UNIX

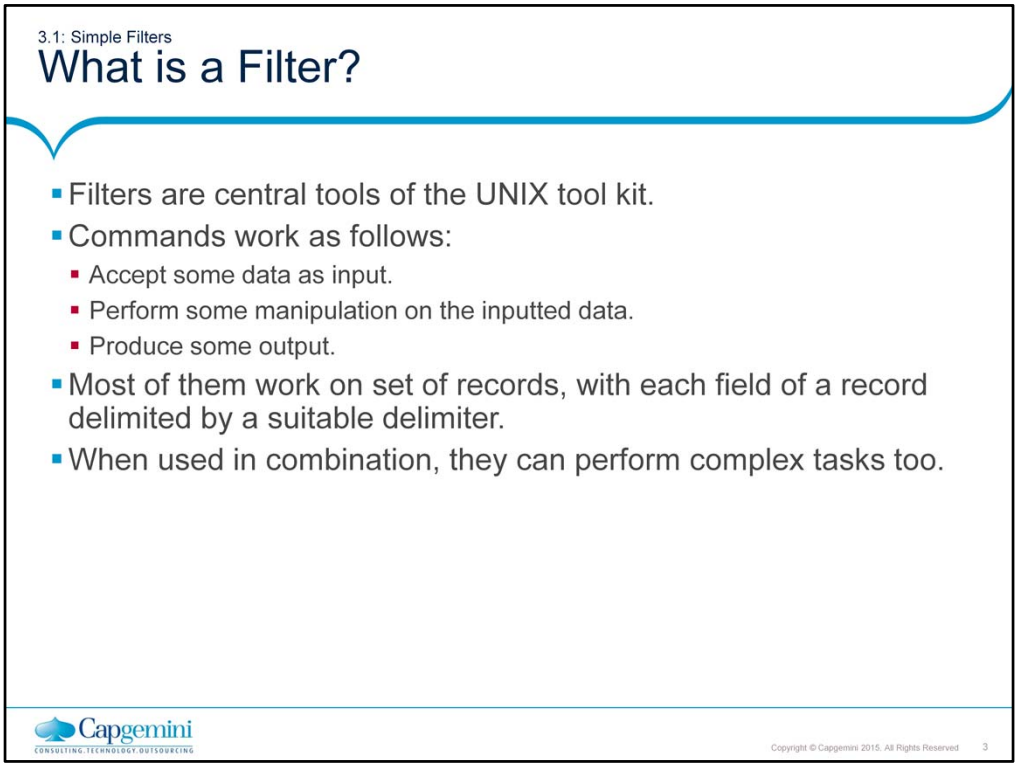
Lesson 03 : Filters

Lesson Objectives

- In this lesson, you will learn:
  - Filter commands in UNIX:
    - Simple Filters
    - Advance Filters

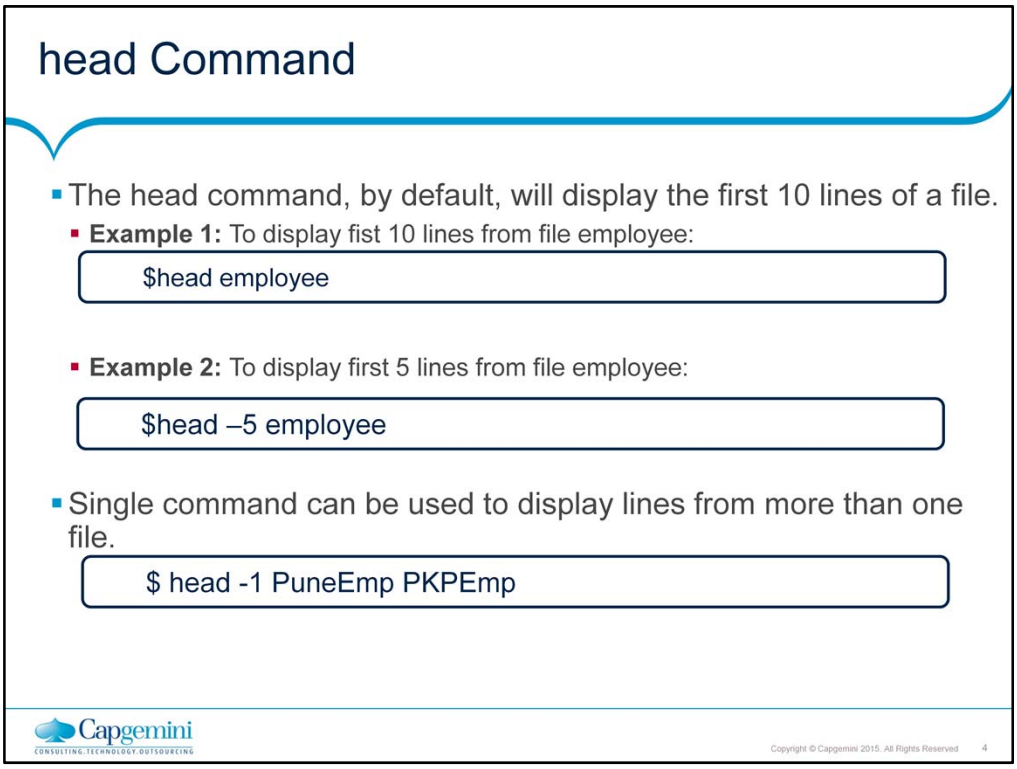
Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 2



Simple Filters:

Filters are central tools of the UNIX tool kit. These commands accept some data as input, perform some manipulation on it and produce some output. Most of them work on set of records, with each field of a record delimited by a suitable delimiter. They serve as useful text manipulators. When used in combination, they can perform complex tasks as well. This lesson discusses some commonly used simple filters.



The head Commands:

These are simple horizontal filters.
Using head, it is possible to display beginning of one or more lines form files. By default, the first 10 lines are displayed. Incase a numeric line count argument is specified, the command would display those many lines from the beginning of the file.

