

**MIT**

**Arts, Commerce  
& Science College**

# CERTIFICATE

## DEPARTMENT OF COMPUTER SCIENCE

*This is certify that Mr. /Miss.* \_\_\_\_\_

*of F.Y. B.Sc.(Computer Science)Exam Seat No*\_\_\_\_\_

*has satisfactory completed his/her practicals in the* \_\_\_\_\_

*as laid down by the Savitribai Phule Pune University for the  
academic Year*\_\_\_\_\_

*Date:*\_\_\_\_\_

*Total no. of Assignments::*\_\_\_\_\_

*Completed Assignments::*\_\_\_\_\_

Subject Teacher

Head of the Dept.

Internal Examiner

External Examiner

# Table of contents

<b>Introduction</b> .....	<b>1</b>
<b>Assignment Completion sheet</b> .....	<b>3</b>
<b>Lab Course II</b>	
<b>Section I</b>	
<b>Exercise 1</b> .....	<b>6</b>
Using basic DOS commands like date, time, dir, copy con, type, ren etc.	
<b>Exercise 2</b> .....	<b>11</b>
Creating the directory structure and Batch file in the DOS	
<b>Exercise 3</b> .....	<b>15</b>
Using Windows XP graphical user interface (GUI).	
<b>Exercise 4</b> .....	<b>21</b>
Using basic Linux commands	
<b>Exercise 5</b> .....	<b>28</b>
Using vi editor	
<b>Exercise 6</b> .....	<b>42</b>
Shell Programming (Writing simple shell scripts, use of conditional structures).	
<b>Exercise 7</b> .....	<b>46</b>
Shell programming (Writing shell scripts using control structures )	
<b>Exercise 8</b> .....	<b>49</b>
Creating simple HTML pages	
<b>Exercise 9</b> .....	<b>54</b>
HTML programming (use of lists, tables, frames, hyperlinks)	
<b>Exercise 10</b> .....	<b>59</b>
HTML programming ( Creation of forms, small case study to create HTML pages using all the above learnt techniques).	
<b>Section II</b>	
<b>Exercise 11</b> .....	<b>64</b>
To create simple tables , with only the primary key constraint ( as a table level constraint & as a field level constraint) (include all data types)	
<b>Exercise 12</b> .....	<b>68</b>
To create more than one table, with referential integrity constraint, PK constraint.	
<b>Exercise 13</b> .....	<b>74</b>
To create one or more tables with Check ,unique and not null constraint	
<b>Exercise 14</b> .....	<b>76</b>
To drop a table from the database and to alter the schema of a table in the Database.	
<b>Exercise 15</b> .....	<b>78</b>
To insert / update / delete records using tables created in previous Assignments. ( use simple forms of insert / update / delete statements)	
<b>Exercise 16</b> .....	<b>82</b>
To understand & get a Hands-on on Select statement	
<b>Exercise 17</b> .....	<b>86</b>
To query table, using set operations (union, intersect)	
<b>Exercise 18</b> .....	<b>89</b>
To query tables using nested queries	
<b>Exercise 19</b> .....	<b>92</b>
To query tables , using nested queries ( use of 'Except', exists, not exists clauses)	
<b>Exercise 20</b> .....	<b>95</b>
Assignment related to small case studies ( Each case study will involve creating tables with specified constraints, inserting records to it & writing queries for extracting records from these tables)	

# Introduction

## 1. About the work book

This workbook is intended to be used by F.Y.B.Sc (Computer Science) students for the two Computer Science laboratory courses in their curriculum. In Computer Science, hands-on laboratory experience is critical to the understanding of theoretical concepts studied in the theory courses. This workbook provides the requisite background material as well as numerous computing problems covering all difficulty levels.

### The objectives of this book are

- 1) Defining clearly the scope of the course
- 2) Bringing uniformity in the way the course is conducted across different colleges
- 3) Continuous assessment of the course
- 4) Bring in variation and variety in the experiments carried out by different students in a batch
- 5) Providing ready reference for students while working in the lab
- 6) Catering to the need of slow paced as well as fast paced learners

## 2. How to use this workbook

This workbook is mandatory for the completion of the laboratory course. It is a measure of the performance of the student in the laboratory for the entire duration of the course.

### 2.1 Instructions to the students

Please read the following instructions carefully and follow them

- 1) You are expected to carry this book every time you come to the lab for computer science practicals
- 2) A file should be maintained separately by each student which should contain the algorithms, flowcharts, written answers, source code as well as the program output.
- 3) You should prepare yourself before hand for the Exercise by reading the material mentioned



Reading



Ready Reference

under icon . Also go through the material given in ready reference icon .

- 4) If the self activity exercise or assessment work contains any blanks such as this \_\_\_\_\_, or , get them filled by your instructor.

5) Instructor will specify which problems you are to solve by ticking box

6) Follow good programming practices like:

- Use appropriate file naming conventions
- Use meaningful variable names
- Use proper Indentation
- Use comments in the program
- Every program should contain in comments prgrammer's name and date

7) You will be assessed for each exercise on a scale of 5

- |                       |   |
|-----------------------|---|
| i) Not done           | 0 |
| ii) Incomplete        | 1 |
| iii) Late Complete    | 2 |
| iv) Needs improvement | 3 |
| v) Complete           | 4 |
| vi) Well Done         | 5 |

### 2.2. Instruction to the Instructors

- 1) Explain the assignment and related concepts in around ten minutes using white board if required or by demonstrating the software

- 2) Fill in the blanks with different values for each student
- 3) Choose appropriate problems to be solved by student by ticking box
- 4) Make sure that students follow the instruction as given above
- 5) After a student completes a specific set, the instructor has to verify the outputs and sign in the provided space after the activity.
- 6) Ensure that students use good programming practices.
- 7) You should evaluate each assignment carried out by a student on a scale of 5 as specified above by ticking appropriate box.
- 8) The value should also be entered on assignment completion page of the respective Lab course

### 2.3. Instructions to the Lab administrator

You have to ensure that appropriate hardware and software is available to each student. The operating system and software requirements on server side and also client side are as given below

- 1) Server Side ( Operating System )
  - a. \* Fedora Core Linux
  - \* Microsoft Windows Server 2003
  - b. Servers Side (software's to be installed)
  - In Linux – C, C++, awk, shell, perl, postgresql/Mysql
  - In WinXP -- MSOffice
- 2). Client Side ( Operating System )
  - a. \* Red Hat Linux and Fedora Core
  - \* Microsoft Windows XP
  - b. Client Side ( software's to be installed )
  - In Linux – C, C++, awk, shell, perl, postgresql/mysql
  - In WinXP -- MSOffice

The detail information about installation and configuring of the server and client are provided in appendix A

## 3. Acknowledgements

The authors wish to express their gratitude to Dr. Narendra Jadhav, Vice Chancellor, University of Pune, for his vision and guidance in bringing out this lab book, a first of its kind. Dr. Pandit Vidyasagar, Director, Board of colleges and university department has played a pivotal role in taking this project to completion. We are indebted to Dr. V. B. Gaikwad, Dean Science Faculty, who extended his wholehearted support to this endeavor. Prof. Arun Gangarde, Chairperson, Board of studies in Computer Science deserves a special mention for his untiring efforts during the entire process.

We appreciate the efforts taken by Prof. Chitra Nagarkar , member, Board of studies in Computer Science during initial phases of the project. We would like to acknowledge the role played by the University authorities and the members of the Board of Studies in Computer Science.

Special thanks to Mr. Achyut Godbole, noted IT personality and renowned author who took a lot of interest in this project.

Our heartfelt thanks to Dr. Sanjay Kadam, CDAC and Ms. Kishori Khadilkar, Patni Computer Systems Ltd., for painstakingly reviewing the entire book and giving valuable inputs. Last but not the least, we thank all the faculty members, who have been involved in this project and shared their expertise.

## Assignment Completion Sheet

Lab Course II		
Section I		
Sr. No	Assignment Name	Grade
1	Using basic DOS commands like date, time, dir, copy con , type, ren etc.	
2	Creating a directory structure in DOS and batch files.	
3	Using Windows XP graphical user interface (GUI).	
4	Using basic Linux commands	
5	Using vi editor	
6	Shell Programming (Writing simple shell scripts, use of conditional structures).	
7	Shell programming (Writing shell scripts using control structures )	
8	Creating simple HTML pages (use of different tags for changing fonts, foreground and background colors etc.)	
9	HTML programming (use of lists, tables, frames, hyperlinks)	
10	HTML programming ( Creation of forms, small case study to create HTML pages using all the above learnt techniques).	
Section II		
11	To create simple tables , with only the primary key constraint ( as a table level constraint & as a field level constraint) (include all data types)	
12	To create more than one table, with referential integrity constraint, PK constraint.	
13	To create one or more tables with Check ,unique and not null constraint	
14	To drop a table from the database and to alter the schema of a table in the Database	
15	To insert / update / delete records using tables created in previous Assignments. ( use simple forms of insert / update / delete statements)	
16	To query the tables using simple form of select statement	
17	To query table, using set operations (union, intersect)	
18	To query tables using nested queries	
19	To query tables , using nested queries ( use of 'Except', exists, not	
20	Assignment related to small case studies ( Each case study will involve creating tables with specified constraints, inserting records to it & writing queries for extracting records from these tables)	

<b>Lab Course I</b>		
<b>Sr. No</b>	<b>Assignment Name</b>	<b>Grade</b>
1	To demonstrate use of data types, simple operators (expressions)	
2	To demonstrate decision making statements (if and if-else, nested structures)	
3	To demonstrate decision making statements (switch case)	
4	To demonstrate use of simple loops	
5	To demonstrate use of nested loops	
6	To demonstrate menu driven programs and use of standard library functions.	
7	To demonstrate writing C programs in modular way ( use of user defined functions)	
8	To demonstrate recursive functions.	
9	To demonstrate use of arrays (1-d arrays ) and functions	
10	To demonstrate use of multidimensional array(2-d arrays ) and functions	
11	To demonstrate use of pointers	
12	To demonstrate concept of strings(strings and pointers)	
13	To demonstrate array of strings.	
14	To demonstrate use of bitwise operators.	
15	To demonstrate structures (using array and functions )	
16	To demonstrate nested structures and Unions	
17	To demonstrate command line arguments and pre-processor directives.	
18	To demonstrate file handling (text files)	
19	To demonstrate file handling (binary files and random access to files)	
20	Problem solving using C	

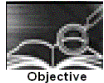
# **Lab Course II**

## **Section I**

## Exercise 1

Start Date

/ /
-----



Using basic DOS commands like date, time, dir, copy con , type, ren etc.



You should read following topics before starting this exercise

1. Read about DOS as an Operating System
2. Command line interface, Internal and external command
3. File and file naming convention.

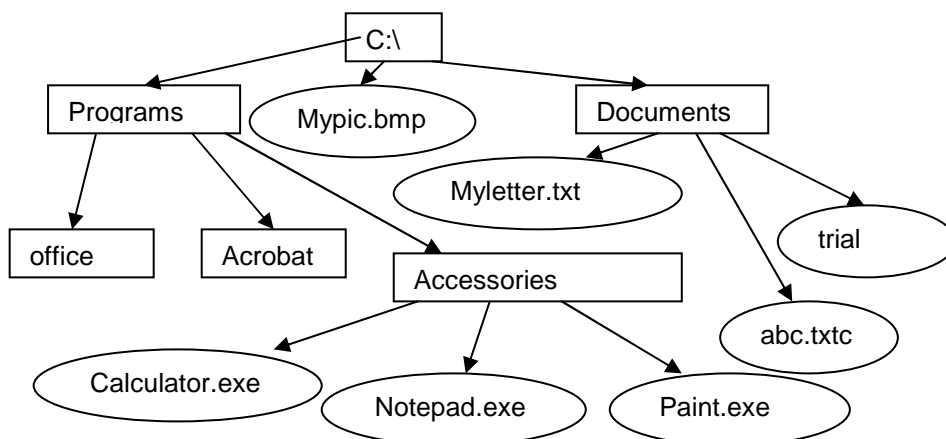


Operating system is an interface between the user and the computer hardware. MS-DOS (Microsoft's Disk Operating System) runs on any of the Intel 8088,80x86 or Pentium class CPU's on a Personal computer platform. The version of MS-DOS that runs on early IBM computers is called PC-DOS.

DOS is a 16-bit, single tasking, single-user operating system. It operates in real mode , meaning that only one program or process can run at a time. There is a 640 KB limit on memory that is accessible to the applications. The applications, directly access and control the hardware like printers bypassing the operating system. DOS has a simple text-based command line interface and it comprises of three files – MSDOS.SYS, COMMAND.COM and IO.SYS.

An operating system allows you to store and access information on a computer which can be letters, favourite music, family pictures, reports etc. Every piece of stored data accessed by the computer is treated as a file and is assigned a unique name. DOS uses a "8.3" filename, which can be up to eight characters long followed by a period and an extension of three characters. It cannot contain characters such as this " / \ [ ] : ; = , .

When the number of files stored on a disk attached to a computer increases, it need to be organized. The disks with large capacity are split into one or more partitions or drives. Each partition, drive or volume is given a name such as 'C' , 'D' etc. Files in a partition are organized into directories which are organized similar to tree structure.



Every file has path starting from root through subdirectories reaching a file. Path of file paint.exe is c:\Programs\Accessories\paint.exe

You can execute DOS command on Windows XP by getting the console with a DOS prompt by executing command program or choosing command prompt from Programs- Accessories.



In Windows XP, Select the following path :

Start -> Programs -> Accessories -> Command Prompt

It will display command prompt as C:\> \_ , by default. The DOS commands can be typed at this prompt.

