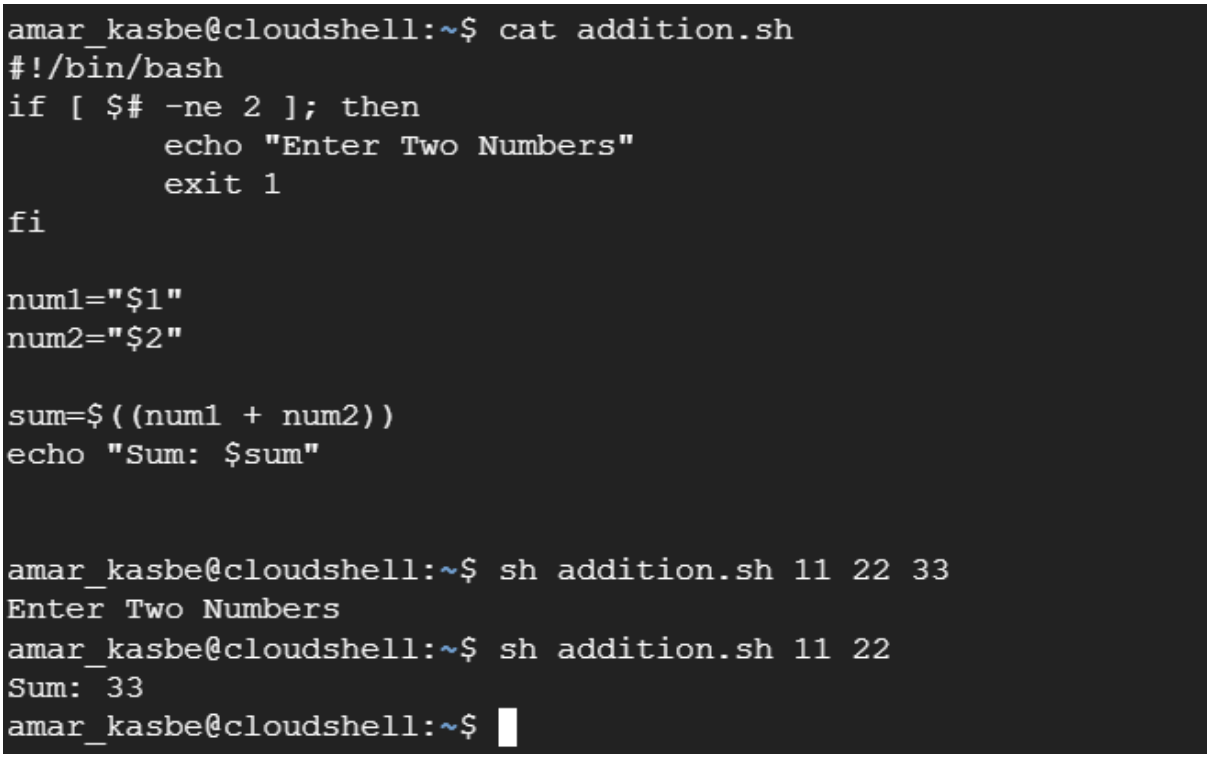
Shell Scripting Day 3

- How to Check the List of Running Processes?

- How to print PID of the current shell?

- Write a Shell Script that adds two numbers if provided as the command Line Argument and if the two numbers are not entered it outputs an Error Message.



