TEKNOTURF

INTRODUCTION TO

UNIX

OPERATING SYSTEM

In this Module you will learn

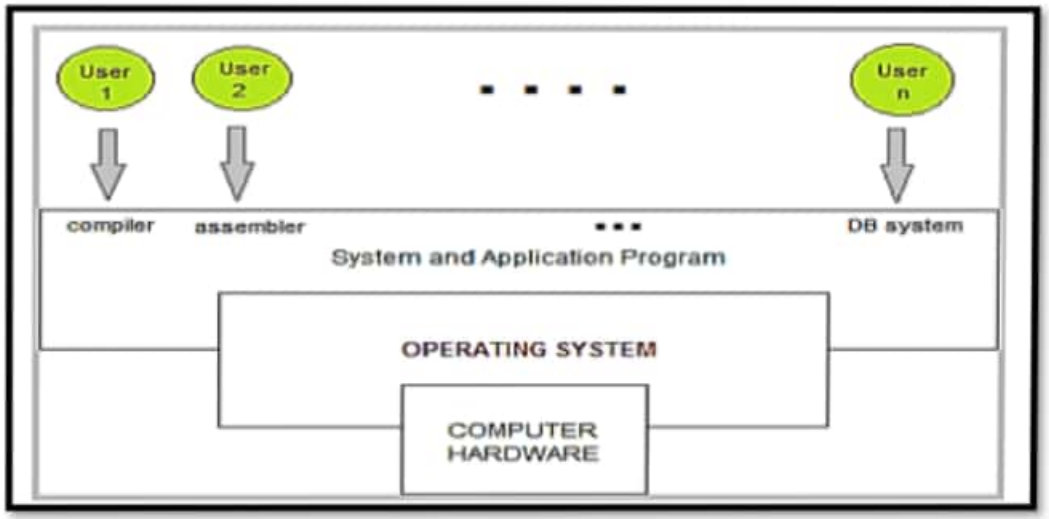
TEKNOTURF

- Introduction to OS
- UNIX Fundamentals
- Basic Commands
  - date
  - passwd
  - cal
  - bc ,dc
  - who , whoami, who am I
  - tty , finger
  - man , info
  - df

Introduction to OS

TEKNOTURF

Operating systems (OS) provide the interface between a computer hardware and the applications that run on it



UNIX Fundamentals

TEKNOTURF

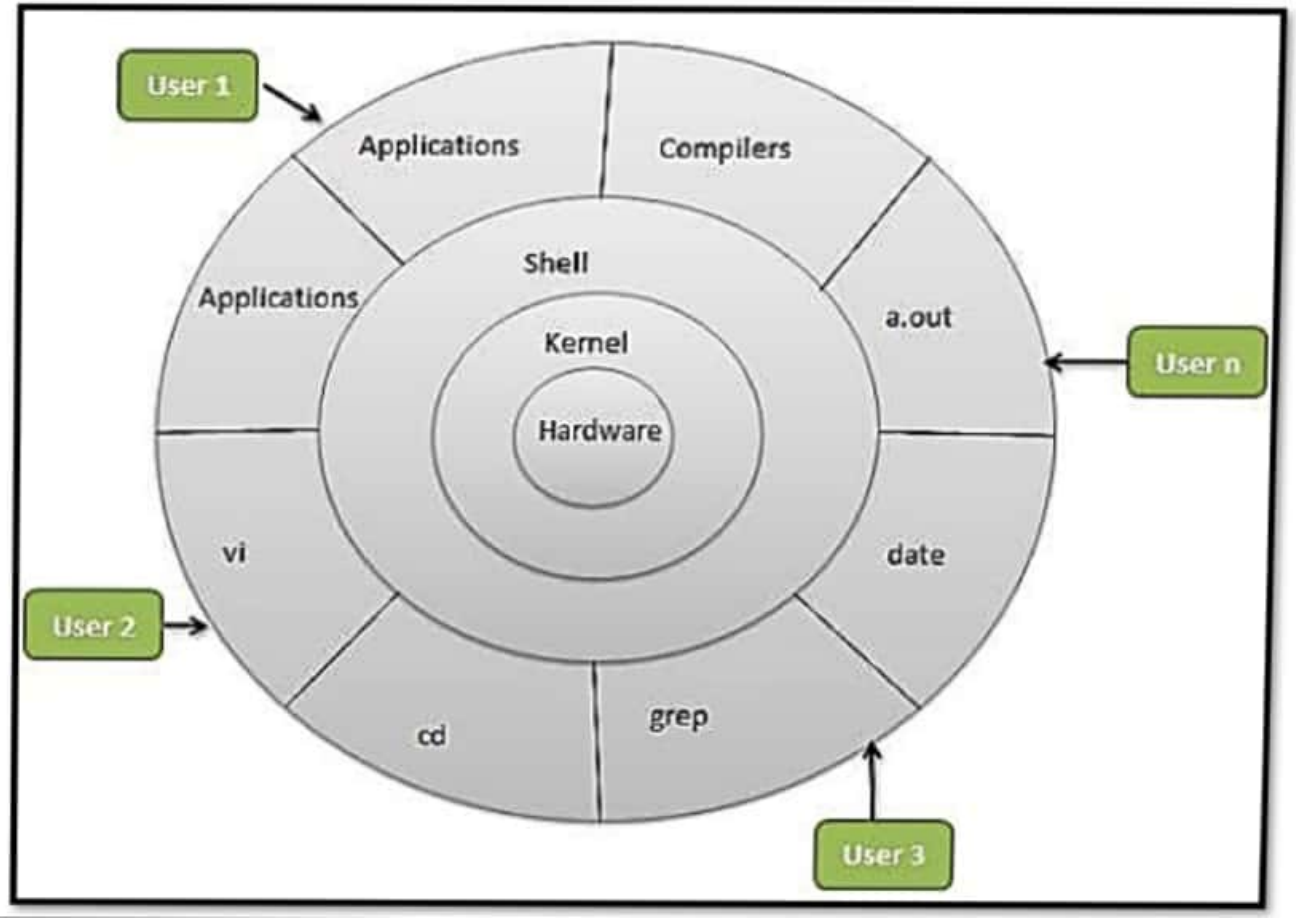
The UNIX System

- UNIX is a multi tasking and multi user Operating system
- It is more secured
- UNIX is distributed freely
- It is an open source software
- UNIX follows “Console - Terminal” architecture
  - Console has the processing capability and many ports
  - Terminals have monitors and keyboards but no processing capabilities
  - The terminals are connected to the port of the console

UNIX Architecture

TEKNOTURF

kernel- shell architecture



Kernel

TEKNOTURF

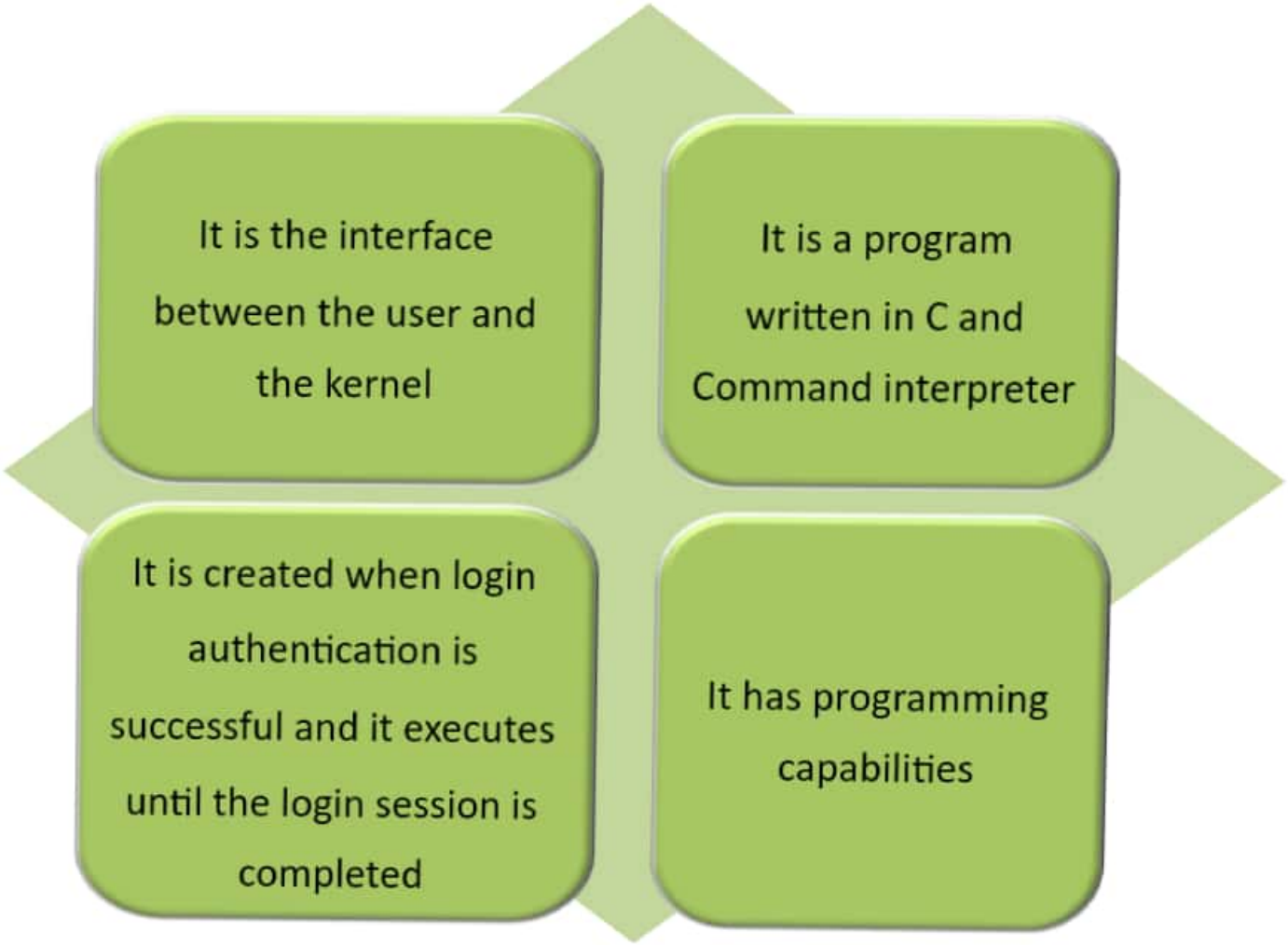
- It is the core of the operating system. It is a collection of routines mostly written in C, which is loaded into memory when system is booted
- It communicates directly with the hardware
- User programs that need to access the resources like hard disk or terminal uses the service provided by the kernel through functions called system calls
- The kernel in turn communicates with the hardware

Features of Kernel

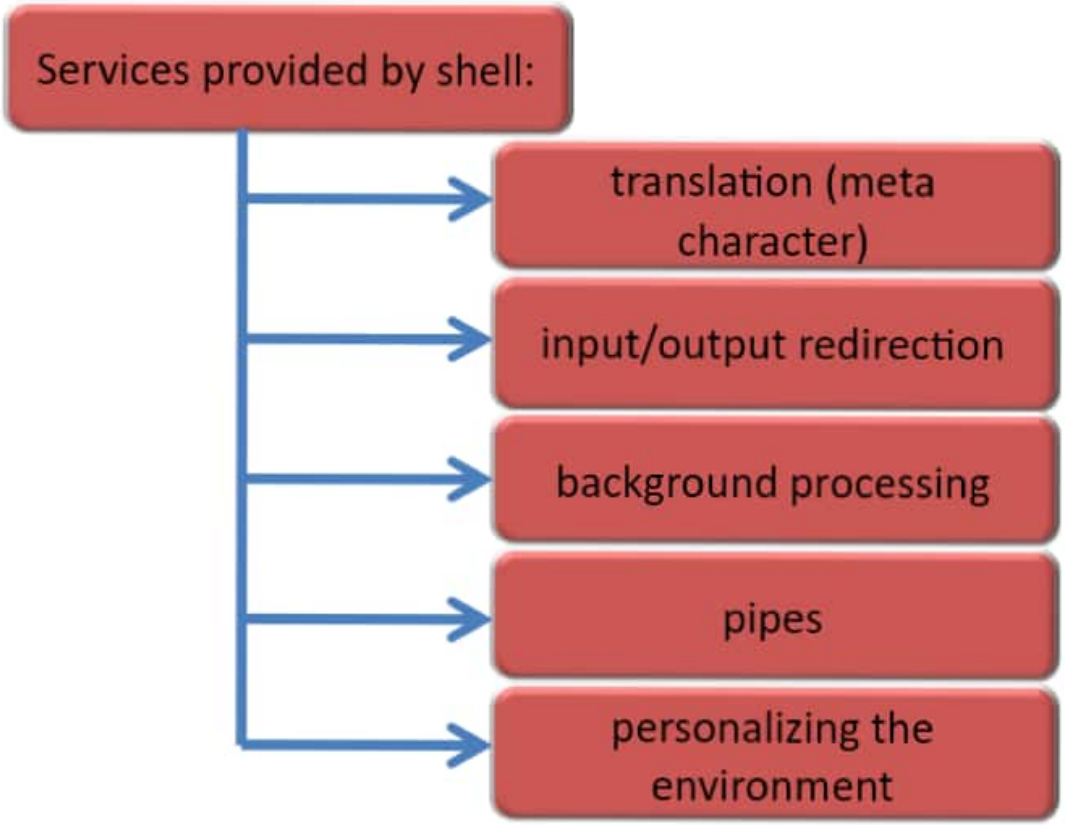
TEKNOTURF

- Manages the files on the disk
- It manages processes(like scheduling, creation, termination) that may run on the system
- It performs the memory management
- It manages the system security
- It manages the network
- It manages the devices like I/O devices i.e. the monitor, keyboard, printer