## 1. Internal Commands

Command	Used for	Format & Example
HELP	Gives help on all DOS commands or a specific command	C:\> help <commandname> C:\> help C:\> help cls
CLS	It clears the screen and the cursor waits in the top left corner of the screen with current working prompt.	C:\> Cls
VER	Displays the current DOS version	C:\> Ver
DATE	Used to display and change the system date. Displays the current date and prompt user to change the date if desired.	C:\> DATE [mm-dd-yy] C:\> Date
TIME	Used to display and change system time.	C:\> TIME [hh:mm:ss:xx] C:\> Time
DIR	List contents of the specified directory	C:\> DIR [drive :] [path] [filename] [.ext] [/option] Drive – specifies the drive name Path - specifies the list of subdirectories to the required directory Filename – the name of the file Ext - specifies the extension of the file. Option – specifies one or more options to be used /p - Page-wise listing /w - Wide-format /s - list the files of subdirectories below specified directory. /o - ordered listing (can be reversed by -) D - chronological order E - extension-wise, then by name G - grouped by subdirectories N – filename-wise, then extension S – file size /a - attributewise listing (can be reversed by -) D – Directories only R - Read-only files H – Hidden files A – Archive files S – System files C:\> dir C:\> dir *.exe C:\> N??.exe Here * and ? are wild-card characters . A wild-card character * can be replaced by any letter or letters while ? can be replaced by any single letter , before executing the command C:\> dir a*.* C:\> dir c:\d2\d21 /p C:\> dir /A:h C:\> dir /o:s /A:R

COPY	Used to copy or append files to other files.	C:\> copy <source> <destination> C:\> copy con a.txt Copies the contents typed at the console to file a.txt. Input has to be terminated by Ctrl+Z. C:\> copy a.txt b.txt C:\> copy a.txt + b.txt c.txt Copies the appended file of a.txt and b.txt to c.txt C:\> Copy c:\d2\d21\c.doc d: Copies files c.doc to floppy in drive d by the same name
DEL	Used to delete a file	C:\> Del filename C:\> Del a.txt
TYPE	Display the contents of the file on the screen or it can be sent to the printer .	C:\> Type <filename> C:\> type a.txt C:\> type a.txt   more C:\> type a.txt > prn
RENAME or REN	Changes name of an existing file.	C:\> ren <old_filename><new_filename> C:\> ren a.txt b.txt
PATH	Used to display the current path or set a new search path for the executable files.	PATH [[drive:] path ][:[drive:] path ...]] C:\> Path Displays the current search path as PATH C:\DOS; C:\system32; D:\PROGRAMS Any executable program will be first searched in DOS, then system32 and then PROGRAMS directory

#### External Commands

ATTRIB	Used to change or display various file attributes.	ATTRIB properties filename + sets - removes the set attributes Properties R : Read only A : Archive S : System file C:\> ATTRIB +R a.txt Makes the file a.txt Read-only C:\> ATTRIB +H C:\D2\D21\a.txt Makes file a.txt hidden file
FORMAT	used to format a disk into sectors and create a File Allocation Table (FAT) which records all files on the disk. Previous disk contents are destroyed.	FORMAT [drive:] [/switches] Switches /I Formats double sided disk as a single sided disk /B leaves room for system files but system files are not copied /Q Quick formatting /S Transfer DOS system files to the formatted disk /U Prompts the user to add a volume label to the disk
CHKDSK	Used to check the disk for errors and displays a status report.	CHKDSK filename option Option /F Automatic correction of errors /V Displays a series of messages indicating the progress C:\> CHKDSK a:
DOSKEY	Used to recall previous commands using up and down arrow keys	



Self-Activity

Type the following commands and explain what the command is used for and give the output of the command

Sr. No	Command	Explanation	Output
1	copy con my.txt		
2	copy my.txt ab.txt		
3	dir *.txt		
4	ren *.txt *.bak		
5	dir		
6	attrib +h ab.bak		
7	dir *.bak		
8	ver		
9	type my.bak		
10	path		
11	cls		
12	help dir		
13	dir /A:h		
14	help attrib		
15	del my.bak		

Signature of the instructor

Date  /  /



Assessment - work

**Set A**

Give the DOS commands to be used to perform following set of tasks

1.

Sr. No	Task	Command
1	Create a file named a.txt containing your name and address	
2	Change the name of the above file as self.txt	
3	Create a copy of the above file as bio.txt	
4	Display the contents of the file self.txt	
5	Change the file attribute to hidden	

2.

Sr. No	Task	Command
1	Create a file named a.txt containing the college details	
2	Change the name of the above file to college.txt	
3	Create a copy of the file by name course.txt	
4	Display the contents of the file course.txt	
5	Change the file attribute to read only	

3.

Sr. No	Task	Command
1	Display the files which have the extension txt	
2	Rename the extension from txt to doc	
3	Remove all the files created starting with	

	the name 1	
4	Chkdsk any of the drives with display and correction options	
5	Set a new search path	

4.

Sr. No	Task	Command
1	Create a file named a.txt containing names of five students	
2	Change the name of the file to b.txt	
3	Create a copy of the file by name copy.txt	
4	Display the contents of the file copy.txt	
5	Change the file attribute to hidden	
6	Display the current path	

Signature of the instructor

Date

**Set B**

1. By pressing the arrow keys, the commands those have been used can be used again. How is it really being done?
2. Create a file, change it into Read only file. Create one more file with the same name. Are both the files existing or any one is only existing? Why?
3. Display the file content pagewise if it goes more than the page.
4. Set the date to 02-30-09. Does the system accept the date? Why?

Signature of the instructor

Date

**Assignment Evaluation**

0: Not done   
1: Incomplete

2: Late Complete   
3: Needs improvement

**Signature**

4: Complete   
5: Well Done

## Exercise 2

Start Date

/ /



Objective

Creating the directory structure and Batch file in the DOS



Reading

You should read following topics before starting this exercise

1. Complete the previous exercise
2. The concepts of Directories and Batch Files



Ready Reference

Directory system is used for organizing the files. The directory is a group of files stored together and identified by a name. The directories are organized in a hierarchical structure i.e. a directory can contain subdirectories which in turn can contain files and / or more directories.

A batch file is a simple text file with an extension .BAT. It contains a set of DOS commands when the name of batch file is typed at the DOS prompt, all the DOS commands within the file are executed one by one

We will study the dos commands for creating and maintaining directory structure

Command	Used for	Format and Example
MKDIR or MD	It creates a new directory – make directory	MD [drive:][path]<directory name>  C:\> md newdir C:\> mkdir c:\>onedir C:\> md c:\onedir\twodir
CHDIR or CD	Changes the current directory to the specified directory	CD [drive] [path] <directory name>  C:\> cd c:\onedir C:\> cd .. – changes to the parent directory
RMDIR or RD	It is used to remove an empty directory i.e. all the files are already deleted in that directory.	RD [drive:] [path] <directory> C:> rd newdir

### Batch file commands

Command	Used for	Format & Example
@	Does not display command on screen	@ date
ECHO	Used to suppress or display commands in the batch file on the screen	Echo on Echo off Echo hello
REM	Used to add comments in a batch file	REM changing the directory
PAUSE	Used to suspend batch file processing and waits for user to press any key before resuming execution.	Pause [remark] Pause changing the directory
GOTO	Redirects batch processing to the command following the specified label.	GOTO label GOTO end The label is written as :label

IF	Checking conditions before executing a command	If [NOT] string==string2 command If [NOT] exist file command If [Not] errorlevel number command.
SHIFT	All parameters are shifted one position to the left.	

AUTOEXEC.BAT (*automatic execution batch file*) is a special batch file, found in the root directory of the boot disk. This file will automatically run before control of the computer gets turned over to the user.

DOS had an AUTOEXEC.BAT that looked like this:

```
@Echo OFF
Path C:\DOS;C:\;C:\BAT;C:\UTILITY;
Prompt $p$g
Set TEMP=C:\Temp
C:\Utility\NumLock -
CD\
CLS
```

This file sets the PATH, defines a prompt and a temporary directory, runs a utility program, changes to the root directory and then clears the screen.



Type in the following set of commands to create a batch file named mydir.bat

```
copy con mydir.bat
echo *** Batch file for creating directories ****
pause
mkdir fy sy ty
chdir fy
@echo off
mkdir morning evening
cd ..\ty
mkdir batch1 batch2
cd ..
^Z
```

Execute the batch file mydir.bat by typing mydir at the prompt. Use dir and cd command to view the directory structure created

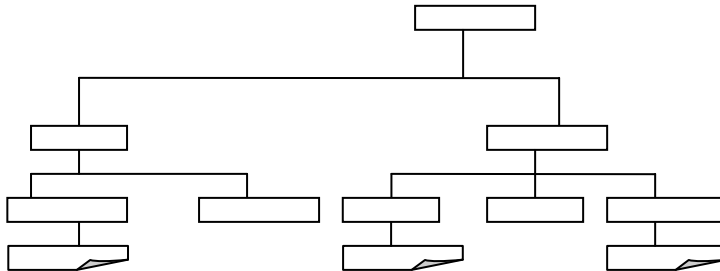
Signature of the instructor

Date



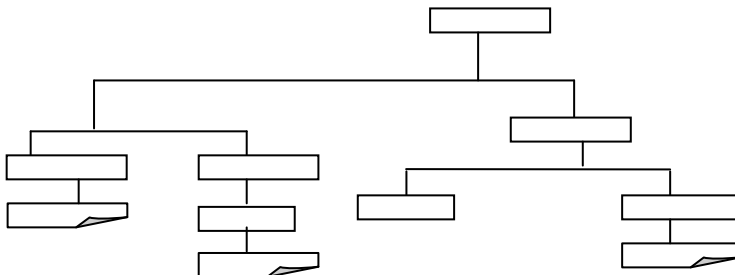
### Set A

1. Create the following Directory Structure in the current directory containing directories and file and also remove it. Write down the commands used for the exercise



Where  is a directory and  is a file.  
Instructor should fill in the blanks with appropriate values.

2. Create the following Directory Structure in the current directory and also remove it. Write down the commands used for the exercise.



Where  is a directory and  is a file.  
Instructor should fill in the blanks with appropriate values.

Signature of the instructor

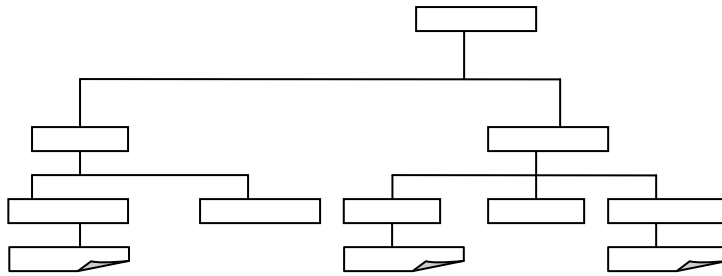
Date  /  /

### Set B

Create batch files to perform the following tasks

1. Accepts two filenames as parameters.
  - (i) If the first file exists, then: Display its contents.  
If second exists, then copy contents of first to second.  
otherwise rename first to second.
  - (iii) If first does not exist, then: Create it.  
If second does not exist, copy contents of first to second  
otherwise delete second file

2. Create the following directory structure by passing dummy parameter to batch file.



Instructor should fill in the blanks with appropriate values.

3. Create a BACKUP directory with two directories TXT and BAT. Copy all batch files with .bat extension to BAT directory, all files with .txt extension to TXT directory. Delete all files with .bat extension. Give appropriate message and pause before deleting the file.

Signature of the instructor

Date

**Set C**

1. Create a new directory with a new.txt file in it. Change the attrib to hidden. Now use the dir command to view the contents of the file. What are the contents you see? Why? Can you use a different command to get the actual directory contents?

Signature of the instructor

Date

**Assignment Evaluation**

0: Not done

2: Late Complete

1: Incomplete

3: Needs improvement

**Signature**

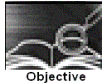
4: Complete

5: Well Done




**Exercise 3**

**Start Date** / /



Using Windows XP graphical user interface (GUI) and Windows explorer.



You should read following topics before starting this exercise



1. Windows XP Operating System Introduction
2. Main keywords associated with Microsoft Windows XP
3. Know the various features of the Graphical user Interface.
4. Know the Windows explorer.





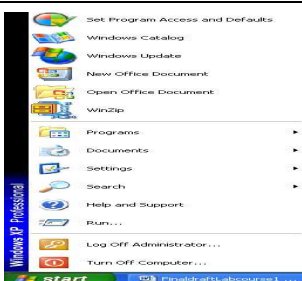





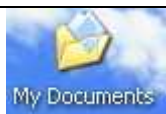

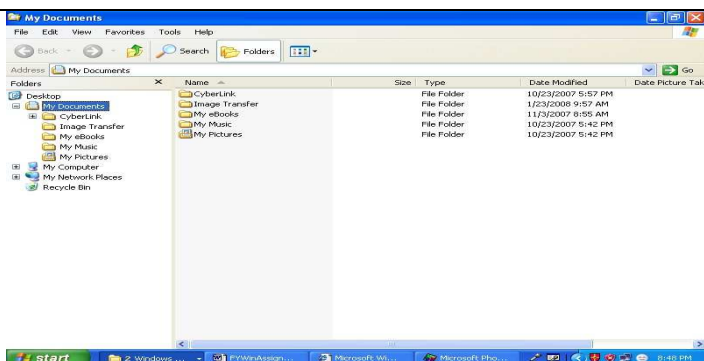
Microsoft has been making OS software utilizing graphical user interfaces since around 1985. Some of the earlier windows versions were Windows 3.1 (1990), Windows 95 (1995), Windows 98 (1998), Windows ME (2000), Windows 2000 (2000), and Windows XP (2001). Windows XP comes in two bundles Windows XP Professional and Windows XP for home users. Windows 2000 and Windows XP are personal operating systems when used as stand alone machines but can be considered network operating systems when connected to a network. An operating system is a collection of programs, which enables the entire pc to work. Some of the tasks that are performed by Windows are:



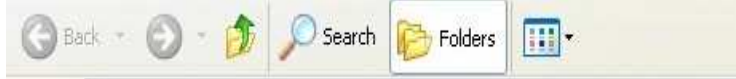


1. Assisting in starting and shutting down of a pc.
2. Controlling and handling the hardware, including RAM, I/O cards etc.
3. Providing a graphics user interface including various features.
4. Provides a platform for applications to execute like Word.
5. File Handling.
6. Provides an interface for various tools like Internet explorer.

