EXPERIMENT NO. 3

Study of UNIX EDITOR- VI and Essential Shell Programming.

Aim: To study about Unix text editor (Vi)
Theory
The vi editor (short for visual editor) is a screen editor which is available on almost all Unix systems. vi has no menus but instead uses combinations of keystrokes in order to accomplish commands.
1. Starting the vi Editor:
There are following way you can start using vi editor:

Command	Description
vi filename	Creates a new file if it already does not exist, otherwise opens an existing file.
vi -R	filename Opens an existing file in read only mode
view filename	Opens an existing file in read only mode.

2. Operation Modes:
While working with vi editor, it comes across the following two modes:
i) Command mode: This mode enables you to perform administrative tasks such as saving files, executing commands, moving the cursor, cutting (yanking) and pasting lines or words, and finding and replacing. In this mode, whatever you type is interpreted as a command.
ii) Insert mode: This mode enables you to insert text into the file. Everything that's typed in This mode is interpreted as input and finally it is put in the file .

The vi always starts in command mode. To enter text, you must be in insert mode. To come in insert mode you simply type i. To get out of insert mode, press the Esc key, which will put you back into command mode.

3. Closing and Saving Files
When editing a file in vi, it is actually editing a copy of the file rather than the original.
The following describes the methods to be used when closing a file, quitting vi, or both.
i) Quitting and Saving a File
The command ZZ (notice that it is in uppercase) will allow you to quit vi and save the edits made to a file. You will then return to a Unix prompt. Note that you can also use the following commands:

:w	to save your file but not quit vi (this is good to do periodically in case of machine crash!).
:q	to quit if you haven't made any edits.
:wq	to quit and save edits (basically the same as ZZ).

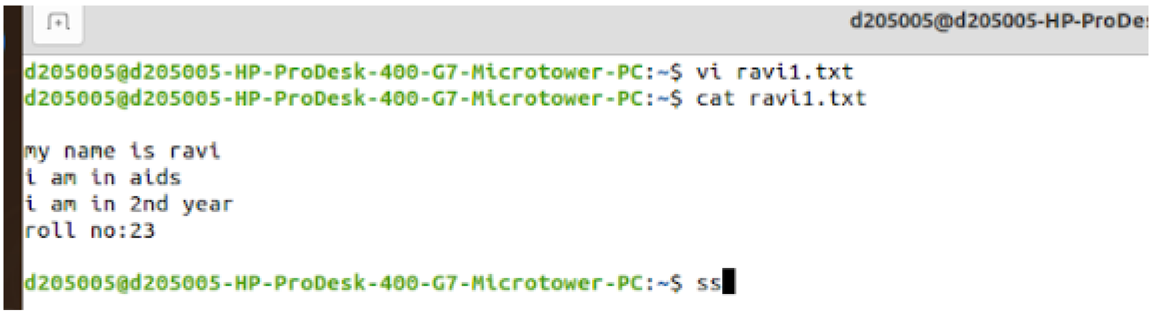
ii) Quitting without Saving Edits
Sometimes, when you create a mess (when you first start using vi this is easy to do!) you may wish to erase all edits made to the file and either start over or quit. To do this, you can choose from the following two commands:

:e!	reads the original file back in so that you can start over.
:q!	wipes out all edits and allows you to exit from vi.

4. Steps for Inserting Text With "vi"
The following shows how to open a file, enter text, and then save the changes to the file.
Step 1: At the command line, type "vi doc1".
Step 2: Type the letter "i". This puts you into insert mode.
Step 3: Enter the following text into your document:
This is my first vi document.
This is the second line of the document.
Step :4 Hit the "<Esc>" key. This takes you back into command mode.
Step 5: Now enter ":wq" to save the changes to the file. This stands for "write and quit". It returns to the UNIX command line.

5. Cursor Movement
It must be in command mode if we wish to move the cursor to another position in the file.
If we have just finished typing text, we are still in insert mode and will need to press ESC to return to the command mode.
i) Moving One Character at a Time
In case of direction keys to move up, down, left and right in the file, the following keys move in the following directions:

l	right one space
h	left one space
k	up one space
j	down one space



```
d205005@d205005-HP-ProDesk-400-G7-Microtower-PC:~$ vi ravi1.txt
d205005@d205005-HP-ProDesk-400-G7-Microtower-PC:~$ cat ravi1.txt

my name is ravi
i am in aids
i am in 2nd year
roll no:23

d205005@d205005-HP-ProDesk-400-G7-Microtower-PC:~$ ss
```

What is SHELL?
A Shell provides you with an interface to the UNIX system. It gathers input from you and executes programs based on that input.

Shell Types

In Unix, there are two major types of shells −

- Bourne shell − If you are using a Bourne-type shell, the $ character is the default prompt.
- C shell − If you are using a C-type shell, the % character is the default prompt.

