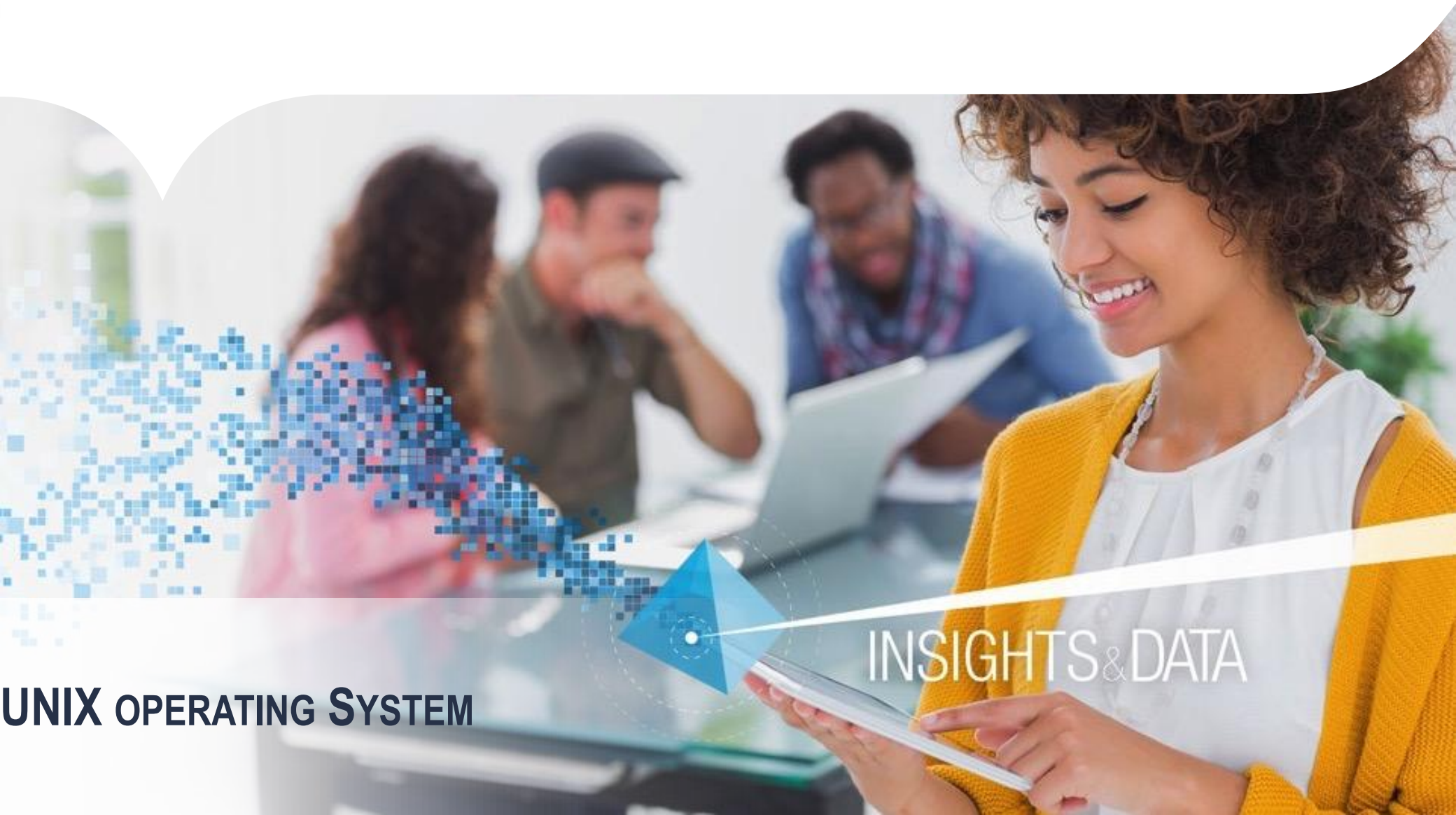
Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

People matter, results count.



UNIX OPERATING SYSTEM

UNIX/LINUX

1: UNIX OPERATING SYSTEM

OBJECTIVES

In this session, you learn about:

- The functions of OS
- The history of Unix
- The features of UNIX
- The Unix architecture
- Process management
- CPU scheduling
- Memory management
- File management

OPERATING SYSTEM (OS)

- OS is a system software
- OS can be defined as an organized collection of software consisting of procedures for operating a computer
- OS provides an environment for execution of programs
- OS acts as an interface between the user and the hardware of the computer system.

OPERATING SYSTEM

- Operating system interacts with user in two ways
  - Operating system commands

    Enables user to interact directly with the operating system.

  - Operating system calls

    Provides an interface to a running program and the operating system. System calls in UNIX are written in C.

HISTORY OF UNIX

- Ken Thompson of AT&T Bell Laboratories designed UNIX in late 1960s.
- Two versions of UNIX that emerged are AT&T Unix and BSD Unix.
- In 1989, AT&T and Sun Microsystems joined together and developed system V release 4 (SVR4).
- Two of the main standards mainly in use are **POSIX** (Portable Operating System Interface) and **X/open** standard. In 1988, MIT formed Xconsortium developed vendor-neutral **Xwindow** System.

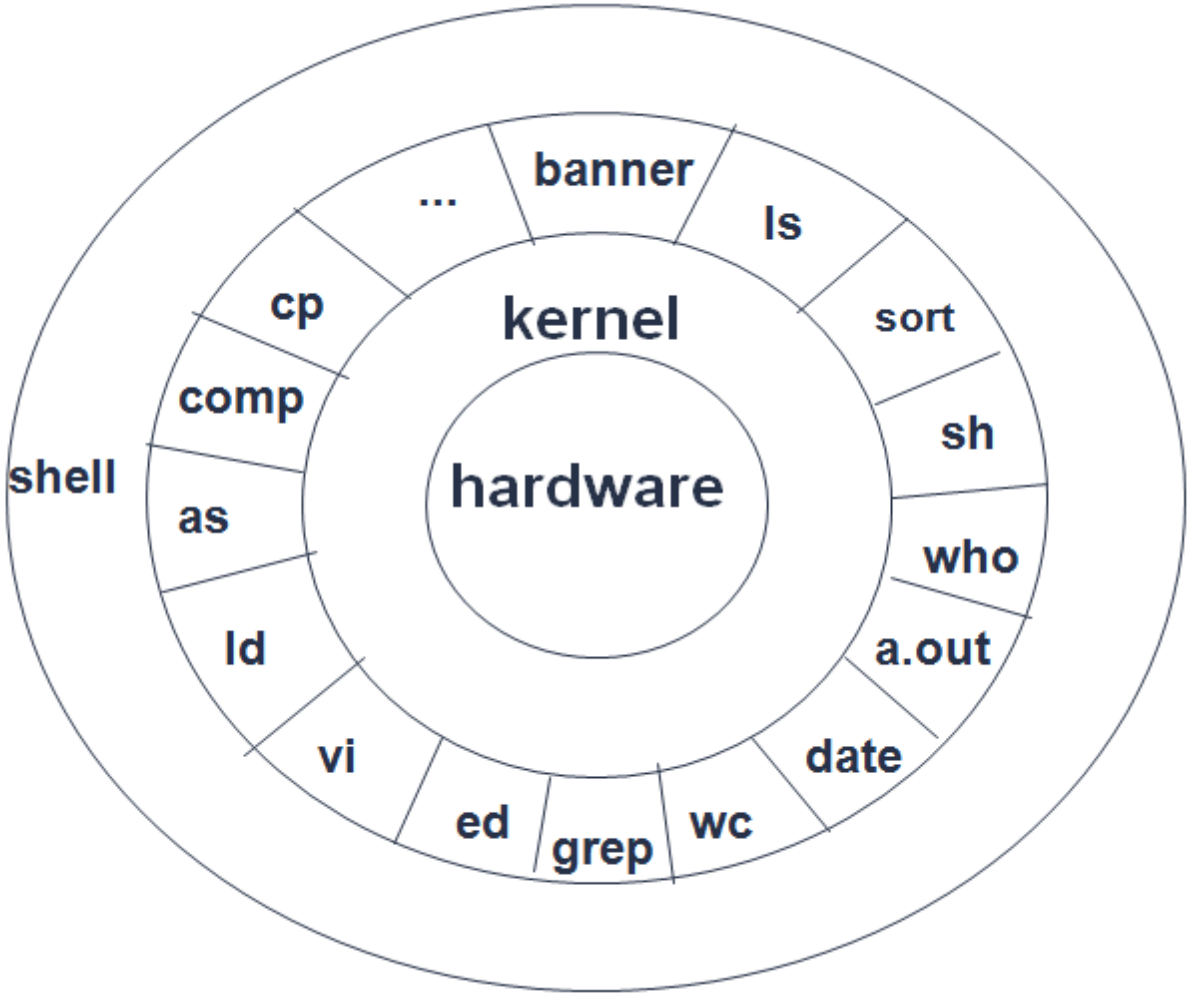
What is Linux?

