EXP NO : 3 Date : 21/03/2019

# SHELL PROGRAMMING - 1

## Aim :

To practice various shell scripting programs

**Question 1 :**

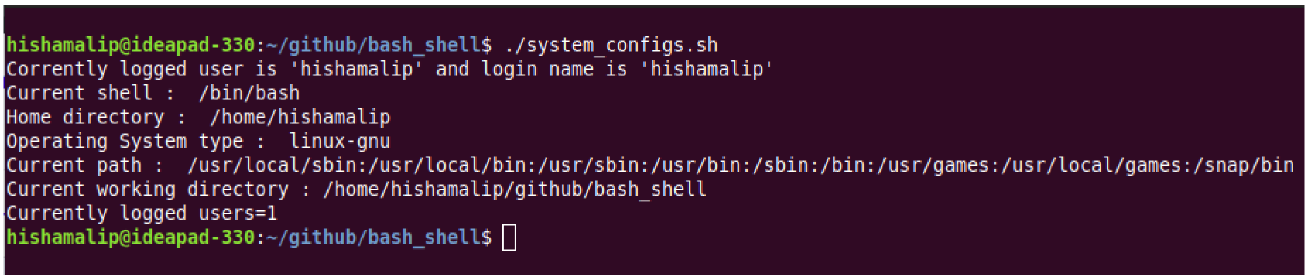
Write a shell script to show various system configuration like

1. Curr ently logged user and his login name
2. Your current shell
3. Your home directory
4. Your operating system type
5. Your current path setting
6. Your current working directory
7. Number of users currently logged in

Program :

```
echo "Corrently logged user is '$USER' and login name is '$LOGNAME'"
echo "Current shell : " $SHELL
echo "Home directory : " $HOME
echo "Operating System type : " $OSTYPE
echo "Current path : " $PATH
echo "Current working directory :" $PWD
echo -e "Currently logged `who --count | sed -n 2p | cut -b 3-` "
```

Output :



```
hishamalip@ideapad-330:~/github/bash_shell$ ./system_configs.sh
Corrently logged user is 'hishamalip' and login name is 'hishamalip'
Current shell :  /bin/bash
Home directory :  /home/hishamalip
Operating System type :  linux-gnu
Current path :  /usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games:/usr/local/games:/snap/bin
Current working directory : /home/hishamalip/github/bash_shell
Currently logged users=1
hishamalip@ideapad-330:~/github/bash_shell$ 
```

---------------------------------------------------------------------------------------------------------------

