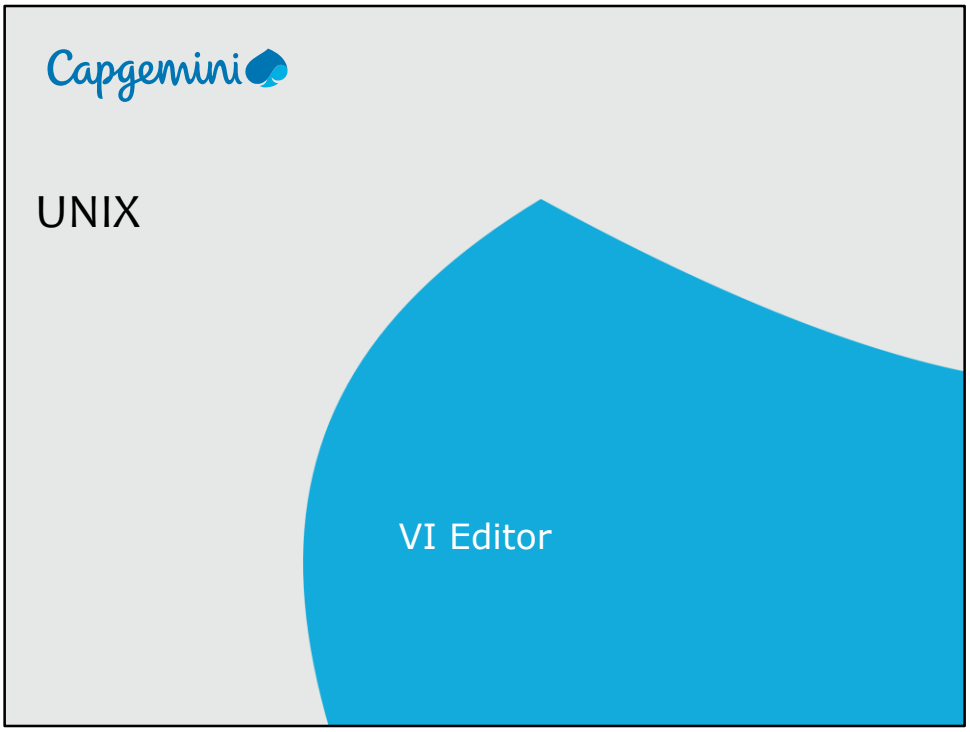
Instructor Notes:



Instructor Notes:

Lesson Objectives

- Different modes of vi editor
  - Input
  - Command
  - Esc mode
- Input mode commands
- Vi editor – Save & Quit
- Navigation commands
- Paging functions
- Search and repeat commands

Instructor Notes:

Lesson Objectives

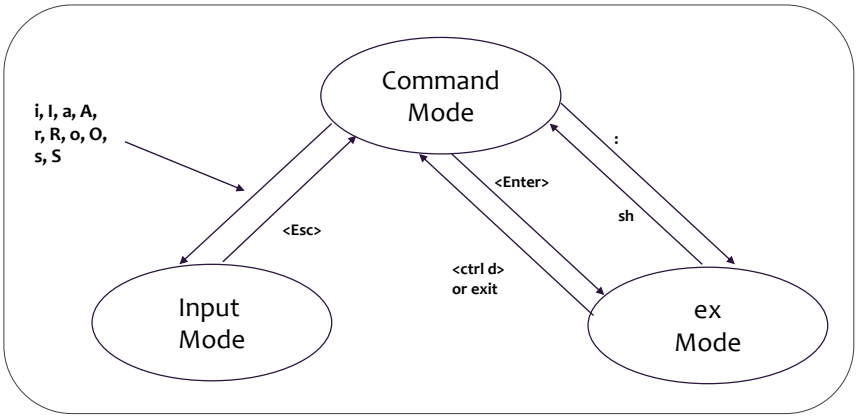
- Vi editor – Other Features
- SED – Introduction to SED
- SED Commands

Instructor Notes:

5.1: Modes of Vi Editor

Introduction

- Three Modes of Vi Editor are:



What is the vi Editor?

Line editors, full screen editors and stream editors are all available on the Unix system. The editor "ed" was developed by Ken Thompson, and was the original editor that accompanied the Unix system. The line editor "ex" was created by William Joy, on the basis of "ed". This chapter discusses the "vi" editor, which is a full screen editor, widely acknowledged as one of the most powerful editors available in any environment.
The vi editor is also created by William Joy, and is in fact simply the visual mode of the line editor "ex". It offers innumerable functions, but the terseness of its commands is considered to be a major handicap.