- An open-source UNIX like operating system
- Initially created by Linus Torvalds for PC architecture
- Ports exist for Alpha and Sparc processors
- Developer community world-wide contribute to its enhancement and growth

FEATURES OF UNIX

- Multi-user, multitasking, timesharing
- Portability
- Modularity
- File structure
- Security
- Strong networking support & advanced graphics

LAYERED ARCHITECTURE



UNIX SYSTEM ARCHITECTURE

- Unix system follows a layered approach. It has four layers
- The innermost layer is the hardware layer
- In the second layer, the kernel is placed
- The utilities and other application programs form the third layer
- Fourth layer is the one with which the user actually interacts.

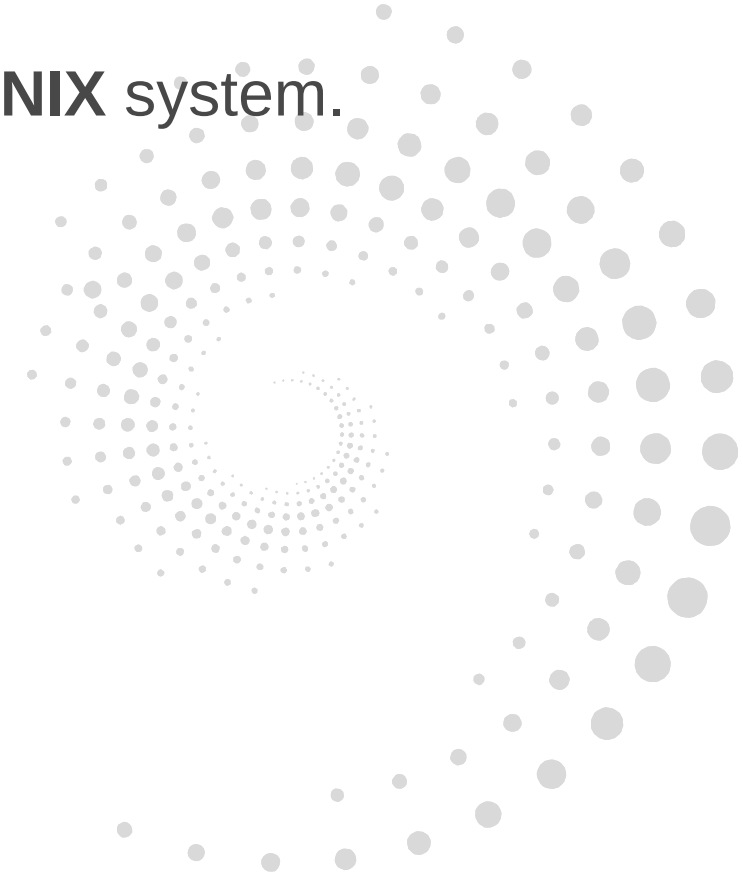
UNIX System Architecture (cont…)

- Kernel is that part of the OS which directly makes interface with the hardware system.

- **Actions:**
  - Provides mechanism for creating and deleting processes
  - Provides processor scheduling, memory, and I/O management
  - Provides inter-process communication.

UNIX System Architecture (cont…)

- A utility program that comes with the **UNIX** system.
- Features of Shell are:
  - Interactive Processing
  - Background Processing
  - I/O Redirection
  - Pipes
  - Shell Scripts
  - Shell Variables
  - Programming Constructs



PROCESS MANAGEMENT

- A process is a program in execution
- Several processes can be executed simultaneously in a **UNIX** system.
- A process is generally created using the "**fork( )**" system call.
- The process that invokes the "**fork( )**" system call is the parent process, and the newly created process is called the child process.

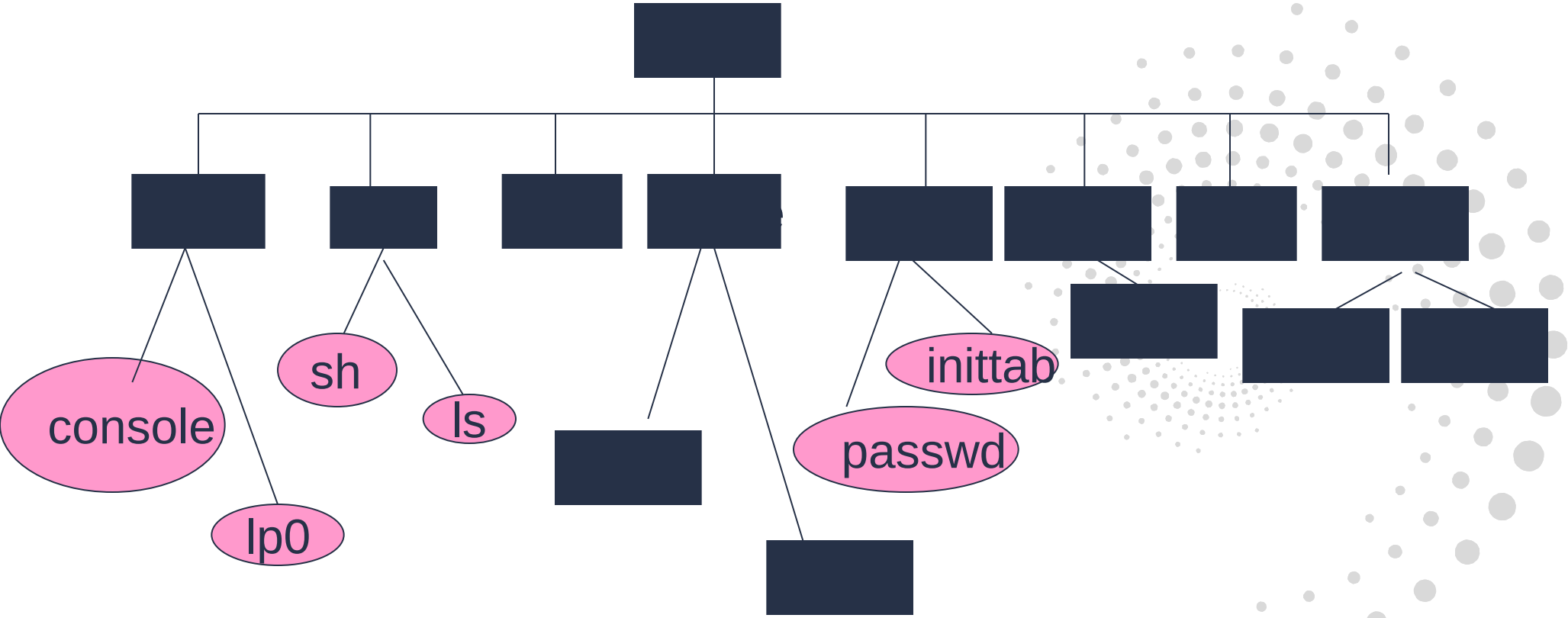
CPU SCHEDULING

- **Unix** uses round-robin scheduling to support its multi-user and time-sharing feature.
- Round-robin fashion of scheduling is considered to be the oldest, simplest and widely used algorithm.
- Every process is given a time slice (10-100 millisec.)