**Main Keywords Associated With Microsoft Windows XP**

Name	Picture & Description
Drives	 Local Disk (C:) <p>Drives are devices used to store data. Most computers have at least two drives: hard drive C:\ (Which is the main storage and a floppy drive or a CD drive (which stores smaller volumes of data) The hard drive is typically designated as C:\ drive and the floppy drive is typically designated as A:\ drive. You will also have other drives typically labelled D:\ or F:\ or H:\ or G:\</p>
Folders / Directory	 <p>Folders are used to organize the data stored on your drives. A Directory is the path given to a folder on a drive. For example a text file called <i>Hello World</i> is located in the <i>My Documents</i> directory on the C:\ drive. It would therefore read <b>"C:\My Documents\Hello World.txt"</b></p>
File Extensions	<p>File Extensions are the ending letters associated with a file and an application that it can be manipulated in. This way Windows knows to tell which program to open the file you want to manipulate. For example a text file has an extension</p>

	<p>of <i>.txt</i>, so a text file created in <i>Notepad</i> called <i>Hello World</i> would look like - <b>Hello World.txt</b>. You do not have to assign a file extension to a file that you create. The program you use will automatically do this for you. All you need to do is give it a filename. Some other common extensions are as follows:</p> <ul style="list-style-type: none"> <li>• .doc = Microsoft Word Document</li> <li>• .xls = Microsoft Excel Document</li> <li>• .ppt = Microsoft PowerPoint Presentation</li> <li>• .mdb = Microsoft Access Database</li> <li>• .bmp = Windows Bitmap Picture</li> <li>• .wav = Sound File</li> <li>• .html or .htm = Internet Document</li> </ul>
<p>Icon</p>	 <p>An Icon is a graphic image. Icons help you to execute the application programs quickly. Commands tell the computer what you want the computer to do. To execute the application program by using an icon, double-click on the icon.</p>
<p>Desktop</p>	 <p>After starting your computer, the desktop is the first thing that you see with some background image displayed on the screen with icons for various programs. The desktop is the area you work in.</p>
<p>Taskbar</p>	 <p>The taskbar is usually located on the bottom of the desktop. The Start button, active program buttons, and the system tray are located on the Taskbar</p>
<p>Start Menu</p>	<div style="display: flex; justify-content: space-around;"> <div data-bbox="416 1547 791 1827">  <p>Start menu</p> </div> <div data-bbox="895 1547 1198 1827">  <p>Classic start menu</p> </div> </div> <p>Start menu guides you how to start with the various application programs that are available on your windows system.</p>


	<p>We can choose the view of start menu by right clicking on the start button → properties → start menu.</p>																														
System Tray	 <p>The System Tray is usually located in the lower right hand corner of the Windows Desktop. The system tray contains a display of the current computer time, and the icons representing the programs activated when Windows first starts up. These are the background running applications required for smooth running of windows.</p>																														
My Computer	 <p>My Computer icon provides access to the different parts of your computer. You can access the different drives (Hard Drive, Floppy Drive, and Network Drives) inside My Computer.</p>																														
Recycle Bin	 <p>When you delete an object, just by pressing Del key, Windows XP sends it to the Recycle Bin. You can restore objects that are located in the Recycle Bin or you can permanently delete them by right clicking on the Recycle Bin and select <b>Empty Recycle Bin</b>.</p>																														
My Documents	 <p>The My Documents folder is nothing more than a regular folder that resides on your Windows Desktop. However, it offers an easy-to-reach location where you can store and retrieve important data, and the icon is always available in Windows explorer and on the desktop. You can double-click My Documents icon, click the File menu, point to New and click Folder to create folders. This is the default destination folder offered by windows system where the entire user created documents gets stored.</p>																														
Internet Explorer	 <p>The Internet Explorer icon launches the Internet Explorer browser. The Internet Explorer browser is what you will use to access the Internet and the World Wide Web.</p>																														
Window	 <p>The screenshot shows a Windows Explorer window titled 'My Documents'. The address bar shows 'My Documents'. The left pane shows the folder tree with 'My Documents' selected. The right pane shows a list of folders: CyberLink, Image Transfer, My eBooks, My Music, and My Pictures. A table below the list provides details for each folder:</p> <table border="1"> <thead> <tr> <th>Name</th> <th>Size</th> <th>Type</th> <th>Date Modified</th> <th>Date Picture Taken</th> </tr> </thead> <tbody> <tr> <td>CyberLink</td> <td></td> <td>File Folder</td> <td>10/23/2007 5:57 PM</td> <td></td> </tr> <tr> <td>Image Transfer</td> <td></td> <td>File Folder</td> <td>11/23/2008 9:57 AM</td> <td></td> </tr> <tr> <td>My eBooks</td> <td></td> <td>File Folder</td> <td>11/3/2007 8:55 AM</td> <td></td> </tr> <tr> <td>My Music</td> <td></td> <td>File Folder</td> <td>10/23/2007 5:42 PM</td> <td></td> </tr> <tr> <td>My Pictures</td> <td></td> <td>File Folder</td> <td>10/23/2007 5:42 PM</td> <td></td> </tr> </tbody> </table>	Name	Size	Type	Date Modified	Date Picture Taken	CyberLink		File Folder	10/23/2007 5:57 PM		Image Transfer		File Folder	11/23/2008 9:57 AM		My eBooks		File Folder	11/3/2007 8:55 AM		My Music		File Folder	10/23/2007 5:42 PM		My Pictures		File Folder	10/23/2007 5:42 PM	
Name	Size	Type	Date Modified	Date Picture Taken																											
CyberLink		File Folder	10/23/2007 5:57 PM																												
Image Transfer		File Folder	11/23/2008 9:57 AM																												
My eBooks		File Folder	11/3/2007 8:55 AM																												
My Music		File Folder	10/23/2007 5:42 PM																												
My Pictures		File Folder	10/23/2007 5:42 PM																												

	Every application when executed opens a window.
Window Title Bar	 Title bar shows the name of the Application we are in.
Menu Bar	 The <b>menu bar</b> contains the menus that will allow us access to all the operations that can be done with a file or folder
Standard Bar	
Tool Bar	 Optional. This bar includes the most commonly used buttons
Minimize, Maximize, Restore, Close	 These are the series of buttons which are present on the top right hand corner of every window.

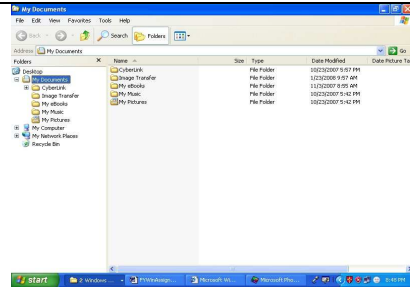
## Windows Explorer

Windows Explorer is the basic shell or user interface or an indispensable tool in the operating system, with the help of which we can organize and control the files and folders of the different storage systems at our disposal such as the hard drive, disk drive, etc. Its properties and characteristics are something we deal with every time we use the computer. The Windows Explorer is also known as the File Manager. Through it we can delete, view, copy, or move files and folders.

## Exploring the explorer

Name	Picture and description
Starting the windows explorer	 <p>The fastest way to get to the explorer is by pressing key combination windows key + e using the modern keyboards or by right clicking the <b>start</b> button and selecting the <b>explore</b> menu option as shown above</p> <p>The other would be click <b>start -&gt; programs -&gt; accessories -&gt; windows explorer</b></p>

## Components of the explorer



The explorer consists of two sections. On the left side there is the directory tree, which is the list of units and folders in the system. Only units and folders appear and no files. On the right side there is another section, which will show the contents of the folder that we have selected in the left section. Depending on the type of view that we have activated, we will see different type of information regarding the files. In detailed view, we see the name, size, type, and date of last modification for each file. The windows explorer view can be customized according to each user. There is a **View menu option** available, with the help of which each user can select his/her own way of displaying files and folders



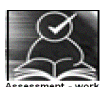
Self-Activity

1. Click on **Start** button. Select **Search -> For Files And Folders**. Search for the file \_\_\_\_\_ and write down the entire path of the file
2. Click on **Start -> Run**. Browse to the \_\_\_\_\_ application in \_\_\_\_\_ folder and execute the application
3. Right click on the desktop and list down the menu items.
4. Right click the **My documents** folder and view its properties
5. Create folder for you in the **My documents** folder. Right click the folder and view its properties and write down its path and size
6. Click on **Start** button. Click control panel and write any three parts
7. Right click on the desktop. Select **New -> Shortcut** and browse to create a new shortcut for \_\_\_\_\_ application.
8. Use All programs in start menu, point to accessories and write down the options available in system tools (If you are in classic start menu change it to start menu before executing this command).
9. Right click on the taskbar and list down different menu items.
10. Open the explorer as directed above and list down all the menu items on the menu bar.
11. Click on the **view** menu option and list out the different appearances of **thumbnails, tiles, icons, list, and details** options.
12. Select the \_\_\_\_\_ folder and write down the information given in the status bar. (Note: If the status bar is not to be seen you would need to make it available by selecting **View** option).
13. Double click the **computer** icon on the desktop. Check which window is opened. Is it similar to the windows explorer? What is the difference between the two?

Instructor should fill in the blanks with appropriate values.

Signature of the instructor

Date



Assessment - Work  
**Set A**

1. Use Notepad option available in accessories to create a file and save it in the folder created by you
2. Use Paint option available in accessories to create an image and save it in the folder created by you
3. Use Disk Defragmenter tool from system tools
4. Use control panel to change the screen saver .
5. Right click the task bar and select the Task Manager option. Name the applications that are currently running.
6. Double click on the time located at the bottom right corner on the system tray. Set the time zone to \_\_\_\_\_. How much is the time difference between \_\_\_\_\_ and Indian time zone?
7. What happens when you select the **Run Desktop cleanup wizard** by right clicking the desktop and selecting the **arrange icons by** option?

Instructor should fill in the blanks with appropriate values.

Signature of the instructor

Date

### Set B

1. Create a new word document in the selected folder through the **File** menu option.
2. By right clicking the newly created file list out the **properties**.
3. Right click the newly created file. Select the \_\_\_\_\_ option (e.g. cut, copy, send to etc.) and perform the specified operation and observe the results.
4. Click **MyComputer** on the left hand side panel. Right click on any drive and select **sharing and security** option. Select the **sharing** tab and do the \_\_\_\_\_ settings.
5. Share the folder created by you by right-clicking on the folder. Use control panel -> administrative tools to see the shared folders. In which option of the administrative tools can you see the shared folders.
6. Customize the entire explorer by selecting / deselecting various toolbars from **view → toolbars**.

Instructor should fill in the blanks with appropriate values.

Signature of the instructor

Date

### Assignment Evaluation

### Signature

0: Not done	<input type="text"/>	2: Late Complete	<input type="text"/>	4: Complete	<input type="text"/>
1: Incomplete	<input type="text"/>	3: Needs improvement	<input type="text"/>	5: Well Done	<input type="text"/>

## Exercise 4

Start Date

/ /



Using basic Linux commands



You should read following topics before starting this exercise

1. UNIX and LINUX operating system
2. cat with options, ls with options, mkdir, cd, rmdir, cp, mv, cal, pwd, wc, grep with options, I/O redirection using >, >>, <, | etc.



### About UNIX and LINUX

The success story of UNIX starts with the failure of the MULTICS project. The project failed and the powerful GE-645 machine was withdrawn by GE. Two scientists at Bell Labs, Ken Thompson and Dennis Ritchie, who were part of the MULTICS team, continued to work and succeeded and named their Operating system UNIX, a pun on MULTICS.

The machine available at Bell Labs was a DEC PDP-7 with only 64 k memory while the Operating system they were developing was meant for a larger machine. The problematic situation was handled with an innovative solution. They developed most part of the software in a higher level language, C, which helped them in porting their Operating system from one hardware to another.

With the growing popularity of UNIX, it was available on a variety of machines, from personal computers to mainframes. The most popular amongst them was UNIX System V from AT&T.

Each big player in the market came up with their own versions of UNIX. IBM had its own version of UNIX called AIX, which was used on high-end servers. Sun's version of UNIX called Solaris was used on Sun workstations. Novell marketed UnixWare along with Netware, its Network operating system.

LINUX is a version of UNIX, which though it resembles UNIX in looks and feels but differs from other versions in the way it was developed and distributed. In contrast to large proprietary UNIX versions, Linux was developed by Linus Torvalds, a Finnish student. He made the source code available and invited partners via the internet in his development effort. He got professional help from all quarters and Linux evolved rapidly. It was made freely available for everyone to use. Linux that was initially meant for Personal computers is now available for a variety of hardware platforms, from mainframes to handheld computers

Linux supports multiple users. Every user need to have an account in order to use the system. One of the users called system administrator (root) is given the charge of creating user accounts and managing the system normally works on the “#” prompt.

You will be given a username and password, using which you can login into Linux operating system. For computer users, the operating system provides a user-command interface that is easy to use, usually called the **Shell**. The user can type commands at the shell **prompt** and get the services of the operating system. Linux operating system shell has the “\$” prompt.

You can open a system terminal that gives you a \$ prompt where you can type in various shell commands.