**Question 2 :**

Write a shell script to show various system configurations like

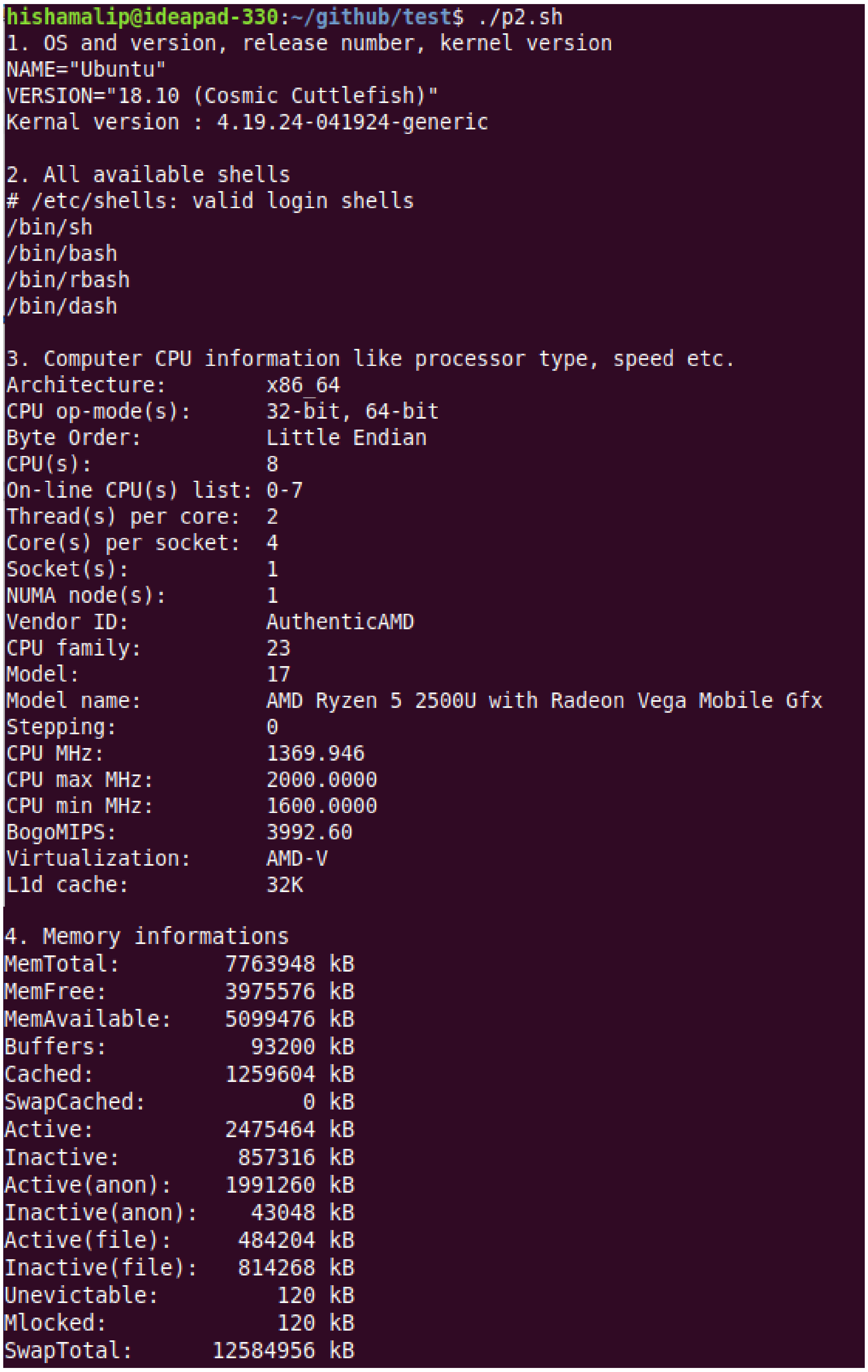
1. your OS and version, release number, kernel version
2. all available shells
3. computer CPU information like processor type, speed etc
4. memory information
5. hard disk information like size of hard-disk, cache memory, model etc
6. File system (Mounted)

Program :

```
echo "1. OS and version, release number, kernel version"
cat /etc/os-release | head -2
echo "Kernal version : `uname -r`"
echo -e "\n2. All available shells"
cat /etc/shells
```

```
echo -e "\n3. Computer CPU information like processor type, speed etc."
lscpu | head -20
echo -e "\n4. Memory informations"
cat /proc/meminfo | head -15
echo -e "\n5. Hard disk information"
sudo lshw -c disk
echo -e "\n6. File system (Mounted)"
lsblk /dev/sda
```

Output :