File Management

- UNIX uses a hierarchical file system with “/” as its root.
- Every non-leaf node of the tree is called as a directory file.
- Every leaf node can either be a file, or an empty directory

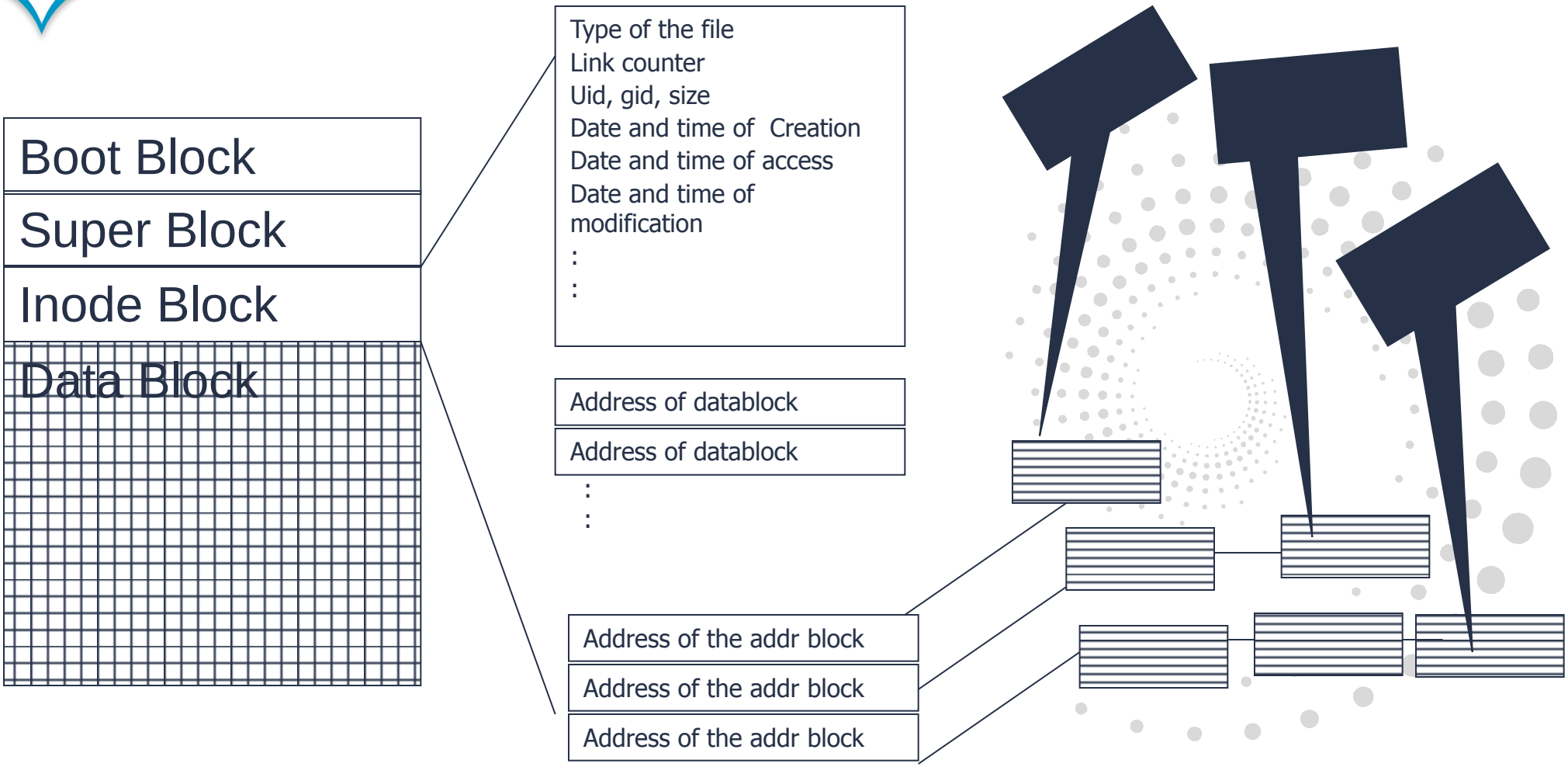
FILE SYSTEM



File System

- File system is the structure in which files are stored on disk
- File in UNIX is sequence of bytes organized in the form of blocks
- The size of each block is 512 bytes (depends on architecture)
- Block size can be decided while creating the file system structure

File System Structure



COMMON UNIX FLAVOURS

- BSD: Berkeley, BSD
- Solaris: Sun Microsystems, Sys 5/BSD
- Ultrix: Digital Equipment Corporation, BSD
- OSF 1: Digital Equipment Corporation, BSD/sys 5
- HPUX: Hewlett-Packard, Sys 5
- AIX: IBM, Sys 5 / BSD
- IRIX: Silicon Graphics, Sys 5
- GNU/Linux: GNU, BSD/Posix

TYPES OF UNIX USERS

- Broad classification of users
  - root (most privileged)
  - Non-root (less privileged)
- Group
  - UNIX allows user IDs to be grouped
  - A single user ID can be member of multiple groups
- Differentiating users with respect to file access
  - Owner
  - Group
  - Others

WORKING WITH UNIX

- User logs in with a valid user ID
- User logs out to terminate the login session

Q & A

QUIZ

1.What is operating system?

a) collection of programs that manages hardware resources
b) system service provider to the application programs
c) link to interface the hardware and application programs
d) all of the mentioned

Option d

2.What is property of UNIX ?

a) Multi User
b) Multi Processes
c) Multi Tasking
d) All of above

Option d

QUIZ

3.In Unix, Which system call creates the new process?

a) fork

b) create

c) new

d) none of the mentioned

Option a

4.Unix system architecture follows ------ approach

a) Structured

b) Layered

c) Hierarchy

d) none

Option b

SUMMARY

In this session, you learned about …

- The functions of OS
- The History of Unix
- The features of Unix
- The Unix Architecture
- Process management
- CPU Scheduling
- File management

UNIX/LINUX

2: : UNIX COMMANDS

OBJECTIVES

In this session, you will learn to:

- **Use the basic Unix commands**
  - pwd
  - date
  - who
  - ls
  - Man
- **Use "man" pages**

SIMPLE COMMANDS

- **pwd**
  - Displays the current working directory.
- **date**
  - Displays the current date and time

SIMPLE COMMANDS

- **who**
  - Displays the names of all the users who have currently logged in
- **who am i**
  - Displays the name of the current user.
- **Whoami**
  - Displays only name of the user

LISTING THE DIRECTORY CONTENTS

- **ls**

Syntax :ls [options] [file….]

options: -l list in long format

-a list all files including those beginning with a dot

-i list inode no of file in first column

-s reports disk blocks occupied by file

-R recursively list all sub directories

-F mark type of each file

-C display files in columns

META CHARACTERS

Meta Characters	Purpose	Example
*	Match with one or more characters or none	$ ls –l *.c file*
?	Match with any single character	$ ls –l file?
[…]	Match with any single character within the brackets	$ ls –l file[abc]
;	Command separator	$ cat file1; cat file2
|	Pipe two commands	$ cat abc | wc
( )	Group commands Useful when the output of the command group has to be redirected	$ (echo "==== x.c ===="; cat x.c) > out
`command`	Execute the command enclosed within back quotes. Useful when the output of a command into a variable in a shell script	count=`expr $count + 1` *assuming count has value 3, this increments the value of count*
'string'	Quote all characters with no substitution (ex. no special meaning for $	echo'expr $count + 1' displays *expr $count + 1*
"string"	Quote all characters with substitution. The characters $\ (back slash) and back quote have special meaning.	echo "expr $count + 1" displays *expr 3 + 1* *assuming the variable count has value 3*