- Create a Script test1.sh which will be executed with arguments: 10 20 Virat Kohli "Rohit Sharma"
  e.g. sh test1.sh 10 20 Kamini Patil "Kamini Patil"
  Use $* and $@ to print the command line arguments in the format below:

  10
  20
  Virat
  Kohli
  Rohit
  Sharma

  10
  20
  Virat
  Kohli
  Rohit Sharma

```
amar_kasbe@cloudshell:~$ vi text1.sh
amar_kasbe@cloudshell:~$ sh text1.sh
Using /

amar_kasbe@cloudshell:~$ sh text1.sh 10 20 Virat Kohli "Rohit Sharma"
Using /10 20 Virat Kohli Rohit Sharma
10 20 Virat Kohli Rohit Sharma
amar_kasbe@cloudshell:~$ cat text1.sh
echo "Using /$*"

for argm in "$*"; do
        echo "$argm"
done
amar_kasbe@cloudshell:~$
```

```
amar_kasbe@cloudshell:~$ vi text1.sh^C
amar_kasbe@cloudshell:~$ cat text1.sh
echo "Using /$*"

for argm in "$*"; do
        echo "$argm"
done
amar_kasbe@cloudshell:~$ vi text1.sh
amar_kasbe@cloudshell:~$ sh text1.sh 10 20 Virat Kohli "Rohit Sharma"
Using /10 20 Virat Kohli Rohit Sharma
10
20
Virat
Kohli
Rohit Sharma
amar_kasbe@cloudshell:~$ cat text1.sh
echo "Using /$@"

for argm in "$@"; do
        echo "$argm"
done
amar_kasbe@cloudshell:~$ sh text1.sh 10 20 Virat Kohli "Rohit Sharma"
Using /10 20 Virat Kohli Rohit Sharma
10
20
Virat
Kohli
Rohit Sharma
amar_kasbe@cloudshell:~$
```

- Write a script to check if the commands below are successful or not in Unix?

a. ls -l <filename that does not exist in the current directory>
b. ls -l <filename that exist in the current directory>

What was the status return when the last command executed was successful and when the last command executed was not successful.

```
amar_kasbe@cloudshell:~/script (citric-biplane-424806-b1)$ cat check3.sh
read -p "Enter File Name:- " status
ls -l $status
status=$?

# Check the exit status of command a
if [ $status -eq 0 ]; then
            echo "Command is successful."
    else
            echo "Command a was not successful. Exit status: $status"
fi

amar_kasbe@cloudshell:~/script (citric-biplane-424806-b1)$ sh check3.sh
Enter File Name:- text.txt
-rw-rw-r-- 1 amar_kasbe amar_kasbe 0 Jun  1 10:27 text.txt
Command is successful.
amar_kasbe@cloudshell:~/script (citric-biplane-424806-b1)$ sh check3.sh
Enter File Name:- aman
ls: cannot access 'aman': No such file or directory
Command a was not successful. Exit status: 2
amar_kasbe@cloudshell:~/script (citric-biplane-424806-b1)$
```

- Write a script to create an array of 10 for storing Employee name and print all array elements and their respective indexes?

```
amar_kasbe@cloudshell:~/script (citric-biplane-424806-b1)$ cat script.sh
#!/bin/bash

employees=("Amar" "Ajay" "vedant" "gaurav" "rutuja" "abhilasha" "mohini" "sagar" "rishikesh" "ronak")

for ((i = 0; i < ${#employees[@]}; i++)); do
    echo "Index $i: ${employees[i]}"
done
amar_kasbe@cloudshell:~/script (citric-biplane-424806-b1)$ sh script.sh
Index 0: Amar
Index 1: Ajay
Index 2: vedant
Index 3: gaurav
Index 4: rutuja
Index 5: abhilasha
Index 6: mohini
Index 7: sagar
Index 8: rishikesh
Index 9: ronak
amar_kasbe@cloudshell:~/script (citric-biplane-424806-b1)$
```