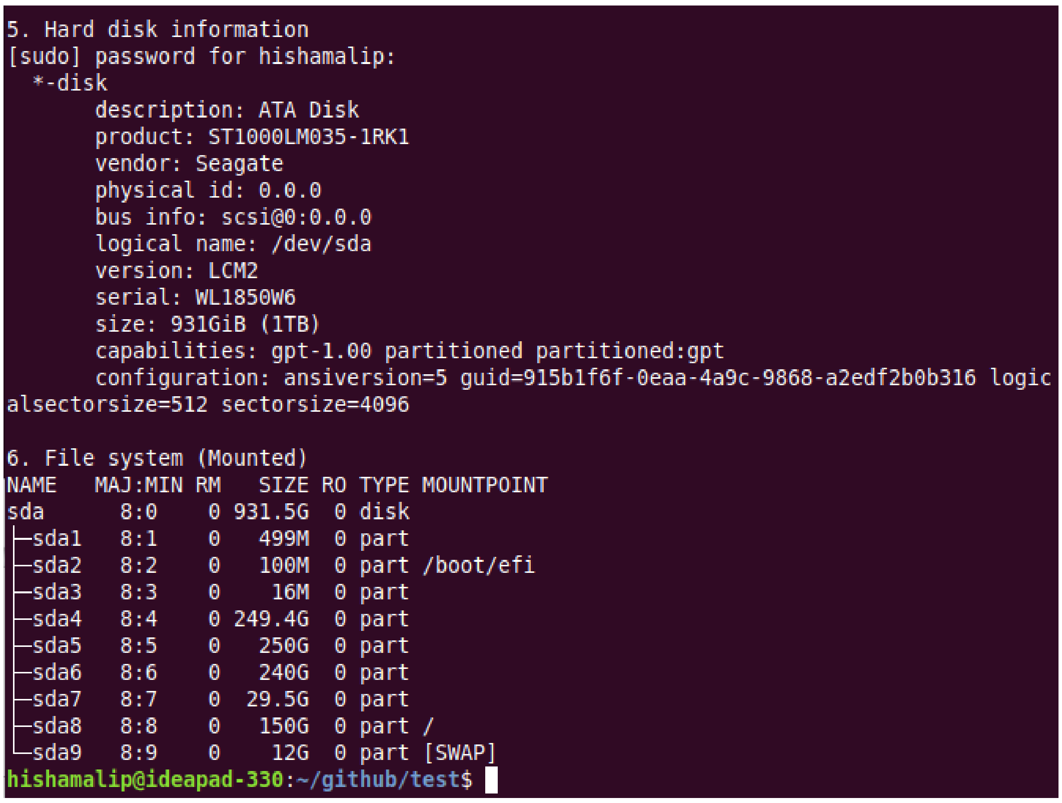


```
hishamalip@ideapad-330:~/github/test$ ./p2.sh
1. OS and version, release number, kernel version
NAME="Ubuntu"
VERSION="18.10 (Cosmic Cuttlefish)"
Kernal version : 4.19.24-041924-generic

2. All available shells
# /etc/shells: valid login shells
/bin/sh
/bin/bash
/bin/rbash
/bin/dash

3. Computer CPU information like processor type, speed etc.
Architecture:        x86_64
CPU op-mode(s):      32-bit, 64-bit
Byte Order:          Little Endian
CPU(s):              8
On-line CPU(s) list: 0-7
Thread(s) per core:  2
Core(s) per socket:  4
Socket(s):           1
NUMA node(s):        1
Vendor ID:           AuthenticAMD
CPU family:          23
Model:               17
Model name:          AMD Ryzen 5 2500U with Radeon Vega Mobile Gfx
Stepping:            0
CPU MHz:             1369.946
CPU max MHz:         2000.0000
CPU min MHz:         1600.0000
BogoMIPS:            3992.60
Virtualization:      AMD-V
L1d cache:           32K

4. Memory informations
MemTotal:        7763948 kB
MemFree:         3975576 kB
MemAvailable:    5099476 kB
Buffers:           93200 kB
Cached:          1259604 kB
SwapCached:            0 kB
Active:          2475464 kB
Inactive:         857316 kB
Active(anon):    1991260 kB
Inactive(anon):    43048 kB
Active(file):     484204 kB
Inactive(file):   814268 kB
Unevictable:         120 kB
Mlocked:             120 kB
SwapTotal:      12584956 kB
```



```
5. Hard disk information
[sudo] password for hishamalip:
  *-disk
       description: ATA Disk
       product: ST1000LM035-1RK1
       vendor: Seagate
       physical id: 0.0.0
       bus info: scsi@0:0.0.0
       logical name: /dev/sda
       version: LCM2
       serial: WL1850W6
       size: 931GiB (1TB)
       capabilities: gpt-1.00 partitioned partitioned:gpt
       configuration: ansiversion=5 guid=915b1f6f-0eaa-4a9c-9868-a2edf2b0b316 logic
alsectorsize=512 sectorsize=4096

6. File system (Mounted)
NAME   MAJ:MIN RM   SIZE RO TYPE MOUNTPOINT
sda      8:0    0 931.5G  0 disk
├─sda1   8:1    0   499M  0 part
├─sda2   8:2    0   100M  0 part /boot/efi
├─sda3   8:3    0    16M  0 part
├─sda4   8:4    0 249.4G  0 part
├─sda5   8:5    0   250G  0 part
├─sda6   8:6    0   240G  0 part
├─sda7   8:7    0  29.5G  0 part
├─sda8   8:8    0   150G  0 part /
└─sda9   8:9    0    12G  0 part [SWAP]
hishamalip@ideapad-330:~/github/test$
```

----------------------------------------------------------------------------------------------------------------

**Question 3 :**

Write a shell script to implement a menu driven calculator with following functions