LISTING THE DIRECTORY CONTENTS

```
$ ls -l

total 6
-rwxr-xr-x    1   user1   projA    12373   Dec 15 14:45  a.out
drwxr-xr-x    2   user2   projD     4096   Dec 22 14:00  awkpro
-rw-r--r--    1   user1   projA    12831   Dec 12 13:59  c
-rw-------    1   user1   projA    61440   Dec 16 11:16  data
-rw-r--r--    1   user3   projC      259   Dec 20 14:29  cs
```

File type

File access permissions

Link count

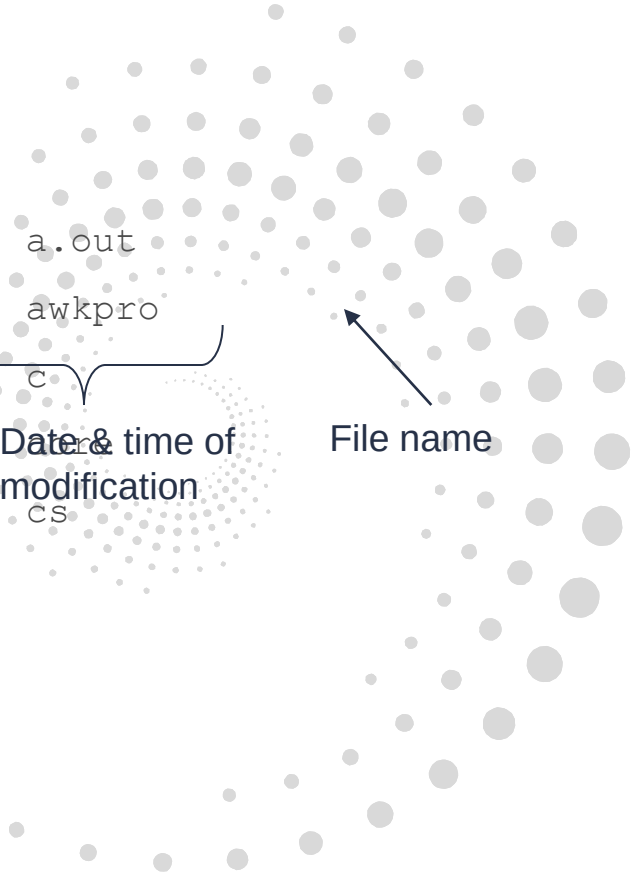
User id

Group id

File size in bytes

Date & time of modification

File name



GETTING HELP ON COMMANDS

- The **Unix** manual, usually called **man pages**, is available on-line to explain the usage of the Unix system and commands.

Syntax:

- **man** [options] command_name

Common Options

-k keyword	list command synopsis line for all keyword matches
-M path	path to man pages
-a show	all matching man pages (SVR4)

- **info** command_name **-** help for commands
- **help -–**command_name**–** gives command syntax

SUMMARY

In this session, you have learned to …

- use the basic Unix commands like
  - pwd
  - date
  - who
  - ls
  - man
- use “man” pages

UNIX/LINUX

3: FILES AND DIRECTORIES

MODULE OUTLINE

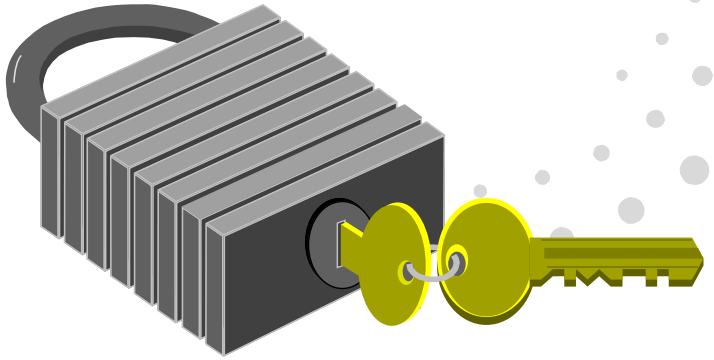
- In this session, you will learn to:
  - set file permissions using the **chmod** command
  - use directory-related commands namely mkdir, rmdir, cd commands
  - use file-related commands namely cp, mv, rm commands
  - create and edit files using the vi editor

File Access Permissions

- Refers to the permissions associated with a file with respect to the following

- **Permission Levels**
  - User (owner) (u)
  - Group (wheel, staff, daemon, etc.) (g)
  - World (guest, anonymous and all other users) (o)

- **Permission Settings**
  - Read (r)
  - Write (w)
  - Execute (x)



File Access Permissions

- No read permission does not allow the user to:
  - List the contents of directory
  - Remove the directory
- No Write permission does not allow the user to :
  - copy files to the directory
  - remove files from the directory
  - rename files in the directory
  - make a subdirectory
  - remove a subdirectory from the directory
  - move files to, and from the directory

File Access Permissions

- No execute permission does not allow the user to:
  - display the contents of a directory file from within the directory
  - change to the directory
  - display a file in the directory
  - copy a file to, or from the directory

CHANGING PERMISSIONS - CHMOD

- **chmod u+x file_name**

Syntax:

chmod <category> <operation> <permission> <filename(s)>

or

chmod <octal number> filename

Octal Number

4 - for read

2 - for write

1 - for execution

$ chmod 744 xyz

this sets read, write and execute permissions for owner, read permission for group and others

DIRECTORY CREATION

Command Syntax

mkdir [OPTION] DIRECTORY

$ mkdir <path>/<directory>

Example:

$ mkdir project1

This creates a directory project1 under current directory

Note: Write and execute permissions are needed for the directory in which user wants to create a directory

DIRECTORY REMOVAL

rmdir command removes directory

Syntax

- rmdir <directory name>

Example:

Removes project1 directory in the current directory

- rmdir project1

Remove multiple directories

rmdir pos1 pos2

Remove the directory recursively

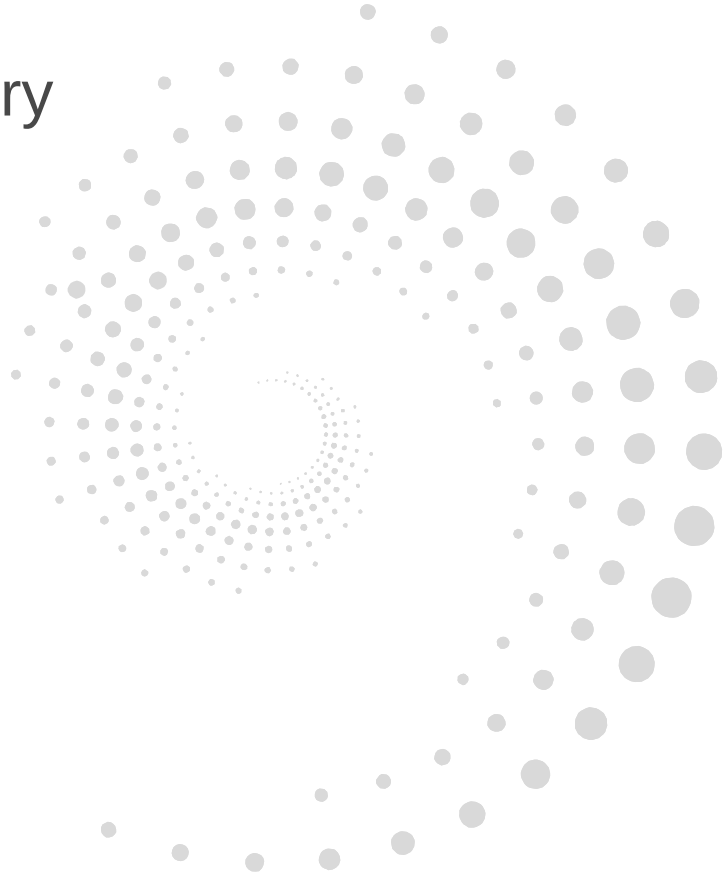
rmdir –p dir1/dir2/dir3

rmdir removes a directory if it is empty and is not the current directory

COMMAND - CD

cd command is used to change the directory

- **cd** - take to the home directory
- **cd ..** - takes to the parent directory
- **cd /** - takes to the root directory