To display first 10 lines of the file bigfile, use the following syntax**:**

```
$ head bigfile
```

Output: cfile1.lst
cfile2.lst
errfile
file1.txt
file2.txt
file3.txt
mail
newdir1
newdir2
newfile.txt

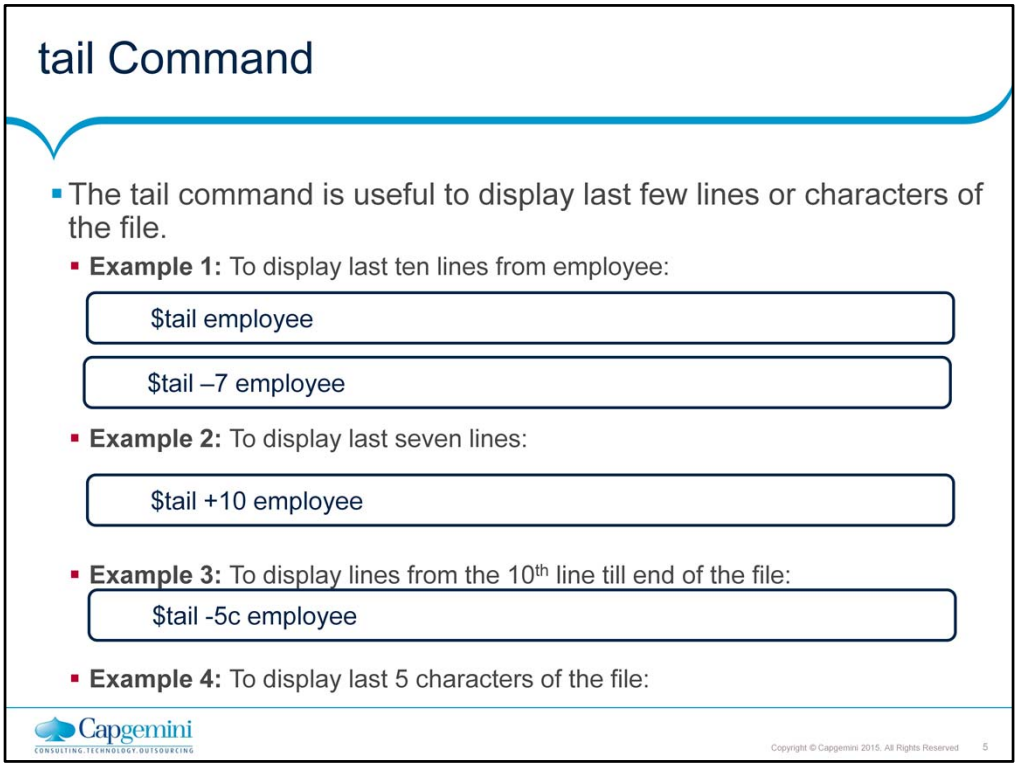
To display first 3 lines of the file bigfile, use the following syntax:

```
$ head -3 bigfile
```

Output:
cfile1.lst
cfile2.lst
errfile

```
$ head -1 listing bigfile
```

Output:
==> listing <==
cfile1.lst
==> bigfile <==
cfile1.lst



The tail Commands :

Using the tail command:
Using tail, the end of file can be displayed – default being last 10 lines.

```
$ tail -2 listing
```

Output:
result
testdir2
mail
newdir1
newdir2
newfile.txt

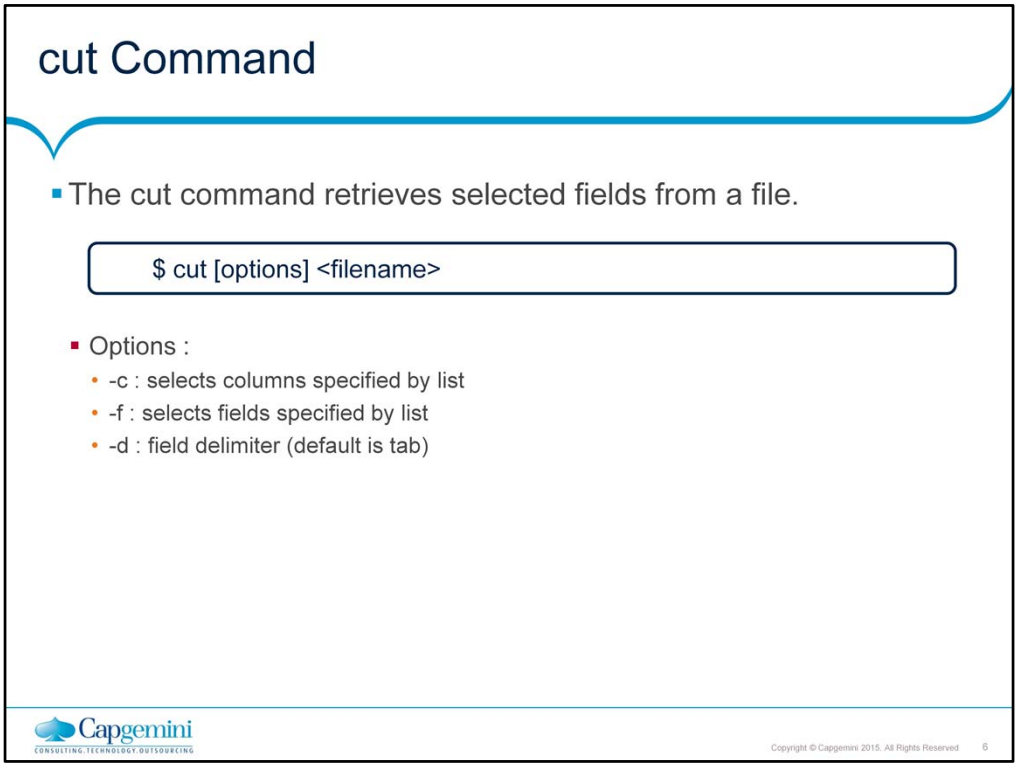
```
$ tail +20 bigfile
```

Output : result
Testdir2

To display last 6 characters from bigfile, use the following syntax:

```
$ tail -6c bigfile
```

le.txt



cut Command:

You can slice the file vertically with the cut command, and paste laterally with the paste command.

The cut command can be used to retrieve specific column information from a file. In case of fixed record formats, the –c (columns) option can be used to specify column positions. If a delimiter has been used, -f (field) in conjunction with –d (delimiter) options can be used for retrieval. The default delimiter is tab.