Shell



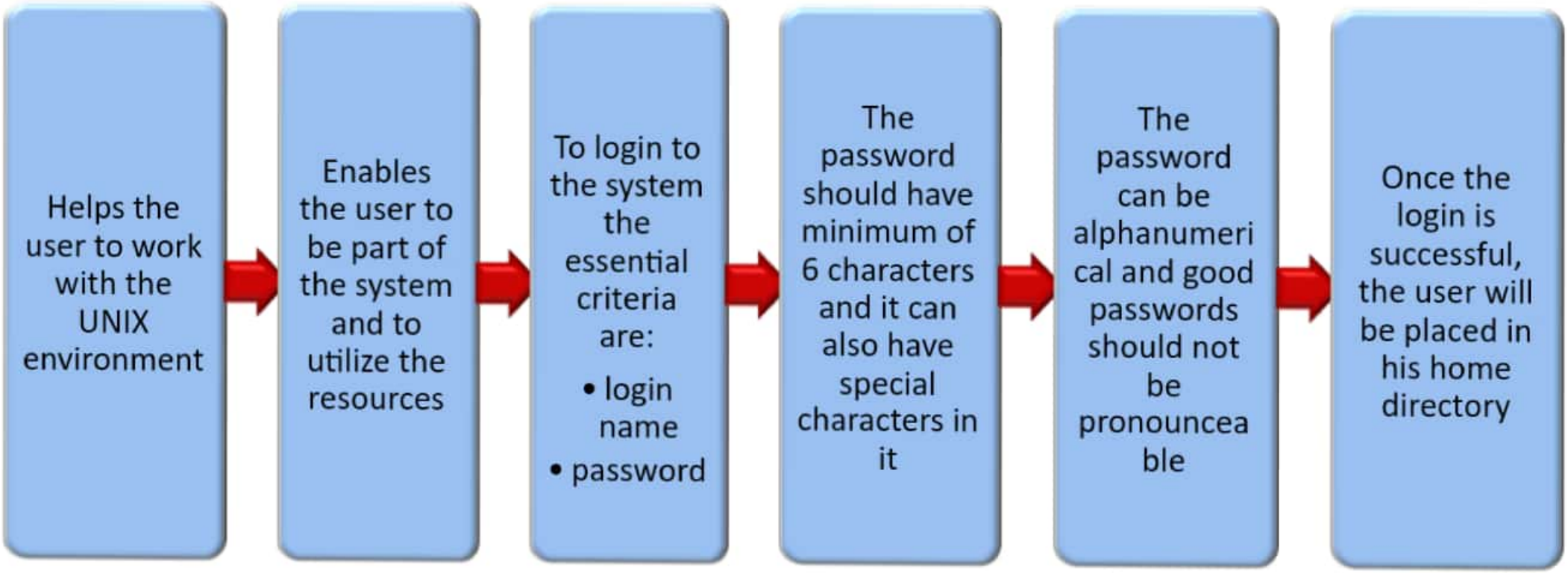
Shell



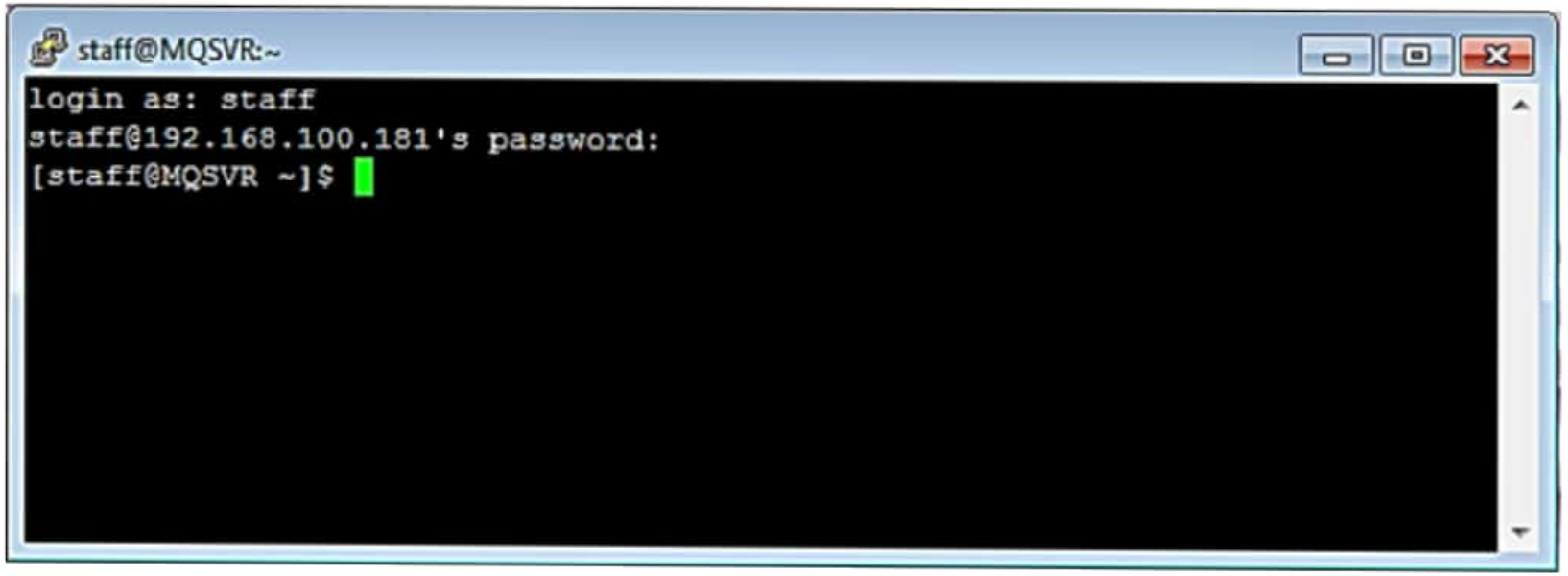
Logging In and Out

TEKNOTURF

Login:



Login



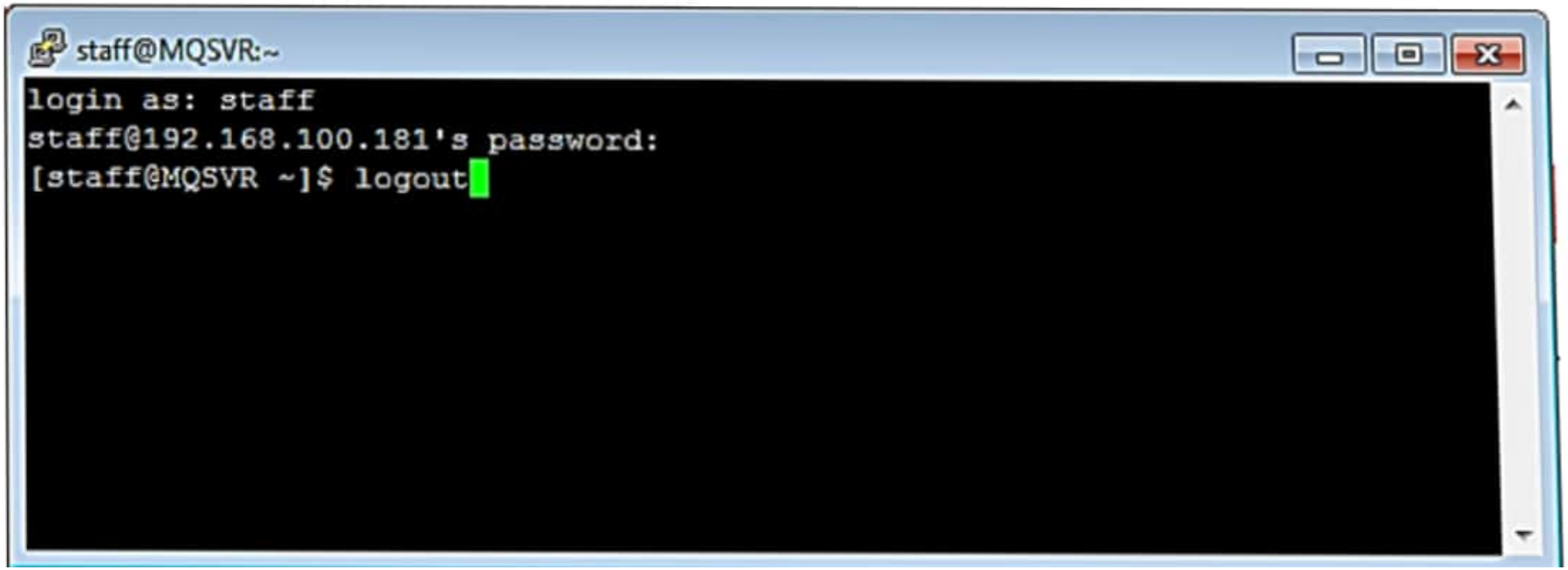
Once the login details are verified, the user is allowed to work in the system. When logged in, the user is presented with a prompt($ or #), which is the shell's way of requesting for a command

Logout

TEKNOTURF

Logout:

- **$** ctrl + d
- $ exit
- $ logout (any of the 3 ways can be used by the user to log out from the system)



Which will inform the kernel about the C program compilation command, which is executed in the prompt?

- [x] System Calls
- [ ] Shell Script
- [ ] Shell
- [ ] C compiler

Correct

That's right! You selected the correct response.

The _____ part of the operating system will interact directly with the computer hardware.

- ○ script
- ○ commands
- ○ shell
- ✔ ◉ kernel

Correct

That's right! You selected the correct response.

Command

TEKNOTURF

Command is a program that performs a particular task

Commands are case and space sensitive

syntax:

- command [option] [argument]

option:

- used to customize the output
- they are preceded with + or -

argument:

- These are inputs given to the command

Command

TEKNOTURF

Command is a program that performs a particular task

Commands are case and space sensitive

syntax:

- command [option] [argument]

option:

- used to customize the output
- they are preceded with + or -

argument:

- These are inputs given to the command

Basic Commands

TEKNOTURF

date

- used to print or set the system date and time

syntax :

- date [-option] [+format] [argument]

example:

- $date prints the current date and time
  - Mon Nov 20 13:40:03 IST 2017
- $ date +%d
  - 06

Options and format to explore:

- -r, --date, %b, %B, %d, %D, %m, %F, %H

Basic Commands

TEKNOTURF

passwd

used to set or change the password of a user

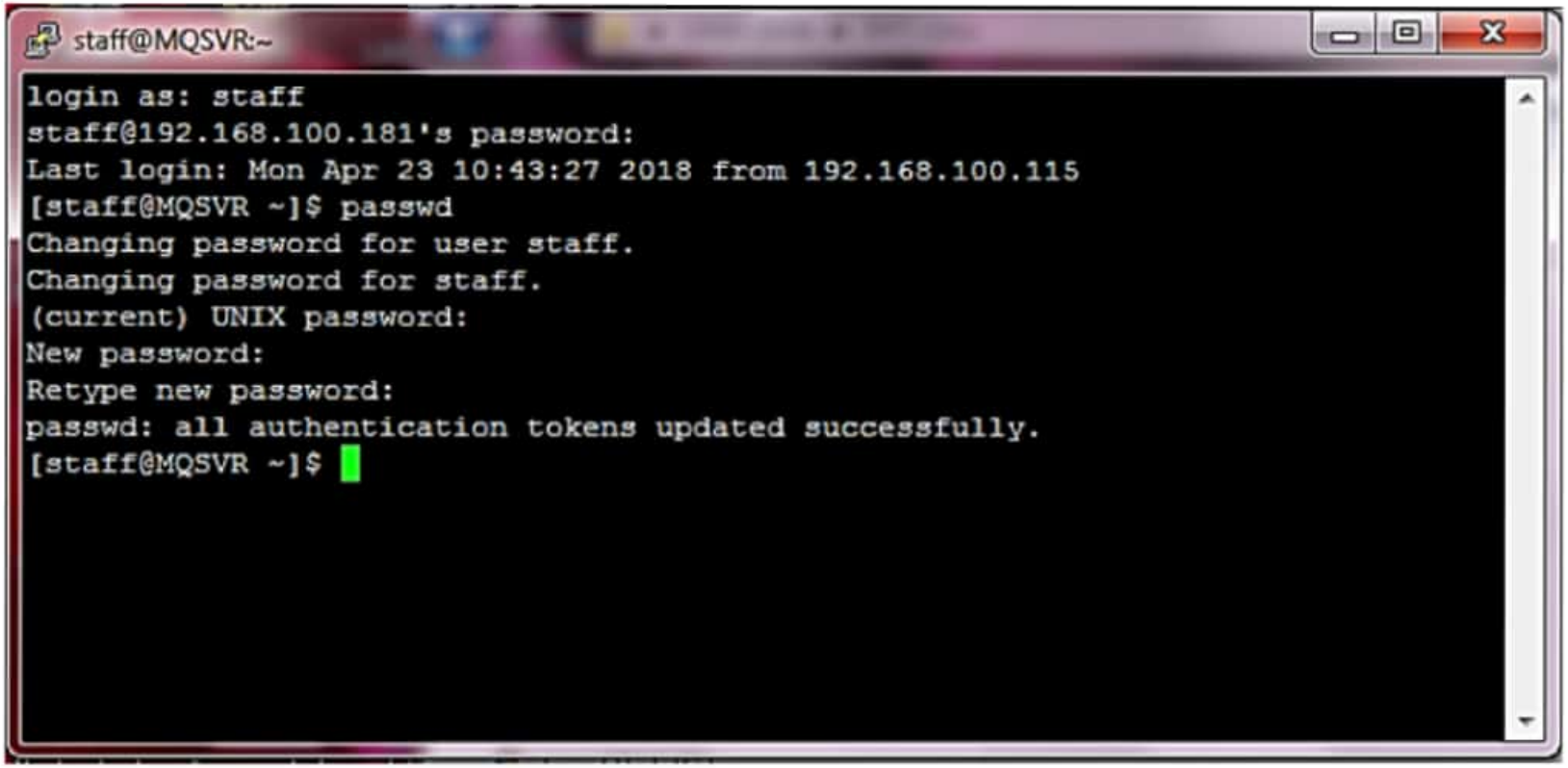
Syntax:

$ passwd

Basic Commands

TEKNOTURF

Example for setting a new password



Basic Commands

TEKNOTURF

cal

displays the calendar of specific month or year

Syntax

cal [-smjy13] [[month] year]

example:

$ cal (displays the current month's calendar)

Basic Commands

TEKNOTURF

$ cal 2000

- displays the year 2000 calendar. When one argument is given it is treated as the year

$ cal 7 2009

- displays the calendar of month July of the year 2009

$ cal -m 7 2009

- Displays the calendar of month July of the year 2009 having Monday as the first day of the week
  - -m is the option and 7, 2009 are arguments

options to explore:

- -s, -m, -j, -y, -1, -3

Basic Commands

TEKNOTURF

bc

- basic calculator
- when given bc, the command expects the input from the keyboard and prints the result
- To quit from the bc command, ctrl+d or quit is used

Example

- $ bc
- 20 + 40
- 60
- ctrl+d
- $

Basic Commands

TEKNOTURF

dc

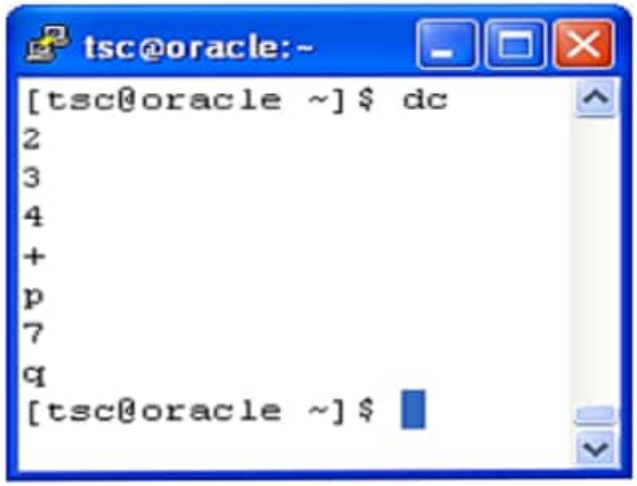
- dc is a reverse-polish desk calculator, which supports unlimited precision arithmetic
- A reverse-polish calculator stores numbers on a stack
- Entering a number pushes it on the stack
- Arithmetic operations pop arguments off the stack and push the results

options:

- p prints the top value of stack
- q quits from dc command

To explore:

- n, P, f,



Basic Commands

TEKNOTURF

who

- shows information about all the users who have logged in
- it displays the name, date, time and terminal number of a user
- **Syntax :**
  - $ who[option]
- **Example:**
  - $ who
    - user1 pts/6 Jul 20 12:27 (192.168.40.182)
    - tsc pts/2 Jul 20 17:48 (192.168.40.146)
- **options to explore**
  - -u, -q, -H

Basic Commands

finger

The finger displays information about the users who have logged in

It displays the Name, Tty, Idle, Login Time, Office, Office Phone

syntax:

$ finger

Example:

```
Login       Name       Tty      Idle  Login Time   Office     Office Phone
User1                  pts/6    1:49  Jul 20 12:27 (192.168.40.182)
tsc                    pts/2          Jul 20 17:48 (192.168.40.146)
```

Basic Commands

TEKNOTURF

whoami

Prints the user name associated with the current effective user id

Syntax

$ whoami

Example:

- $ whoami
- tsc

Basic Commands

TEKNOTURF

who am i

- displays the information about the current user
- displays the real user name, date, time and terminal number of a user

Syntax:

$who am i

Example:

- $ who am i
  tsc pts/5 Jul 6 14:04 (192.168.40.146)

Basic Commands

TEKNOTURF

tty

used to know the terminal number in which the user is connected

Syntax:

$ tty

Example:

- $ tty
- /dev/pts/5

Basic Commands

TEKNOTURF

df:

df displays the amount of disk space available on the Filesystem containing the given file name argument

If no file name is given, the space available on all currently mounted Filesystems is shown

Syntax :

$ df [options] [filename]

options to explore:

-H, -i

Basic Commands

TEKNOTURF

man

- It is an online manual page used as a help for the UNIX commands
- **syntax:**
  - $ man <command>

info

- It is another manual page for command help
- **syntax:**
  - $ info < command>

Review

TEKNOTURF

- cal - It displays calendar
- date - It displays current date and time in system
- clear - It clears the terminal screen
- df - It displays number of free disk blocks and files
- who am i -It gives information about currently logged in user
- whoami - It displays effective current user name
- who - It lists currently logged in users in a system
- man - It displays manual reference page for commands

What is the option to display the calendar using Sunday as the first day of the week.

- [x] -s
- [ ] -j
- [ ] -y
- [ ] -m

Correct

That's right! You selected the correct response.

Syntax of a Unix command is:

- command options arguments
- command options [arguments]
- ✔ command [+/-options] [arguments]
- command [+/-options] arguments

Correct

That's right! You selected the correct response.

In df command, -H is used to print the disk space size in human readable format.

- ○ True
- ✔ ◉ False

Correct

That's right! You selected the correct response.

Which command should be used to know more about the command "whoami"?

- whoami ?
- help whoami
- ✔ man whoami

Correct

That's right! You selected the correct response.

Summary

TEKNOTURF

- Introduction to OS
- UNIX Fundamentals
- Basic Commands

TEKNOTURF

THANK YOU

TEKNOTURF

FILE SYSTEM

Objective

TEKNOTURF

- File System
- File Types
- File Permissions
- File Commands

File System

TEKNOTURF

File System(Files and directories)

Directory commands

- mkdir, cd
- rmdir ls

File commands

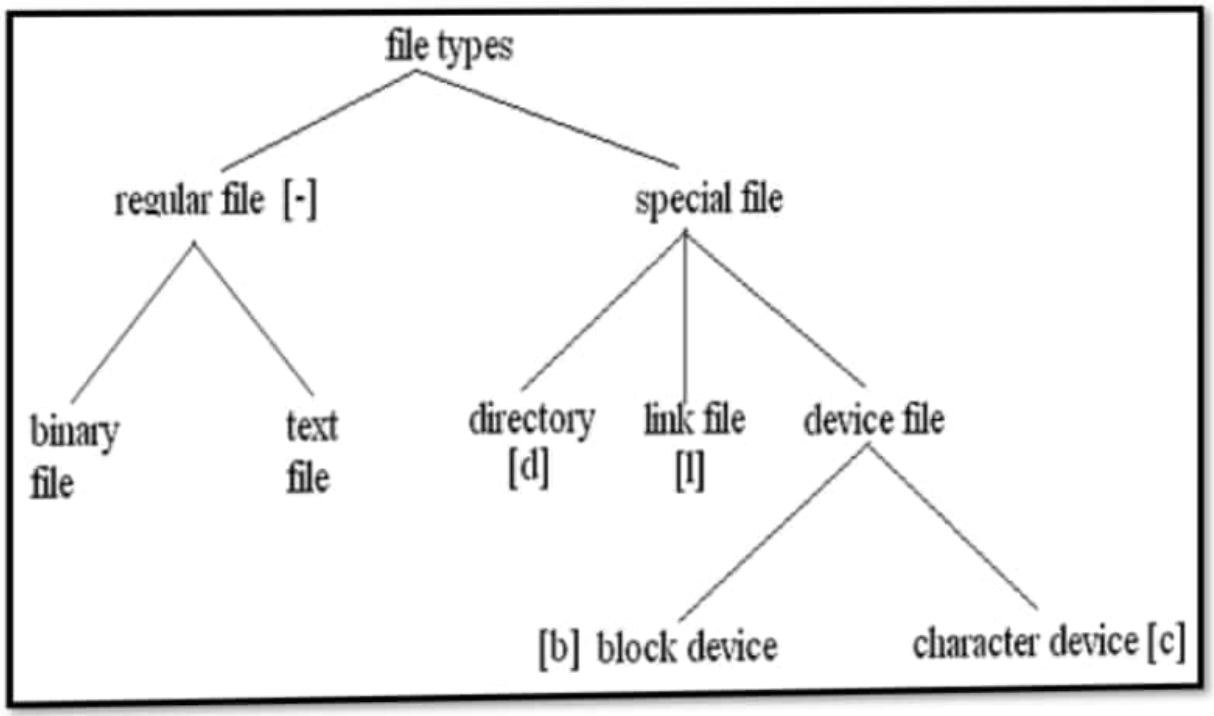
- touch , cat, vi editor
- rm ,cp, mv
- link

UNIX File System

TEKNOTURF

- **File System:**
  - It is the collection of files and directories organized in a hierarchical structure(inverted tree structure)
- **File:**
  - Container for storing information.
  - UNIX imposes no internal structure on a file's content. The user is free to structure and interpret the file content
    - Example: The employee data can be stored as
      - 1:amit:25000:developer (or)
      - 1 amit 25000 developer
  - UNIX treats directories (folder for storing filenames and other directory names) and devices(hard disk, memory, printer etc.) as files only

File types



File types

- These files can contain text data or binary data
- The contents of the text file are readable by the user
- The contents of the binary file are unreadable by the user and readable by the system or an application
- The binary files may be executed
- The files are denoted by (-)

Ordinary or regular files

- Contains the information about the files and the subdirectories
- It has no data
- The directory has 2 entries for each file or subdirectory it contains
  - filename
  - a unique number to identify the file in the hard disk (inode number)

Directory files

File types

TEKNOTURF

Device files

- Represents hardware devices and are present in the /dev directory
- *Character special files* or *character devices* relate to devices through which the system transmits data, one character at a time. Denoted by (c)
- *Block special files* or *block devices* correspond to devices through which the system moves data in the form of blocks These devices often represent addressable devices such as hard disk, drives. Denoted by (b)

File System Elements

TEKNOTURF

Rules for a file name

- Filename can consists up to 255 characters
- UNIX has no concept called file name extension
- It can have any ASCII character except the / and the NULL character
- The recommended characters are
  - alphabetic characters and numerals
  - period (.) , hyphen (-) and underscore (_)
  - can't have a file name having 2 consequent periods
- File names starting with (.) are hidden from the user
- The file names are case sensitive

Summary on File Types:

TEKNOTURF

1.Regular file

- binary file
- text file

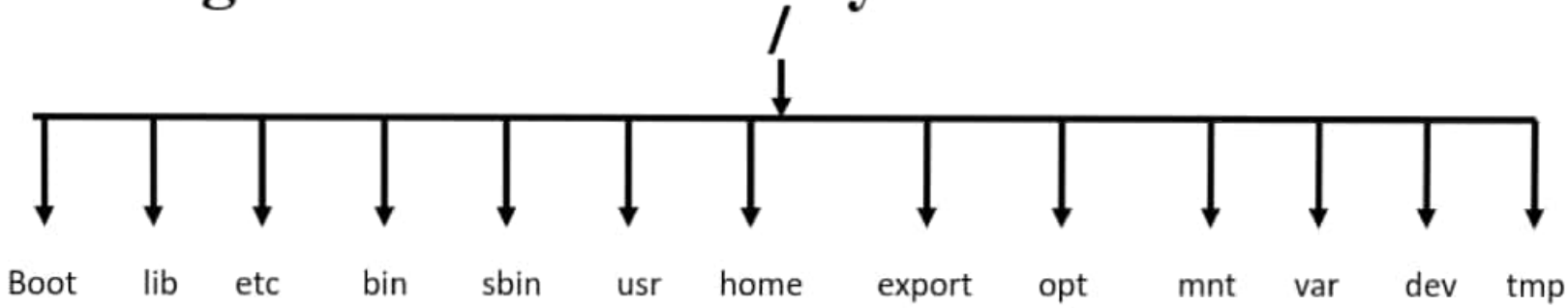
2.Special file

- Directory file
- Link file
- Device file
  - Block device file
  - Character device file

Symbols used to identify the file type

- (-) Regular file
- (d) Directory file
- (l) Link file
- (b) Block device file
- (c) Character device file

File System Single Rooted Hierarchy



/boot	-	Boot image of the kernel
/lib	-	Library files needed for the operating system and applications
/etc	-	Configuration files storage area of system and applications
/bin	-	Executables that can be executed by normal users
/sbin	-	Executables for super user (root)
/usr	-	Application's storage area
/home	-	Area of mount the remote home directories of users
/export	-	System's normal users home directory
/opt	-	optional (provides storage for large, static application software packages)
/mnt	-	Common mounting area
/var	-	Variables area (www, mail, log, etc…)
/dev	-	Device files directory
/tmp	-	System temporary area, Globally accessible

Special Files: DIRECTORY

TEKNOTURF

- NAME INODE NUMBER
- NAME is the file name or subdirectory name
- INODE NUMBER is the unique number given to the file or subdirectory in a given file system

Directory contents:

- An *inode* is a *structure* in a file system on UNIX operating systems that stores all the information about a file except its name and its actual data
- The inode table is a data structure that contains a list of inodes
- Inode will not contain inode number. It is just an index of the entry in the inode table

INODE:

INODE Entries

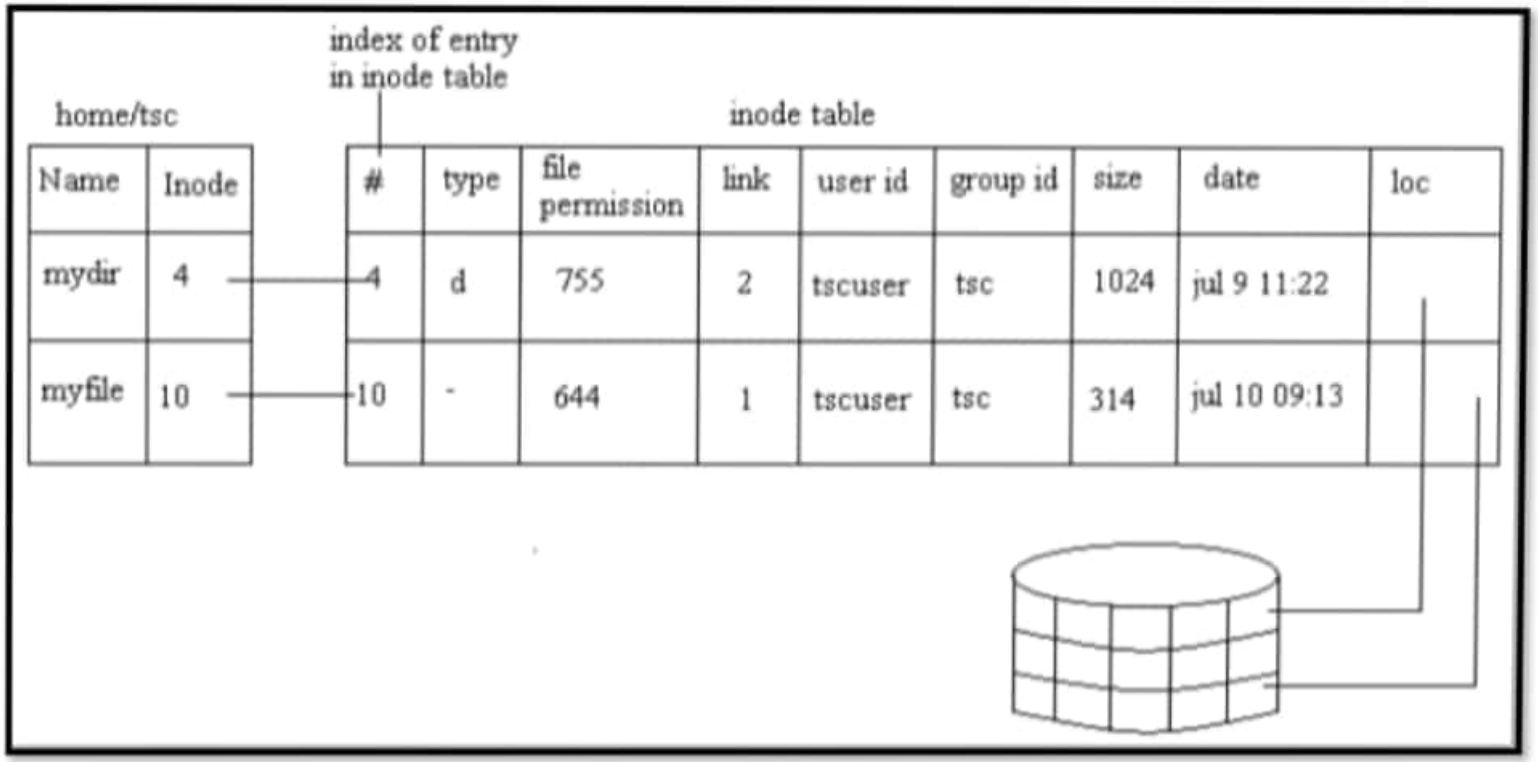
TEKNOTURF

Type	Type of the file (- d b l)
Permission	File Access Permissions
Ref. count	Hard link count
User	Owner of the file
Group	Primary group info
Size	size of the file
Disk Address	Location of a file in the hard disk
Time	created/modified/accessed time