- Accept the index number to be deleted from the array created in the previous example as command line argument to the script, Remove the elements and display the entire array again.

```
# Function to remove element at a specific index
remove_element() {
    index=$1
    if [ $index -ge 0 ] && [ $index -lt ${#employees[@]} ]; then
        unset employees[$index]
    else
        echo "Index out of range or invalid"
    fi
}

# Check if an index is provided as command-line argument
if [ $# -eq 1 ]; then
    remove_element $1
else
    echo "Usage: $0 <index>"
    exit 1
fi

# Display the updated array
echo "Updated Employee List:"
for ((i = 0; i < ${#employees[@]}; i++)); do
    echo "Index $i: ${employees[i]}"
done

cat: 3: No such file or directory
amar_kasbe@cloudshell:~/script (citric-biplane-424806-b1)$ sh script111.sh 3
Updated Employee List:
Index 0: Amar
Index 1: Ajay
Index 2: vedant
Index 3:
Index 4: rutuja
Index 5: abhilasha
Index 6: mohini
Index 7: sagar
Index 8: rishikesh
amar_kasbe@cloudshell:~/script (citric-biplane-424806-b1)$
```

- Write 5 functions below:

a. addition() # Function to add 2 numbers

```
amar_kasbe@cloudshell:~/script (citric-biplane-424806-b1)$ sh add.sh
Addition of 5 and 10 is 15
amar_kasbe@cloudshell:~/script (citric-biplane-424806-b1)$ cat add.sh
addition() {
        num1=$1
        num2=$2
        sum=$((num1 + num2))
        echo "Addition of $num1 and $num2 is $sum"
}

addition 5 10

amar_kasbe@cloudshell:~/script (citric-biplane-424806-b1)$
```

b. subtraction() #Function to subtract second number from first number

```
amar_kasbe@cloudshell:~/script (citric-biplane-424806-b1)$ sh sub.sh
Substaction of 50 and 10 is 40
amar_kasbe@cloudshell:~/script (citric-biplane-424806-b1)$ cat sub.sh
sub(){
        num1=$1
        num2=$2
        sub=$((num2-num1))
        echo "Substaction of $num2 and $num1 is $sub"
}
sub 10 50
amar_kasbe@cloudshell:~/script (citric-biplane-424806-b1)$
```

c. multiplication() #Function to multiply 2 number

```
amar_kasbe@cloudshell:~/script (citric-biplane-424806-b1)$ sh mul.sh
Multiplication of 10 and 3 is 30
amar_kasbe@cloudshell:~/script (citric-biplane-424806-b1)$ cat mul.sh
mul(){
        num1=$1
        num2=$2
        mul=$(($num1*$num2))
        echo "Multiplication of $num1 and $num2 is $mul"

}

mul 10 3
amar_kasbe@cloudshell:~/script (citric-biplane-424806-b1)$
```

d. division() #Function to divide second number by first number

```
amar_kasbe@cloudshell:~/script (citric-biplane-424806-b1)$ sh did.sh
Divison of 64 and 2 is 32
amar_kasbe@cloudshell:~/script (citric-biplane-424806-b1)$ cat did.sh
div() {
        num1=$1
        num2=$2
        div=$((num1/num2))
        echo "Divison of $num1 and $num2 is $div"
}

div 64 2
amar_kasbe@cloudshell:~/script (citric-biplane-424806-b1)$
```

e. Write a function to display a menu as below,

******** calculator ********

0. exit"
1. addition"
2. substraction"
3. multiplication"
4. division"

Enter the choice :

Ask the user to enter the choice and then 2 numbers to perform the calculation as per the choice provided by the user. If the user enters 0 as choince display the message "Closing the calculator"

```
                echo "Error: Division by zero"
            else
                echo "$num1 / $num2 = $(($num1 / $num2))"
            fi
            ;;
        *)
            echo "Invalid choice"
            ;;
    esac
}

display_menu
calculator

amar_kasbe@cloudshell:~/script (citric-biplane-424806-b1)$ sh call.sh
******** calculator ********
0. exit
1. addition
2. subtraction
3. multiplication
4. division
Enter your choice: 2
Enter two numbers: 10 50
10 - 50 = -40
amar_kasbe@cloudshell:~/script (citric-biplane-424806-b1)$ sh call.sh
******** calculator ********
0. exit
1. addition
2. subtraction
3. multiplication
4. division
Enter your choice: 1
Enter two numbers: 5 10
5 + 10 = 15
amar_kasbe@cloudshell:~/script (citric-biplane-424806-b1)$
```