LINUX system will usually offer a variety of shell types:

- sh or Bourne Shell: the original shell still used on UNIX systems and in UNIX-related environments. It is available on every Linux system for compatibility with UNIX programs.
- bash or Bourne Again shell: the standard GNU shell, is the standard shell for common users on Linux and is a *superset* of the Bourne shell.
- csh or C shell: the syntax of this shell resembles that of the C programming language.
- tcsh or Turbo C shell: a superset of the common C shell, enhancing user-friendliness and speed.

- ksh or the Korn shell: A superset of the Bourne shell

All LINUX commands are case sensitive single words optionally having arguments. One of the argument is options which starts with “-“ sign immediately followed by one or more characters indicating option. The wild-cards or metacharacters “\*” and “?” have similar meaning as in DOS. The “\*” character matches any number of characters while “?” matches a single character. The backquote “ ` ” is another metacharacter. Shell executes the command enclosed in backquote in its place. Any wild-card is escaped with a \ character to be treated as it is

### Shell Variables

There are number of predefined shell variables called system or environment variables which are set by the system when the system boots up. Some important system variables are

PATH	It contains set of paths where the system searches for an executable file
HOME	It is the home or login directory where the user is placed initially
PS1	It is the primary shell prompt which is usually \$
PS2	It is the secondary shell prompt which is usually >

### Linux Files and directories

Linux defines three main types of files. Linux treats all devices also as files.

Ordinary or regular file	A file containing data or program
Directory file	A file containing the list of filenames and their unique identifiers
Special or device file	A file assigned to a device attached to a system

Linux files may or may not have extensions. A file can have any number of dots in its name. Linux file names are case sensitive. The root directory represented by / is the topmost directory file containing number of subdirectories which in turn contains subdirectories and files

### Shell Commands

The following is the list of shell commands

Command	Used for	Example
date	Displays both date and time The command can be used by the system administrator to change date and time.	\$date Format specifiers can be used as arguments %m month in integer format %h Name of the month %d Day of the month %y Last two digits of the year %H hours %M Minutes %S Seconds \$date +%H \$date +%h %m"
cal	Displays the calendar	\$cal 8 2007 Displays the calendar for the month august of year 2007 \$cal aug Displays the calendar for the month august of current year



cat	Displays the contents of the files used with the command	<p>\$cat</p> <p>Displays immediately what is typed when you hit enter key</p> <p>\$ cat &gt; abc.txt</p> <p>Whatever number of lines typed till you press ^D are placed in abc.txt file</p> <p>\$cat abc.txt</p> <p>Displays contents of file abc.txt</p>
ls	Displays the contents of current directory. A single dot ( . ) stands for the current directory while a double dot( .. ) indicates the parent directory	<p>\$ls</p> <p>lists all files in the current directory</p> <p>\$ls -a</p> <p>Lists also the hidden files</p> <p>\$ls -l</p> <p>Lists the permission information along with other information such as date of last modification, size in blocks etc. The first column of the output exhibits the file type and permissions.</p> <p>File type: -, d, b respectively for ordinary, directory and block device file.</p> <p>Permissions are of the form r, w, x, - i.e. read, write, execute and none respectively.</p> <p>There are three groups of rwx. Owner, group and public.</p>
mkdir	Creates specified directory in the current directory, fails if a file or directory by that name is already present or user is not having permissions to create a directory	<p>\$mkdir bin</p> <p>Creates bin directory</p> <p>\$mkdir dir1 dir2 dir3</p> <p>Creates three directories dir1, dir2 and dir3</p>
cd	Switches to specified directory, fails if user is not having permissions to access the directory	<p>\$cd /</p> <p>Switches to root directory</p> <p>\$cd</p> <p>Changes to HOME directory</p>
rmdir	Removes specified directory fails if the directory is not empty	<p>\$rmdir dir1</p> <p>Removes dir1 directory</p> <p>\$rmdir dir2 dir3</p> <p>Removes dir2 and dir3 directories</p>
cp	Creates an exact copy of a file with a different name	<p>\$cp abc.txt xyz.txt</p> <p>Copies abc.txt into a new file named xyz.txt</p> <p>\$cp abc.txt bin</p> <p>Copies abc.txt into a new file with the same name in bin directory</p>
mv	It renames a file or moves a group of files to a different directory	<p>\$mv xyz.txt pqr</p>
rm	Deletes specified file. It can be used	<p>\$rm pqr</p>

	with wildcards * and ? as in DOS, to delete all files of a specified type	
pwd	Displays the path of your present working directory	\$pwd displays the directory in which you are currently working
wc	Counts words, lines and characters or bytes	\$wc -c abc.txt Displays the number of bytes in the file abc.txt \$wc -l abc.txt Displays the number of lines in the file abc.txt \$wc -w abc.txt Displays the number of words in the file abc.txt \$wc abc.txt Displays the number of bytes, words and lines in the file abc.txt
grep	The syntax is grep options pattern filename It displays the lines in the file in which the pattern is found	\$grep Agarwal names.txt Displays lines in the names.txt where the string "Agarwal" is present \$grep -n Agarwal names.txt Displays lines along with line numbers in the names.txt where the string "Agarwal" is present
man	Offers help on the shell command	\$man ls Shows entire manual page of Linux manual pertaining to ls command
passwd	It is used to change the password	\$passwd When invoked by an ordinary user asks for the old password and then demands typing and retyping of new password #passwd user1 Used by administrator to change the passwd of user1
echo	Displays its arguments compressing the spaces. To preserve the spaces the words should be placed within quotes	\$echo \$HOME \$echo \$PATH \$echo eats up the spaces \$echo The date to-day is `date` \$echo You can multiply using \*
who	Displays list of users currently logged in	\$who
tail	Displays last lines of the file	\$tail -3 abc.txt Displays last three lines of file abc.txt
head	Displays top lines of the file	\$head -5 abc.txt Displays top five lines of file abc.txt

## Redirection and pipes

The most of the above commands take some input, do some processing and give the output or give error message in case there is some error. For example the cat command is usually given as `$cat filename`. Here cat command takes input from file named filename and gives output on the console. If the file is not present then it gives appropriate error message. By default the cat command writes the output or error message to the console. If we just type cat command without any filename, it will wait for user to type characters that means, it by default is expecting input also from console. The default files where a command reads its input, sends its output and error messages are called standard input(stdin), standard output(stdout) and standard error(stderr) respectively.

By default all the above three files are attached with the terminal on which the command is executing. Therefore, every command, by default, takes its input from the keyboard and sends its output and error messages to the display screen. Redirection is used to detach default file from the command and attach some specific file. Pipes allow you to send output of one command as input to the other command. The commands that are connected via a pipe are called filters

Command	Symbol	Description	Format & Examples
Input Redirection	<	It detaches the keyboard from the standard input of command and attaches specific file	<code>\$cat &lt; abc.txt</code> Takes its input from abc.txt and the output by default is on console. The effect is same as <code>\$cat tempfile</code>
Output Redirection	>	It detaches the console from the standard output of command and attaches specific file	<code>\$cat &gt; file1</code> Takes its input from keyboard by default and writes the output to file1, effectively whatever typed at the keyboard goes into tempfile <code>\$cat file1 abc.txt &gt; file2</code> The contents of file1 and abc.txt will be concatenated and send to file2 <code>\$cat file1 &gt; /dev/lp0</code> The contents of file file1 will be sent to printer instead of console
Output Redirection without overwriting	>>	In output redirection the file is cleared before writing to it. The >> is used so that output is appended and not overwritten	<code>\$cat file1 &gt; file1</code> The file1 contents will be cleared <code>\$cat file2 &gt;&gt; file2</code> The file2 will have its contents appended to it
Pipe		The pipe character   is used between two commands so that output of first command is send as input to the second command	<code>\$ ls -l   grep "abc"</code> Displays the line in the output of ls -l containing pattern abc



Execute all the commands given in the example column of all the tables above in the same order and understand the usage of the commands

Signature of the instructor

Date  /  /



### Set A

1 Using cat command, create a file named 'names.txt' containing at least ten names and addresses of your friends ( firstname , surname, street name, cityname ). Type the following commands and explain what the command is used for and give the output of the command

Command	Explanation	Output
wc -lw names.txt		
mkdir ass1 ass2		
cp names.txt ass2		
cp names.txt list		
tail -3 list		
rmdir ass2		
cd ass2		
rm names.txt		
cd		
pwd		
ls -l		
mv list list.txt		
grep ___ names.txt		

2 Using cat command create a file named college.txt containing at least ten names and location of colleges ( collegename, place , pincode ). Type the following commands and explain what the command is used for and give the output of the command

Command	Explanation	Output
mkdir s1 s2 s3 s4		
cp college.txt coll		
cp college.txt coll s1		
head -5 coll		
grep -n _____ college.txt		
rmdir s3 s4		
cd s1		
rm coll		
pwd		
cd		
mv coll xy.txt		
rm *.txt		
ls -a		

Signature of the instructor

Date  /  /

### Set B

Give the commands to perform the following actions and give the output

- 1 List the last three lines of the file \_\_\_\_\_
- 2 Create a file named \_\_\_\_\_containing abc.txt appended to itself
- 3 Display the current month(string) and year
- 4 Display the home directory followed by path
- 5 Write the contents of directory to a file
- 6 Append at the end of a file no of lines and the name of the file
- 7 Create a file named Manualcp containing manual for cp command

Signature of the instructor

Date  /  /

**Set C**

Give the commands to perform the following actions and verify by executing the command

- 1 Display the number of lines containing pattern "\_\_\_\_" in first five lines of the file \_\_\_\_\_
- 2 Display the calendar of current month
- 3 Store the number of users logged-in in a file \_\_\_\_\_
- 4 Create a file containing first three and last three lines of a file.
- 5 Create a file containing word count of each and every file in the current directory plus a total at the end.
- 6 Create a single file containing the data from all .txt files in the current directory.

Signature of the instructor

Date

**Assignment Evaluation**

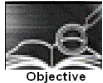
**Signature**

0: Not done	<input type="text"/>	2: Late Complete	<input type="text"/>	4: Complete	<input type="text"/>
1: Incomplete	<input type="text"/>	3: Needs improvement	<input type="text"/>	5: Well Done	<input type="text"/>

## Exercise 5

Start Date

/ /



### Using vi editor



You should read following topics before starting this exercise

1. Three modes in which vi editor works
2. Commands in vi input mode for inserting, replacing, saving and quitting.
3. Commands in vi for deleting, paging and scrolling,
4. Undoing last editing instructions, search and replace



Editor vi was developed by the University of California at Berkeley and is also supplied with the Berkeley distribution of the UNIX system. We are dividing the discussion into three parts – Introduction to vi, useful commands of vi and advanced and miscellaneous vi commands. We will first look at the table exhibiting the summary of vi Commands and then we will see the detailing of the vi commands.

Sr. No.	Command	Meaning	Sr. No.	Command	Meaning
<b>Using vi command</b>			<b>Delete and change</b>		
*1.	vi file	Edit file	*16	dd	Delete line
*2.	vi -r file	Recover file from crash	*17	cc	Change line
<b>Basic Cursor motions</b>			18	D	Delete from cursor to EOL
*3.	h j k l	←, ↓, ↑, →,	19	C	Change from cursor to EOL
4.	CR	Down line to first non-blank	*20	x	Delete character
5.	0 (Zero)	Beginning of line	*21	s	Change character
6.	\$	End of line (EOL)	22	S	Change line
<b>Screen Control</b>			*23	rchr	Replace current chr with <i>chr</i>
7.	^U ^D	Up or Down half page	24	R	Overprint change
8.	^B ^F	UP or Down whole page	<b>Word Commands</b>		
9.	^L	Reprint page	*25.	w	Next word
<b>Character input modes</b>			*26.	b	Back word
*10.	a	Append after cursor	*27.	e	End of word
11.	A	Append at end of line	*28.	dw	Delete word
*12.	i	Insert before cursor	*29.	cw	Change word
13.	I	Insert before first non-blank	<b>Generic commands</b> <i>object</i> is any cursor motion: w for word; b back word; h,j,k,l for left, down, up, right; /string for up to <i>string</i> etc.		
*14.	o	Add lines after current line	*30.	dobject	Delete object
15.	O	Add lines before current line	*31.	cobject	Change object
<b>Search</b>			<b>Control Commands</b>		
32.	/string/	Search for string	*42.	:w	Write file
33.	?string?	Reverse search for	*43.	:wq	Write file and quit

		string			
*34	n	Repeat last / or ?	*44.	:q	Quit
*35	N	Reverse of n	*45.	:q!	Quit (override check)
<b>Miscellaneous</b>			*46.	:ed-cmd	Run the ed command <i>ed-cmd</i>
*36.	u	Undo previous command	*47.	:num	Go to line <i>num</i>
37.	U	Restore entire line	48.	ZZ	Same as :wq
*38.	Yobject	Save object in temp buffer			
39.	Y	Save line(s) in temp buffer			
*40.	p	Put saved buffer after cursor			
41.	P	Put saved buffer before cursor			

\* Indicates all the characters of the command are in lower case.



#### Entering into vi

You can run the vi command just as normal Unix/Linux command. As a result you will get the screen, printing about the file name, number of lines and number of characters at the *bottom* of your screen.

\$ vi names ↵ where (↵) is an enter key throughout this documentation.

```

_
~
~
~
~
~
~
~
~
~
~
~
~
~
"names" [New File]           0,0-1           All

```

Linux is a case sensitive operating system. **“c” is not “C”**

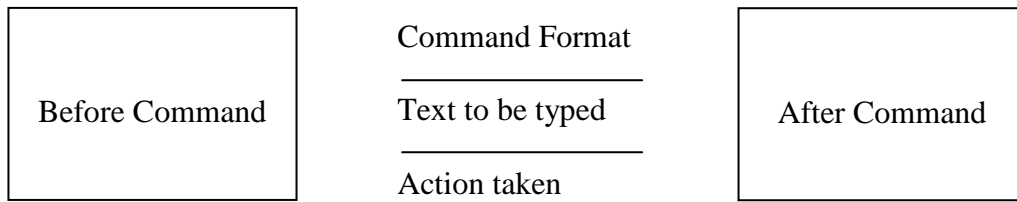
The cursor is shown as `_` and is placed in the upper left corner when vi starts. A `~` in the first column indicates the file doesn't have enough lines to fill up the screen. The bottom line is the message line.