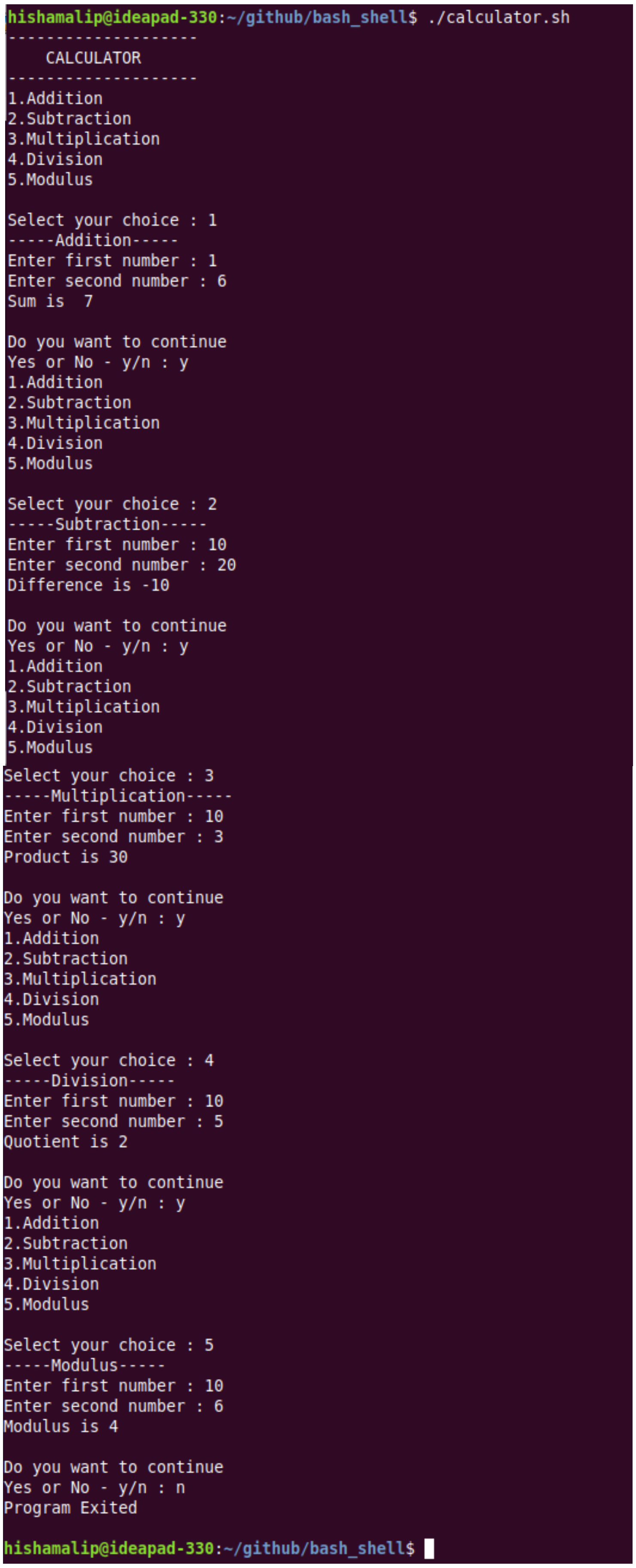
1. Addition
2. Subtraction
3. Multiplication
4. Division
5. Modulus

Program :

```
function input() #function to read two numbers
{
        read -p "Enter first number : " num1
        read -p "Enter second number : " num2
}
function prompt() #function to prompt a message
{
        echo "Do you want to continue"
        read -p "Yes or No - y/n : " flag
        if [ $flag = "n" ] ; then
                echo -e "Program Exited\n"
                exit
        elif [ $flag != "y" ] ; then
                echo "Invalid input"
                prompt
        fi
}
```

```
function addition() #function for addition
{
        echo "-----Addition-----"
        input
        result=`expr $num1 + $num2`
        echo -e "Sum is  $result \n"
}
function subtraction() #function for subtraction
{
        echo "-----Subtraction-----"
        input
        result=$(expr $num1 - $num2)
        echo -e "Difference is $result \n"
}
function multiplication() #function for multiplication
{
        echo "-----Multiplication-----"
        input
        result=`expr $num1 \* $num2`
        echo -e "Product is $result \n"
}
function division() #function for division
{
        echo "-----Division-----"
        input
        result=$(expr $num1 / $num2)
        if [ $num2 -eq 0 ]; then
                echo -e "Can't define. \n"
        else
                echo -e "Quotient is $result \n"
        fi
}
function modulus() #function for modulus
{
        echo "-----Modulus-----"
        input
        result=`expr $num1 % $num2`
        echo -e "Modulus is $result \n "
}
echo "--------------------"
echo "    CALCULATOR"
echo "--------------------"
while true ; do
        echo -e "1.Addition\n2.Subtraction\n3.Multiplication\n4.Division\n5.Modulus\n"
        read -p  "Select your choice : " ch
                case  $ch in
                        1)      addition;;
                        2)      subtraction;;
                        3)      multiplication;;
                        4)      division;;
                        5)      modulus;;
                        *)      echo "Invalide Choice";;   #default case
                esac
        prompt
done
```