FILE-RELATED COMMANDS

File Operation	Command
Copying a file	cp
Moving a file	mv
Removing a file	rm
Displaying a file and concatenating files	cat

COMMAND - CP

Used to copy files across directories

Syntax

cp <source file> <new file name>

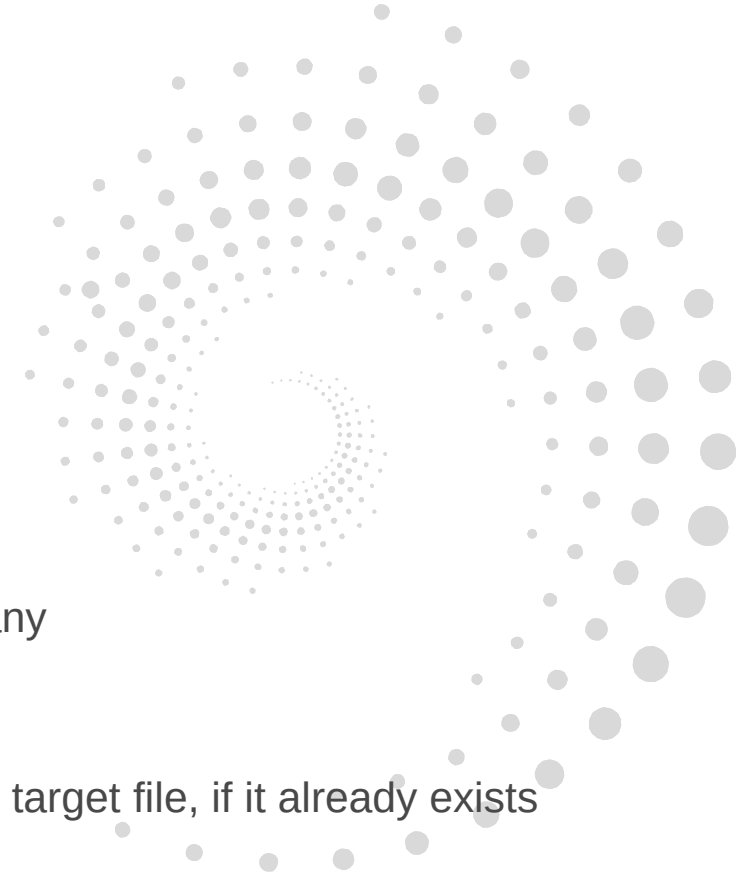
Example

cp file1 file2

COMMAND - CP

Options to cp

- -p
  - Copies the file and preserves the following attributes
    owner id
    group id
    permissions
    last modification time
- -r
  - recursive copy; copy subdirectories under the directory if any
- -i
  - interactive; prompts for confirmation before overwriting the target file, if it already exists



COMMAND - MV

Used to move a file, or rename a file

Preserves the following details

- owner id
- group id
- permissions
- Last modification time

-f suppresses all prompting (forces overwriting of target)

-i prompts before overwriting destination file

Used to remove a file

- **Syntax** : rm file(s)

-f suppresses all prompting

-i prompts before deleting destination file

-r will recursively remove the file from a directory (can be used to delete a directory along with the content)

Caution: Use “i” option along with “r” to get notified on deletion

COMMAND – CHOWN & CHGRP

$ ls –l

-rwxr-xr-x 1 user1 training 12373 Dec 15 14:45 a.out

-rwxr-xr-x 3 user1 faculty 4096 Dec 24 11:56 awkpro

$chown user2 a.out

$ls –l

-rwxr-xr-x 1 user2 training 12373 Dec 15 14:45 a.out

-rwxr-xr-x 3 user1 faculty 4096 Dec 24 11:56 awkpro

$ chgrp training awkpro

$ls –l

-rwxr-xr-x 1 user2 training 12373 Dec 15 14:45 a.out

-rwxr-xr-x 3 user1 training 4096 Dec 24 11:56 awkpro

COMMAND - UMASK

umask value is used to set the default permission of a file and directory while creating

umask command is used to see the default mask for the file permission

Default **umask** value will be set in the system environment file like /etc/profile

umask 022 will set a mask of 022 for the current session

- The file permission after setting this **umask** value will be 644
- And the directory permission will be 755

Linking files:

- Hard Link (in the same filesystem)
  - $ ln /usr/bin/clear /usr/bin/cls
  - Hard link uses the same inode number
- Soft Link (in different filesystems also used to link directories)
  - $ ln –s /usr/bin/clear /home/user1/cls

QUIZ

1.Copying a file use

a) cp

b) cd

c) dir

d) mv

2.Default permissions of a file can be changed with ?

a)group

b)chperm

c)chmod

d)chall

Option A

Option C

VI EDITOR

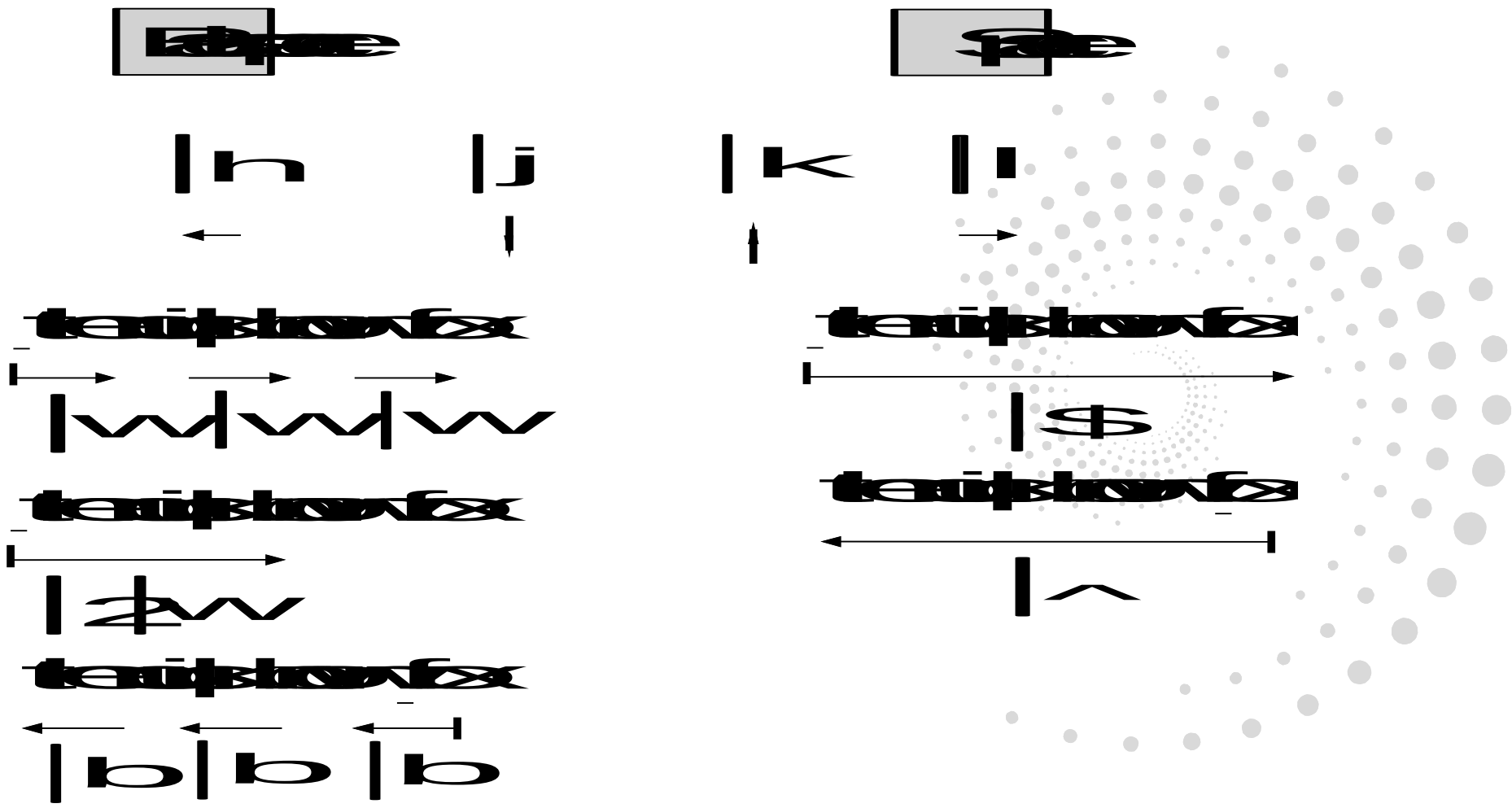
- **vi** is a visual editor used to create and edit text files.
  - A screen-oriented text editor
  - Included with most UNIX system distributions
  - Command driven