The Bourne Shell has the following subcategories −

- Bourne shell (sh)
- Korn shell (ksh)
- Bourne Again shell (bash)
- POSIX shell (sh)

The different C-type shells follow −

- C shell (csh)
- TENEX/TOPS C shell (tcsh)

SHELL Scripts:
When a group of commands have to be executed regularly, they should be stored in a file, and the file itself executed as a shell script or shell program. Use the .sh extension for shell scripts.
Use your **vi** editor to create the shell script, script.sh. The script runs three echo commands and shows the use of variable evaluation and command substitution. It also prints the calendar of the current month.

```
#script.sh: Sample shell script
echo “Today’s date: `date`”
echo “This month’s calendar:`cal`”
echo “My shell: $SHELL”
```

# comment character that can be placed anywhere in a line.
To run the script, make it executable first and then invoke the script name:

```
$chmod +x script.sh
$sh script.sh
```

read : MAKING SCRIPT INTERACTIVE
The **read** statement is the shell’s internal tool for taking input from the user, i.e., making scripts interactive. It is used with one or more variables. Input supplied through the standard input is read into these variables. When you use a statement like

```
read name
```

the script pauses at that point to take input from the keyboard. Whatever you enter is stored in the variable name. Since this is a form of assignment , no $ is used before name

Example: Taking Input from user

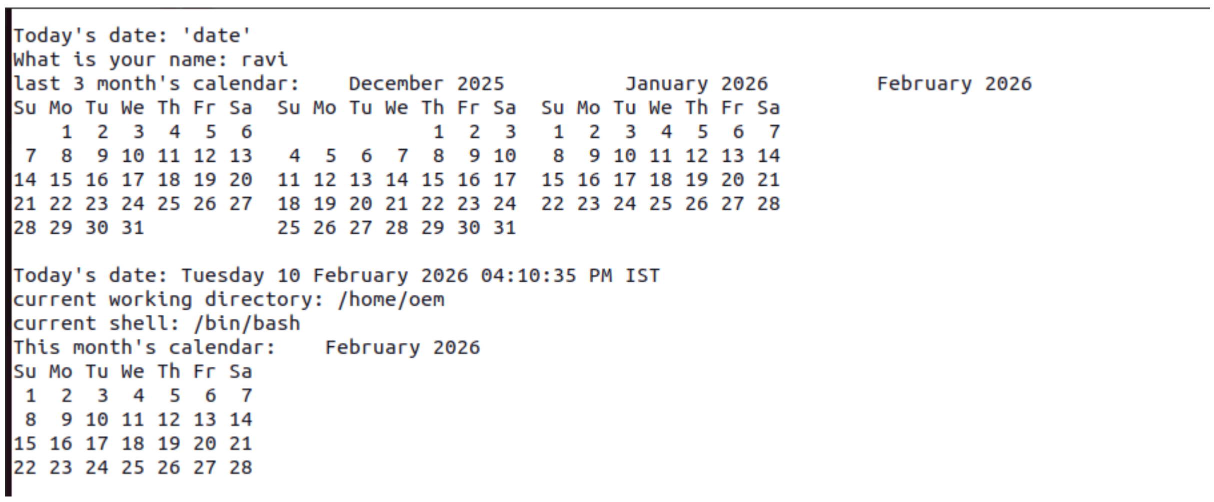
```
echo “Enter the pattern to be searched : \c”
read pname
echo “Enter the file to be used : \c”
read flame
```

Example shows how to add two numbers −

```
val=`expr 2 + 2`
echo "Total value : $val"
```

Exercise:
Write a shell script to DISPLAY HELLO WORLD.
Write a shell script to display today’s date, last 3 months calendar, current working directory and current shell.

```
echo "Hello world"
echo "Today's date : `date`"
echo " Last 3 months calendar : `cal -B2`"
echo "current working directory : `pwd`"
echo "current shell : $SHELL"
```



```
Today's date: 'date'
What is your name: ravi
last 3 month's calendar:    December 2025          January 2026         February 2026
Su Mo Tu We Th Fr Sa  Su Mo Tu We Th Fr Sa  Su Mo Tu We Th Fr Sa
    1  2  3  4  5  6               1  2  3   1  2  3  4  5  6  7
 7  8  9 10 11 12 13   4  5  6  7  8  9 10   8  9 10 11 12 13 14
14 15 16 17 18 19 20  11 12 13 14 15 16 17  15 16 17 18 19 20 21
21 22 23 24 25 26 27  18 19 20 21 22 23 24  22 23 24 25 26 27 28
28 29 30 31           25 26 27 28 29 30 31

Today's date: Tuesday 10 February 2026 04:10:35 PM IST
current working directory: /home/oem
current shell: /bin/bash
This month's calendar:    February 2026
Su Mo Tu We Th Fr Sa
 1  2  3  4  5  6  7
 8  9 10 11 12 13 14
15 16 17 18 19 20 21
22 23 24 25 26 27 28
```

Post Lab
https://www.tutorialspoint.com/unix/unix-basic-operators.htm

- Unix Shell Operators −
  - Arithmetic Operators
  - Relational Operators
  - Boolean Operators
  - String Operators
  - File Test Operators

- Unix Shell conditional statements
- Unix Shell loops

1. **Write a shell script that calculates the sum of integers from 1 to N using a loop.**
2. **Write a shell script to FIND THE GIVEN NUMBER IS EVEN OR ODD**
3. **Write a shell script to display even numbers within range 1 to 100**

Conclusion:-