Modes of vi editor:

The vi editor works in 3 modes: Input, Command and ex mode. The relation between the three modes is depicted in the figure in the slide above.
Command Mode, Input Mode, ex Mode

Instructor Notes:

5.2: Input Mode Commands

Contents

Command	Function
i	Insert text to left of cursor
I	Inserts text at beginning of line
a	Appends text to right of cursor
A	Appends text at the end of line
o	Opens line below
O	Opens line above

What is the Input Mode?

The Input mode is used to insert, append, replace or change text. A summary of input mode commands are given below:

Command	Function
• i	Inserts text to left of cursor
• I	Inserts text at beginning of line
• a	Appends text to right of cursor
• A	Appends text at end of line
• o	Opens line below
• O	Opens line above
• rch	Replaces single character at cursor with character ch
• R	Replaces text from cursor to right
• s	Replaces single character at cursor with any number of characters
•S	Replaces entire line

Instructor Notes:

Contents (contd..)

Command	Function
r	Replaces single character under cursor with character (no<Esc>)
R	Replace text from cursor to right

Add the notes here.

Instructor Notes:

5.3: Vi Editor – Save & Quit

Description

- From input mode to command mode press <Esc>
- From command mode:

To Save	: w
To Quit	: q
To Quit without saving	: q!
To save & quit	: wq
or	: x

Saving and Quitting: The Last Line Mode

vi uses the ZZ command to save and quit editor. The 'ex' mode, also referred to as last line mode, can also be used.
To switch from command mode to ex mode, a colon (:) is pressed, which appears as ex prompt in the bottom line. Any ex command can be entered at this prompt. Following commands can be used for saving and quitting from the ex mode:

Command	Function
• w	Write buffer into disk and remain in editing mode
• x	Save and quit the editor
• wq	Write and quit editor
• q	Quit editor

The Repeat Factor

A number can be prefixed to any command: most commands will interpret the instruction to repeat the command that many times. Hence this number is called as the repeat factor. For example, to insert a series of 30 asterisks in a line, 30i* can be used.
The repeat factor can be used with input as well as command mode.

Instructor Notes:

5.4: Navigation Commands

Overview

Command	Function
h	Moves cursor left
j	Moves cursor down
k	Moves cursor up
l	Moves cursor right
^	Moves cursor to beginning of first
$	Moves cursor to end of line
b	Moves cursor backwards to beginning of word
e	Moves cursor forward to end of word
w	Moves cursor forward to beginning of word

Navigation (Cursor Movement commands):

Command	Function
• h (or backspace)	Move cursor left
• j	Move cursor down
• k	Move cursor up
• l (or spacebar)	Move cursor right
• ^	Move cursor to beginning of first word of line (no repeat factor)
• 0 or |	Move cursor to beginning of line (no repeat factor with 0)
• $	Move cursor to end of line
• b	Move cursor back to beginning of word
• e	Move cursor forward to end of word
• w	Move cursor forward to beginning of word

Instructor Notes:

5.5: Paging Functions

Details

Command	Function
<Control-f>	Full page forward
<Control-b>	Full page backward
<Control-d>	Half page forward
<Control-u>	Half page backward

Paging Functions:

Command	Function
• <Ctrl-f>	Full Page forward
• <Ctrl-b>	Full Page backward
• <Ctrl-d>	Half Page forward
• <Ctrl-u>	Half Page backward
• <Ctrl-l>	Redraw page screen (no repeat factor)

Instructor Notes:

5.6: Search and Repeat Commands

Details

Commands	Functions
/pat	Searches forward for pat
?pat	Searches backward for pattern pat
n	Repeats search in the same direction along which the previous search was made (no repeat factor)
N	Repeats search in a direction opposite to that which the previous search was made (no repeat factor)

Search and repeat commands:

Command	Function
• */pat*	Searches forward for pattern *pat*
• ?*pat*	Searches backward for pattern *pat*
• n	Repeats search in same direction as previous search (no repeat factor)
• N	Repeats search in opposite direction as previous search (no repeat factor)
• f*ch*	Moves cursor forward to first occurrence of character ch in current line

Instructor Notes:

5.7: Vi Editor – Other Features

Using set command

- **Set command is used to customize the behavior of the VI editor**
- **Some of the useful commands**

Sr no.	Command	Description
1.	:set autoindent or :set ai	To set autoindent on
2	:set number or :set nu	To Displays lines with line numbers on the left side
3	:set smd or :set showmode	To show the actual mode of the editor that you are in at the bottom line.
4.	:set wm=x or :set wrapmargin=x	To automatically wrap the word on next line, x will be any nonzero value. (:set wm=2 sets the wrap margin to 2 characters)