- Categories of commands include
  - Cursor movement
  - Editing commands
  - Search and replace commands

- The vi editor is invoked by the following command:

  $ vi filename

NAVIGATION



EDITING COMMANDS

- **Text insertion / replacement**

  - i - inserts text to the left of the cursor
  - a - inserts text to the right of the cursor
  - I - inserts text at the beginning of the line
  - A - appends text at end of the line
  - o - opens line below
  - O - opens line above
  - R - replaces text from cursor to right
  - s - replaces a single character with any number of characters
  - S - replaces entire line

EDITING COMMANDS

- **Deletion:**

  - x - to delete character at cursor position
  - 3x - to delete 3 characters at cursor position
  - dw - to delete word
  - 2dw - to delete 2 word
  - dd - to delete a line
  - 2dd - to delete 2 lines

EDITING COMMANDS

- **Yanking / Copy & Paste**
  - Y - copy line into buffer
  - 3Y - copy 3 lines into buffer
  - p - Paste buffer below cursor
  - P - Paste buffer above cursor

- **Save and quit**
  - :w - to save
  - :x - save and quit
  - :q - cancel changes
  - :q! - cancel and quit

SEARCH & REPLACE COMMANDS

The following commands are applicable for vi editor in Linux

/pat	searches for the pattern **pat** and places cursor where pattern occurs.
/	repeat last search
:%s/old/new/g	to change every occurrence in the whole file.
:#,#s/old/new/g	where #,# are replaced with the numbers of the two lines.

Ie: From which line to which line we want to replace with new string.

QUIZ

1. what does Y ?

A) copy line into buffer

B) copy 3 lines into buffer

C) Paste buffer below cursor

D) Paste buffer above cursor

Option A

2. what does 2dd?

A) copy line into buffer

B) copy 3 lines into buffer

C) Paste buffer below cursor

D) Paste buffer above cursor

Option A

SUMMARY

- **In this session, you have learned how to …**
  - use file permissions using the **chmod** command
  - use directory-related commands namely mkdir, rmdir, cd commands
  - use file-related commands namely cp, mv, rm commands
  - create and edit files using the vi editor

UNIX/LINUX

4: UNIX UTILITIES

OBJECTIVES

In this session, you will learn how to:

- use the Unix utilities such as

  cat, **echo**, **touch**, **more**, **file**, **wc**, **cmp**, **comm**, **find**

- employ redirection operators

- use filters such as

  sort, **grep**, **cut**, **head**, **tail**, **tr**, and **paste**

- use backup commands

  zip/gzip and **tar**

CAT

- **cat** command takes the input from the keyboard, and sends the output to the monitor

- We can redirect the input and output using the redirection operators

$ cat > file1

Type the content here

press <ctrl d>

$ cat file1

Displays the content of the file

$cat >> file1

This will append standard input to the content of file1

ECHO & READ

- **echo** command is used to print output to the screen

  echo “This is an example”

  This is an example

  x=10

  echo $x

  10

- **read** command allows to read input from user and assign it to the variable specified.

  read x

TOUCH

- **touch** is used to change the time stamp of the file

Syntax:

touch [options] file

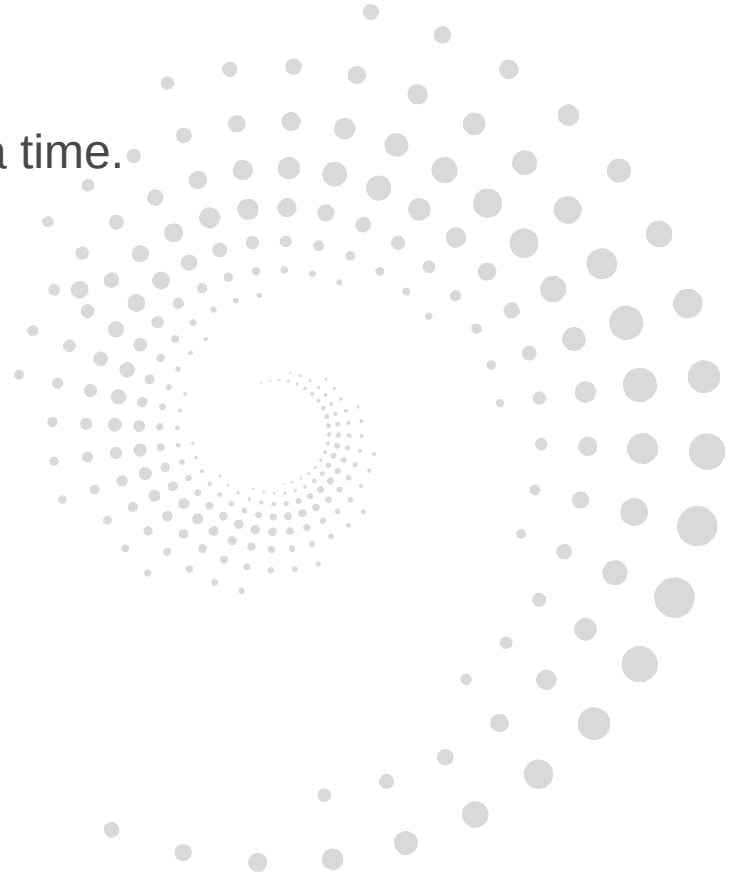
- **Options:**
  - **-a** to change the access time
  - **-m** to change the modification time
  - **-c** no create if not exists

- **touch <file>** will change the time of change of the file if the file exists

- If the file does not exist, it will create a file of zero byte size.

