```
#!/bin/bash
clear
echo "This is information provided by mysystem.sh. Program starts now."
echo "Hello, $USER"
echo
echo "Today's date is `date`, this is week `date +"%V"`."
echo
echo "These users are currently connected:"
w | cut -d " " -f 1 - | grep -v USER | sort -u
echo
echo "This is `uname -s` running on a `uname -m` processor."
echo
echo "This is the uptime information:"
uptime
echo
echo "That's all folks!"
```

```
#!/bin/bash
clear
printf "This is information provided by mysystem.sh.
Program starts now.\n"
printf "Hello, $USER.\n\n"
printf "Today's date is `date`, this is week `date +"%V"`.\n\n"
printf "These users are currently connected:\n"
w | cut -d " " -f 1 - | grep -v USER | sort -u
printf "\n"
printf "This is `uname -s` running on a `uname -m` processor.\n\n"
printf "This is the uptime information:\n"
uptime
printf "\n"
printf "That's all folks!\n"
```

```
#!/bin/bash -xv
# This script clears the terminal, displays a greeting and gives information
# about currently connected users. The two example variables are set and displayed.
clear
# clear terminal window
echo "The script starts now."
echo "Hi, $USER!"
echo
# dollar sign is used to get content of variable
echo "I will now fetch you a list of connected users:"
echo
w        # show who is logged on and what they are doing
echo
echo "I'm setting two variables now."
COLOUR="black"     #set a local shell variable
VALUE="9"   #set a local shell variable
```

```
echo "This is a string: $COLOUR"
echo "And this is a number: $VALUE"
echo
echo "I'm giving you back your prompt now."
echo
```

Read the following files (do not make any changes) and try to understand:

/etc/profile
/etc/bashrc
~/.bash_profile
~/.bash_login
~/.profile
~/.bashrc
~/.bash_logout

See the values of the following variables: -

CDPATH
HOME
IFS
MAIL
MAILPATH
OPTARG
OPTIND
PATH
PS1
PS2

and also these: -

auto_resume
BASH
BASH_ENV
BASH_VERSION
BASH_VERSINFO
COLUMNS
COMP_CWORD
COMP_LINE
COMP_POINT
COMP_WORDS
COMPREPLY
DIRSTACK
EUID
FCEDIT
FIGNORE
FUNCNAME
GLOBIGNORE
GROUPS
histchars

HISTCMD
HISTCONTROL
HISTFILE
HISTFILESIZE
HISTIGNORE
HISTSIZE
HOSTFILE
HOSTNAME
HOSTTYPE
IGNOREEOF
INPUTRC
LANG
LC_ALL
LC_COLLATE
LC_CTYPE
LC_MESSAGES
LC_NUMERIC
LINENO
LINES
MACHTYPE
MAILCHECK
OLDPWD
OPTERR
OSTYPE
PIPESTATUS
POSIXLY_CORRECT
PPID
PROMPT_COMMAND
PS3
PS4
PWD
RANDOM
REPLY
SECONDS
SHELLOPTS
SHLVL
TIMEFORMAT
TMOUT
UID

and finally, look at these special variables: -

$*
$@
$#
$?
$-
$$
$!

$0
$_

Accept the challenge, if you are brave, to code these programs and enjoy !.

```
#!/bin/bash
#Drawing a Special Pattern
MAX_NO=0
echo -n "Enter Number between (5 to 9) : "
read MAX_NO
if ! [ $MAX_NO -ge 5 -a $MAX_NO -le 9 ] ; then
   echo "WTF... I ask to enter number between 5 and 9, Try Again"
   exit 1
fi
clear
for (( i=1; i<=MAX_NO; i++ ))
 do
     for (( s=MAX_NO; s>=i; s-- ))
    do
       echo -n " "
    done
    for (( j=1; j<=i;  j++ ))
     do
      echo -n " ."
      done
     echo ""
 done
###### Second stage ######################
for (( i=MAX_NO; i>=1; i-- ))
do
    for (( s=i; s<=MAX_NO; s++ ))
    do
       echo -n " "
    done
    for (( j=1; j<=i;  j++ ))
    do
     echo -n " ."
    done
    echo ""
done
echo -e "\n\n\t\t\t Whenever you need help, Tecmint.com is always there"
```

```
#!/bin/bash
# Colourful pattern
clear
echo -e "33[1m Hello World"
# bold effect
echo -e "33[5m Blink"
# blink effect
echo -e "33[0m Hello World"
# back to normal
echo -e "33[31m Hello World"
# Red color
echo -e "33[32m Hello World"
# Green color
echo -e "33[33m Hello World"
# See remaining on screen
echo -e "33[34m Hello World"
```

```
echo -e "33[35m Hello World"
echo -e "33[36m Hello World"
echo -e -n "33[0m"
# back to normal
echo -e "33[41m Hello World"
echo -e "33[42m Hello World"
echo -e "33[43m Hello World"
echo -e "33[44m Hello World"
echo -e "33[45m Hello World"
echo -e "33[46m Hello World"
echo -e "33[0m Hello World"
```

```
#!/bin/bash

    date;
    echo "uptime:"
    uptime
    echo "Currently connected:"
    w
    echo "--------------------"
    echo "Last logins:"
    last -a |head -3
    echo "--------------------"
    echo "Disk and memory usage:"
    df -h | xargs | awk '{print "Free/total disk: " $11 " / " $9}'
    free -m | xargs | awk '{print "Free/total memory: " $17 " / " $8 " MB"}'
    echo "--------------------"
    start_log=`head -1 /var/log/messages |cut -c 1-12`
    oom=`grep -ci kill /var/log/messages`
    echo -n "OOM errors since $start_log :" $oom
    echo ""
    echo "--------------------"
    echo "Utilization and most expensive processes:"
    top -b |head -3
    echo
        top -b |head -10 |tail -4
    echo "--------------------"
    echo "Open TCP ports:"
    nmap -p- -T4 127.0.0.1
    echo "--------------------"
    echo "Current connections:"
    ss -s
    echo "--------------------"
    echo "processes:"
    ps auxf --width=200
    echo "--------------------"
    echo "vmstat:"
    vmstat 1 5
```

Pre-requisites:-

```
[root@midstage ~]# yum install pinentry-gui
[root@midstage ~]# apt-get install pinentry-gui
```

Program is as follows: -

```
#!/bin/bash
echo "Welcome, I am ready to encrypt a file/folder for you"
```

```
echo "currently I have a limitation, Place me to thh same folder, where a file to
be
encrypted is present"
echo "Enter the Exact File Name with extension"
read file;
gpg -c $file
echo "I have encrypted the file successfully..."
echo "Now I will be removing the original file"
rm -rf $file
```

```
# Check Disk Space and Sends an Email Alert

MAX=95
EMAIL=USER@domain.com
PART=sda1
USE=`df -h |grep $PART | awk '{ print $5 }' | cut -d'%' -f1`
if [ $USE -gt $MAX ]; then
  echo "Percent used: $USE" | mail -s "Running out of disk space" $EMAIL
fi
```