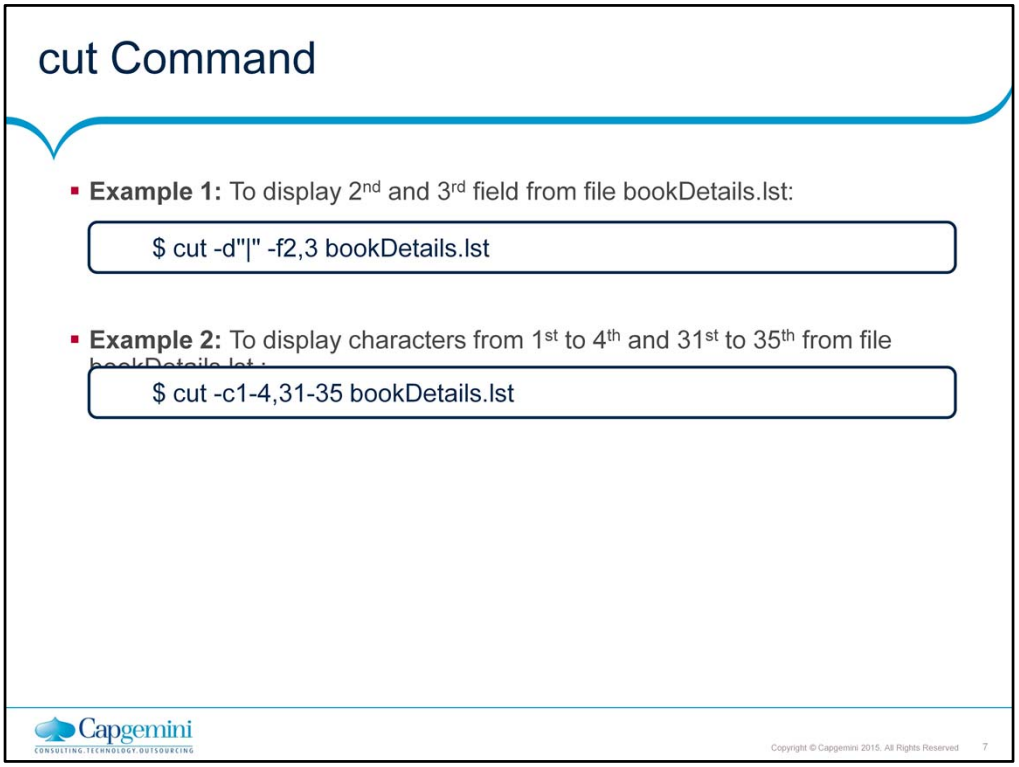
```
$ cat bookDetails.lst
```

Output:
```
1001|Unix for You              |375
1002|Learning Unix             |250
1003|Unix Shell Programming    |450
1004|Unix Device Drivers       |375
1005|Advanced Unix Concepts    |450
```

The following command will display first 4 characters followed by 31st to 35th characters:

```
$ cut -c1-4,31-35 bookDetails.lst
```

Output:
```
1001|375
1002|250
1003|450
1004|375
1005|450
```



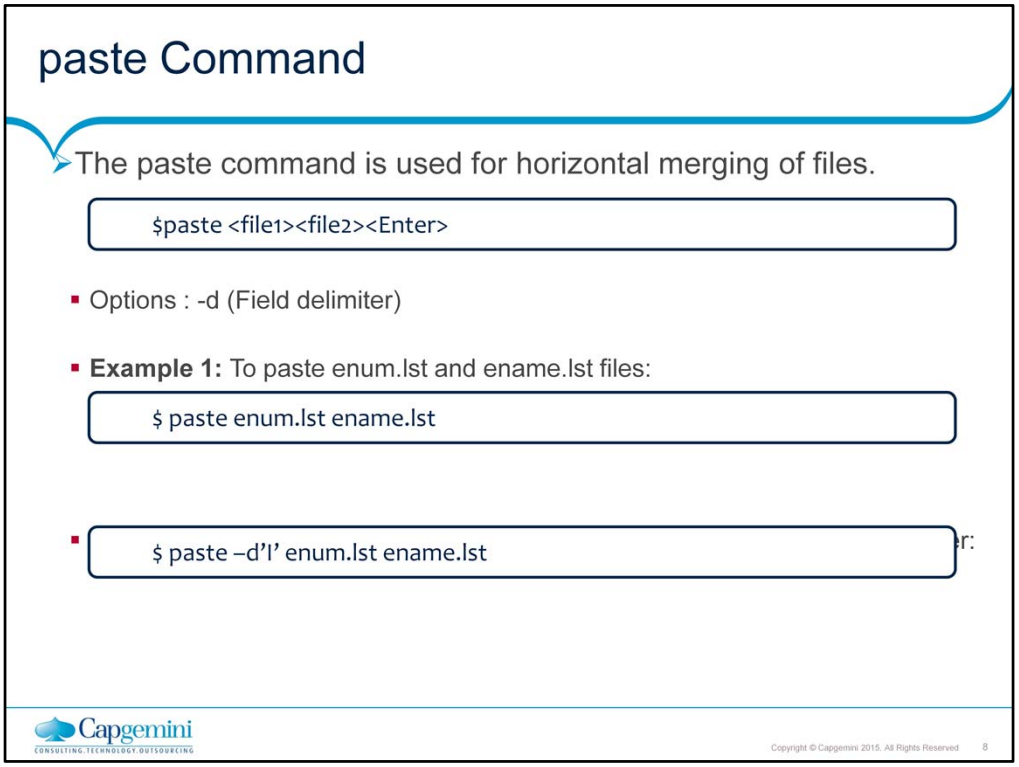
cut Commands:

To display 2nd and 3rd field from file bookDetails.lst :
In the given command on the above slide, the –d option specifies field delimiter is |.
Hence it will consider that in bookDetails file there are 3 fields separated by '|' character.
The –f option will specify to display 2nd and 3rd field.

```
$ cut -d"|" -f2,3 bookDetails.lst
```

Output:

```
Unix for You                |375
Learning Unix               |250
Unix Shell Programming      |450
Unix Device Drivers         |375
Advanced Unix Concepts      |450
```



paste Command:

Several files can be pasted laterally, with specific delimiters, with the paste command.

```
$  cat enum.lst
```