General Purpose Utilities

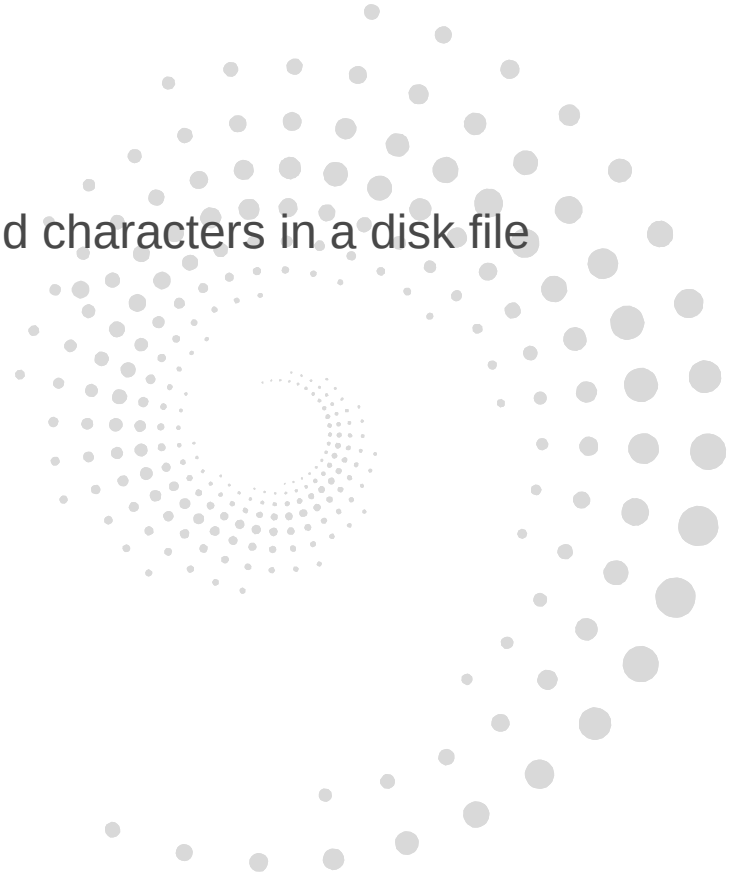
- **more**
  - Allows user to view one page-full of information at a time.
- **file**
  - Used to display the type of the file
  - Eg: $file a
  - A is ACII file
  - $file abc
  - Abc is directory
- **tty**
  - Prints the terminal's name



General Purpose Utilities

- **Wc**

  - A filter used to count the number of lines, words, and characters in a disk file or from the standard input.
  - -l - displays the number of lines
  - -w - displays the number of words
  - -c - displays the number of characters



General Purpose Utilities

- **cmp**
  - Returns the offset and the line number of the first position where the two files differ.

GENERAL PURPOSE UTILITIES

- **Diff**

  - Indicate the differences between the files
  - **a** Lines added
  - **d** Lines deleted
  - **c** Lines changed

FIND

- Lets user to search set of files and directories based on various criteria
- **Syntax: find [path...] [expression]**
- **[path]**
  - where to search
- **[expression]**
  - What type of file to search (specified with –type option)
  - What action to be applied (–exec, –print, etc.)
  - Name of the files (specified as part of –name option, enclosed in “ “)
- **Example**
  - find . –name “*.c” -print

lists all files with .c extension from the current dir & its subdirectories

FIND

- Finding files on the basis of file size
  - – size [+ –]n[bc]

n represents size in bytes (c) or blocks (b) of 512 bytes

find . –size 1000c	lists all files that are exactly 1000 bytes in size
find . –size +1000c	lists all files that are more than 1000 bytes in size
find . –size –1000c	lists all files that are less than 1000 bytes in size

FIND

- Finding files on the basis of access time (atime) or modified time (mtime)
  - – atime [+-]n
  - – mtime [+-]n

n represents number of days (actually 24 * n hours)

find . –atime 2	lists files accessed exactly 2 days ago
find . –atime +2	lists files accessed more than 2 days ago
find / –mtime –2	lists files modified less than 2 days ago

PR

- pr
  - Used to display a file in a format to be printed.
  - Breaks up a file into pages with a header, text and footer area

- **Options**
  - -l to alter the length of the file
  - -h to set the header
  - -t to suppress the header and the footer
  - -n to set the line number

STANDARD FILES

- Standard Input file
  - Keyboard, file descriptor is 0
- Standard Output file
  - Monitor, file descriptor is 1
- Standard Error file
  - Monitor, file descriptor is 2

I/O Redirection

< file	redirect standard input from file
> file	redirect standard output to file
2> file	redirect standard error to file
2>&1	merge standard error with standard output

$ **cat > abc**

$ **ls –l > outfile**

$ **cat xyz abc > outfile 2> errfile**

$ **cat xyz abc > outfile 2>&1**

FILTERS

- Filters are programs that takes its input from the standard input file, process it, and sends it to the standard output file.

- Commonly used filter commands
  - sort
  - grep
  - cut
  - head
  - tail
  - paste

FILTERS

Sorts the contents of the given file based on the first char of each line.

-n	numeric sort (comparison made according to strings numeric value)
-r	reverse sort
-t	specify delimiter for fields
+num	specify sorting field numbers
+num [-num]	to specify the range
-k n	– specified field on which we want sort n represents column number
-u	it supresses all duplicate lines (it is like distinct clause)

EXAMPLES

Eg: $ sort -k 5 -t"|" emp.dat – sorts on 5th column delimated by “|”

$ sort –u emp.dat – supreses duplicate lines

$ sort -k 5 -t"|" -r emp.dat – Sorts on 5th column delimated by “|” in reverse order

QUIZ

1.Which command is used to extract specific columns from the file?

A. cat

B. cut

C. grep

D. paste

E. None of the above

Option B

2.The field separator in cut command is specified with

A. -a option
B. -d option
C. -r option
D. -x option

Option B

QUIZ

3.Which option will be used with touch command to change the access time?

Option A

A. -a

B. -b

C. -t

D. -h

4.Which command is used to concatenate all files beginning with the string 'emp' and followed by a non-numeric characters?

Option B

A. more [emp][!0-9]
B. cat emp[!0-9]
C. cat emp[x-z]
D. cat emp[a-z]

GREP

- **grep** -Global Regular Expression Printer is used for searching regular expressions

- **Syntax:**
  - grep <options> <pattern> <filename(s)>

GREP OPTIONS

- **-c** displays count of the number of occurrences
- **-n** displays line numbers along with the lines
- **-v** displays all lines except lines matching pattern
- **-i** Ignores case for matching

EXAMPLES

- **$ grep -c "Desai" emp.dat – Count number of lines in which**
  Pattern appears
- **]$ grep -ni "desai" emp.dat – Prints along with line number**
  - 1:Desai |Analyst |5600 |01/09/06 |Pune
  - 7:Desai |Analyst |5600 |01/09/06 |Pune
- **grep -vni "desai" emp.dat**
  - **Displays all lines except matching pattern**

2:Patil |Consultant |6700 |09/09/30 |Hydrabad

3:Anant |Project Mgr |8900 |01/04/06 |Pune

4:Ranad |Programmer |2000 |04/04/06 |Pune

5:John |Analyst |8000 |03/06/87 |Banglore

6:Scott |Programmer |8900 |04/02/90 |Pune

PATTERNS

- ***** - matches 0 or more characters
- **[^pqr]** - Matches a single character which is **not p** ,**q** or **r**
- **^pqr** -Matches **pqr** at the beginning of the line
- **pqr$** -Matches pqr at the end of the line
- "**.**" - Matches any one character
- **** - ignores the special meaning. grep "New\[abc\]" filename

FILTER COMMAND - HEAD

Displays the first **n** lines of the file

- $ head -3 file1

FILTER COMMAND - TAIL

Displays the last n lines of a file

$ tail -3 file1

Can also specify the line number from which the data has to be displayed till the end of file

$ tail +5 file1

FILTER COMMAND - TR

tr - translate filter used to translate a given set of characters.