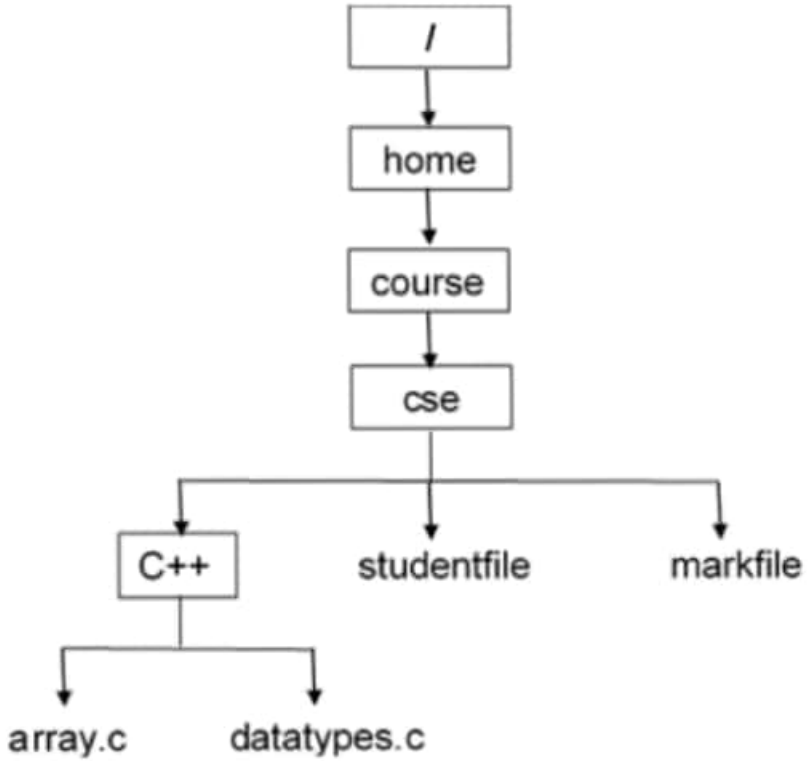
The part of the inode information can be viewed using the command

$ stat <filename>

Contents of inode



Sample Directory structure:



Path names

TEKNOTURF

Path:

- The sequence of file names, separated by / describes the path
- The path is used to locate the file in the system

Types of path:

- **Absolute/Full path:**
  - Starts from the root always (/)
  - e.g **/home/course/cse/C++**
- **Relative path:**
  - Starts from the current directory where the user is located
  - The path is said to be relative unless it starts with /
  - e.g: **./C++**

Where Am I

pwd

The Print Working Directory is used to print the current working directory of the user

Directory commands

TEKNOTURF

Creating a directory:

mkdir

- make directory
- **syntax:**
  - mkdir <directory name>
- **example:**
  - $mkdir sample

Directory commands

TEKNOTURF

Deleting a directory:

rmdir

- remove directory
- **syntax:**
  - rmdir <directory name>
- **example:**
  - $rmdir sample

constraints on removing a directory:

- The directory should be empty
- The directory name must not be the current working directory, and it is any parent directories name

Directory commands

TEKNOTURF

Traversing from one directory to other:

- **cd**
- change directory
- It changes the current directory to the directory specified as argument for the cd command

syntax:

- cd [<directory name>]

example:

- $cd sample (relative path) now the current working directory is sample
- $ cd /home/tech/cap/capuser/sample (full or absolute path)

Directory commands

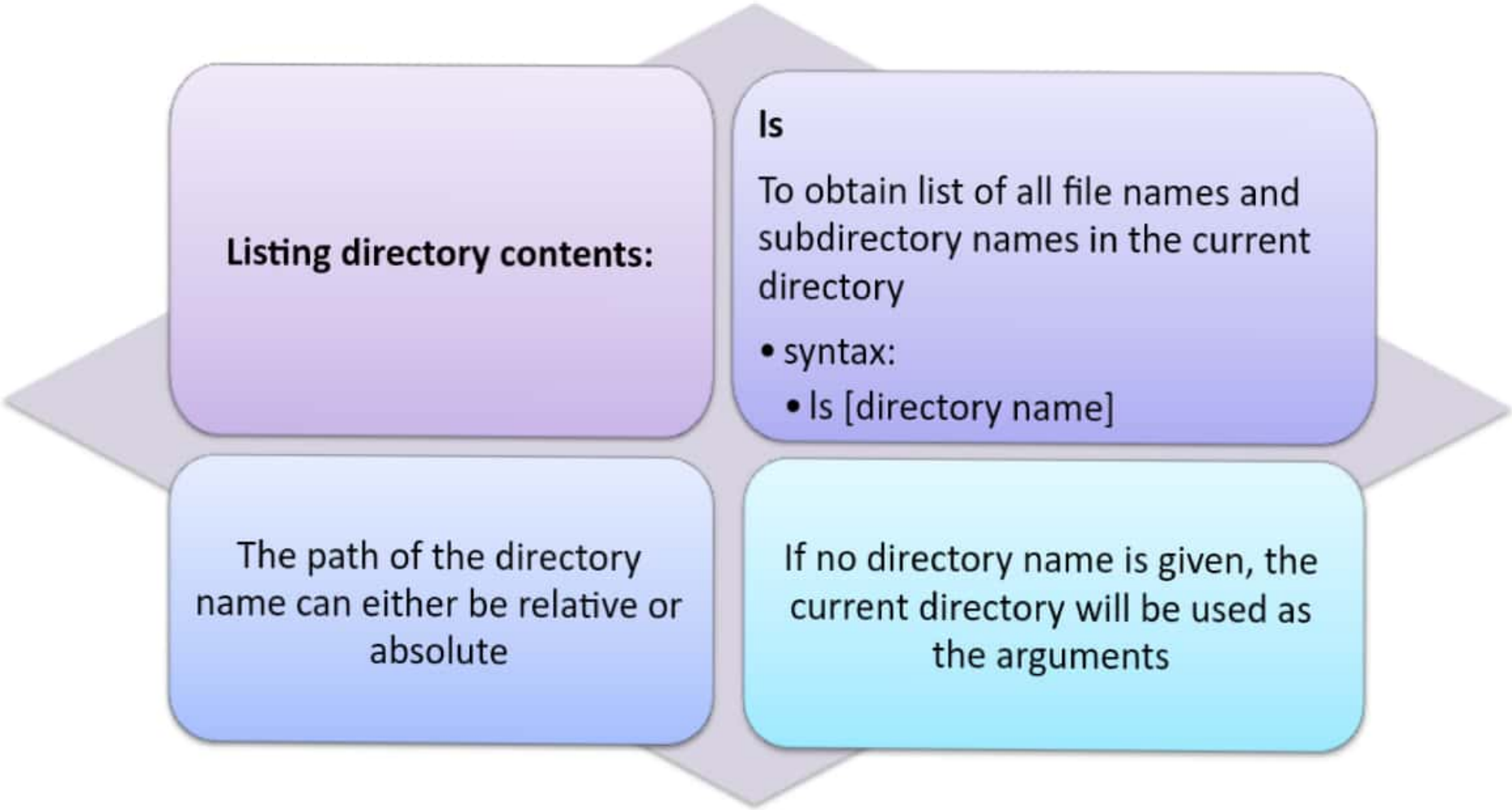
TEKNOTURF

Traversing from one directory to other:

The other ways of navigating across the directory :

- cd - takes the user to their home directory
- cd ~ - same as cd
- cd . - remains in the current directory - No change
- cd .. - takes the user one level above the current directory
- cd - - takes the user to the previous visited directory

Directory commands



Directory commands

Listing directory contents:

example:

1) $ ls

o/p: Cdir markfile studentfile

2) $ ls Cdir/

o/p: array.c datatype.c

options in ls command:

-a - list all the files including the hidden files

-l - long listing of the directory contents. Gives information about the file or directory like name, permission, size, date etc

-i - displays the inode number of the files

options to explore:

-d, -R, -r, -h

By default, the reference count for any regular file is 1

- [x] True
- [ ] False

Correct

That's right! You selected the correct response.

By default, the cp command will not copy the meta data of the file, but the mv command preserves the metadata.

- [x] True
- [ ] False

Correct

That's right! You selected the correct response.

Which directory in the file structure holds the printer details?

- /etc
- /printer
- ✔ /dev
- /lib

Correct

That's right! You selected the correct response.

File commands

creation of files:

cat:

used to create file and display the content of the file

- **creation:**
  - syntax:
    - cat > filename
    - when the command is given, the system waits for the user to enter the contents of the file
    - To save and exit from the command, ctrl+d is given
- **example:**

```
$ cat > samplefile
    hi this is a test file
    [ctrl+d]
$
```

File commands

TEKNOTURF

cat command is also used for concatenating files and printing the result on the standard output

syntax:

cat [<filename[s]>]

displays the content of the file, if the file already exists

Example:

$ cat samplefile

hi this is a test file

$ cat samplefile testfile

hi this is a test file

this is the content

(it prints the content of samplefile and then the testfile)

File commands

TEKNOTURF

deleting files

rm:

- removes files or directories
- syntax:
  - rm <filename[s]>

examples:

$ rm testfile -deletes the file

$ rm -r mydir -recursive delete (Deletes the entire subtrees along with the specified directory)

$ rm -i mydir -interactive deletion

$ rm -rf testfile -forceful deletion (Deletes the directory forcefully even if the directory is not empty)

File commands

TEKNOTURF

copying file:

cp:

- copies a file or a group of files
- It creates an exact image of the file on the disk
- It can have different names, if it is in the same directory
- same or different name in another location
- Has a different inode number

syntax:

- cp <source> <destination>
  - There can be more than one source file
  - If there is more than one source file, then the destination should be a directory

File commands

TEKNOTURF

cp:

example:

$ cp testfile sample (sample can be a file or directory)

$ cp testfile1 testfile2 sample (sample should be a directory)

options to explore:

-r, -i, -f

File commands

TEKNOTURF

moving files:

mv:

- moving a file/directory or a group of files/directories
- renaming file or directory

syntax:

- mv <source> <destination>
  - If there is more than one source, the destination should be a directory.

example:

- mv testfile sample

options to explore

- -i, -f

File commands

TEKNOTURF

linking files:

ln

make links between files

Types of links:

- **hard link**
  - alias name for the file
  - The inode number remains the same
  - The reference count of the file increases by 1
  - Only a single copy of the file is maintained, others are just references pointing to the original file

Restrictions in hard link:

- Can't be created for directories
- Can't be referred across file systems

File commands

TEKNOTURF

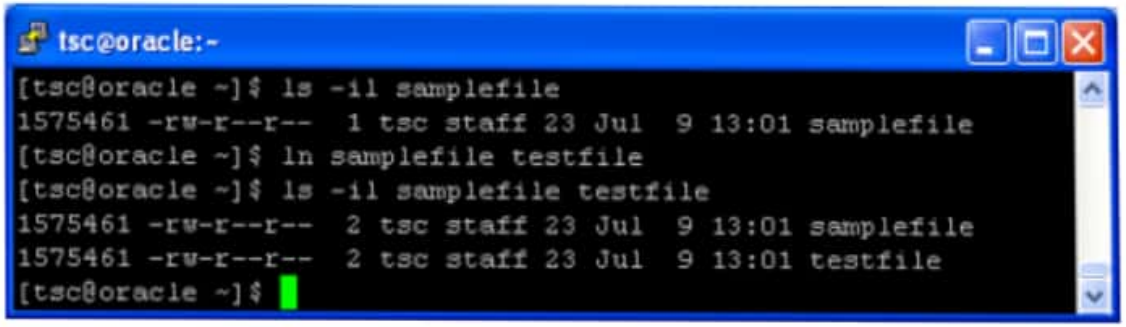
Hard link continuation

syntax:

- ln <sourcefile>[<destinationfile>]
  - If link name is not given then the source name is taken as the link name, if the link is not created in the same directory

example:

$ ln samplefile testfile



File commands

soft or symbolic link

- used for creating shortcuts for the files and directories
- Inode numbers are different
- The soft link file's content is only the path of the original file
- The type of the file for soft link is denoted by l

Syntax

- ln -s <sourcefile> <destinationfile>

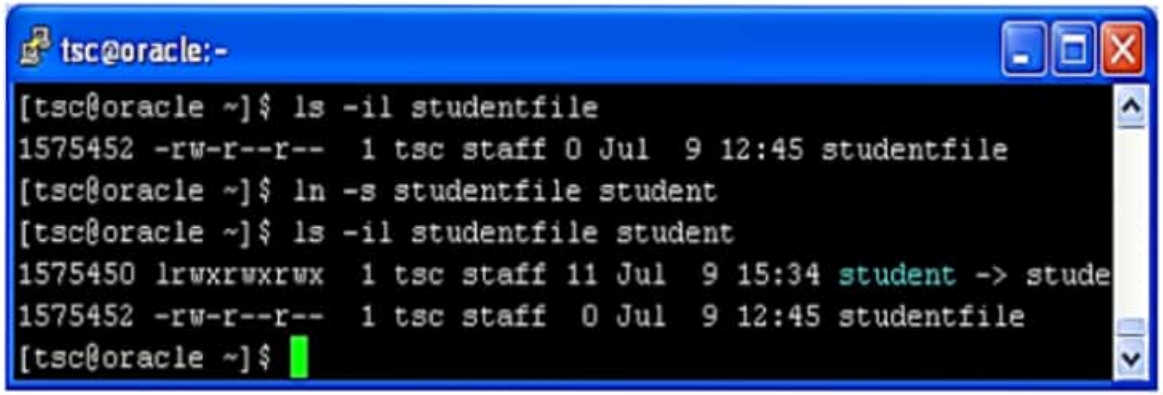
example:

- $ ln -s studentfile student

File commands

TEKNOTURF

Example of soft link



If a link is created using the command “ln myfile linkfile”, what will be the reference count for the files?

- 3
- ✔ 2
- 1
- 0

Correct

That's right! You selected the correct response.

Create a hard link “hlink.txt” and a soft link “slink.txt” for “myfile.txt”. Now delete the source file “myfile.txt”. What will happen to the link files?

- ○ able to access the file “slink.txt” but not “hlink.txt”
- ○ can able to access both the link files
- ○ can’t able to access both the link files
- ✔ ◉ able to access the file “hlink.txt” but not “slink.txt”

Correct

That's right! You selected the correct response.