Output:
1010
2021
3718
4135
5765

```
$ cat ename.lst
```

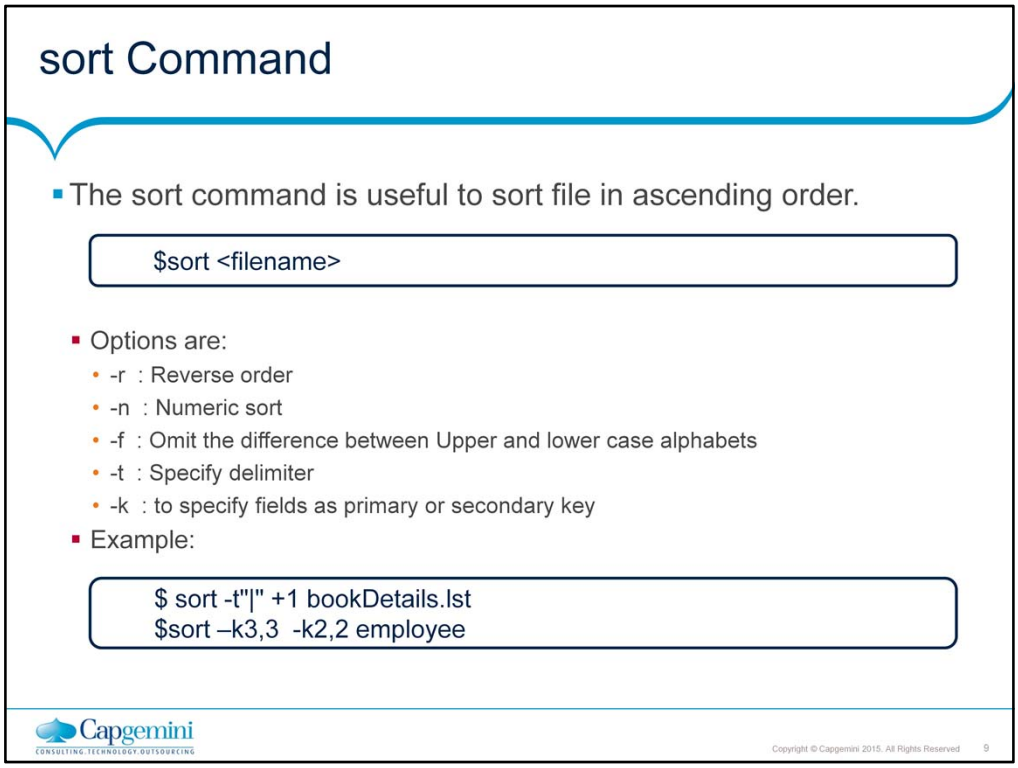
Output:
Abc
Zxc
Qwe
Jkl
Uio

```
$ paste enum.lst ename.lst
```

Output:
1010 Abc
2021 Zxc
3718 Qwe
4135 Jkl
5765 Uio

```
$ paste -d"|" enum.lst ename.lst
```

Output:
1010|Abc
2021|Zxc
3718|Qwe
4135|Jkl
5765|Uio



sort Command:

Sorting a file with the sort command:

The sort command sorts a file (which may or may not contain fixed length records) on line by line basis. Default sorting is in the ascending ASCII order, which can be reversed by using the –r option.

Sorting can be done on one or more fields by specifying the delimiter using –t option. It is also possible to specify character positions within fields.

Using the –m option, it is also possible to merge any number of sorted files.

Since the sorting is done on the basis of ASCII collating sequence, incase of sorting of numbers, -n option needs to be used.

```
$ sort -t"|" +1 bookDetails.lst
```

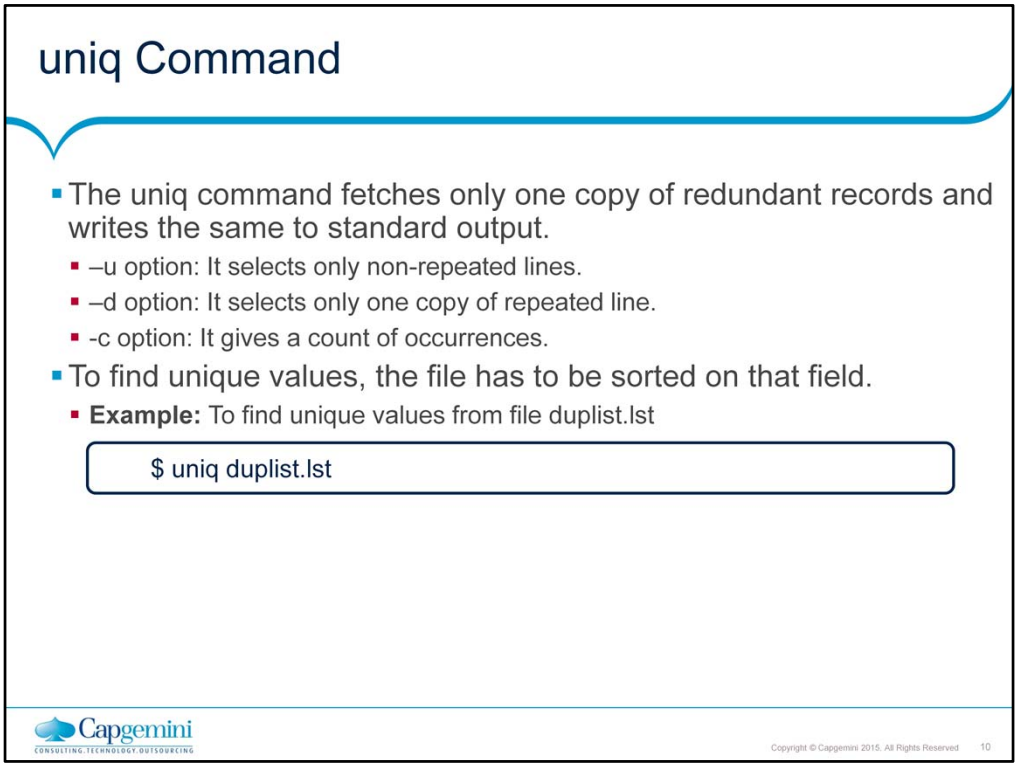
Output:
```
1005|Advanced Unix Concepts   |450
1002|Learning Unix            |250
1004|Unix Device Drivers       |375
1003|Unix Shell Programming    |450
1001|Unix for You              |375
```

To sort file employee on 3rd field as primary key and 2nd field as secondary key, use the following syntax:

```
$ sort –t"|" –k3,3 -k2,2 Employee
```

To consider only 3rd and 4th character from 2nd field for sorting employee file, use the following syntax:

```
$sort –t"|" –k2.3,2.4 employee
```



uniq Command:

The **uniq** command requires a sorted file as input. It fetches only one copy of redundant records and writes the same to standard output.