The vi editor uses two types of mode to deal with the file operations: insert mode and command mode.

The vi editor allows getting into insert mode by pressing a respective characters cause to enter into insert mode. It also allows getting into command mode by pressing ESC key or `：“` character sequence. The ESC key is to come out from the current activity and `：“` character behaves like a prompt where you execute the commands of vi editor. We are discussing vi editor command in the control command section and into other few sections.

**N.B. vi editor follows two modes – insert mode and the command mode.**

We are following the demo in the sequence shown below.



- **Commands related to insert mode**
- **Adding text**

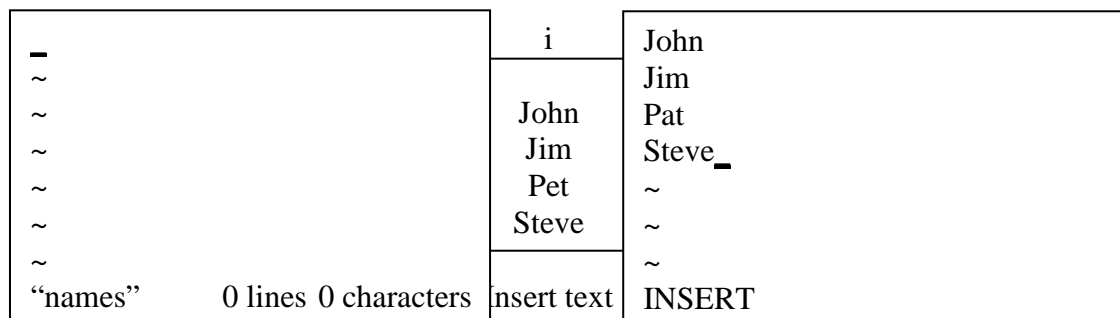
Getting more comfortable with the moving around the screen, we are now trying to add some text. To add the text into the file through vi editor you need to enter into the insert mode. The insert mode of the vi editor follows two scenarios.

Using the “i” (Insert) command

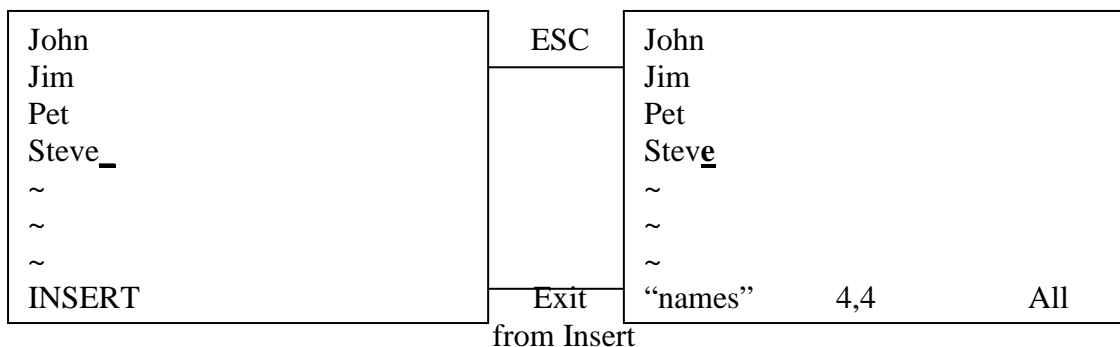
Using the “a” (append) command

#### Inserting a text using “i” (insert) command

By pressing the “i” character you can have the insert mode of the vi editor. The characters typed by you are placed *before* the current character position. To come out from the insert mode, required to press an ESC key.

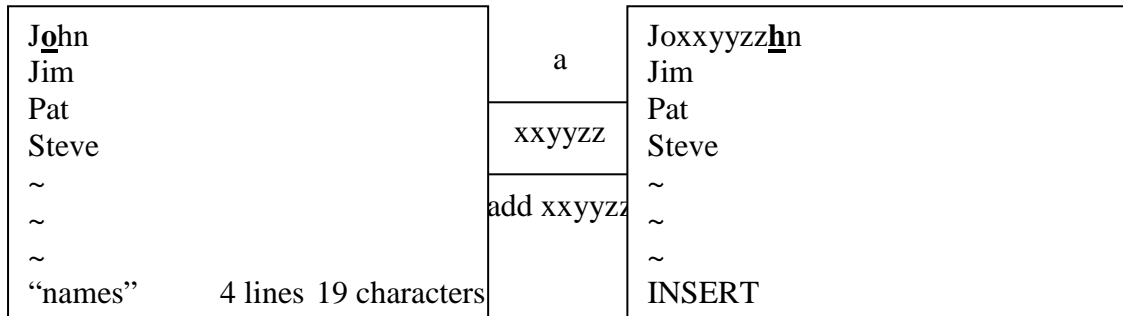


After the ESC is pressed, the cursor moves back to the last character inserted, just as with the “a” command.

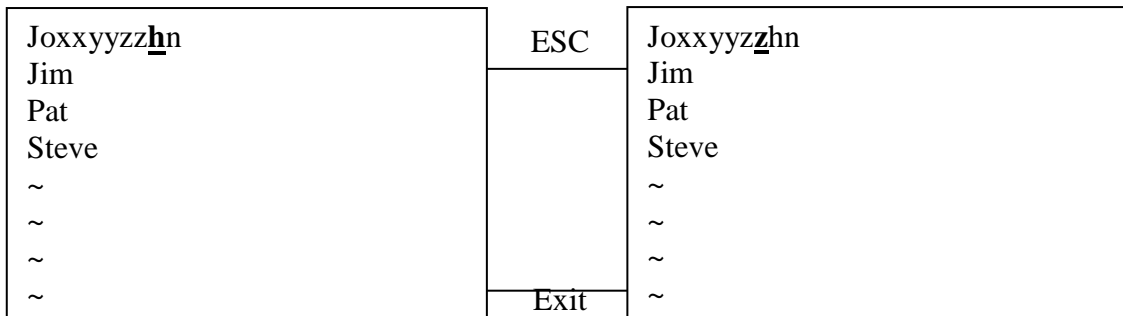


To add text you position the cursor over a character and press an “a”. This puts you in a special mode of operations called “insert mode”. Now every thing typed is appended to the text *after* the character the cursor was positioned over:



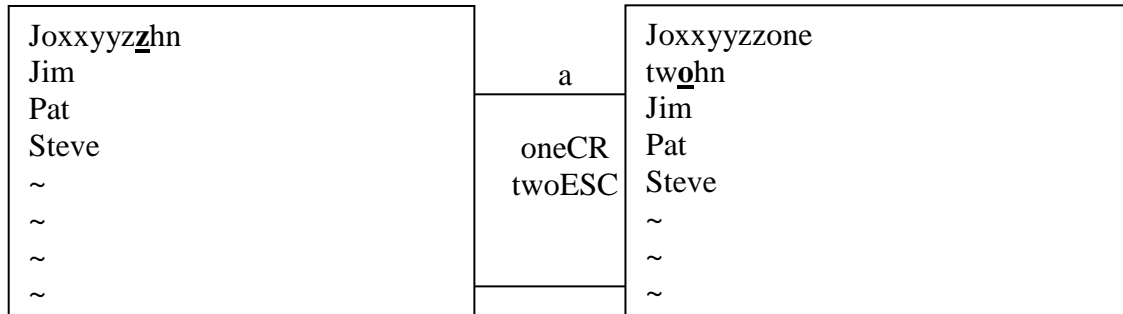


When you are done adding text, you press the ESC key. When you press ESC key, the cursor moves back to the last character you entered.



from Append

You can even put ↵ RETURNS (CR) in the added text, and new lines appear.



*embedded CR*

The appending started between the z and hn of the first line, causing the hn to be carried to the next line when the CR i.e. (↵) was pressed.

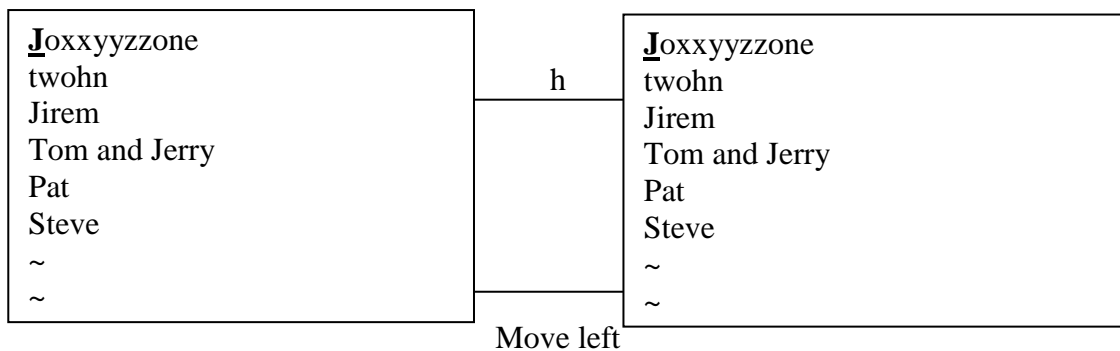
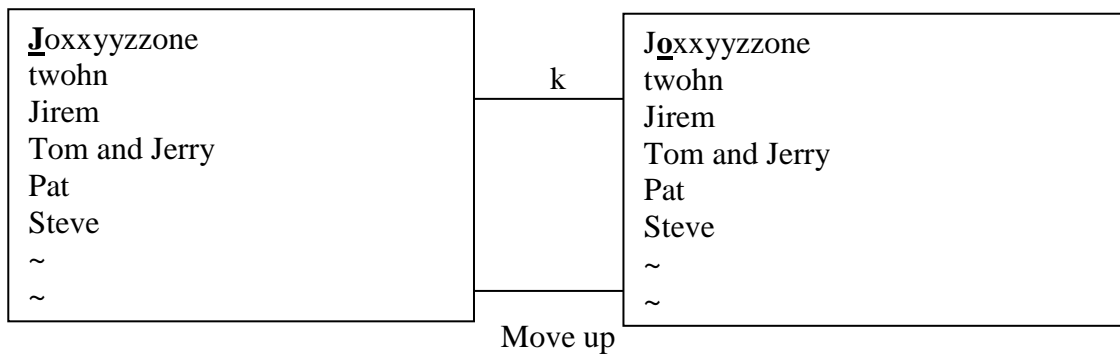
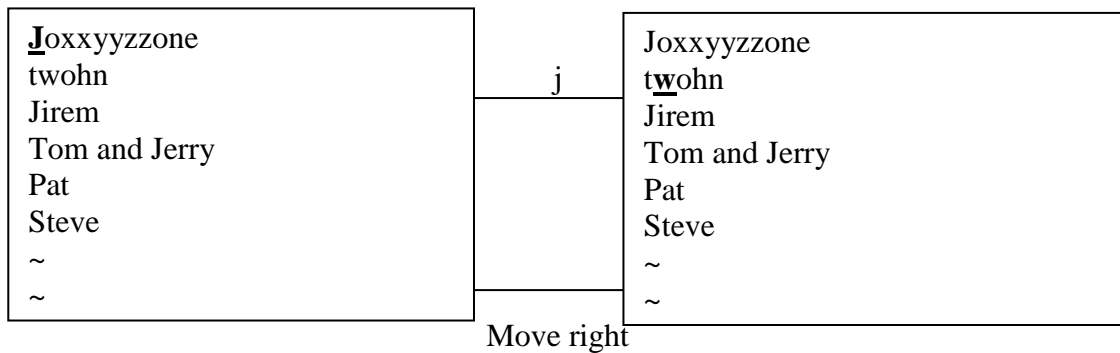
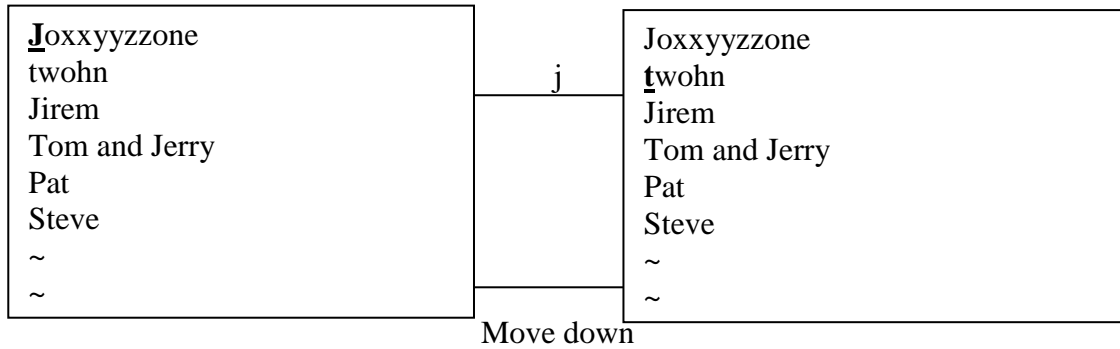
1. Perform the following changes to your file. Specify the command and the resulting text as a answer.

Action	Command typed	Result
Change Jim in line 3 to Jirem		
Insert a new line "Tom and Jerry" after line number 3.		
Insert a new line at the end		

### Moving Around

This is very essential to know about how to move the cursor around the screen to make additions or changes. The basic screen motion commands are h,j,k and l, situated next to each other on the

right side of the keyboard. The motions for h,j,k and l are left, down, up and right, respectively. H = ←, j = ↓, k = ↑, l = →.



