## Assignment No 01

**AIM:**
Assignment to learn and understand Shell Programming

**Problem Statement**
Write a program to handle students database with following options a) Create Database b) View Database c) Insert a record d) Delete a record e) modify a record f) Result of a particular student g) exit

## THEORY:

Shell is an utility program which runs over the kernel. It acts as a command interpreter for the kernel and is the interface between user and the kernel. Thus it can be compared with command processor of MS-DOS which interprets, manages and co-ordinates the execution of commands by user.

To begin with the, kernel displays shell at prompt after authorized login. The input from the user decode the command line data and reaches for program. If the program is found, the shell retrieves it and submit it to the kernel to execute. Finally the kernel delivers the output. If the command is not found, it signals to the kernel to display "command not found" at the terminal. The shell accepts the kernel rely and in both cases, displays the next prompt. The cycle continues until it is terminated by Ctrl + D or logout. Login message is again displayed.

The /bin and /user/bin are the directories where all commands are located.

Types of shell :

1) **sh or Bourne Shell**: the original shell still used on UNIX systems. This is the basic shell, a small program with few features. While this is not the standard shell, it is still available on every Linux system for compatibility with UNIX programs.
2) **bash or Bourne Again shell**: the standard GNU shell, intuitive and flexible. Probably most advisable for beginning users while being at the same time a powerful tool for the advanced and professional user. On Linux, bash is the standard shell for common users. Commands that work in sh, also work in bash. However, the reverse is not always the case.
3) **csh or C shell**: the syntax of this shell resembles that of the C programming language. Sometimes asked for by programmers.
4) **tcsh or Turbo C shell**: a superset of the common C shell, enhancing user friendliness and speed.
5) **ksh or the Korn shell**: sometimes appreciated by people with a UNIX background. A superset of the Bourne shell; with standard configuration a nightmare for beginning users.
6) **Restricted Shell (rsh) :** This is the restricted version of bash shell, It is used for guest login and in service installation where user must be restricted to work only in their own limited environments.

The file /etc/shells gives an overview of known shells on a Linux system. To switch from one shell to another, just enter the name of the new shell in the active terminal. The system finds the directory where the name occurs using the PATH settings, since a shells is an executable files (program), the current shell activates it and it gets executed. A new prompt is usually shown, because each shell has its typical appearance.
A set of commands that has to be performed repeatedly is grouped into a batch file in DOS. Shell scripts are also similar to DOS batch files. Use any editor like vi or gedit to write shell script. .sh extension is used to identify shell script file.

**$ vi sample.sh**
#
# My first shell script

#
clear
echo “Welcome to Shell Programming"

After writing shell script set execute permission for your script :
**$ chmod +x sample.sh**
After saving the above script, you can run the script :
**$ ./sample.sh**

clear is used to clear the screen and echo to print message or value of variables on screen.

if condition is used for decision making in shell script. test command or [ expr ] is used to see if an expression is true, and if it is true it return 0, otherwise returns nonzero for false.

Following script determine whether given argument number is positive.

```
$ vi sample.sh
if test $1 -gt 0
then
echo "$1 number is positive"
fi
```

The output is
**$ ./sample.sh 5**
5 number is positive

The various mathematical operators are used such as eq, ne, lt, le, gt and ge. The other operator used are : ! for Logical NOT, -a for AND, -o for OR. The above program can be modified as :

```
$ vi sample.sh
if [ $1 -gt 0 ]
then
echo "$1 number is positive"
else
        ------------------------
fi
```

**Multi-level if else loop is as follows:**

```
$ vi sample.sh
if [ $1 -gt 0 ]; then
        echo "$1 is positive"
elif [ $1 -lt 0 ] then
        echo "$1 is negative"
elif [ $1 -eq 0 ] then
        echo "$1 is zero"
else
        echo "Opps! $1 is not number, give number"
fi
```

For loop can be used as follows:

```
$ vi sample.sh
for i in 1 2 3 4 5
```

```
do
        echo "Welcome $i times"
done
```

**$ vi sample.sh**

```
for ((  i = 0 ;  i <= 5;  i++  ))
do
        echo "Welcome $i times"
done
```

The case construct is used for the execution of the shell script based on our choice. Some examples is as follows :

**$ vi sample.sh**

```
echo "Menu"
echo " 1. Your Current working directory"
echo "2. Today's date"
echo "3. List of users logged in"
echo "your Choice"
read choice
case $choice in
    1)      pwd;;
    2)      date;;
    3)      who;;
    *)  echo "invalid Choice"
esac
```

**$ vi sample.sh**

```
echo "Menu"
echo "1. Displays a long listing of file"
echo "2. Displays long listing of files including hidden files"
echo "3. Delete a file"
echo "Your Choice : "
read choice
case $choice in
        1)  ls –l ;;
        2)  ls –al;;
        3)  echo "Enter the name of file to be deleted"
            read file
            rm $file
            echo "File is deleted"
        *) echo "Invalid Choice"
esac
```

**$ vi sample.sh (it uses break and while loop)**

```
while true
do
echo "Menu"
echo "1. Displays a long listing of file"
echo "2. Displays long listing of files including hidden files"
echo "3. Delete a file"
echo "Your Choice :  (Press w to quit)"
read choice
case $choice in
        1)      ls –l ;;
```

```
        2)      ls –al;;
        3)      echo "Enter the name of file to be deleted"
                read file
                rm $file
                echo "File is deleted";;
        w)      break;;
        *)      echo "Invalid Choice"
esac
done
```

**$ vi sample.sh (it uses continue and while loop)**

```
ans="Y"
while [ $ans = y –o $ans = Y ]
do
echo "Menu"
echo "1. Displays a long listing of file"
echo "2. Displays long listing of files including hidden files"
echo "3. Delete a file"
echo "Your Choice :  "
read choice
case $choice in
        1)      ls –l ;;
        2)      ls –al;;
        3)      echo "Enter the name of file to be deleted"
                read file
                rm $file
                echo "File is deleted";;
        *)      echo "Invalid Choice"
esac
echo "Do You want to continue (y or Y)"
read ans
if [ $ans = y –o $ans = Y ]
then
        continue
else
        exit
fi
done
```

**CONCLUSION –**

Thus we have studied and implemented shell scripting language for various programs.

---

FAQ's – Shell Programming and introduction to operating system

1. Differentiate between
    a. DOS and Windows
    b. UNIX and Linux
    c. Windows and Linux
2. Specify at least five commands with description in UNIX?
3. Write a short note on
    a. Wild card patterns
    b. Redirection
    c. Pipes and filters
    d. File access permission.

4. Describe in detail what is shell with block diagram of UNIX?
5. What is the use of shell programming?

---