Output :



```
hishamalip@ideapad-330:~/github/bash_shell$ ./calculator.sh
--------------------
    CALCULATOR
--------------------
1.Addition
2.Subtraction
3.Multiplication
4.Division
5.Modulus

Select your choice : 1
-----Addition-----
Enter first number : 1
Enter second number : 6
Sum is  7

Do you want to continue
Yes or No - y/n : y
1.Addition
2.Subtraction
3.Multiplication
4.Division
5.Modulus

Select your choice : 2
-----Subtraction-----
Enter first number : 10
Enter second number : 20
Difference is -10

Do you want to continue
Yes or No - y/n : y
1.Addition
2.Subtraction
3.Multiplication
4.Division
5.Modulus
Select your choice : 3
-----Multiplication-----
Enter first number : 10
Enter second number : 3
Product is 30

Do you want to continue
Yes or No - y/n : y
1.Addition
2.Subtraction
3.Multiplication
4.Division
5.Modulus

Select your choice : 4
-----Division-----
Enter first number : 10
Enter second number : 5
Quotient is 2

Do you want to continue
Yes or No - y/n : y
1.Addition
2.Subtraction
3.Multiplication
4.Division
5.Modulus

Select your choice : 5
-----Modulus-----
Enter first number : 10
Enter second number : 6
Modulus is 4

Do you want to continue
Yes or No - y/n : n
Program Exited

hishamalip@ideapad-330:~/github/bash_shell$
```

**Question 4 :**

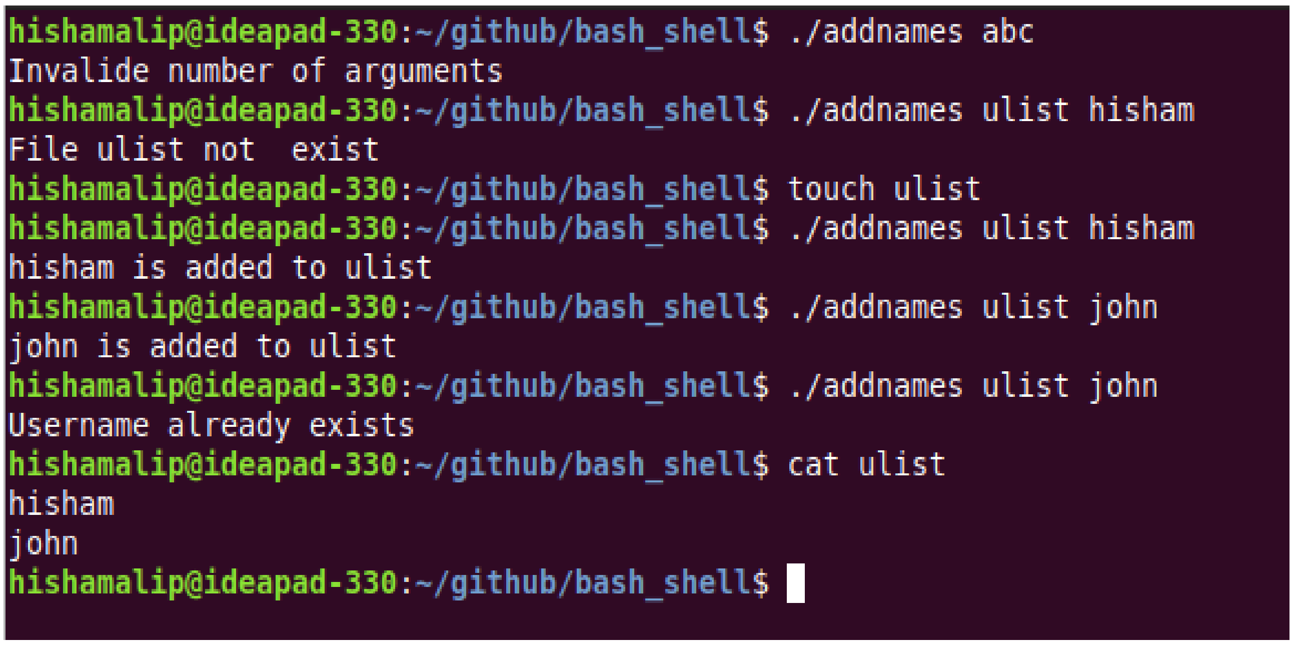
Write a script called addnames that is to be called as follows ./addnames ulist username. Here ulist is the name of the file that contains list of user names and username is a particular student's username. The script should

1. check that the correct number of arguments was received and print a message, in case the number of arguments is incorrect
2. check whether the ulist file exists and print an error message if it does not
3. check whether the username already exists in the file. If the username exists, print a message stating that the name already exists. Otherwise, add the username to the end of the list.

