vi Editor in UNIX

saloni1297
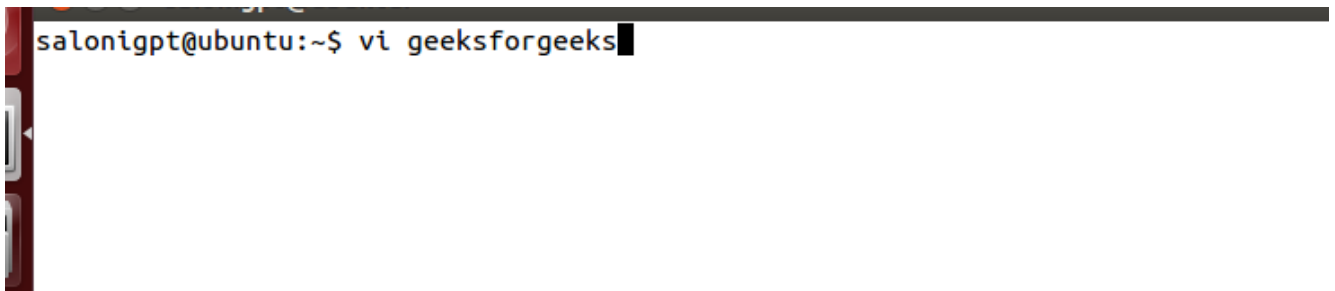
Read   Discuss

The default editor that comes with the UNIX operating system is called vi (visual editor). Using vi editor, we can edit an existing file or create a new file from scratch. we can also use this editor to just read a text file.
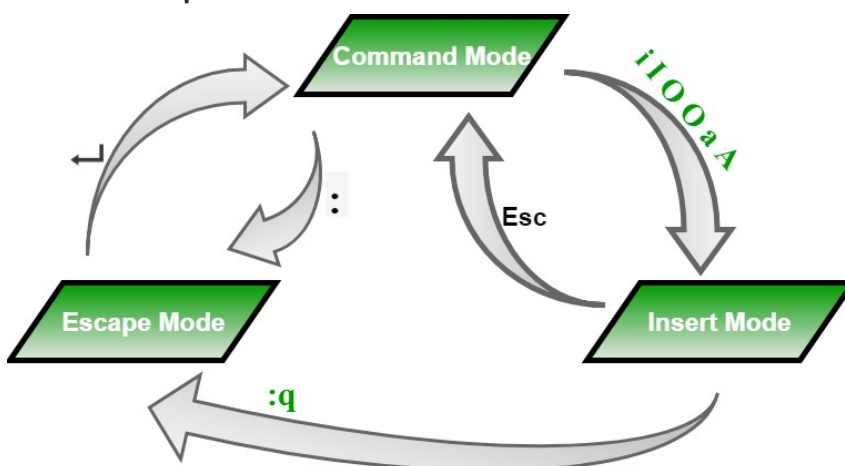
Syntax:

```
vi filename
```

Input:

```
salonigpt@ubuntu:~$ vi geeksforgeeks
```

Output:

Modes of Operation in vi editor: There are three modes of operation in vi:

- **Command Mode:** When vi starts up, it is in Command Mode. This mode is where vi interprets any characters we type as commands and thus does not display them in the window. This mode allows us to move through a file, and to delete, copy, or paste a piece of text. To enter into Command Mode from any other mode, it requires pressing the **[Esc]** key. If we press [Esc] when we are already in Command Mode, then vi will beep or flash the screen.
- **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.

- **Last Line Mode(Escape Mode):** Line Mode is invoked by typing a colon [:], while vi is in Command Mode. The cursor will jump to the last line of the screen and vi will wait for a command. This mode enables you to perform tasks such as saving files, executing commands.

Starting the vi Editor

There are following way you can start using vi editor :

Commands and their Description

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

Moving within a File(Navigation): To move around within a file without affecting text must be in command mode (press Esc twice). Here are some of the commands can be used to move around one character at a time.