- Create a text file employee.txt as shown in the presentation today.

a. Write an awk command to print the last line.

```
amar_kasbe@cloudshell:~/script (citric-biplane-424806-b1)$ cat <<EOF > employee.txt
Employee ID | Name         | Department   | Salary
-----------------------------------------------
1001        | John Smith   | Engineering  | 60000
1002        | Alice Johnson| Marketing    | 55000
1003        | Michael Brown| Sales        | 58000
1004        | Emily Davis  | Engineering  | 62000
1005        | David Miller | HR           | 50000
EOF
amar_kasbe@cloudshell:~/script (citric-biplane-424806-b1)$ awk 'END {print}' employee.txt
1005        | David Miller | HR           | 50000
amar_kasbe@cloudshell:~/script (citric-biplane-424806-b1)$
```

b. Write an awk command to print rows with department "account" only.

```
amar_kasbe@cloudshell:~/script (citric-biplane-424806-b1)$ awk -F '|' '{print $3}' employee.txt
 Department

 Engineering
 Marketing
 Sales
 Engineering
 HR
amar_kasbe@cloudshell:~/script (citric-biplane-424806-b1)$
```

c. Write an awk command to print columns below:
serial number, employee name, employee salary

```
amar_kasbe@cloudshell:~/script (citric-biplane-424806-b1)$ awk -F '|' '{print $1 $2 $4}' employee.txt
Employee ID  Name          Salary
-----------------------------------------------
1001         John Smith    60000
1002         Alice Johnson 55000
1003         Michael Brown 58000
1004         Emily Davis   62000
1005         David Miller  50000
amar_kasbe@cloudshell:~/script (citric-biplane-424806-b1)$
```

d. Do you know any alternatives to the awk commands used in a, b and c above ? If Yes, please list the commands.

Ans - Yes, we can also use the “tail” and “grep” commands to filter.

- Write a script to accept year from user and check whether leap year or not.

```
amar_kasbe@cloudshell:~/script (citric-biplane-424806-b1)$ sh leap_yr.sh
Enter a year: 2025
2025 is not a leap year.
amar_kasbe@cloudshell:~/script (citric-biplane-424806-b1)$ sh leap_yr.sh
Enter a year: 2000
2000 is a leap year.
amar_kasbe@cloudshell:~/script (citric-biplane-424806-b1)$ cat leap_yr.sh
#!/bin/bash

is_leap_year() {
    year=$1

    # Check if the year is divisible by 4 and not divisible by 100, or divisible by 400
    if [ $((year % 4)) -eq 0 ] && [ $((year % 100)) -ne 0 ] || [ $((year % 400)) -eq 0 ]; then
        echo "$year is a leap year."
    else
        echo "$year is not a leap year."
    fi
}


read -p "Enter a year: " user_year


is_leap_year $user_year

amar_kasbe@cloudshell:~/script (citric-biplane-424806-b1)$
```

- Write a shell script to accept the name from the user and check whether user entered name is file or directory. If name is file display its size and if it is directory display its contents.

```
#!/bin/bash

# Accept name from the user
read -p "Enter a file or directory name: " name

# Check if the name is a file
if [ -f "$name" ]; then
    echo "$name is a file."
    # Display the size of the file
    echo "Size of $name: $(du -h "$name" | cut -f1)"
# Check if the name is a directory
elif [ -d "$name" ]; then
    echo "$name is a directory."
    # Display the contents of the directory
    echo "Contents of $name:"
    ls "$name"
else
    echo "The entered name does not exist or is not a valid file or directory."
fi

amar_kasbe@cloudshell:~/script (citric-biplane-424806-b1)$ sh check.sh
Enter a file or directory name: employee
The entered name does not exist or is not a valid file or directory.
amar_kasbe@cloudshell:~/script (citric-biplane-424806-b1)$ ls
add.sh  amar  cal1.sh  cal.sh  check.sh  did.sh  employee.txt  emp.sh  leap_yr.sh  mul.sh  script1.sh  script.sh  status.sh  sub.sh  text  text.txt
amar_kasbe@cloudshell:~/script (citric-biplane-424806-b1)$ sh check.sh
Enter a file or directory name: employee.txt
employee.txt is a file.
Size of employee.txt: 4.0K
amar_kasbe@cloudshell:~/script (citric-biplane-424806-b1)$ sh check.sh
Enter a file or directory name: amar
amar is a directory.
Contents of amar:
amar_kasbe@cloudshell:~/script (citric-biplane-424806-b1)$
```