The **–u** option can be used to select only non-repeated lines, while the **–d** option can be used to select only one copy of repeated line. It is also possible to get a count of occurrences with the **–c** option.

Example 1:

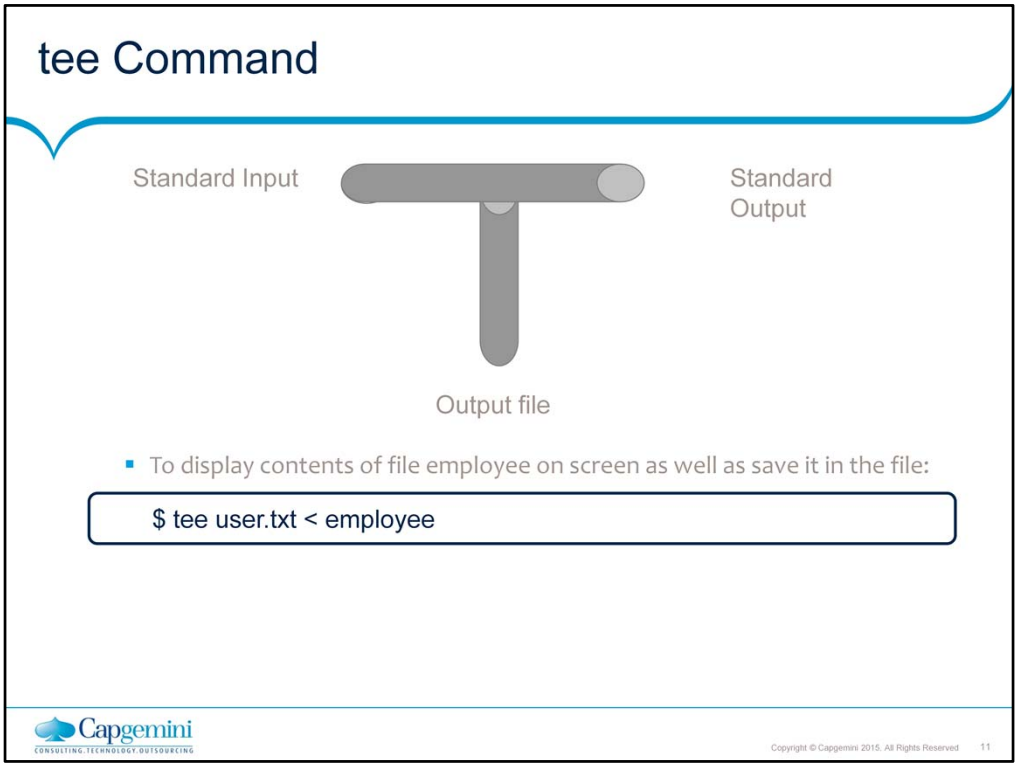
```
$ cat  duplist.lst
```

Output:
```
1
34
30
34
1
23
23
4
```

Example 2:

```
$ sort -n duplist.lst | uniq
```

Output:
```
1
4
23
30
34
```



tee Command:

The tee command copies the standard input to the standard output and also to the specified file.

If it is required to see the output on screen as well as to save output to a file, the **tee** command can be used. The **tee** command uses both standard input and standard output.

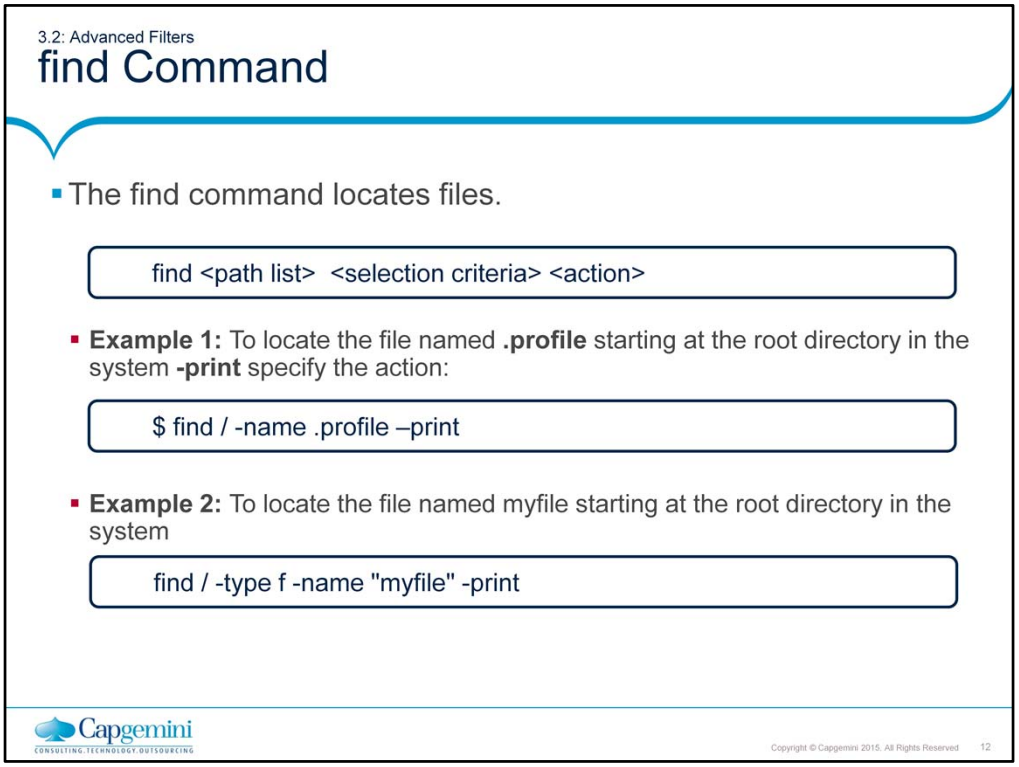
To display list of users and it's count both on screen, use the following:

```
Who| tee /dev/tty  | wc -l
```

If we give command as **who|wc –l** , it will display only number of users on screen. However, in the above command **tee** will save the o/p to /dev/tty file which is terminal device file. Hence the o/p of **who** will be displayed on screen and also will be transferred as i/p to **wc** command.