If a link is created using the command “ln myfile linkfile”, what will be the reference count for the files?

- 3
- ✔ 2
- 1
- 0

Correct

That's right! You selected the correct response.

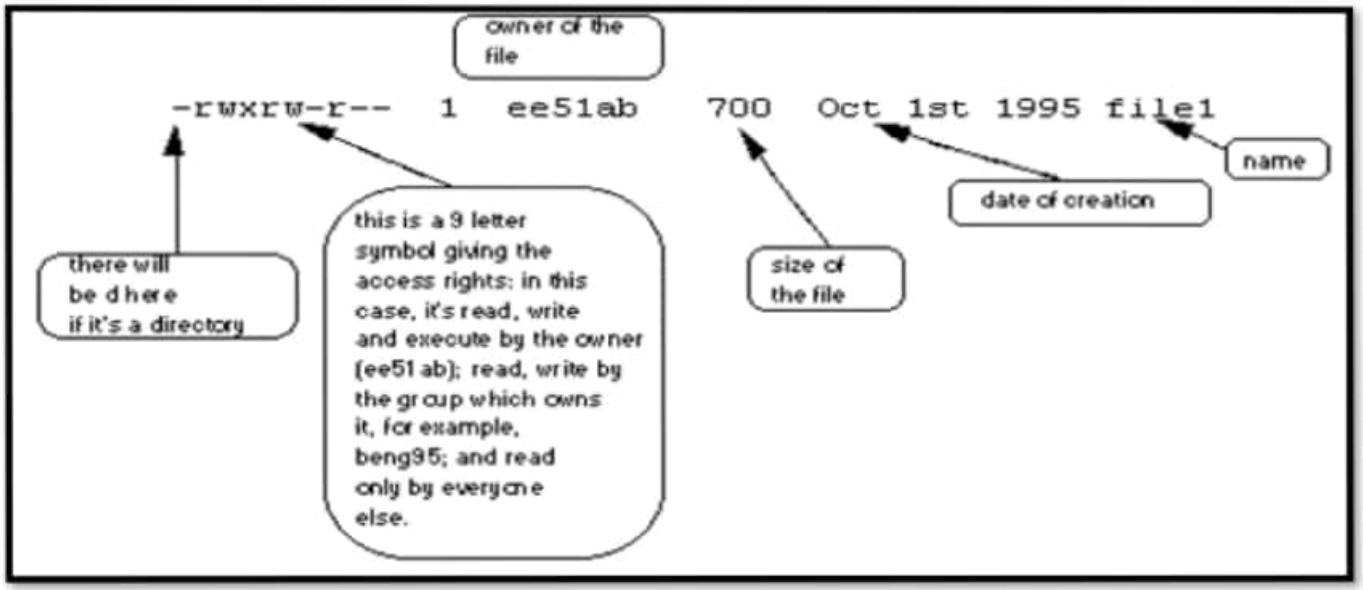
Permission

TEKNOTURF

Permission is used to describe the file's read, write and execute rights available to three categories of users – user, group and others

Permission can be altered only by the owner of the file with chmod

The permission of a file is displayed in the 1st column of the ls -l command



Permission continued ...

TEKNOTURF

Users of a file:

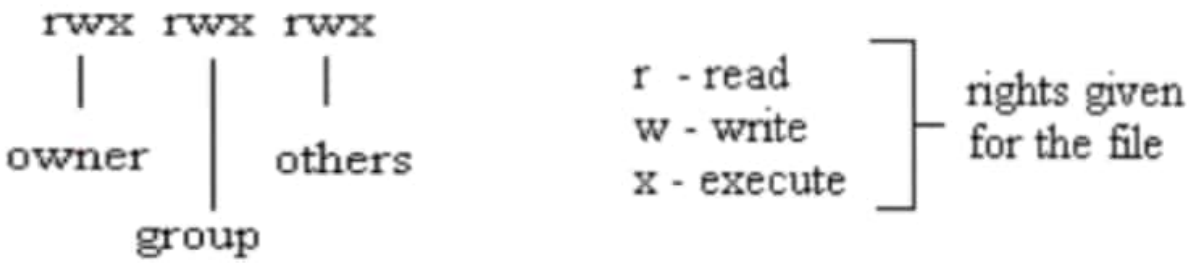
- owner:
  - Every file created in the UNIX system will have one user as a owner
- group:
  - Every user belongs to at least one primary group
- others:
  - The users who don't belong to the file's group and are not the owners are called others. Generally they are termed as public

id → Command displays the user and group id

```
$ id
uid=555(tsc) gid=503(staff) groups=503(staff)
```

Permission continued ...

chmod:
change file access permissions



syntax:
chmod <permission> <filename>

Permission continued ...

TEKNOTURF

Permission for file:

read: reads the contents of the file (example : cat, vi)
write: modifies the contents
execute: allows to execute the file if it is executable

Permission for directory:

read: reads the contents of directory (example : ls)
write: modifies the directory content (example: cp, mv, create and remove a file)
execute: open or enter the directory (example: cd)

Permission continued ...

TEKNOTURF

- **symbolic notation**
  - Used to assign or remove particular rights from a particular category of user
- **octal notation**
  - Need not know what the current permissions are
  - All the permissions for all the category of users should be explicitly set

Types of assigning permission

Permission continued ...

TEKNOTURF

symbolic notation:

- r read
- w write
- x execute
- u owner
- g group
- o others
- + add permission
- - remove permission
- = assign permission
- a assign permission to all the user (u g o)

Permission continued ...

TEKNOTURF

example:

- chmod u+x inputfile → (add execute permission to the owner)
- chmod a+w inputfile → (add write permission to all the users)
- chmod u+x, g-r inputfile → (add execute permission to the owner and remove read permission from the group)
- chmod go-rw inputfile → (remove the read and write permission from the group and others)
- chmod ugo=r inputfile → (assign read permission to all the users. After executing this command only, read permission will be present for all the users)

Permission continued ...

TEKNOTURF

- **octal notation:**

	binary notation	octal notation
read	100	4
write	010	2
execute	001	1

- To grant all the permissions to a category (read, write, execute) : 4+2+1 = 7

example:

- chmod 644 inputfile (644 → 6 for user, 4 for group, 4 for others)
- chmod 622 inputfile

Permission continued ...

TEKNOTURF

umask

- umask stands for user mask
- umask tells the restricted permission for each user
- the default umask value is 022
- if the umask value is 022 then
  - the default file permission is 644
  - the default directory permission is 755
- Every file or directory created will have this default permission
- The maximum default file permission is 666
- The maximum default directory permission is 777

What is the bare minimum permission required to get into a directory?

- [x] execute
- [ ] read
- [ ] write

Correct

That's right! You selected the correct response.

What is the bare minimum permission required to get into a directory?

- [x] execute
- [ ] read
- [ ] write

Correct

That's right! You selected the correct response.

If the umask value is set to 0002, then what will be the default permissions set for the files?

- 773
- ✔ 664
- 777
- 666

Correct

That's right! You selected the correct response.

If the umask value is set to 0002, then what will be the default permissions set for the files?

- 773
- ✔ 664
- 777
- 666

Correct

That's right! You selected the correct response.

What would happen if we try to remove a read only file, rm filename?

- file cannot be deleted
- Error in command
- ✔ file will be deleted with confirmation - interactively
- file will be deleted without interaction

Correct

That's right! You selected the correct response.

Summary

TEKNOTURF

- File System
- File Types
- File Permissions
- File Commands

TEKNOTURF

THANK YOU

TEKNOTURF

FILTERS

OBJECTIVE

TEKNOTURF

- Pipes
  - tee
- Filter
  - Simple Filters
    - grep
    - cut
  - Advanced Filters
    - awk
    - sed

PIPES

TEKNOTURF

- Used to combine two or more commands
- The commands are separated by a vertical bar (|)
- The standard output of one command is sent to the standard input of another command
- No intermediate outputs can be viewed
- If error occurs in the 1st command, then the error is sent to the error stream instead of sending it to the next command

Pipes

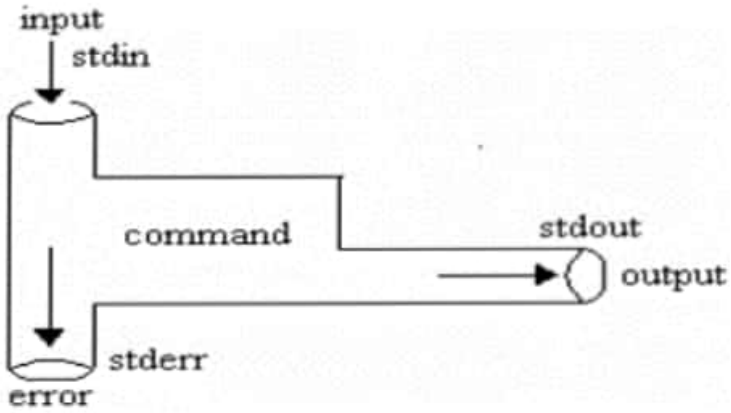
TEKNOTURF

syntax:

command1 | command2

example:

$ ls -l | wc -l



The output of ls -l is sent as input to the wc command and the number of lines is displayed in the output stream

tee

TEKNOTURF

- In pipe the intermediate output can't be viewed
- tee is a command which saves the output in a file as well as writes the output to the standard output

example:

$ who | tee userlist.txt

The tee command displays the output of who in the monitor and also saves the output in userlist.txt

$ who | tee /dev/tty | wc -l

This command will display the output of the who command in the screen and counts the number of lines with the wc command and displays the count also

the tee command should have the argument which is the file name.

/dev/tty – it is a special file which indicates the user's terminal

Predict the output of this command, who | wc -l .

- [x] It displays the number of users logged in
- [ ] Displays an error message
- [ ] It displays the number of characters in logged in list
- [ ] It displays the users who are all logged in

Correct

That's right! You selected the correct response.

In the following example, the option “-i” is used for the case insensitive search

```
$ls -l | grep -i "carol.*aug"
```

- [x] True
- [ ] False

Correct

That's right! You selected the correct response.

Filters

TEKNOTURF

A command is referred to as a filter if it can read the input, alter it in some way, and write its output to standard output stream

When a program performs operations on input and writes the result to the standard output, it is called a filter. One of the most common uses of filters is to restructure output

UNIX has a large number of filters. Some useful ones are the commands

- awk
- grep
- sed

Simple Filters

TEKNOTURF

grep:

- used to match patterns in a file
- used for searching file
- scans its input for a pattern and displays lines containing the pattern
- grep acts as filter
- grep can be used along with pipe command
- It does not change the content of the file

Filters using grep

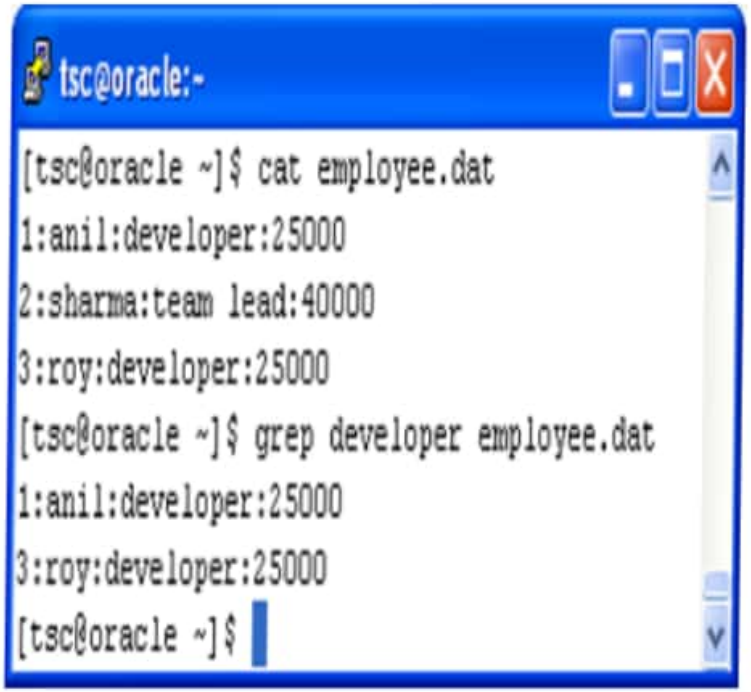
TEKNOTURF

syntax:

- grep [options] pattern [filename[s]]
  - searches for pattern in one or more filename(s), or the standard input if no filename is specified

in the example, “developer” is the pattern to be searched and “employee.dat” is the filename where the search should happen

When there is more than 1 word in the pattern, enclose the pattern within double quotes



Filters using grep

- Options in grep:
  - -i → ignoring the case
  - -v → selects all lines except those containing the pattern
  - -n → displays the line number along with the matching content
  - -c → displays the count
  - -l → displays the filenames containing the pattern
- options to explore:
  - -x, -f

Filters using grep continued...

TEKNOTURF

Basic regular expression

- * zero or more occurrence of the previous character

  example : g* → g gg ggg ggg.....
- . single character
- [ ] single character within the group
- [^] single character which is not part of the group
- ^$ lines containing nothing
- ^pat pattern pat as the beginning of the line
- pat$ pattern pat at the end of the line

Filter using cut

- splitting a file vertically
- remove sections from each line of files
- cut can be used along with pipe

syntax:

- cut option[s] file[s]

options in cut:

- -f → fields to be cut
- -d → delimiter

(both -f and -d are used when there is a special character that separates the column.)