- Write a Program to find the greatest of three numbers

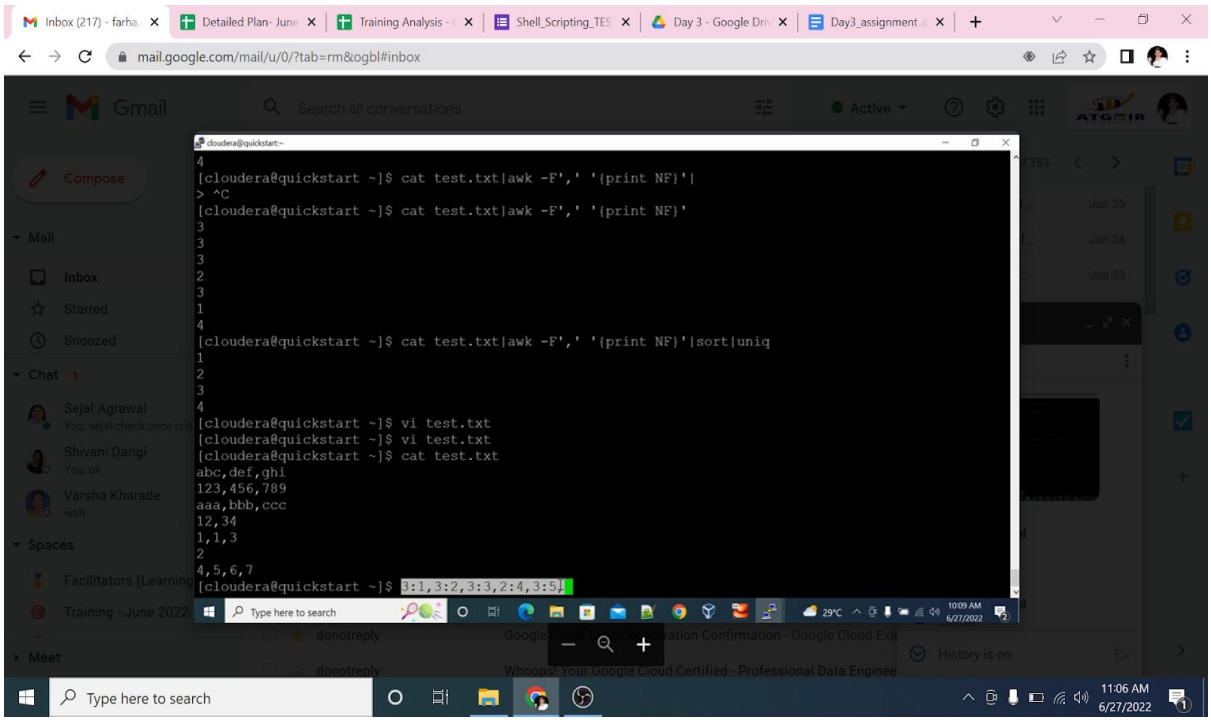
```
amar_kasbe@cloudshell:~/script (citric-biplane-424806-b1)$ cat great.sh
find_greatest() {
    if [ $1 -gt $2 ] && [ $1 -gt $3 ]; then
        echo "$1 is the greatest."
    elif [ $2 -gt $1 ] && [ $2 -gt $3 ]; then
        echo "$2 is the greatest."
    else
        echo "$3 is the greatest."
    fi
}


read -p "Enter first number:-" num1
read -p "Enter second number:-" num2
read -p "Enter third number:-" num3

find_greatest $num1 $num2 $num3
amar_kasbe@cloudshell:~/script (citric-biplane-424806-b1)$ sh great.sh
Enter first number:-45
Enter second number:-85
Enter third number:-100
100 is the greatest.
amar_kasbe@cloudshell:~/script (citric-biplane-424806-b1)$
```