You can precede these keys with numbers, which allows you to move more than one column or line at a time. Command is *nj*, *nh*, *nk* or *nl*. For example

3j – move 3 lines down.

3h – move 3 columns left.

3k – move 3 lines up.

3l – move 3 columns right.

If you try to move past the beginning or end of file, vi will “beep” at you.

**Consider the file contents given below.**

```

Joxxyzzzone
twohn
Jirem
Tom and Jerry
Pat
Steve
~
~

```

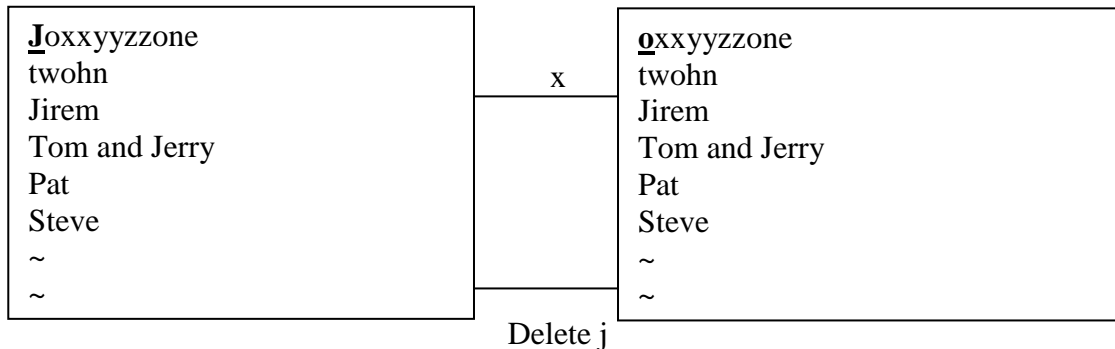
2. Perform the following operation with your file by specifying the command and the resulting text as answer.

Action (from current cursor position)	Command typed	Result
Move __ lines down		
Move __ columns right		
Move __ columns left		
Move __ lines up		

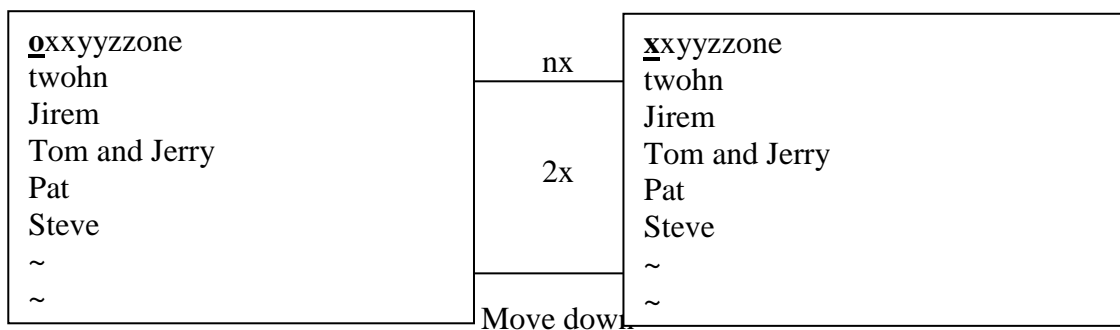
• **Deleting Text**

Here we are focusing on how to delete a text. There are two commands that delete text in vi: *x* and *d*.

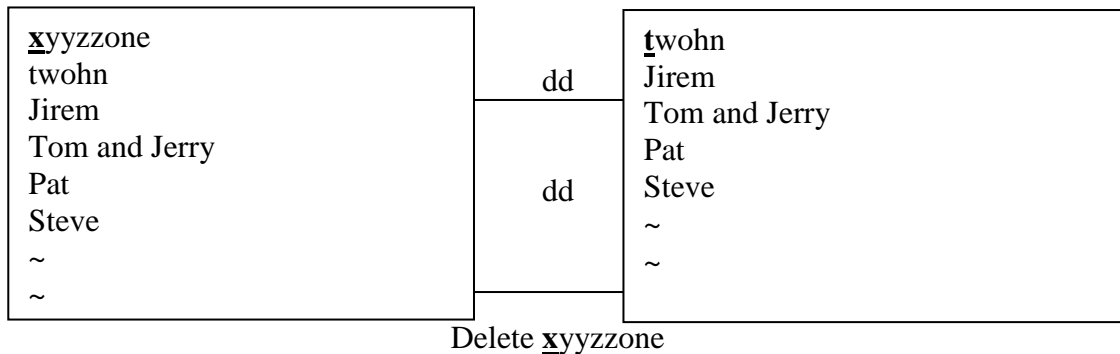
To delete one character, you use the “*x*” command. “*x*” deletes the character at the end current cursor position, moving the rest of the line left into the void created by the deleted character.



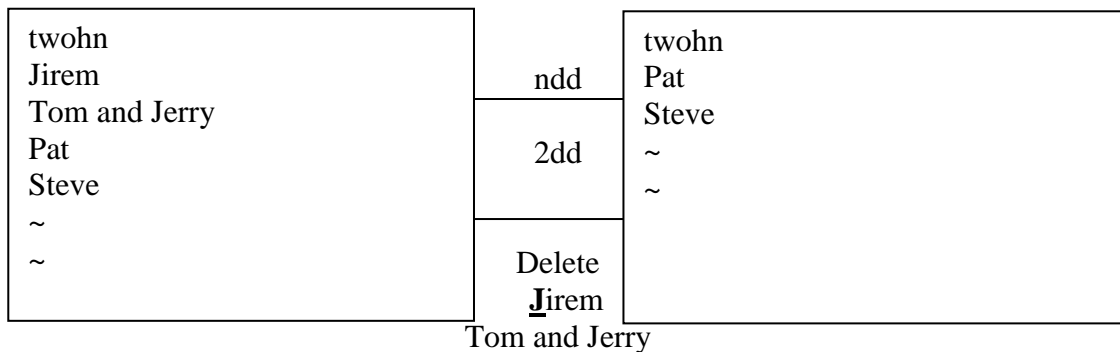
The “*x*” command can be preceded by a number to indicate how many characters you want to delete. You will get the “beep”, if you are trying to delete nonexistent characters.



Sometimes you want to delete the entire line. The “x” command will get rid of all the characters on a line, but it won’t get rid of the line itself. To delete a line, you use the dd command, a special case of a more general delete. It can be preceded by a number to indicate the number of lines to delete.



The “dd” command can be preceded by a number to indicate how many lines you want to delete. Highlight the cursor at the beginning of Jirem.

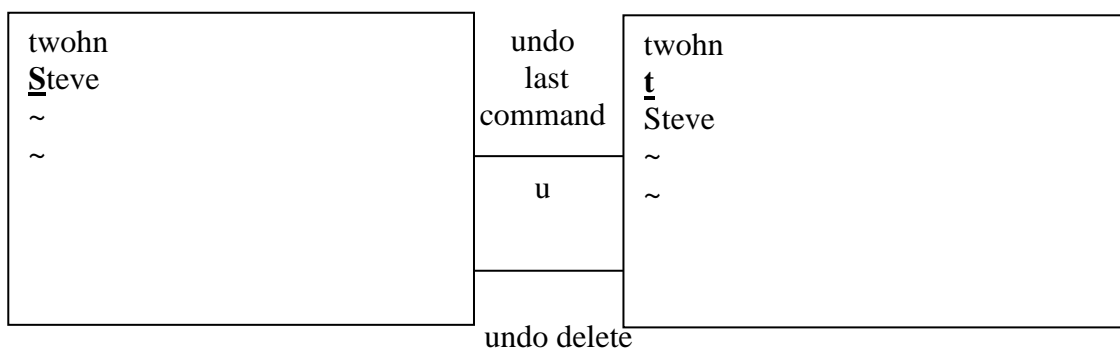


3. Perform the following operation by specifying the command and the resulting text as answer.

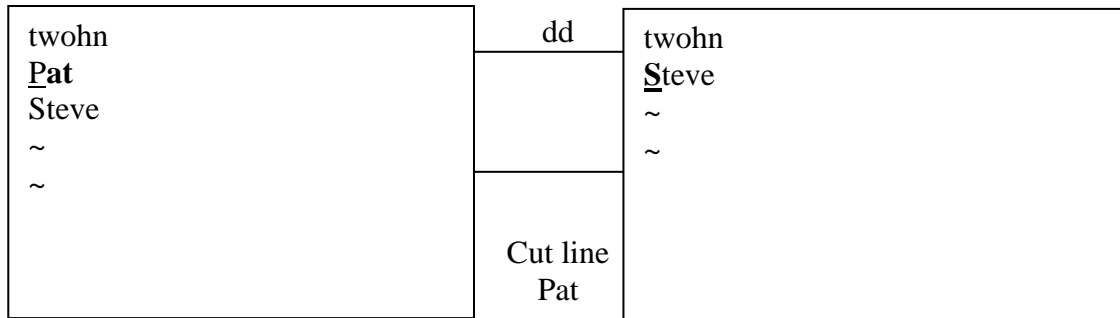
Action (from current cursor position)	Command typed	Result
Delete 2 characters		
Delete 3 characters from 3 <sup>rd</sup> line		
Delete 1 <sup>st</sup> line		
Delete 4 <sup>th</sup> line		

### Miscellaneous Command

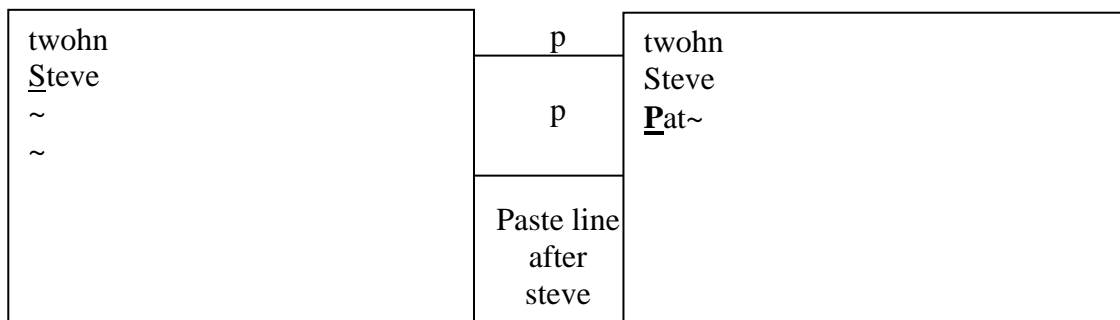
If we need to undo the activity, it is achieved by means of “u” command.



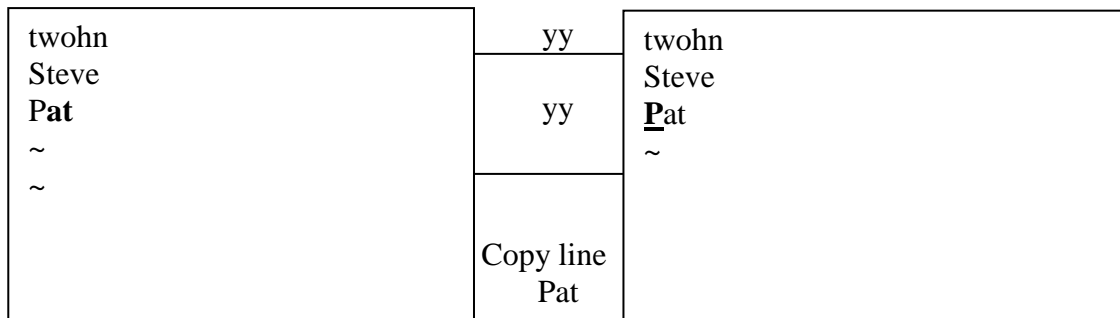
Cut the line(s) from the desired cursor position and paste those lines to the desired cursor position



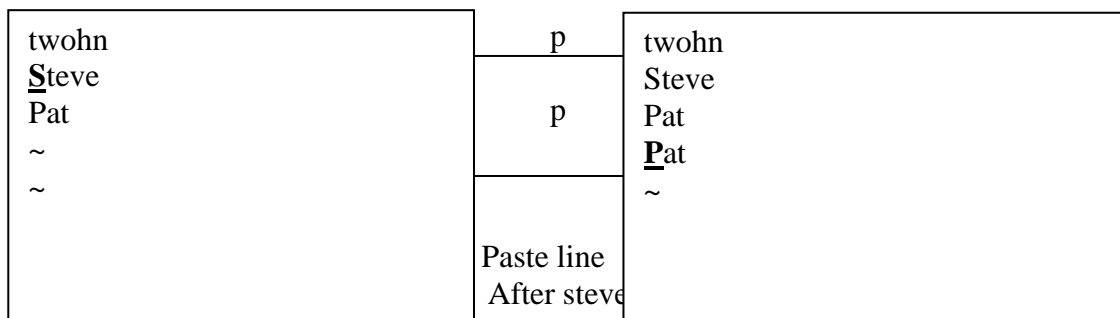
The “p” character is used to paste the line(s) before the desired cursor position.