Commands and their Description

- *k : Moves the cursor up one line.*
- *j : Moves the cursor down one line.*
- *h : Moves the cursor to the left one character position.*
- *l : Moves the cursor to the right one character position.*
- *0 or | : Positions cursor at beginning of line.*
- *$: Positions cursor at end of line.*
- *W : Positions cursor to the next word.*
- *B : Positions cursor to previous word.*
- *(: Positions cursor to beginning of current sentence.*
- *) : Positions cursor to beginning of next sentence.*
- *H : Move to top of screen.*
- *nH : Moves to nth line from the top of the screen.*
- *M : Move to middle of screen.*
- *L : Move to bottom of screen.*

- *colon along with x* : *Colon followed by a number would position the cursor on line number represented by x.*

Control Commands(Scrolling): There are following useful commands which can used along with **Control Key**:

Commands and their Description:

- *CTRL+d* : *Move forward 1/2 screen.*
- *CTRL+f* : *Move forward one full screen.*
- *CTRL+u* : *Move backward 1/2 screen.*
- *CTRL+b* : *Move backward one full screen.*
- *CTRL+e* : *Moves screen up one line.*
- *CTRL+y* : *Moves screen down one line.*
- *CTRL+u* : *Moves screen up 1/2 page.*
- *CTRL+d* : *Moves screen down 1/2 page.*
- *CTRL+b* : *Moves screen up one page.*
- *CTRL+f* : *Moves screen down one page.*
- *CTRL+l* : *Redraws screen.*

Editing and inserting in Files(Entering and Replacing Text): To edit the file, we need to be in the insert mode. There are many ways to enter insert mode from the command mode.

- *i* : *Inserts text before current cursor location.*

- *A* : Inserts text at end of current line.
- *o* : Creates a new line for text entry below cursor location.
- *O* : Creates a new line for text entry above cursor location.
- *r* : Replace single character under the cursor with the next character typed.
- *R* : Replaces text from the cursor to right.
- *s* : Replaces single character under the cursor with any number of characters.
- *S* :Replaces entire line.

Deleting Characters: Here is the list of important commands which can be used to delete characters and lines in an opened file.

- *X Uppercase: Deletes the character before the cursor location.*
- *x Lowercase : Deletes the character at the cursor location.*
- *Dw : Deletes from the current cursor location to the next word.*
- *d^ : Deletes from current cursor position to the beginning of the line.*
- *d$: Deletes from current cursor position to the end of the line.*
- *Dd : Deletes the line the cursor is on.*

Copy and Past Commands: Copy lines or words from one place and paste them on another place by using the following commands.

- *Yy : Copies the current line.*
- *9yy : Yank current line and 9 lines below.*
- *p : Puts the copied text after the cursor.*
- *P : Puts the yanked text before the cursor.*

Save and Exit Commands of the ex Mode : Need to press **[Esc]** key followed by the colon (:) before typing the following commands:

- *q : Quit*
- *q! : Quit without saving changes i.e. discard changes.*
- *r fileName : Read data from file called fileName.*
- *wq : Write and quit (save and exit).*
- *w fileName : Write to file called fileName (save as).*
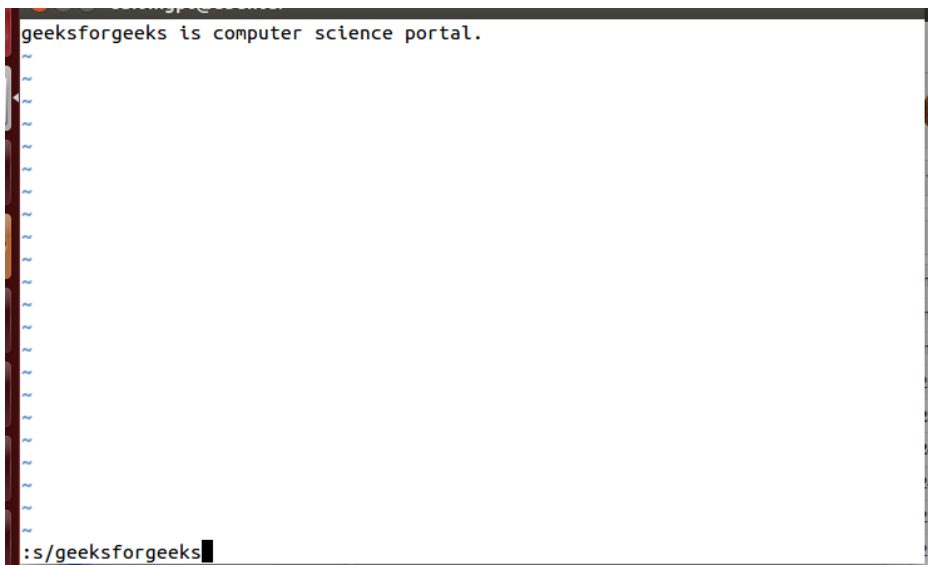- *w! fileName : Overwrite to file called fileName (save as forcefully).*

Searching and Replacing in (ex Mode): vi also has powerful search and replace capabilities. The formal syntax for searching is:

```
:s/string
```

For example, suppose we want to search some text for the string **"geeksforgeeks"** Type the following and press ENTER:
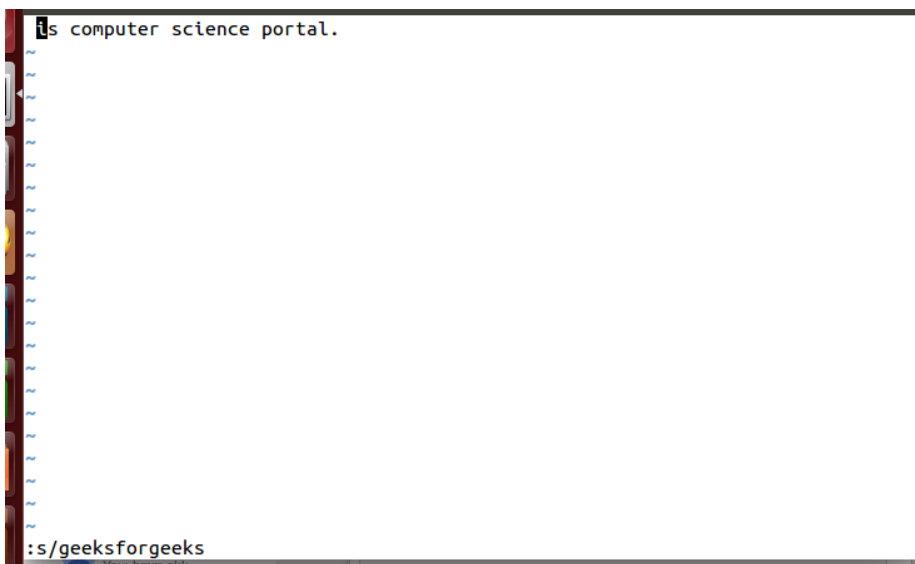
```
:s/geeksforgeeks
```

Input:

Output: finding the first match for "geeksforgeeks" in text will then be highlighted.

The syntax for replacing one string with another string in the current line is:

```
:s/pattern/replace/
```

Here "pattern" represents the old string and "replace" represents the new string. For example, to replace each occurrence of the word "geeks" in a line with "geeksforgeeks" type:

    :s/geeksforgeeks/gfg/

Input:

Output:

The syntax for replacing every occurrence of a string in the entire text is similar. The only difference is the addition of a "%" in front of the "s":

    :%s/pattern/replace/

Thus repeating the previous example for the entire text instead of just for a single line.

```
:%s/gfg/geeksforgeeks/
```

Block delete commands in (x mode): need to press ESC and then commands will be followed by colon(:).