- Write a Program to find whether a given number is positive or negative

```
amar_kasbe@cloudshell:~/script (citric-biplane-424806-b1)$ cat check1.sh
#!/bin/bash

check_number() {
    if [ $1 -gt 0 ]; then
        echo "$1 is positive."
    elif [ $1 -lt 0 ]; then
        echo "$1 is negative."
    else
        echo "$1 is zero."
    fi
}

read -p "Enter a number: " number

check_number $number

amar_kasbe@cloudshell:~/script (citric-biplane-424806-b1)$ sh check1.sh
Enter a number: -15
-15 is negative.
amar_kasbe@cloudshell:~/script (citric-biplane-424806-b1)$ sh check1.sh
Enter a number: 0
0 is zero.
amar_kasbe@cloudshell:~/script (citric-biplane-424806-b1)$ sh check1.sh
Enter a number: 5
5 is positive.
amar_kasbe@cloudshell:~/script (citric-biplane-424806-b1)$
```

- Commands for below:
    1. List the directory contents date wise sorted.
    Ans => ls -lt
    2. List the directory contents size wise sorted.
    Ans => ls -lS
    3. List the contents of sub directory.
    Ans => ls <subdirectory_path>

- For the given file contents write a shell script that prints Number of field delimiters : Line Number
    Note: in this file the field delimiter is comma(,)



```
amar_kasbe@cloudshell:~/script (citric-biplane-424806-b1)$ cat sample.csv
John,Doe,30,New York
Jane,Smith,25,Los Angeles
Michael,Johnson,40,Chicago
amar_kasbe@cloudshell:~/script (citric-biplane-424806-b1)$ cat script12.sh
#!/bin/bash

file=$1

if [ -z "$file" ]; then
    echo "Usage: $0 <filename>"
    exit 1
fi

if [ ! -f "$file" ]; then
    echo "File $file not found!"
    exit 1
fi

line_num=1

while IFS= read -r line; do
    num_delimiters=$(echo "$line" | awk -F ',' '{print NF-1}')
    echo "$num_delimiters : $line_num"
    ((line_num++))
done < "$file"

amar_kasbe@cloudshell:~/script (citric-biplane-424806-b1)$ sh script12.sh sample.csv
3 : 1
3 : 2
3 : 3
amar_kasbe@cloudshell:~/script (citric-biplane-424806-b1)$
```

- write a shell script to count total no of arguments. If total arguments are 2 then come out of the script else print the arguments.

```
amar_kasbe@cloudshell:~/script (citric-biplane-424806-b1)$ cat check_arg.sh
total_arguments=$#

if [ $total_arguments -ne 2 ]; then
        echo "Total number of arguments: $total_arguments"
        echo "Arguments: $@"
else
        echo "Total number of arguments is 2. Exiting the script."
        exit 0
fi

amar_kasbe@cloudshell:~/script (citric-biplane-424806-b1)$ sh check_arg.sh 15 4 45
Total number of arguments: 3
Arguments: 15 4 45
amar_kasbe@cloudshell:~/script (citric-biplane-424806-b1)$ sh check_arg.sh 15 4
Total number of arguments is 2. Exiting the script.
amar_kasbe@cloudshell:~/script (citric-biplane-424806-b1)$
```