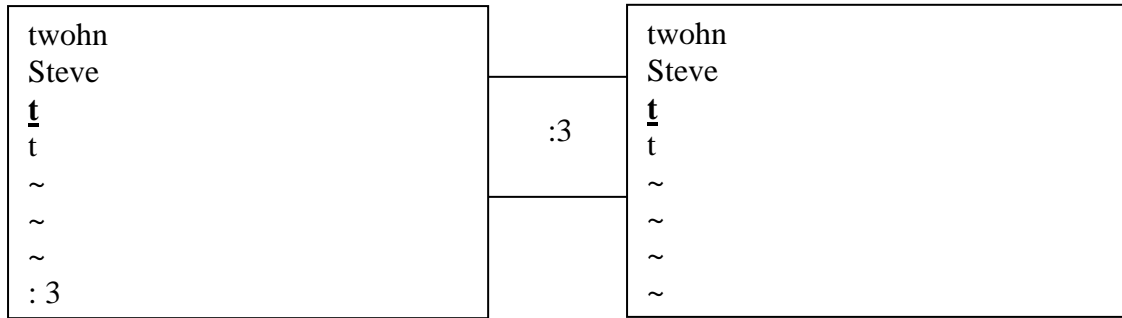
Copy the line(s) from the desired cursor position and paste those lines to the desired cursor position



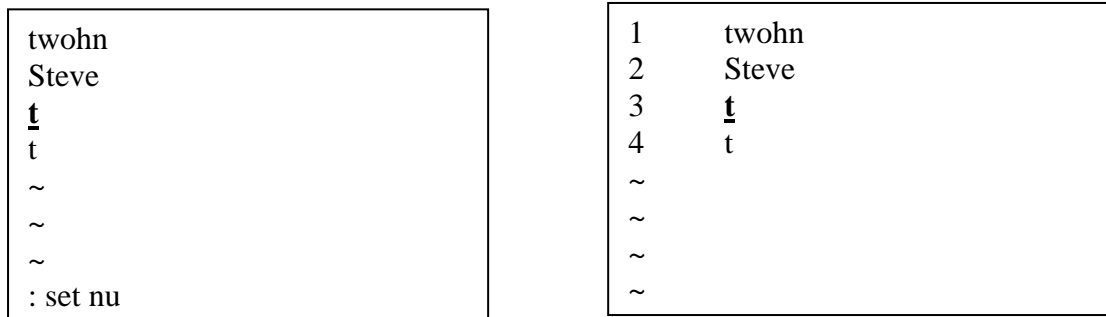
The “p” character is used to paste the line(s) before the desired cursor position.



**:num** –command moves the cursor to the specified line number scrolling if necessary.



**:set nu** – command allows you to show the line numbers for the line(s) present in the screen editor.

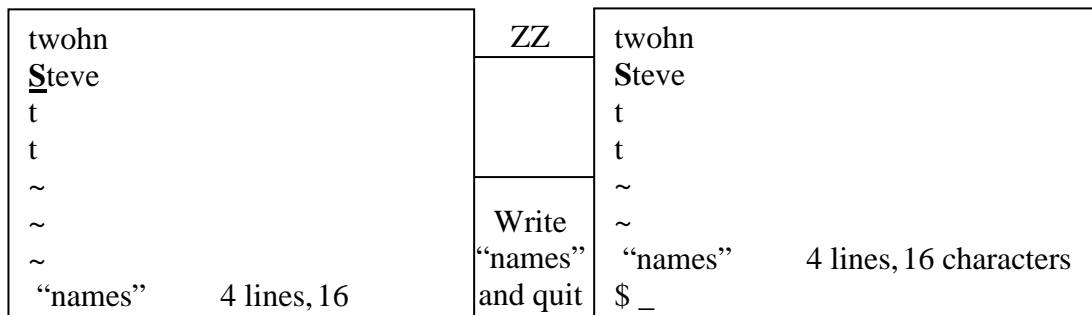


4. Perform the following operation by specifying the command and the resulting text as answer.

Action (from current cursor position)	Command typed	Result
Cut line 2, 3 and put those after twohn		
Copy line 3 and put it after line 4.		
Undo all the changes		
Locate the content at line 2		

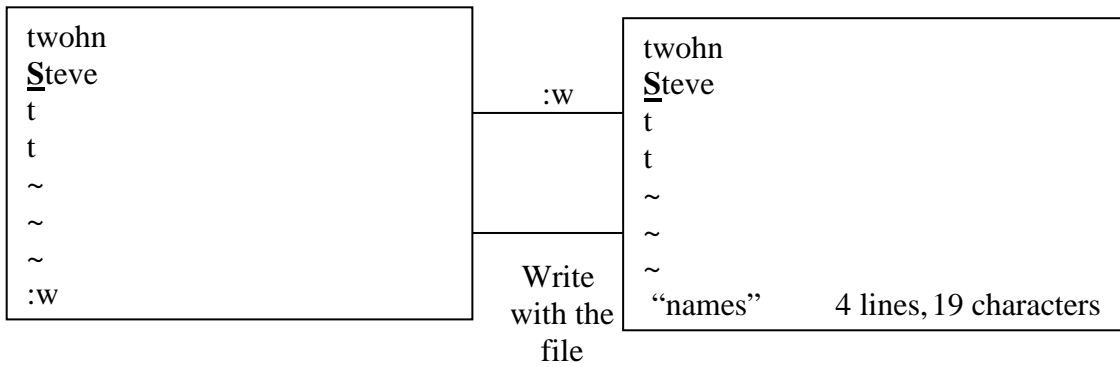
- **Control Commands**
- **Saving the file**

The vi editor also changes a copy of the file that must be written before the file is actually changed. There are several ways to write file in vi editor, but the easiest way is through the “ZZ” command that automatically write the file and quit, putting you back on to the shell prompt.

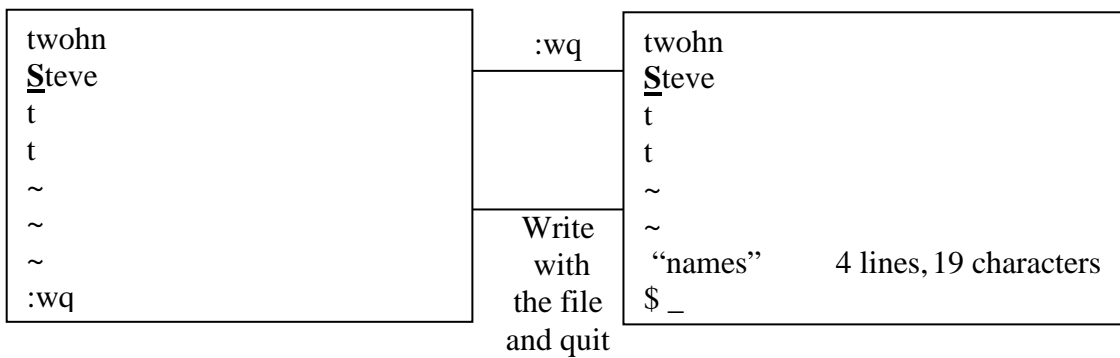


**Usage of control commands through “:” prompt of vi editor.**

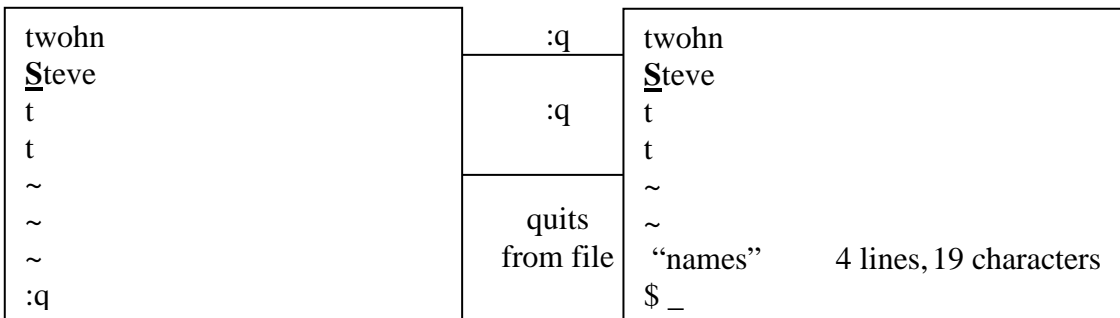
1. `:w` – command write the file without quitting vi editor.



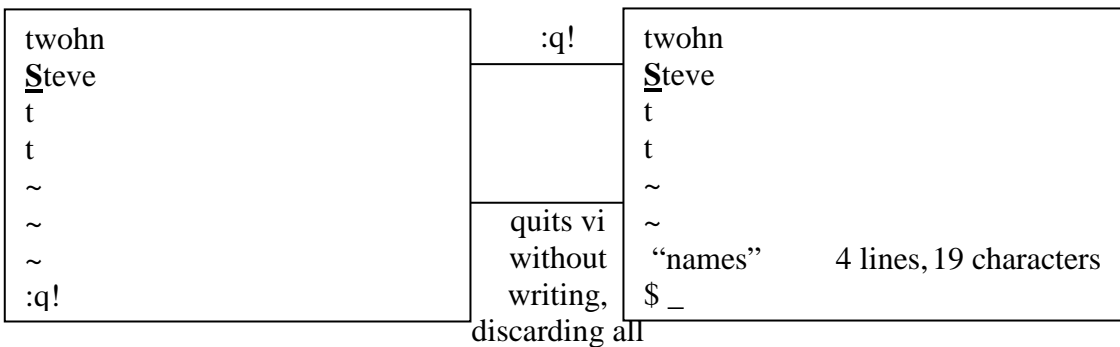
`:wq` – command write the file and put you back on to the shell prompt.



2. `:q` – command quits vi editor.



3. `:q!` – command quits vi without writing, discarding all changes.



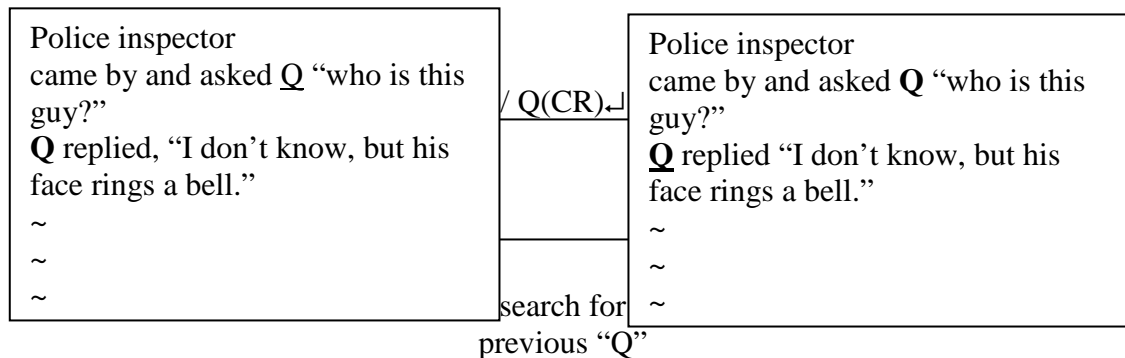
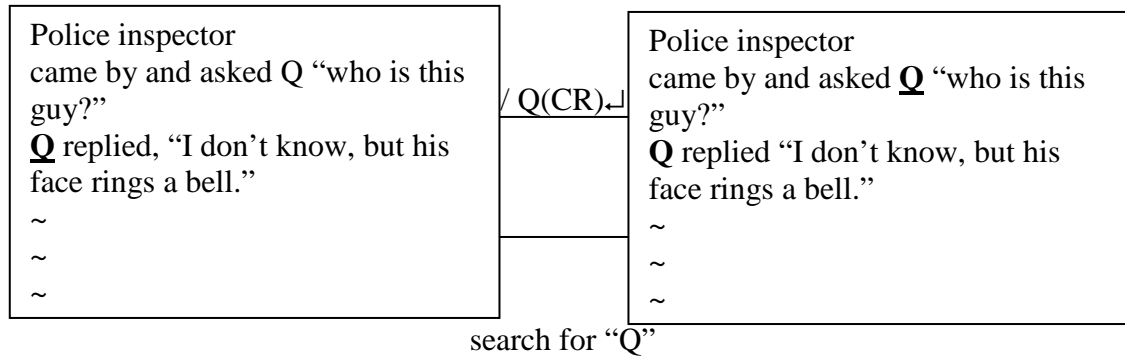
4. Perform the following operation by specifying the command and the resulting text as answer.

Action (from current cursor position)	Command typed	Result
Create the file called "test " to test the other commands containing text as shown below		
Apply operations covered so far on the file		

```
Police inspector
came by and asked Q "who is this
guy?"
Q replied, "I don't know, but his
face rings a bell."
~
~
~
```

### String Searching

The vi editor can search for strings, by typing in a "/" followed by the string you want to search for followed by a CR (↵). The vi editor then scans for the next occurrences of the strings.



5. Perform the following operation by specifying the command and the resulting text as answer.

Action (from current cursor position)	Command typed	Result
Search for keyword guy		

### Word Commands

The vi editor knows about objects called words that are simply letters and numbers separated by blank, tabs or punctuation marks. The vi editor allows you to move from word to word, delete them and change them with simple commands.



### 1. w command

Moves the cursor to the next word.

Police inspector came by and asked Q “who is this guy?” Q replied, “I don’t know, but his face rings a bell.” ~ ~ ~	w	Police inspector came by and asked Q “who is this guy?”
	w	Q replied “I don’t know, but his face rings a bell.”
	go to next word	~ ~ ~

### 2. b command

Moves the cursor backward a word.

Police inspector came by and asked Q “who is this guy?” Q replied, “I don’t know, but his face rings a bell.” ~ ~ ~	b	Police inspector came by and asked Q “who is this guy?”
	b	Q replied “I don’t know, but his face rings a bell.”
	go back to word	~ ~ ~

### 3. e command

Moves the cursor to the end of a word.

Police inspector came by and asked Q “who is this guy?” Q replied, “I don’t know, but his face rings a bell.” ~ ~ ~	e	Police inspector came by and asked Q “who is this guy?”
	e	Q replied “I don’t know, but his face rings a bell.”
	go to end of word	~ ~ ~

### Deleting and Changing text

The vi editor provides you with several ways to delete and change text. One method of deleting text is with the “d” command. The “d” command is *always* followed by another character that specifies what will be deleted.