To edit the behavior of vi editor. There are many options which can be used with :set command

To get the list of all options in set command use
:set all
Some more commands

Sr no.	Command	Description
1.	:set noautoindent or :set noai	To unset autoindention
2.	:set nomesg	Turn off messages, so that nobody can bother you while using the editor.
3.	:set warn	To warns you if you have modified the file, but haven't saved it yet.
4.	:set tabstop=x or :set ts=x	To set the tabstop to x spaces (:set tabstop=8 the tab key will display 8 spaces)
5.	:set ignorecase or :set ic	To set ignore case by default while searching
6.	:set noignorecase or :set noic	To unset ignore case option
7.	:set linelimit=1048560	To set the maximum file size to edit
8.	:set list	To display hidden character like tabs or end of the line
9.	:set nolist	To display hide character like tabs or end of the line

Instructor Notes:

5.7: Vi Editor – Other Features

Details

- Joining line:
  - J - to join current line with next line
  - 4J - to join 4 lines from current line
- Undo last Instruction - u
- Reverse all changes made to current line – U
- Using set command

iGate Sensitive

Operators

vi uses a number of operators which can be used along with commands to perform complex editing functions. The most commonly used operators are:

d – delete
c – change
yy – yank (copy)
! – filter to act on text

Operators can work only when combined with a command or itself. The operators also take a repeat factor.

Some samples of using operators:

Command	Function
d$ or D	Deletes from cursor to end of line
5dd	Deletes five lines
d/*endif*	Deletes from cursor up to the first occurrence of the string *endif* in the forward direction
d30G	Deletes from cursor up to line number 30
df.	Deletes from cursor to first occurrence of a dot
c0	Changes from cursor to beginning of line
c$ or C	Changes from cursor to end of line
3cw or c3w	Changes three words

Instructor Notes:

5.8: SED – Introduction to SED

- SED("Stream EDitor") is a non-interactive stream oriented editor for filtering and transforming text.
- It reads input line by line, applying the operation which has been specified via the command line (or a sed script), and then outputs the line in a terminal or file.
- When to use SED?
  - To automate editing actions to be performed on one or more files.
  - To simplify the task of performing the same edits on multiple files.
  - To write conversion programs.

Instructor Notes:

5.9: SED Commands

Invoking SED using Command Line

- Syntax of SED Command

  sed *options sed-script* filename

  - sed-script -> sed can use regular expressions for manipulating text on the input file.
  - Options:
    - -n Suppress the default output.
    - -e Script is an edit command for sed . Used to specify multiple instructions by preceding with –e.

Instructor Notes:

5.9: SED Commands

Invoking SED using script file

- Create a script file with long editing instructions to perform task on an input file.
- The sed command will then be used as:

  sed –f scriptfile file

  For Example,

  sed –f sedsrc text
  - *sedsrc* – script file contains editing instructions.
  - *text* – input file consists of data.

The sed command will then be used as:

sed –f scriptfile file

All the editing command that we need to execute are placed in a file, as shown:

```
$ cat sedsrc
s/ WB/, West Bengal/
s/ BH/, Bihar/
s/ MH/, Maharashtra/
```

The following command reads all of the substitution commands in the sedsrc and applies them to each line in the input file "text":

$ sed –f sedsrc text

```
Sidd B-1/250 Kalyani, West Bengal
Tito A-3/11 Thane, Maharashtra
Rayn D-17 LakeTown, West Bengal
Miter C/268 G.B.Road, Bihar
```

The above command will display the output in the Terminal.

The following command used for Redirecting the output to a file:

$ sed –f sedsrc text > newtext

Instructor Notes:

5.9: SED Commands

Substitute Command

/s Command

- The substitute command changes all occurrences of the regular expression into a new value

Syntax:

sed 's/old/new' file

For Example:

sed 's/Hi/Hello' data

would substitute the occurrence of the word hi to hello in "data" file.

Let there be a text file called "text". The content of the file is as shown:
Sidd B-1/250 Kalyani WB
Tito A-3/11 Thane MH
Rayn D-17 LakeTown WB
Miter C/268 G.B.Road BH

The substitution command in sed:
$ sed 's/WB/WestBengal/' text

Two lines are affected by the instruction but in the above example all lines will be displayed. Enclosing the instruction in single quotes is not mandatory but its required if the substitution command contains spaces:
$ sed 's/ WB/, West Bengal/' text

***g option*:** Used to make the command replace in all the instance of the word instead of first occurrence of the word in each input line.
$ sed 's/rat/cat/g' temp
cat cat
For example if we want to change 'rat' to 'cat' in lines that contain the word 'dog' we say:
$ sed '/dog/s/rat/cat/g' temp
$ sed 's/rat/cat/4' # replaces only 4th instance in a line