Program :

```
if [ $# -ne 2 ] ; then
        echo "Invalide number of arguments"
        exit
else
        file1=$1
        if [ ! -f $file1 ] ; then
                echo "File $1 not  exist"
        elif  [ -f $file1  ]&&[ $file1 != "ulist" ] ; then
                echo "The file "\"$file1"\" cant be used for this operation"
        else
                uname=$2
                uname_check="$(cat ulist|grep -w $uname)"
                if [ "$uname_check" == "$uname" ] ; then
                        echo "Username already exists"
                else
                        echo $uname >> $file1
                        echo "$uname is added to $file1"
                fi
        fi
fi
```

Output:



```
hishamalip@ideapad-330:~/github/bash_shell$ ./addnames abc
Invalide number of arguments
hishamalip@ideapad-330:~/github/bash_shell$ ./addnames ulist hisham
File ulist not  exist
hishamalip@ideapad-330:~/github/bash_shell$ touch ulist
hishamalip@ideapad-330:~/github/bash_shell$ ./addnames ulist hisham
hisham is added to ulist
hishamalip@ideapad-330:~/github/bash_shell$ ./addnames ulist john
john is added to ulist
hishamalip@ideapad-330:~/github/bash_shell$ ./addnames ulist john
Username already exists
hishamalip@ideapad-330:~/github/bash_shell$ cat ulist
hisham
john
hishamalip@ideapad-330:~/github/bash_shell$ 
```

--------------------------------------------------------------------------------------------------------------------

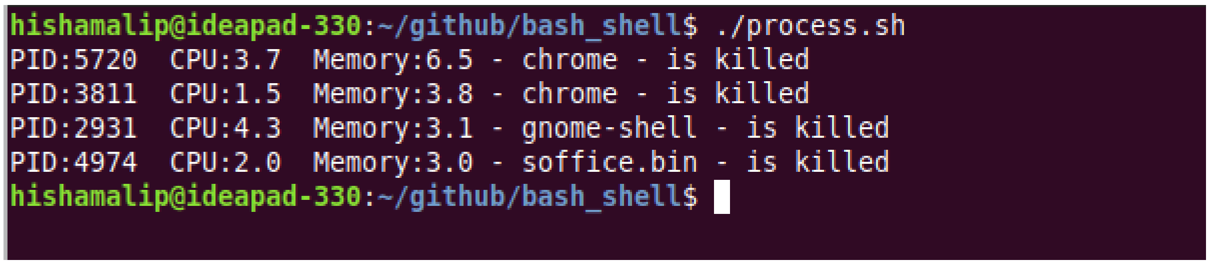
**Question 5 :**

Write a Shell script which starts on system boot up and kills every process which uses more than a specified amount of memory or CPU.

Program :

```
ps -e -o pmem=,pcpu=,pid=,comm= | sort -r -k 1 | while read memsize cpusize pid command ; do
            xcpu=1
            xmemory=1
            check="$(echo "$cpusize>$xcpu" | bc || echo "$memsize>$xmemory" | bc)"
            if [ $check -eq 1 ] ; then
                  echo -e  "PID:$pid  CPU:$cpusize  Memory:$memsize - $command-is killed"
                  kill $pid
            fi
      done
```

Output :



```
hishamalip@ideapad-330:~/github/bash_shell$ ./process.sh
PID:5720  CPU:3.7  Memory:6.5 - chrome - is killed
PID:3811  CPU:1.5  Memory:3.8 - chrome - is killed
PID:2931  CPU:4.3  Memory:3.1 - gnome-shell - is killed
PID:4974  CPU:2.0  Memory:3.0 - soffice.bin - is killed
hishamalip@ideapad-330:~/github/bash_shell$ 
```

-----------------------------------------------------------------------------------------------------------

## Result :

Practiced various shell scripting programs and output is verified.