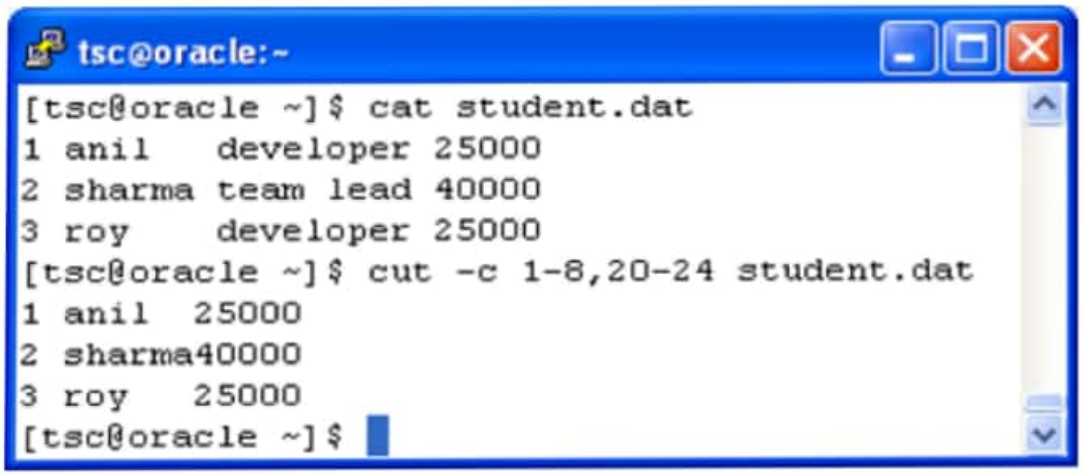
- -c → characters. Used when the number of columns are not equal in each line

Filter using cut

TEKNOTURF

example:

-c is used when the file contains fixed length record

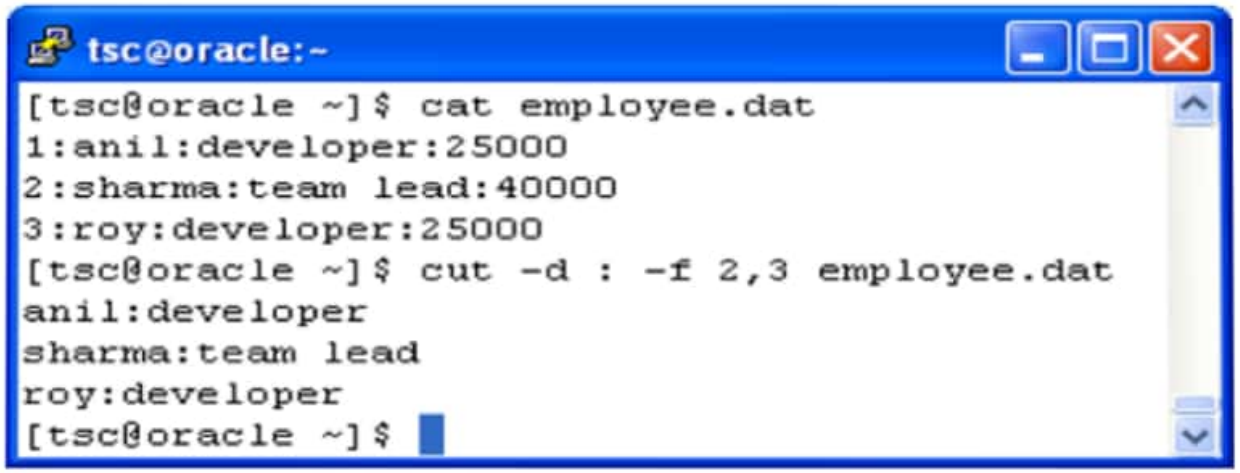


Filter using cut

TEKNOTURF

Example:

- -d and -f are used when the columns are separated by a delimiter
- In the given example ":" is the delimiter



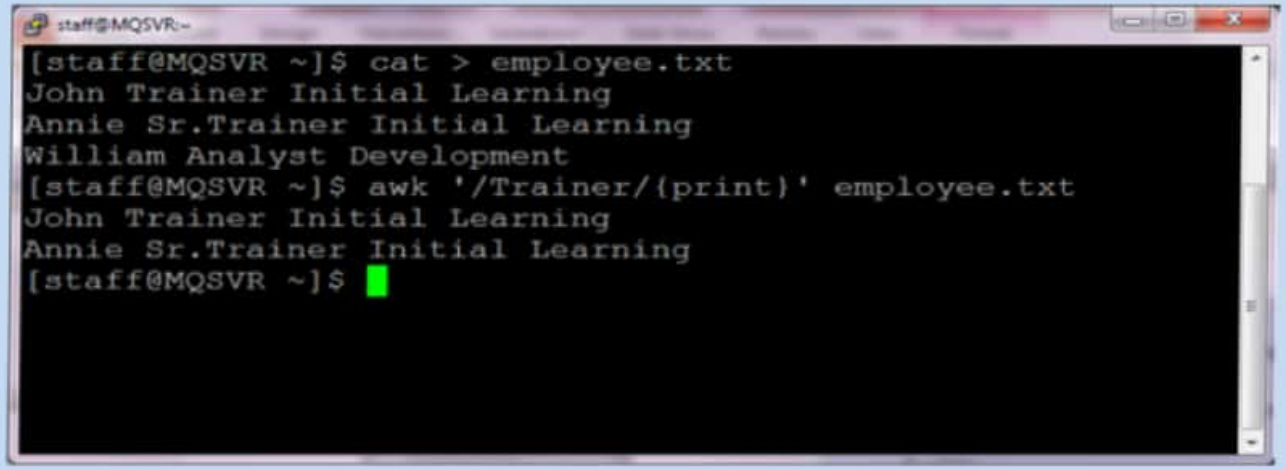
Advanced Filters

TEKNOTURF

awk

- It is an interpreted programming language, which focuses on processing text
- It is used to execute complex pattern-matching operations on streams of textual data
- **Syntax**

  awk [-F *fs*] [-v var=*value*] ['*prog*' | -f *progfile*] [*file* ...]



Summary

TEKNOTURF

- Pipes
  - tee
- Filter
  - Simple Filters
    - grep
    - cut
  - Advanced Filters
    - awk
    - sed

TEKNOTURF

THANK YOU

TEKNOTURF

VI EDITOR

Overview

TEKNOTURF

Can we write and execute different programming languages like C,C++,etc. in UNIX environment?

Definitely! In windows you use notepad to write programs. Likewise in UNIX, you can use VI Editor.

Objective

TEKNOTURF

- Introduction to vi Editor
- Input mode Commands
- Save & Quit
- Cursor movement Commands
- Paging Functions
- Search and Repeat Commands
- Introduction to SED
- SED Commands

Introduction to vi Editor

TEKNOTURF

vi Editor

- The **vi** editor lets a user to create new files or edit existing files
- **Syntax:**
  - vi <filename>
  - If the file doesn't exist, a new file is created. If the file exists then the file is opened for editing
  - When vi command is given without an argument, the editor is opened for inserting data, which can be saved later
- **Modes of vi editor:**
  - Command mode
  - Input mode
  - Last line mode

Modes of vi

TEKNOTURF

- **command mode**
  - When given vi, the file is opened in the command mode by default
- Some of the available options are
  - r - replace one character
  - x - delete text at cursor
  - dd - delete entire line
  - 5dd - delete 5 lines
  - yy - copy a line
  - *n*yy - copy *n* lines
  - P - paste above current line
  - p - paste below current line

Input Mode

TEKNOTURF

Options to switch from command mode to input or text mode:

- i- insert text at cursor
- a- insert text after cursor
- A- append text at the end

Input mode

- This mode allows the user to insert or modify or append text.
- To change to command mode press **<Esc>**

Last Line Mode

TEKNOTURF

Last line mode

- This is invoked from the command mode
- When the user types : the cursor moves to the last line of the screen

options given in last line mode:

- :w - save
- :wq - save and quit
- :q! - quit without save
- :w <filename> - saves a copy of the file (save as in windows)
- :set nu - sets line number
- :set ai - set auto indent

Cursor Movement Commands

TEKNOTURF

j or <return> [or down-arrow]	Move cursor down one line
k[or up-arrow]	Move cursor up one line
h or<backspace> [or left-arrow]	Move cursor left one line
l or<space>[or right-arrow]	Move cursor right one line
0(zero)	Move cursor to start of current line(the one with the arrow)
$	Move cursor to end of the current line
w	Move cursor to beginning of next word
b	Move cursor back to beginning of preceding word
:0<return> or 1G	Move cursor to first line in file
:n<return>or nG	Move cursor to line n in file
:$<return>or G	Move cursor to last line in file

Paging Functions

TEKNOTURF

:.=	Returns line number of current line at bottom of screen
:=	Returns the total number of lines at bottom of the screen
^g	Provides the current line number, along with the total number of lines,in the file at the bottom of the screen

Searching Commands

TEKNOTURF

/string	Search forward for occurence of string in text
?string	Search backward for occurence of string in text
n	Move to next occurence of search string
N	Move to next occurence of search string in opposite direction

The current line needs to be printed five times in the same file. Identify the command that will execute this task?

- [x] yy
- [ ] yc
- [ ] yw
- [ ] xx

Correct

That's right! You selected the correct response.

In Vi editor, a wrong file has been opened for editing. What command should be used to close the file without saving it?

- s
- ✔ q
- sq
- wq

Correct

That's right! You selected the correct response.

In cursor movement command, the w option is used to move the cursor to the beginning of the next word.

- [x] True
- [ ] False

Correct

That's right! You selected the correct response.

Introduction to SED

TEKNOTURF

- sed command is commonly used to replace string in Unix or UNIX based OS. That is why it is more commonly used as 'sed replace'
- sed is stream editor in Unix. It is used to parse and transforms texts
- sed was developed in the year 1973
- sed uses compact programming language; it was built for command line processing
- sed is also used to make programs which can change files

SED Commands

TEKNOTURF

SED with &

- The option & is used to append to the search pattern

Syntax:

- sed 's/search_pattern/& append_pattern/' file.txt

Example:

- The below sed command will find the search pattern '123' and append '456' to it
- $ sed 's/123/&456/' file.txt

SED Commands

TEKNOTURF

SED with s

- S stands for substitution. It replaces the searched string with the new string

Syntax:

- sed 's/search_pattern/replaced_pattern/' file.txt

Example:

- The below command will replace the string 'ABC' with 'ZYX' In the given sed command, sed will replace only the first occurrence of the pattern in each line
- $ sed 's/ABC/ZYX/' file.txt

SED Commands

TEKNOTURF

SED with g

g works as global replacement when used with sed command. The following command will replace all the searched pattern with the replaced pattern

Syntax:

- $ sed 's/searched_pattern/replaced_pattern/'g file.txt

Which sed command deletes the specified address range?

- [address range]/y
- ✔ [address range]/d
- [address range]/p
- [address range]/s

Correct

That's right! You selected the correct response.

‘sed’ is an interactive editor.

- ○ True
- ✔ ◉ False

Correct

That's right! You selected the correct response.

Which is the correct syntax for sed on command line?

- sed [filename] [options] '[command]'
- sed '[command]' [options] [filename]
- ✔ sed [options] '[command]' [filename]
- sed '[command]' [filename] [options]

Correct

That's right! You selected the correct response.

Summary

- vi Editor Introduction
- Input mode Commands
- Save & Quit
- Cursor movement Commands
- Paging Functions
- Search and Repeat Commands
- Introduction to SED
- SED Commands

TEKNOTURF

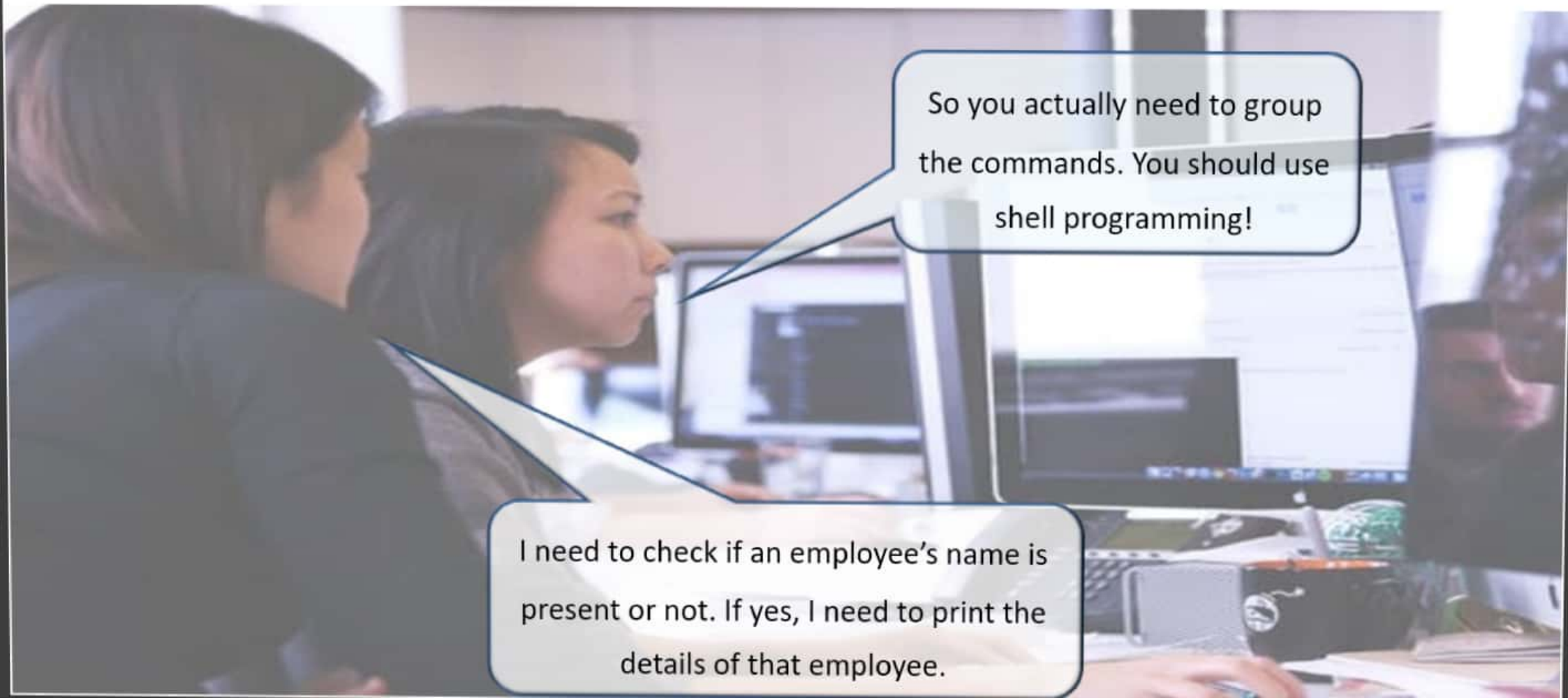
THANK YOU

TEKNOTURF

BOURNE SHELL

Overview

TEKNOTURF



Objective

- Shell Introduction
- Shell Types
- Working of Shell
- Meta Character
- Shell Redirections
- Command Substitution