In the following command, **sort** will sort the file and o/p will be transferred to **tee** command. It will display o/p on screen as well as store it in file **sorted_file.txt**. The o/p will be given to **uniq** command which will give count of lines in the file. The head will find first 12 lines and store it in **top12.txt** file.

```
$sort somefile.txt | tee sorted_file.txt |  uniq -c |  head 12 > top12.txt
```



find Command (locating files with find):

The **find** command is used to find files matching a certain set of selection criteria. The **find** command searches recursively in a hierarchy, and also for each pathname in the pathname-list (a list of one or more pathnames specified for searching).

The syntax of the find command is given in the following format:

```
$ find <path list>  <selection criteria> <action>
```

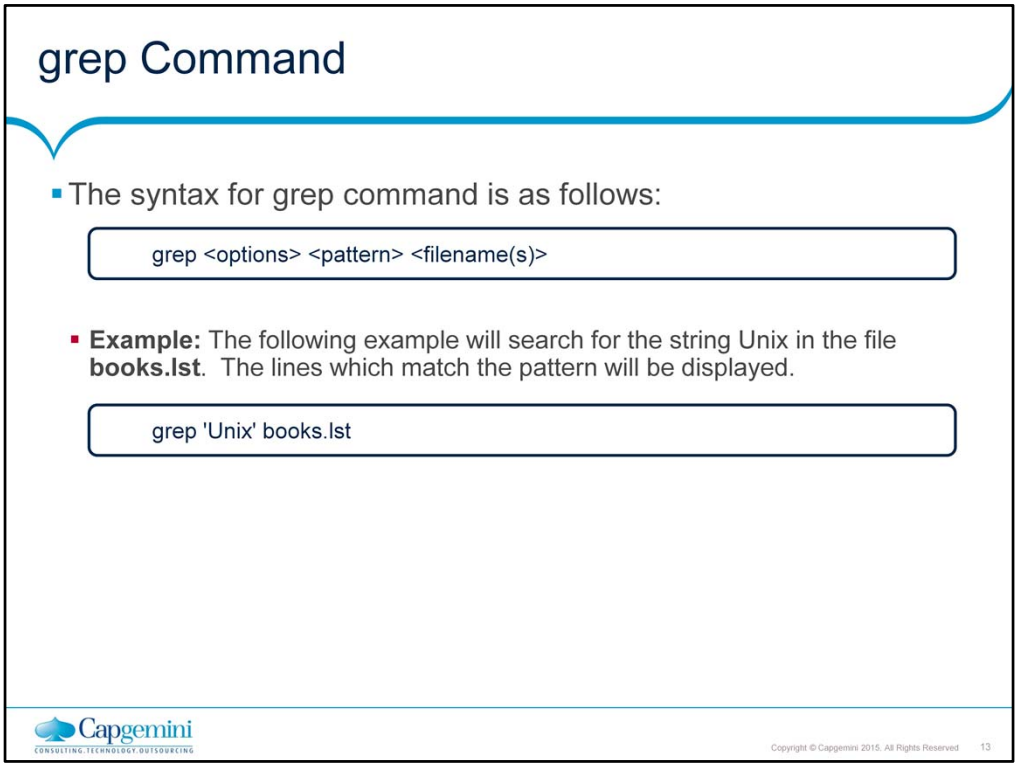
The find command first looks at all the files in the directories specified in the path list. Subsequently, it matches the files for one or more selection criteria. Finally it takes action on those selected files.

```
$ find / -name .profile -print
```

The above command will locate the **.profile** files in the system.

```
$find . –name *stat
```

The above command will locate all file names ending with **stat**.



grep Command:

The **grep** command is used to locate a pattern / expression in a file / set of files. There are many options that are available for obtaining different types of outputs.

The syntax for the grep command is as follows:

```
grep <options> <pattern> <filename(s)>
```

The grep command scans the file(s) specified for the required pattern, and outputs the lines containing the pattern. Depending on the options used, appropriate output is printed. The grep command compulsorily requires a pattern to be specified, and the rest of the arguments are considered as file names in which the pattern has to be searched.

grep Command

- Options of grep:
  - c : It displays count of lines which match the pattern.
  - n : It displays lines with the number of the line in the text file which match the pattern.
  - v : It displays all lines which do not match pattern.
  - i : It ignores case while matching pattern.
  - -w : It forces grep to select only those lines containing matches that form whole words

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

grep Command

- **Example 1:** To print all lines containing “rose” regardless of case:

```
$grep -i rose flower.txt
```

- **Example 2:** To print all lines containing “rose” as a word:

```
$grep -w rose flower.txt
```

- **Example 3:** To print all lines not containing “rose”:

```
$grep -v rose flower.txt
```

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 15

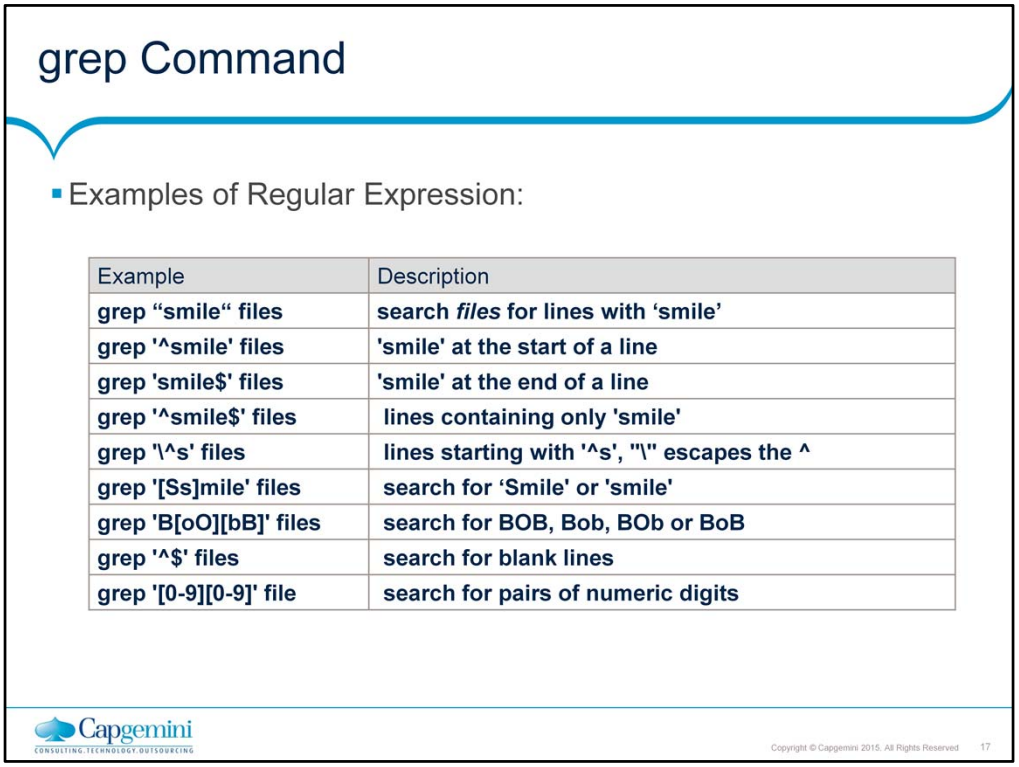
grep Command

- Regular Expression:

Expression	Description
^ (Caret)	match expression at the start of a line, as in ^A.
$ (Question)	match expression at the end of a line, as in A$.
\ (Back Slash)	turn off the special meaning of the next character, as in \^.
[ ] (Brackets)	match any one of the enclosed characters, as in [aeiou]. Use Hyphen "-" for a range, as in [0-9].
[^]	match any one character except those enclosed in [ ], as in [^0-9].
. (Period)	match a single character of any value, except end of line.
* (Asterisk)	match zero or more of the preceding character or expression.
\{x,y\}	match x to y occurrences of the preceding.
\{x\}	match exactly x occurrences of the preceding.
\{x,\}	match x or more occurrences of the preceding.

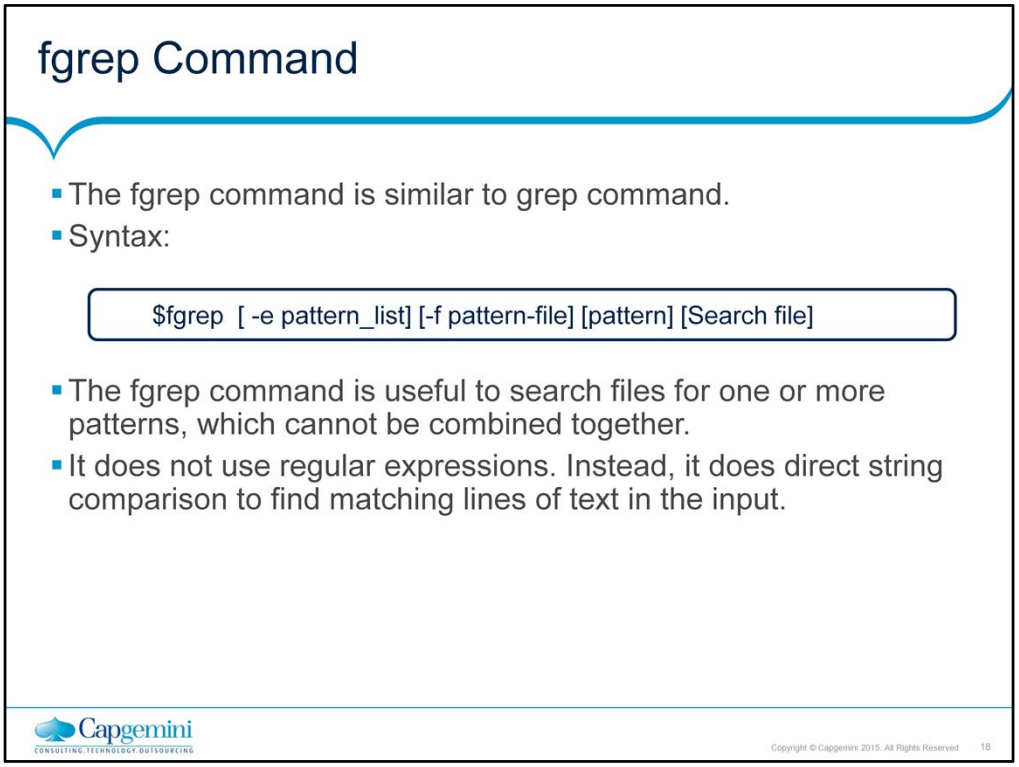
Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 16



grep Command:
Some more examples:

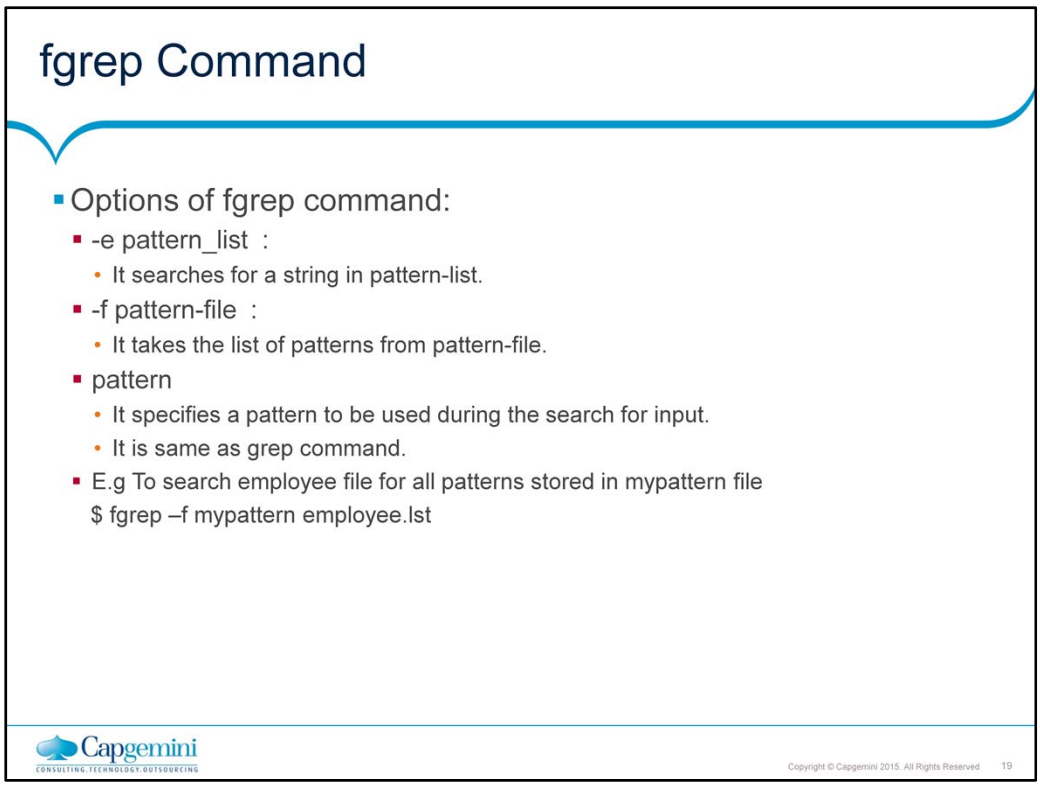
Example	Description
grep '^From: ' /usr/mail/$USER	list your mail
grep '[a-zA-Z]'	any line with at least one letter
grep '[^a-zA-Z0-9]	anything not a letter or number
grep '[0-9]\{3\}-[0-9]\{4\}'	999-9999, like phone numbers
grep '^.$'	lines with exactly one character
grep "'smug"'	'smug' within double quotes
grep "'*smug"*'	'smug', with or without quotes
grep '^\.'	any line that starts with a Period "."
grep '^\.[a-z][a-z]'	line start with "." followed by 2 lowercase letters