Example :

tr [a-z] [A-Z] < filename

This converts standard input read from lower case to upper case.

option **-s** can be used to squeeze the repeated characters.

Eg: $ tr "A-Z" "a-z" < emp.dat – translates all upper to lower case

$ tr "|" "~" < emp.dat

FILTER COMMAND - TR

Useful options for tr

- **-s char**

  Squeeze multiple contiguous occurrences of the character into single char

- **-d char**

  Remove the character

  $ tr -d "D" < emp.dat – Remove D char from file wherever it occurs

COMMAND PIPING

- Allows the output (only the standard output) of a command to be sent as input to another command.
- Multiple pipes may appear in one command line.

Example:

- $ cat * **|** wc

$ cat fil1 **|** head **|** wc -l

- ***Eg:$ cat emp.dat |head -6|wc -l***
- ***6***

FILTER COMMAND – TEE

- **tee** command allows the normal output to the standard output, as well as to a file
- Useful to capture intermediate output of a long command pipeline for further processing, or debugging purpose.

- Example
  - who | tee userlist
  - cat - | tee file1 | wc –l

Eg: $ ls -l |tee filelst|wc -l |tee knt

– sends list of files to **filelst** and counts number lines stores in **knt** file plus displays on std output device

FILTER COMMAND – CUT

Used to extract specified columns of a text

Option remark:

-c	used to extract characters
-d	Delimiter for fields
-f	Field no.

Examples:

$ cut -c2-5 file1 – displays from 2nd char to 5 from file1

Eg: $ cut -d "|" -f2 emp.dat – displays contents of 2nd column which is separated by "|"

FILTER COMMAND – PASTE

Paste is used to fix two cut portions of the file vertically

-s Pastes the contents of file2 below file1

-d Specify delimiter

$ paste **-d”|”** file1 file2

Eg: $ paste -d "|" -s a.dat b.dat – displays one after another

EXAMPLES

$ cut –d"|" –f1,2 emp.dat > a.dat

Desai |Analyst

Patil |Consultant

Anant |Project Mgr

$ cut -d"|" -f3,5 emp.dat > b.dat

5600 |Pune

6700 |Hydrabad

8900 |Pune

$ paste -d "|" a.dat b.dat

Desai |Analyst |5600 |Pune

Patil |Consultant |6700 |Hydrabad

Anant |Project Mgr |8900 |Pune

COMPRESSION UTILITIES

gzip,

Usage is very similar to compress and pack utilities in **Unix**: **gzip** [-vc] filename

where -v displays the compression ratio.

-c sends the compressed output to standard output and leaves the original file intact.

Eg: $ gzip -vc emp.dat > temp.dat

emp.dat: 41.8%

gunzip

gunzip can uncompress files originally compressed with compress.

TAPE ARCHIVE - TAR

- Tar is an archiving utility to store and retrieve files from an archive, known as tarfile.
- Though archives are created on a tape, it is common to have them as disk files as well.
  - tar c|t|x [vf destination] source...

- **Options:**
  - c create new archive
  - t list contents of archive
  - x extract files from archive
  - r append files to the archive
  - d delete files from archive
  - –f filename name of the archive
  - v verbose listing

TAPE ARCHIVE - TAR

- Examples:

Create a new tar file containing all .dat files (assuming a.dat, b.dat and c.dat exist)

- **$ tar –cf mytar *.dat**
- *Eg. $ tar -cf backfls *.dat* – creates backfls which containts all dat files

- **$ tar -tvf backfls – lists all files from archive file**
  - -rw-rw-r-- slclab/slclab 125 2006-09-12 10:13:01 a.dat
  - -rw-rw-r-- slclab/slclab 112 2006-09-12 10:13:15 b.dat
  - -rw-rw-r-- slclab/slclab 280 2006-09-11 17:07:19 emp.dat
  - -rw-rw-r-- slclab/slclab 210 2006-09-12 10:07:39 final.dat
  - -rw-rw-r-- slclab/slclab 189 2006-09-12 10:26:41 temp.dat

EXAMPLE

- $ tar -xwf backfls – To extract files from archive files

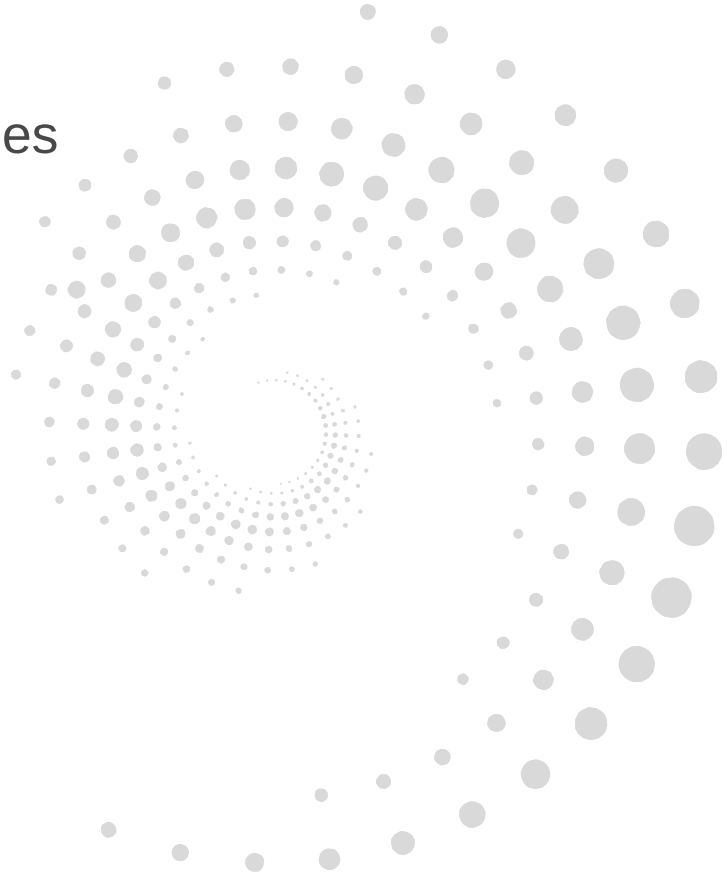
  extract `a.dat'?y

  extract `b.dat'?n

  extract `emp.dat'?n

  extract `final.dat'?n

  extract `temp.dat'?n



Q & A

QUIZ

1. Which symbol will be used with grep command to match the pattern pat at the beginning of a line?

A. ^pat

B. $pat

C. pat$

D. pat^

E. None of the above

Option A

2. Which command is used to change protection mode of files starting with the string emp and ending with 1,2, or 3?

A. chmod u+x emp[l-3]

B. chmod 777 emp*

C. chmod u+r ??? emp

D. chmod 222 emp?

E. None of the above

Option A

QUIZ

Which command in UNIX is used for description of any command ?

a) help

b) man

c) detail

d) shortdetal

Option b

Which command is used to extract specific columns from the file?

a) cat

b)cut

c)grep

d)paste

Option b

SUMMARY

- **In this session, you have learned to:**
  - use the Unix Utilities like

    cat, echo, touch, more, file, wc, cmp, comm, find
  - employ redirection operators
  - use filters like

    sort, grep, cut, head, tail, tr, and paste
  - backup commands

    zip/gzip and tar

Capgemini

CONSULTING.TECHNOLOGY.OUTSOURCING

People matter, results count.

About Capgemini

With more than 125,000 people in 44 countries, Capgemini is one of the world's foremost providers of consulting, technology and outsourcing services. The Group reported 2012 global revenues of EUR 10.3 billion.

Together with its clients, Capgemini creates and delivers business and technology solutions that fit their needs and drive the results they want. A deeply multicultural organization, Capgemini has developed its own way of working, the Collaborative Business Experience™, and draws on Rightshore®, its worldwide delivery model

www.capgemini.com/bim