TEKNOTURF

Shell basics Introduction

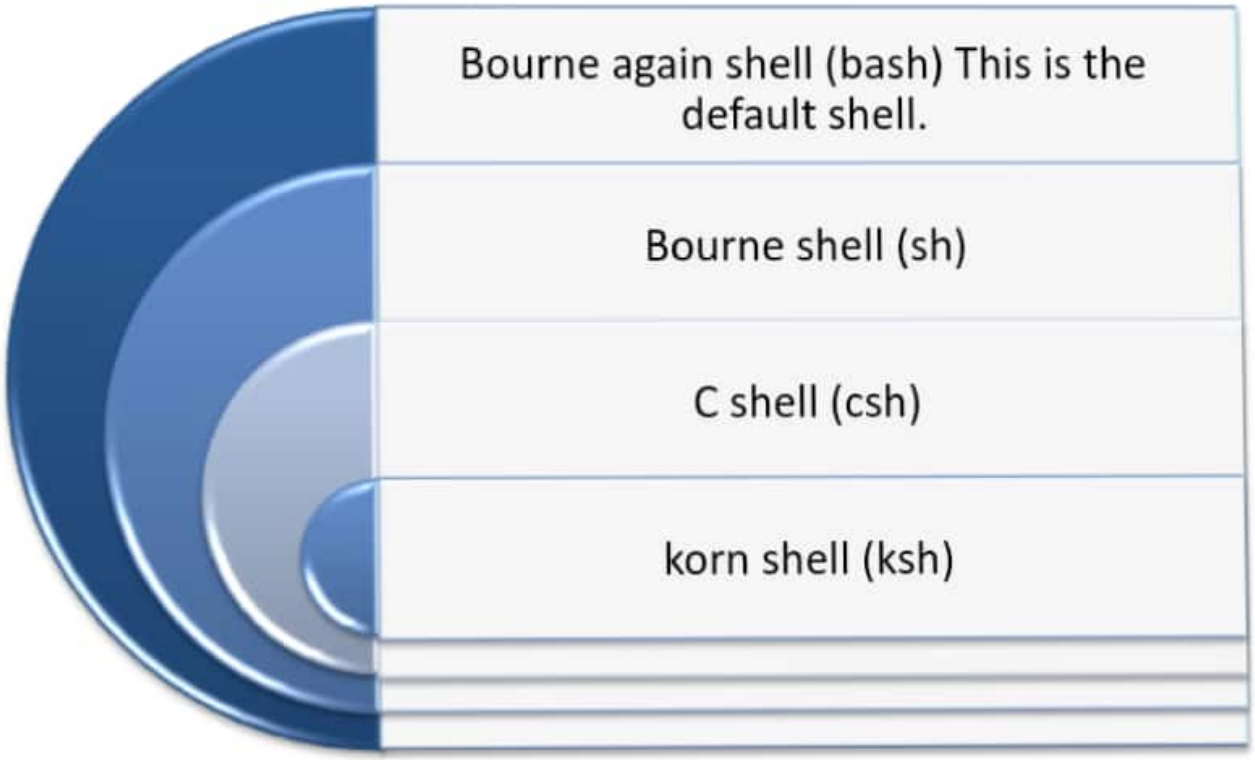
TEKNOTURF

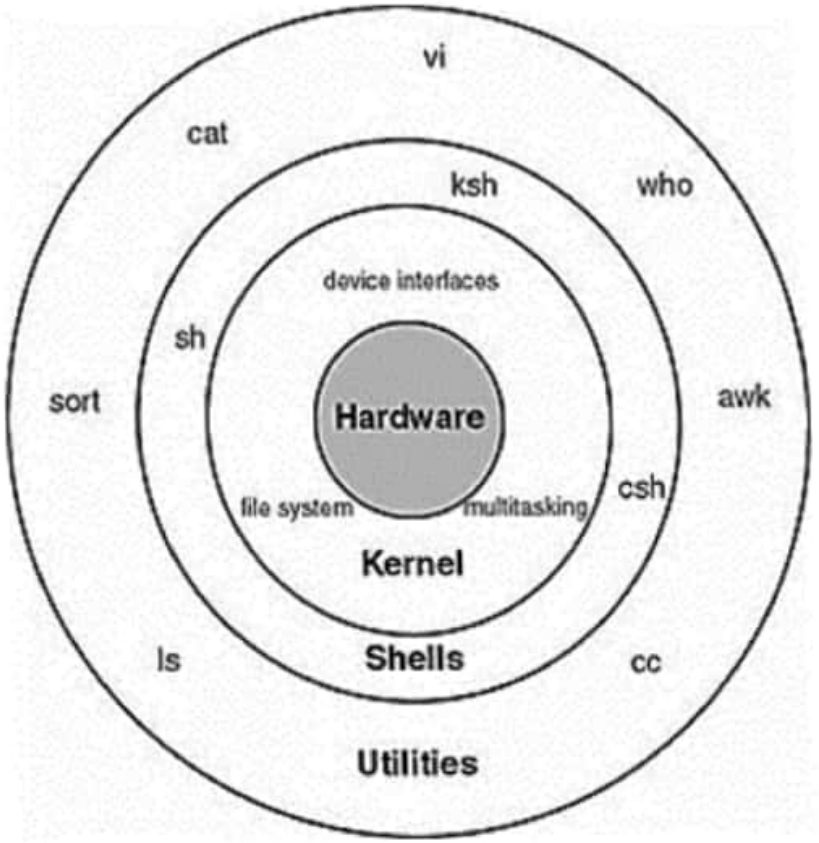
- Shell supports execution of scripts
- When a group of commands have to be executed regularly, they should be written in a file, which is the shell script or shell program and the script will be executed by the shell
- A shell program runs in interpretive mode. It is not compiled to a separate executable file. Each statement is loaded into memory when it is to be executed
- For the programmer's understandability, the script file may have extension as “sh”
  - For a good programming practice, the extension is followed
    - E.g testfile.sh

Shell

TEKNOTURF

Types of Shell





Working of Shell

TEKNOTURF

Comments

Single line comment

is used to comment a line in the script

Example: # echo “hai”

Multiple line comment

<< comment
- statements
- comment

Example: <<comment
- echo “this is”
- echo “test code”
- comment

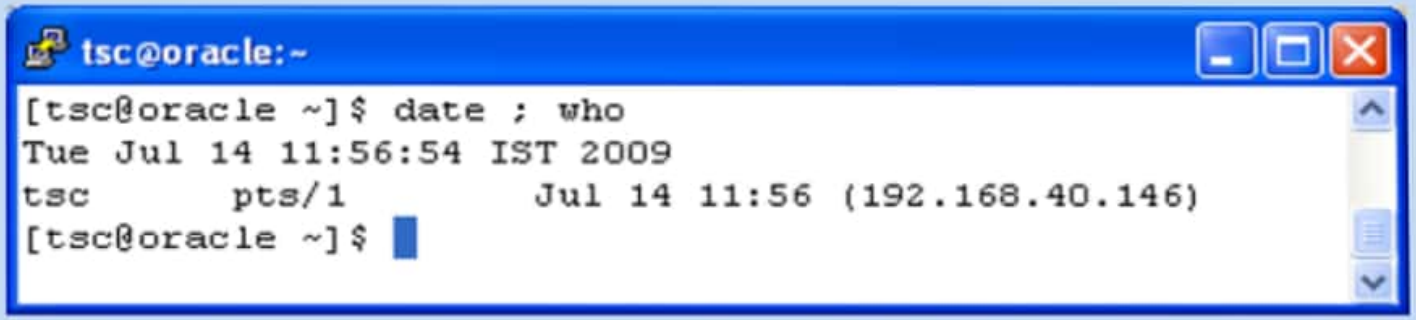
Basic elements in Shell programming:

TEKNOTURF

command grouping:

;

- executing multiple commands in a single line
- Example:



This executes the date command 1st and then the who command.

Basic elements in Shell programming:

TEKNOTURF

{ }

- used for grouping commands and executes in the current shell. The last statement in every group should be terminated with a semicolon
- There should be a blank space after the open braces and before the close braces
- This is used when one of the statements in the group needs to update the current environment

- Example:
  - $ { cd mydir ; pwd; }
    - /home/teknoturf/staff/tsc/mydir
- After executing this command, the user is placed in the mydir directory

Basic elements in Shell programming:

TEKNOTURF

( )

- used for grouping commands in a single line and executing the grouped command in the sub-shell(child process)

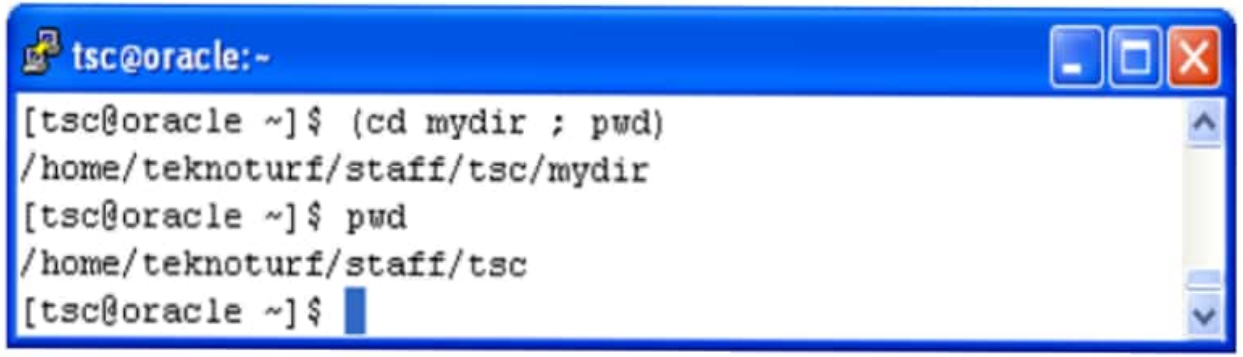
example:

- Without parenthesis, these commands execute in the same shell

Basic elements in Shell programming:

TEKNOTURF

With parenthesis, these commands execute in the sub-shell. So once executed, the sub-shell(child process) is destroyed.



Basic elements in Shell programming:

TEKNOTURF

Conditional execution:

- These are for logical testing

&& :

- This will test for "and" condition
- The second command will be executed only when the 1st command is successful

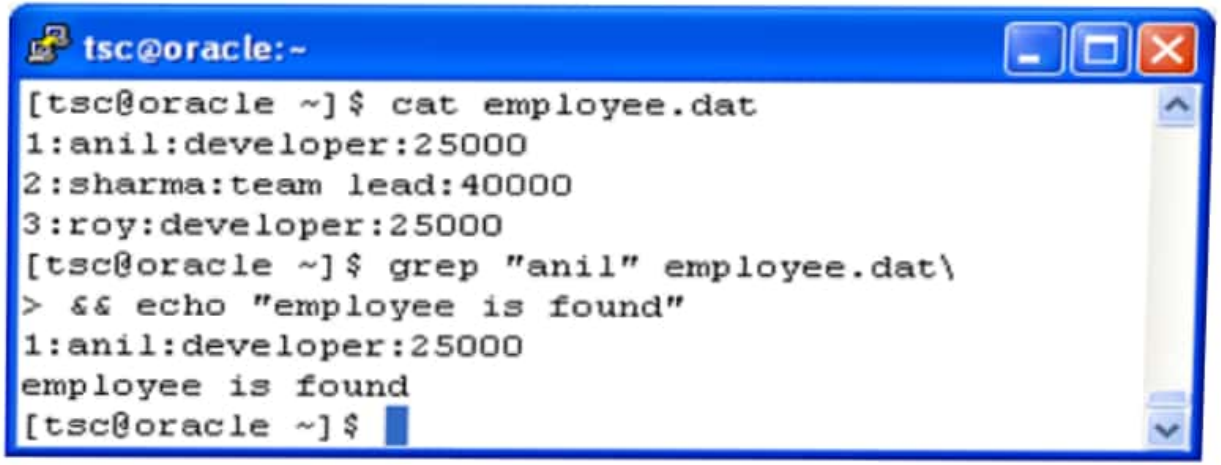
Syntax:

- command1 && command2

Basic elements in Shell programming:

&& example:

check whether the employee “anil” is present and if present, display his information



Basic elements in Shell programming:

TEKNOTURF

Conditional execution:

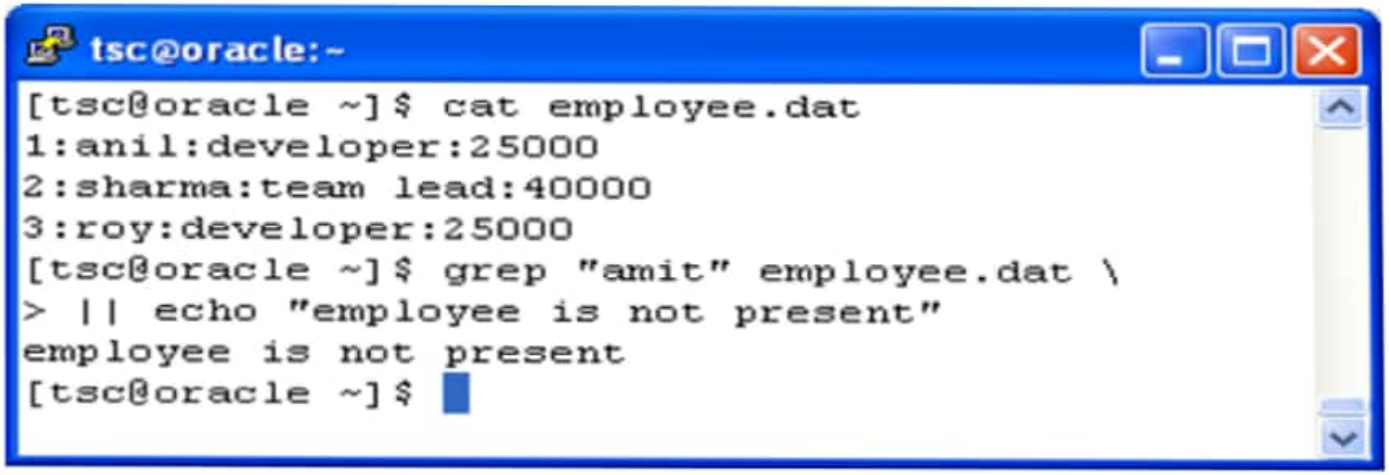
||

- This will test “or” condition
- The second command will be executed only when the 1st command fails
- **Syntax:**
  - **command1 || command2**

Basic elements in Shell programming:

TEKNOTURF

Logical OR (||) example:



Basic elements in Shell programming:

TEKNOTURF

Variables:

- Variables provide the ability to store and manipulate information within a shell program.
- The data is treated as string.