Instructor Notes:

5.9: SED Commands

Multiple Instructions in SED Command

There are three ways to specify multiple instructions on the command line:

- Separate instructions with semicolon
  - sed 's/ WB/, West Bengal/; s/ BH/, Bihar/' text
- Precede each instruction by –e
  - sed –e 's/ WB/, West Bengal/' -e 's/ BH/, Bihar/' text
- Use the multiline entry capability
  - sed `
    s/ WB/, West Bengal/
    s/ BH/, Bihar/' text

There are three ways to specify multiple instructions on the command line:
1. Separate instructions with semicolon
sed 's/ WB/, West Bengal/; s/ BH/, Bihar/' text

2. Precede each instruction by –e
sed –e 's/ WB/, West Bengal/' -e 's/ BH/, Bihar/' text

3. Use the multiline entry capability
sed '
s/ WB/, West Bengal/
s/ BH/, Bihar/' text

It is very easy to make mistake in the instruction or omit a required element.

$ sed 's/ WB/, West Bengal' text
sed: command garbled: s/ WB/, West Bengal

Notice the error message. Sed usually display any line that it cannot execute, but it does not tell what is wrong with the command. Here a slash at the end is missing.

Instructor Notes:

5.9: SED Commands

Other options

-n option
- Suppresses the display of all input lines with print command 'p'
- For example
  - $ *sed –n 's/WB/WestBengal/p' text* - prints only the affected lines

d command
- Used to delete all lines and also to delete specific lines by either using regular expression or line number.
  - **For Example:** $ sed d temp # deletes all lines

-i option
- Used to substitute for the current given file. i.e the original file is changed.

$ sed = temp # number each line of a file.

The –n option suppresses the automatic output. When using this option each line needed to produce output must contain the print command, **p.**
$ sed –n 's/WB/WestBengal/p' text
Sidd B-1/250 Kalyani WestBengal
Rayn D-17 LakeTown WestBengal

Here only the lines that were affected were printed.
For printing only line 2 and 3, the command used is:
$ sed –n 2,3p text
If –n option is not present all the lines will get printed and line from 2 to 3 will get printed twice.
d used to delete all lines and also to delete specific lines by either using regular expression or line number.
$ sed d temp # deletes all lines
$ sed '$d' temp #delete last line.
$ sed '1d' temp #delete first line.
$ sed '/^$/d' temp #delete all blank lines.

# number each line of a file (simple left alignment).
$ sed = temp

Instructor Notes:

5.9: SED Commands

Other options

-i option

- Used to edit content and save for the given file. In this case original file is changed.
  - Ex: $sed –i 's/^/\t' file
- If back up of original file is to be maintained then extension to –i option can be used. Extension used can be anything. Its just acts like another file which contains the original content.
  - Ex: $sed –i.temp 's/^/\t/' emp

-i option is used to substitute for the given file. i.e. original file is actually changed.
If any copy without changes is to be maintained then –I with some extension can be used as shown above.

For ex: sed –i 's/^/\t/' emp
Above command will insert tab at beginning of the file. File 'emp' is changed. If you want to keep a copy of original file an extension can be used with option as shown below.
Ex: $sed –i.temp 's/^/\t/' emp

Now, emp.txt will have original file and emp is changed according to the command.

Instructor Notes:

5.9: SED Commands

More Commands

Sl.No	Command	Description
1	sed 10q temp	print first 10 lines of file(emulates behavior of "head")
2	sed q temp	print first line of file(emulates "head -1")
3	sed '$!d' temp # method 1 sed -n '$p' temp # method 2	Prints last line of a file(emulates "tail -1")
4	sed '$!N;s/\n/ /' temp	join pairs of lines side-by-side (like "paste")
5	sed '$!N; s/^\(.*\)\n\1$/\1/; t; D' temp	Delete all lines except duplicate lines (emulates "uniq -d").
6	sed '1,10d' temp	delete the first 10 lines of a file

Instructor Notes:

Summary

- In vi editor:
  - esc key is used to change the mode.
  - esc - $ is used to move cursor at the end of the file.
  - wq is used to write (save) and quit from the file.
  - q! is used to quit without saving.
- SED

  Commands used to process the data.
  - Command line instruction
  - Script file based instruction

Instructor Notes:

Review Questions

- What command is used to copy the lines in vi editor?
- _____ command search for the pattern in vi editor in forward direction?
- What is the <control b> command used for?
- VI editor is stream Oriented?
  - True
  - False