fgrep Command:

The **fgrep** command can also accept multiple patterns from command line as well as a file. However, it does not accept regular expressions – only fixed strings can be specified. The **fgrep** command is faster than **grep** and **egrep**, and should be used while using fixed strings.

The **egrep** and **fgrep** commands to some extent overcome the limitations of **grep**. However, the principal disadvantage of the grep family of filters is that there are no options available to identify fields. Also it is very difficult to search for an expression in a field. This is where the **awk** command is very useful.



fgrep Command:
Example using fgrep:

```
$ cat stud.lst
```

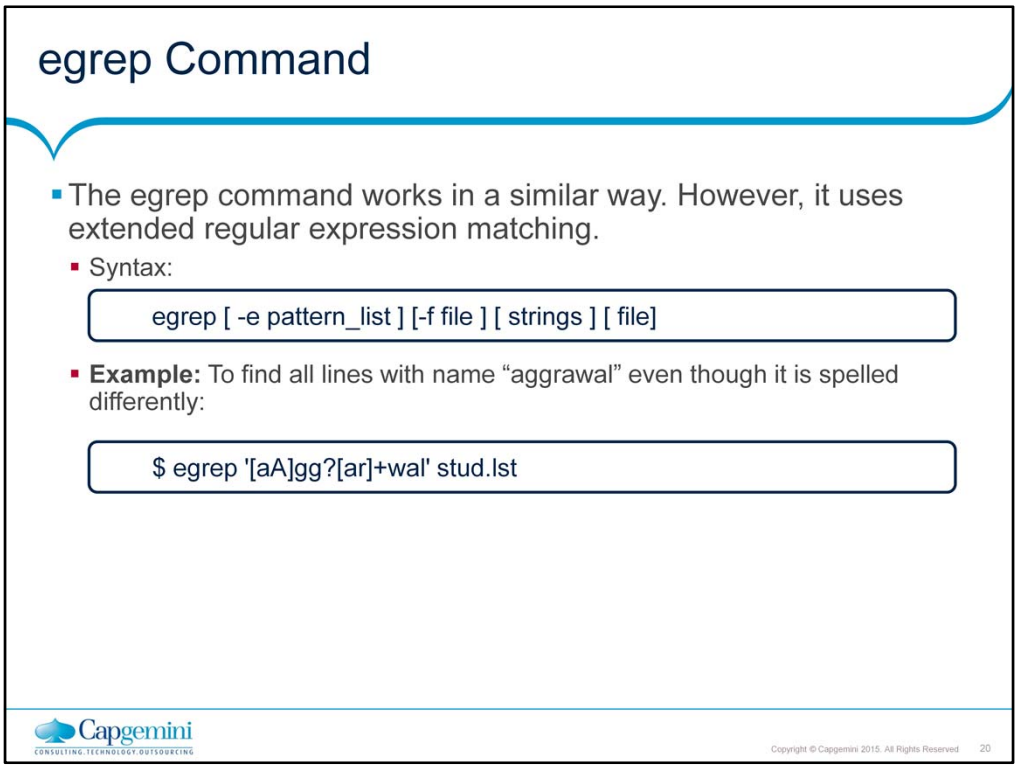
Output:
```
R001|Pratik Sharma      |425
R002|Pallavi V.         |398
R003|Pratibha Aggarwal  |400
R004|Preeti Agrawal     |390
R005|Prerana Agarwal    |421
R006|Pranita aggarwal   |380
```

```
$cat mypattern
```

Output:
```
Pratik
Pratibha
```

```
$ fgrep –f mypattern stud.lst
```

Output:
```
R001|Pratik Sharma      |425
R003|Pratibha Aggarwal  |400
```



egrep Command (extending grep):

The egrep command offers all the options of the grep command. In addition, it is possible to specify alternative patterns. The table given below gives the extended regular

Expression	Significance
ch+	Match with 1 or more occurrences of character ch
ch?	Match with 0 or more occurrences of character ch
exp1|exp2	Match with expressions exp1 or exp2
(a1|a2)a3	Match with expression a1a3 or a2a3

Some examples of using egrep are given:

```
$ cat stud.lst
```

Output:
```
R001|Pratik Sharma      |425
R002|Pallavi V.         |398
R003|Pratibha Aggarwal  |400
R004|Preeti Agrawal     |390
R005|Prerana Agarwal    |421
R006|Pranita aggarwal   |380
```

```
$ egrep '[aA]gg?[ar]+wal' stud.lst
```

Output:
```
R003|Pratibha Aggarwal  |400
R004|Preeti Agrawal     |390
R005|Prerana Agarwal    |421
R006|Pranita aggarwal   |380
```

Summary

- In this lesson, you have learnt:
  - The head and tail filter commands filter the file horizontally.
  - The cut and paste commands filter the file vertically.
  - -m option of sort command is used to merge two sorted files.
  - The tee command helps us to send o/p to standard o/p as well as to file.
  - grep, fgrep, and egrep commands use to search files for some pattern.

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 21

Review Questions

- Question 1: ___ command to display directory listing on screen as well as store it in dirlist.lst.
- Question 2: ___ filter commands filter file vertically?
- Question 3: ___ filter commands filter file horizontally?

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 22