1. **:1d** delete the line 1.
2. **:1,5d** deletes the lines from 1 to 5.
3. **:10,$d** deletes lines from 10th line to the last of the file. ($ means last line of file).
4. **:.,$d** deletes lines from present line to that last line. (. means the present line).
5. **:.-3,.d** deletes the lines from present line and above 2 lines (deletes 3 lines including the cursor line).
6. **:.,.+4d** deletes the lines from present cursor line followed 3 lines (total 3 lines).
7. **:16** deletes the 16 line of the file.

Block copy command in (x mode): need to press ESC and then commands will be followed by colon(:).

1. **:1,5 co 10** copies the lines from 1 to 5 after 10th line.
2. **:1,$ co $** copies the lines from 1 to last line after last line.
3. **:.,.+5 co 8** copies lines from present to 5 lines after 8th line.
4. **:-3,. co 10** copies the lines from present cursor line above 3 lines after 10th line.

Block moving commands in (x mode): need to press ESC and then commands will be followed by colon(:).

1. **:1,5 mo 9** moves line from 1 to 5 after 9th line.
2. **:1,$ mo $** moves lines from 1 to $ after last line.
3. **:.,.+5 mo 10** moves line from present line ans next 5 lines after 10th line onwards.
4. **:-3,. mo 10** moves present line and above 3 lines after 10th line.

Reference: http://www.linfo.org/vi/

Last Updated : 18 Nov, 2022   24

1.   grep command in Unix/Linux

2.   Sed Command in Linux/Unix with examples

3.   SORT command in Linux/Unix with examples

4.   Soft and Hard links in Unix/Linux

5.   Commands in Unix when things go wrong

6.   AWK command in Unix/Linux with examples

7.   tr command in Unix/Linux with examples

8.   Wget command in Linux/Unix

9.   Absolute and Relative Pathnames in UNIX

10.   Process Control Commands in Unix/Linux

Previous   Next

Article Contributed By :

saloni1297
@saloni1297

Vote for difficulty

Current difficulty : Medium

| Easy | Normal | Medium | Hard | Expert |

Improved By :   AbhinavKinagi,   lucifer2411

Article Tags :   linux-command,   Linux-text-processing-commands,   Linux-Unix

GeeksforGeeks

A-143, 9th Floor, Sovereign Corporate Tower,
Sector-136, Noida, Uttar Pradesh - 201305

feedback@geeksforgeeks.org

Company

About Us

Careers

In Media

Contact Us

Terms and Conditions

Privacy Policy

Copyright Policy

Third-Party Copyright Notices

Advertise with us

Languages

Python

Java

C++

GoLang

SQL

R Language

Android Tutorial

Data Structures

Array

String

Linked List

Stack

Queue

Tree

Graph

Algorithms

Sorting

Searching

Greedy

Dynamic Programming

Pattern Searching

Recursion

Backtracking

Web Development

HTML

CSS

JavaScript

Bootstrap

ReactJS

AngularJS

NodeJS

Write & Earn

Write an Article

Improve an Article

Pick Topics to Write

Write Interview Experience

Internships

Video Internship

GATE CS Notes

Operating Systems

Computer Network

Database Management System

Software Engineering

Digital Logic Design

Engineering Maths

Data Science With Python

Data Science For Beginner

Machine Learning Tutorial

Maths For Machine Learning

Pandas Tutorial

NumPy Tutorial

NLP Tutorial

Interview Corner

Company Preparation

Preparation for SDE

Company Interview Corner

Experienced Interview

Internship Interview

Competitive Programming

Aptitude

Python

Python Tutorial

Python Programming Examples

Django Tutorial

Python Projects

Python Tkinter

OpenCV Python Tutorial

GfG School

CBSE Notes for Class 8

CBSE Notes for Class 9

CBSE Notes for Class 10

CBSE Notes for Class 11

CBSE Notes for Class 12

English Grammar

UPSC/SSC/BANKING

SSC CGL Syllabus

SBI PO Syllabus

IBPS PO Syllabus

UPSC Ethics Notes

UPSC Economics Notes

UPSC History Notes