Types of variable:

- System or environment variable
- User defined variable

Basic elements in Shell programming:

TEKNOTURF

System Variables

- These are standard variables which are always accessible
- These variables are declared in the environment area
- The shell provides the value for these variables
- These variables are used by the system to govern the environment
- The user is allowed to change the values of these variables

Basic elements in Shell programming:

TEKNOTURF

Various system variables are

PS1, PS2	PATH	HOME	LOGNAME	SHELL	TERM
→ system prompts	→ path which the shell must search in order to execute any command or file	→ stores the default working directory of the user	→ login name	→ name of the default working shell	→ defines the name of the terminal on which the user is working

Basic elements in Shell programming:

TEKNOTURF

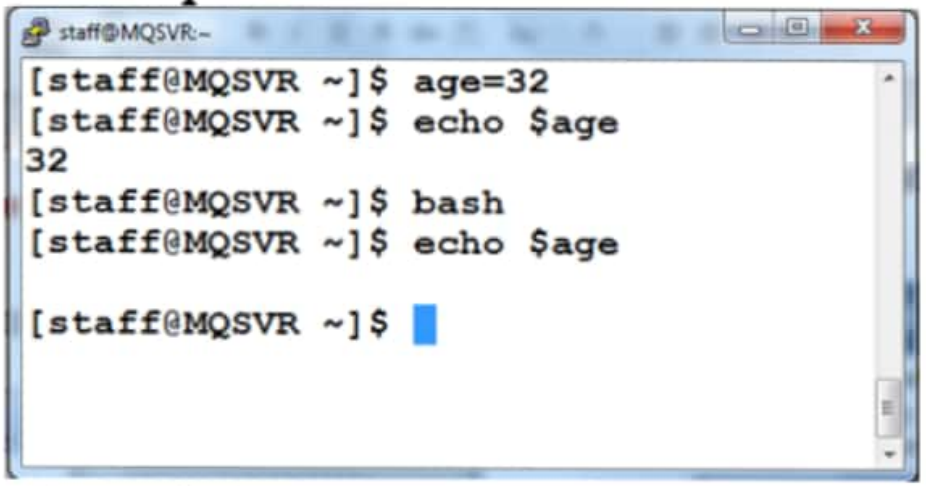
user-defined variable:

- These are created, used and maintained by the user in shell programming

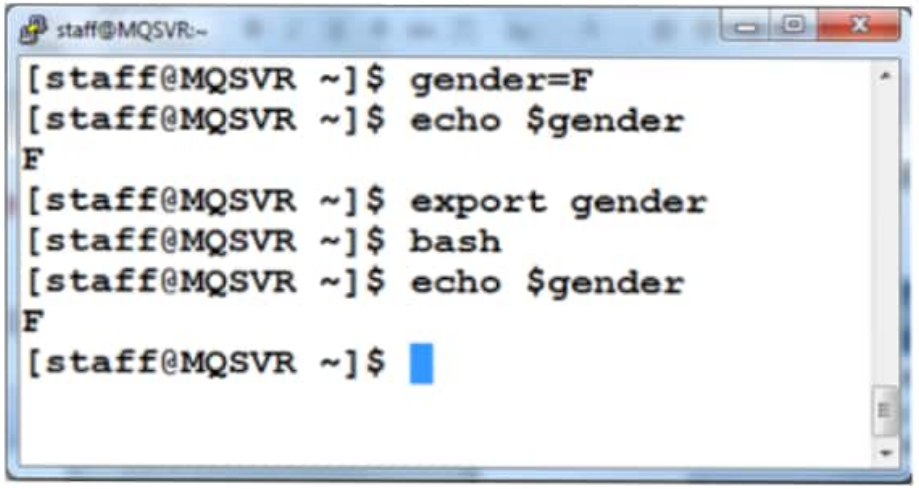
export:

- User defined variables are for the current shell
- To access the user defined variable in the child process, these variables should be placed in the environment area
- This is done by the command export

Example 1:



Example 2:



Basic elements in Shell programming:

TEKNOTURF

Assigning value to the variable:

= is used to assign value to the variable

Example 1: **$ name=tsc**

- name is the variable name
- tsc is the value
- no space between the variable name, assignment operator and the value.

Example 2: **$ name="Teknoturf school of computing"**

- Value should be enclosed within Double quotes when there is more than 1 word.

Basic elements in Shell programming:

TEKNOTURF

Listing of variables:

- $ env → displays all the system or environment variables
- $ set → displays all the variables available in the current shell
- $ echo $<variable-name> → displays the value of the specific variable

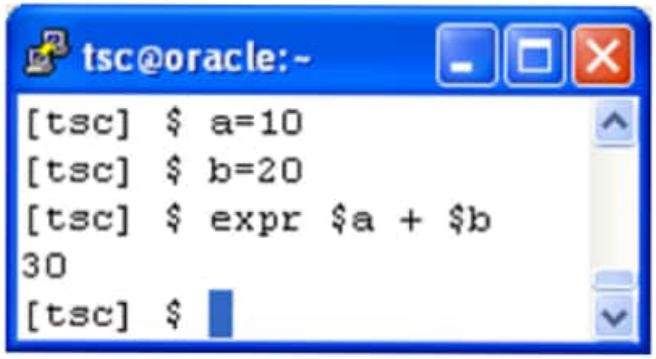
Basic elements in Shell programming:

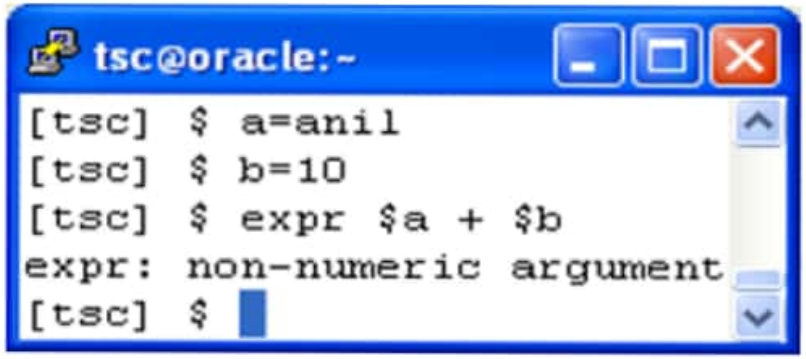
TEKNOTURF

Evaluate expression:

- To carry out arithmetic operations, **expr** command is used.
- **expr** can do only the integer operations.

Example:





Basic elements in Shell programming:

TEKNOTURF

Test (or) [ ]

Used to check whether the statement is true or false

test returns true or false status

It works in 3 ways

- Numeric comparison
- String comparison
- Checks file's attributes

$? is used to display the status returned by the previous command

Operators should be surrounded by spaces

Basic elements in Shell programming:

TEKNOTURF

Numeric comparison

OPERATOR	MEANING
-eq	Equal to
-ne	Not equal to
-gt	Greater than
-ge	Greater than or equal to
-lt	Less than
-le	Less than or equal to

Basic elements in Shell programming:

TEKNOTURF

Numeric comparison Example:

$ x=5 ; y=10

$ test $x -eq $y ; echo $?

output : 1 (for not equal)

or

$ [$x -eq $y] ; echo $?

output : 1 (for not equal)

When the test condition returns true, $? prints 0

Basic elements in Shell programming:

TEKNOTURF

String comparison

OPERATOR	MEANING
S1=S2	Checks for equality
S1!=S2	Not equal
-n string	String is not null
-z string	String is null

Example:

$ employee1=Tom ; employee2=Amit

$ test $employee1 = $employee2 ; echo $?

output : 1 (represents not equal)

File Comparison

TEKNOTURF

operator	Meaning
•-e file	True if file exists.
•-f *file*	*file* exists and is a regular file
•-r *file*	*file* exists and is readable
•-w file	file exists and is writable
•-x file	file exists and is executable
•-d file	file exists and is a directory
•-s file	file exists and the size is greater than 0
•file1 -nt file2	file1 is newer than file2
•file2 -ot file2	file1 is older than file2
•file1 -ef file2	file1 is linked to file2

•(-nt , -ot , -ef are available only in bash and korn shell)

Basic elements in Shell programming:

TEKNOTURF

File comparison example:

- $ touch testfile
- $ ls -l testfile

  -rw-r--r-- 1 tsc staff 0 Jul 15 15:10 testfile

- $ [-f testfile] ; echo $?

  0 (it is an ordinary file)

- $ [-x testfile] ; echo $?

  1 (it is not executable)

Which file should be updated so that the changes made will get reflected in all the sessions?

- [x] .bash_profile
- [ ] .bashrc
- [ ] /etc/profile

Correct

That's right! You selected the correct response.

Create a new file “new.txt”, which is a concatenation of “file1.txt” and “file2.txt”. Which command would do the given task?

- mv file[12].txt new.txt
- ✔ cat file1.txt file2.txt > new.txt
- cp file.txt file2.txt new.txt
- ls file1.txt file2.txt | new.txt

Correct

That's right! You selected the correct response.

Which one of the following statements is true about variables in shell script?

- to extract the contents of a variable, we have to provide the variable
- all of the mentioned
- variables are case in-sensitive
- ✔ variables do not require declaration before assigning value to them

Incorrect

You did not select the correct response.

By default, all the variables used in shell are treated as _____ .

- character
- decimal
- ✔ string
- number

Correct

That's right! You selected the correct response.

Which command is used to remove a variable from the list of variables to redefine it?

- [x] unset
- [ ] clear
- [ ] delete
- [ ] remove

Correct

That's right! You selected the correct response.

Meta-character

TEKNOTURF

A meta character is a character that has a special meaning to the Shell.

Type	Metacharacter
Filename substitution	? * [ ] [!]
I/O Redirection	> >> < <<
Process Execution	; ( ) & && ||
Quoting metacharacters	\ " " ' '
positional parameters	$1 to $9
Special characters	$0 $* $@ $# $! $$ $-

Metacharacter continued

TEKNOTURF

Filename substitution metacharacter

- * → it is a wild card character, which represents none or any number of characters
- $ ls a* lists all files beginning with character 'a'
- ? → stands for 1 character
  - $ ls ?? lists all files whose file names are 2 character long
- [ ] any one character from the enclosed list
  - $ ls [kdgp]* lists all files whose 1st character is k , d , g or p
- [!] any one character except those enclosed in the list
  - $ ls [!d – m]* lists all files whose 1st character is anything other than the alphabet in the range d to m

Quoting Metacharacter

TEKNOTURF

double quote:

- They allow *all* shell interpretations to take place inside them.
- Example:
  - $ name=tom
  - $ echo “The name is $name”
  - output : The name is tom

Quoting Metacharacter

TEKNOTURF

single quote:

- anything enclosed in single quotes remains unchanged.
- Example:
  - $ name=tom
  - $ echo 'The name is $name'
  - output: The name is $name

Quoting Metacharacter

TEKNOTURF

back quotes

- To execute command and replace the output in place of the command
- Example:
  - $ echo "Today is date"

    Output : Today is date
  - $ echo "Today is `date`"

    Output : Today is Wed Jul 15 13:18:30 IST 2009

Shell Redirection

TEKNOTURF

- Any process that starts executing in the shell has 3 files which are given by the shell
- These files are called streams
- streams are
  - standard input – represents the input which is normally the keyboard
  - standard output – represents the output which is normally the monitor
  - standard error – represents the error which is normally the monitor
- These streams are assigned with numbers which are called file descriptors

Redirection

TEKNOTURF

File Descriptor:

- a file descriptor is an abstract key for accessing a file
- a file descriptor is an integer value
- There are 3 standard file descriptors which every process should have

integer value	name
0	standard input(stdin)
1	standard output(stdout)
2	standard error(stderr)

Redirection

Redirection is the switching of a *standard stream* of data so that it comes from a source other than its default source or goes to some destination other than its default destination

Redirection is used to change the default stream to any other stream

Example: redirected to or from a file

Types of redirection

- Input redirection
- Output redirection
- Error redirection

Redirection

TEKNOTURF

Input redirection:

- Input redirection makes the command to get the input from other devices apart from the standard input
- The symbol for input redirection is **<**

example:

- with standard input stream:

```
$ bc
2 + 3
5
[ctrl+d]
```

Redirection

TEKNOTURF

with input redirection: to avoid user interaction

- $ bc < inputfile (or) bc 0 < inputfile
  5
- note: the content of inputfile is 2 + 3
  0 is the file descriptor and is optional

How it works:

- On seeing the < , the shell opens the file, inputfile for reading
- unplugs the standard input stream from its default and assigns it to inputfile
- bc reads the contents from the file, manipulates it and displays the output to the screen

Redirection

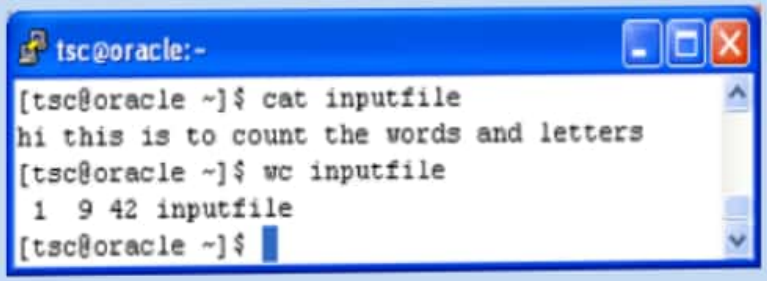
TEKNOTURF

Output redirection:

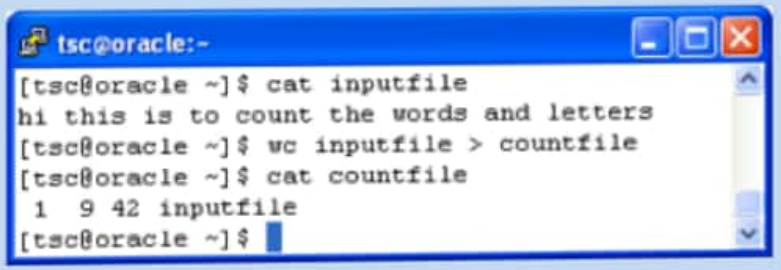
- The output of the command is normally displayed in the monitor which is the standard output stream
- The *output redirection operator* (>) can be used to redirect standard output from the monitor to any other device.

Example

with standard redirection



with output redirection



- 1> can also be used. 1 is the file descriptor for output stream
- The output instead of displaying in the monitor is being redirected to countfile

TEKNOTURF

How it works:

- On seeing >, the shell opens a new file named countfile if it doesn't exist or it overwrites the content, if the file already exists
- Unplugs the default standard output stream and assigns it to countfile
- command gets executed and the output is written to the countfile

Output redirection with append:

- symbol >>
- Shell opens the file and moves the cursor to the end of the file. No overwriting occurs

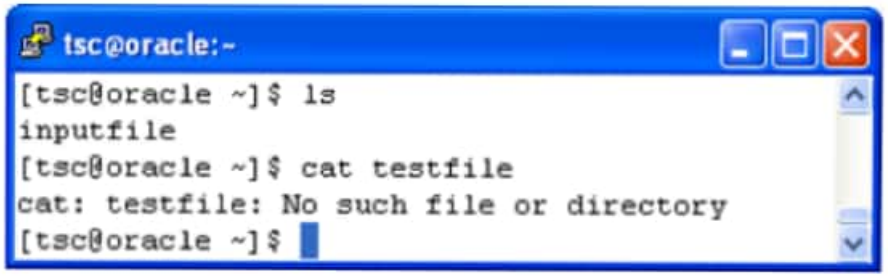
Shell Redirection

Error redirection:

- The default standard error stream is the monitor.
- standard error can be redirected from its default to any other device by using the *standard error redirection operator*, 2>.
- 2> overwrites the file content and 2>> appends to the already existing file
- Both output and error redirection works the same.

Example:

with standard error



with error redirection