**Dw command** – command is use to delete a word.

Police inspector came by and asked Q “who is this gu <u>y</u> ?” Q replied, “I don’t know, but his face rings a bell.” ~ ~ ~	dw	Police inspector came by and asked Q “who is this gu <u>y</u> ?” Q replied “I don’t know, but his face rings a bell.” ~ ~ ~
	dw	
	delete word	

6. **cw command** – command is use to change a word.

Police inspector came by and asked Q “who is this gu <u>y</u> ?” Q replied, “I don’t know, but his face rings a bell.” ~ ~ ~	cw	Police inspector came by and asked Q “who is this gu <u>y</u> ?” Q replied “I don’t know, but his face rings a bell.” ~ ~ ~
	cwy?ESC	
	change word Enter change	

Signature of the instructor

Date  /  /



**Set A**

1. Create a file by name \_\_\_\_\_ at least 25 lines long using vi editor's input commands – “a” and “i”. Also try the replace mode by examining the toggle feature of “i” character.
2. Create a file by name \_\_\_\_\_ at least 25 lines long using vi editor's input commands – “a” and “i”. Also try search command on the file.

Signature of the instructor

Date  /  /

**Set B**

1. Create a file name \_\_\_\_\_ containing five lines and execute the following set of commands of vi editor and describe the result on the paper.

Sr. No.	Command
1	^U
2	^B
3	o

4	O
5	n
6	N
7	dw

2. Create a file name \_\_\_\_\_ containing five lines and execute the following set of commands of vi editor and describe the result on the paper.

Sr. No	Command
1	cc
2	D
3	C
4	s
5	S
6	rchr
7	R

Signature of the instructor

Date  /  /

**Assignment Evaluation**

**Signature**

0: Not done       2: Late Complete       4: Complete   
 1: Incomplete       3: Needs improvement       5: Well Done

## Exercise 6

Start Date

/ /
-----



Objective

To understand shell programming and use of different conditional statements in shell programming.



Reading

You should read following topics before starting this exercise

1. What is a shell and different shells in UNIX?
2. LINUX commands
3. Shell programming statements, operators and conditional statements



Ready Reference

The set of internal commands provided by shell can be combined into a language with its own variables, operators, conditional statements and loops called shell programming language. It helps in combining basic shell commands into a complex service commonly required by users. The UNIX system administrator should be an accomplished shell programmer. Shell programs run in interpretive mode ,i.e., one statement at a time.

Shell program is stored in a file usually with .sh extension.

The shell program can be executed in one of the following ways

- a) using sh command along with the file name for example `$ sh myshell.sh`  
In case the program accepts filename as command line argument then  
`$sh myshell.sh file1`
- b) Make the file executable by using chmod command and then typing the filename at \$ prompt for example  
`$ chmod +x myshell.sh`  
`$ myshell.sh`  
In case the program accepts two integers as command line argument then  
`$ myshell.sh 45 36`

The command line arguments specified to a shell procedure are assigned to certain special variables or positional parameters such as \$0, \$1 etc.. \$0 stores the filename of the shell script, while \$1 is first argument, \$2 is second argument and so on. \$\* stores, the entire list of arguments, as a single string. \$# stores the total number of arguments passed to the script. The positional parameter \$?, Stores the exit status of the last command. It has the value 0 if the command succeeds, and a non-zero value if the command fails.

Different operators used in shell expressions

Meaning	Example
Number of arguments greater than 3	<code> \$# -gt 3</code>
Value of a less than or equal to 0	<code> \$a -le 0</code>
Value of a less than 3 and greater than or equal to 5	<code> \$a -lt 3 -a \$a -ge 5</code>
Value of choice equal to "y" or "Y"	<code> [ \$choice ="y" -o \$choice ="Y" ]</code>
Number of arguments not equal to 2	<code> \$# -ne 2</code>
If not number of arguments equals 3	<code> ! \$# -eq 3</code>
True if name is not null string	<code> -n \$name</code>
True if string name is null string	<code> -z \$name</code>
True if string name is same as abc	<code> \$name = "abc"</code>
True if string name is not same as abc	<code> \$name != "abc"</code>

Different statements used in shell script are

Statement	Usage	Example
read	to accept input from user.	<code> read n  read name</code>

echo	to display output to user.	echo " Give your name" echo "Enter first number"
expr	It is used to do arithmetic operations as also convert string to integer.	sum=`expr \$a + \$b` x=`expr \$x + 1`
Test [ ]	Evaluates expression on its right or evaluates expression within square brackets	x=5; y=7; test \$x -eq \$y ; echo \$?  read choice If [ \$choice ="y" -o \$choice ="Y" ] then exit
If - then fi	For conditional branching	If grep "\$1" \$2 echo "pattern found"
if - then - else - fi	For Two-way conditional branching	If [ \$# -eq 1 ] then cat \$1 else echo " wrong no of arguments" fi
If - then - elif - then - else - fi	Nested if statements	If [ \$# -eq 3 ] ; then # semicolon separator is required # as if and then are on same line grep "\$1" , \$2 > \$3 elif [ \$# -eq 2 ] grep "\$1" \$ 2 else echo " wrong no of arguments " fi
case - esac	Multiple branching	read answer case \$answer in [yY]*) exit ;; #matches Yes yes [nN]*) echo No ;; *) echo "Invalid response" esac

#### Different statements used in testing file status

Test	Meaning
-e filename	true if file exists
-f filename	true if file exists and is a regular file
-r filename	true if file exists and is readable
-w filename	true if file exists and is writable
-x filename	true if file exists and is executable
-d filename	true if file exists and is a directory
-s filename	true if file exists and has size >0.

#### Sample programs

Sr. No	Program statement	Program code
1	An interactive program that accepts month name and checks with current date if the person is late	#The program accepts the date echo "enter the date" read dt a=`date +%d` # a stores the day value of current date as string a=`expr \$a + 0` # converting to integer # note space before and after + if [ \$a -gt \$dt ] # note space before and after brackets

		<pre> then echo "You are late by \$a -\$dt days" fi </pre>
2	A command line program that accepts only two arguments and outputs sum and product of the two	<pre> # program accepts two arguments if test \$# -ne 2 ; then # semicolon separator is required as if and then # are on the same line echo "wrong number of arguments" else tot=`expr \$1 + \$2` # * is escaped to be treated as mult operator # and not as a wild character prod=`expr \$1 \* \$2` echo The total is \$tot echo The product is \$prod fi </pre>
3	A interactive program that accepts filename and checks whether it is regular file or directory	<pre> echo "Enter the filename" read fname # checks if value entered is null If [ -z \$fname ] ; then # semicolon separator is required as if and then #are on the same line echo you have not entered filename elif [ ! -e \$fname ] ; then ; echo file does not exist elif [ -f \$fname ] ; then ; echo \$fname is regular elif [ -d \$fname ] ; then ; echo \$fname is directory else echo \$fname is a special file fi </pre>



Type the examples given for different statements in files with .sh extension and execute them

1. Type the sample program 1, execute it for different date values and modify it to a program that decides the file as late by accepting both month and date. Modify the program to one that accepts value as command line arguments

2. Type the sample program 2, execute it for different values and modify it to a program that prints quotient and divisor of command line arguments. Modify the program to one that accepts values interactively from user.

3. Type the sample program 3, execute it for different values and modify it to a program that checks for a regular file if it is readable or writable giving appropriate message.

Signature of the instructor

Date



**Set A**

1. Write a shell script to accept a file name, check if it is regular & show it's contents. (use cat command)

2. Write a shell script to accept a file name, check if it is regular & display number of words in a file. (use wc command)

3. Write a shell script to accept a name, check if it is directory & display its contents. (use ls command)

4. Write a shell script to accept a file name, and accept a pattern and display lines from the file in which the pattern is present. (use grep command)

5. Write a shell script to accept a name, and create a copy of it named as this name-(hyphen)copy in the same directory . (use cp command)

6. Write a shell script to display “ Good Morning”, “ Good afternoon” , and “Good evening” depending on the hour (use date command)

Signature of the instructor

Date

**Set B**

1. Write a shell script to accept argument string , and display present working directory if argument string is “current” ,display parent directory if argument string is “parent” and display the contents of root directory if argument string is “root” (use pwd, cd and ls command)

2. Write a shell script to accept an extension name such as txt and display the contents of all files with this extension, if there exists a file with this extension or give appropriate message (use cat with wild cards and ls)

3. Write a shell script to accept as argument an extension name such as .txt and move the contents of all files with this extension to a directory by the same name (use mkdir and mv)

4. Write a shell script to accept a file name , and display file details if the file exists and a suitable message if it does not. (use grep and ls)

Signature of the instructor

Date

**Set C**

1. Write a shell script which accepts a filename, displays menu with following options, accepts user choice as number and takes appropriate actions

Number	Menu option	Expected Action
1	Contents	Display the file contents
2	Size in blocks	Display the file Size in blocks
3	Number of words	Display the number of words in file
4	Last five lines	Display last five lines of the file
5	First ten lines	Display first ten lines of the file

2. Write a shell script that displays menu with following options, accepts user choice as number and takes appropriate actions

Number	Menu option	Expected Action
1	No of users	Displays the No of users logged in
2	Current user	Display the login id of user logged i
3	Current Directory	Display the present working directory
4	Home Directory	Display the home directory of logged in user
5	Current Path	Display the path

3. Write a shell script that displays menu with different DOS commands, accepts user choice as letters of the command and executes appropriate linux command after accepting required arguments as given below.

Number	Menu option	Linux command
1	Dir	use ls
2	Copy	Accept filenames and use cp command
3	Type	Accept filename and use cat command
4	delete	Accept filename and use rm command
5	date	Use date

Signature of the instructor

Date  /  /

**Assignment Evaluation**

**Signature**

0: Not done       2: Late Complete       4: Complete

1: Incomplete       3: Needs improvement       5: Well Done



**Exercise 7**

**Start Date** / /



Shell programming using control structures and writing shell scripts



You should read following topics before starting this exercise

1. Linux commands
2. Shell programming statements and loops



Shell provides following loop structures

Loop structure	Syntax	Example
While	while [condition] do commands done	i=`date +%m` i=`expr \$i + 0` # store the value of current month in i while [ \$i -gt 0 ] do mkdir file\$i # decrementing the value of i i=`expr \$i - 1` done
Until	until [condition] do commands done	i=1 # checking if file\$i is not a directory until [ ! -d file\$i ] do cp abc.txt file\$i i=`expr \$i + 1` done
For	for variable in list do commands done	for i in 1 2 3 4 do echo "deleting all files in directory file\$i" # displaying the contents of directory ls file\$i done echo " job over "

Some shell commands are specially useful when writing shell programs.

We will consider some of them

Command	Used for	Example
set	Allows the arguments to be stored as \$1, \$2 and so on	\$set 23 45 \$echo "\\$1 is \$1 and \\$2 is \$2" \$set `date` \$echo "\\$1 is \$1 and \\$2 is \$2" \$echo \$*
shift	Shifts the arguments to the left, When executed once \$2 becomes \$1, \$3 becomes \$2 and so on.	\$echo \$1 \$2 \$3 \$shift \$echo \$1 \$2 \$3
cut	Used to slice a file vertically, -c is used for cutting columns, -f is used for cutting	\$ls -l > dirfile

	fields , -d is used for specifying delimiter	\$cut -c 1-10, 17-20 dirfile \$cut -d ' ' -f 1,6 dirfile
--	--	---



Type the examples given above for “while” , “until” and “for” and execute them in that order. Use shell commands to verify the outcome.

Write the outcome when you execute the following set of commands at shell prompt

1	name=date \$name `\$name`
2	set `date` shift cal "\$5"
3	set `wc abc.txt` shift echo the number of characters is \$2
4	set `who` shift echo My terminal is \$1
5	who > userlist cut -d ' ' -f 1,3 userlist

Signature of the instructor

Date



**Set A**

1. Write a shell script which prints file name followed by first line of each file in the current directory.

2. Write a shell script which checks if any of the strings in the output of date command are present in the dirfile

3. Write a shell script which accepts directory names till a valid directory name is given. It should give appropriate message if directory is not present.

4. Write a shell script to print the information as to how many files and how many directories are present in current directory.

Signature of the instructor

Date

**Set B**

1. Write a shell script to print the information of all files in current directory in the following format

- Name of the file - followed by name of the file
- Directory - followed by yes or no
- Date of last modification - followed by date of last modification
- Size – followed by file size

2. Write a shell script that accepts name from the user and creates a directory by that name, then creates a text file in that directory and stores in it, the data accepted from user(till ^z), and displays the number of characters stored in the file. The program stops if directory name given is null.

Signature of the instructor

Date

**Assignment Evaluation**

**Signature**

0: Not done

2: Late Complete

4: Complete

1: Incomplete

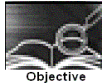
3: Needs improvement

5: Well Done

## Exercise 8

Start Date

/ /



Creating simple HTML pages.



You should read following topics before starting this exercise

1. Internet and web
2. web browsers and web servers
3. HTML tags



### Internet and the Web

The internet is a collection of connected computers that communicate with each other. The Web is a collection of protocols (rules) and software's that support such communication.

In most situations when two computers communicate, one acts as a server and the other as a client, known as client-server configuration.

Browsers running on client machines request documents provided by servers. Browsers are so called because they allow the user to browse through the documents available on the web servers. A browser initiates the communication with a server, requesting for a document. The server that is continuously waiting for a request, locates the requested document and sends it to the browser, which displays it to the user

The most common protocol on the web is HyperText Transfer protocol(HTTP)

The most commonly used browsers are Microsoft Internet Explorer (IE). Netscape browser and Mozilla. The most commonly used web servers are Apache and Microsoft Internet Information server(IIS).

### HTML

HyperText Markup Language is a simple markup language used to create platform-independent hypertext documents on the World Wide Web. Most hypertext documents on the web are written in HTML.

You will need a simple text editor to write html codes. For example you can use notepad in windows and in Linux operating system. You will need a browser to view the html code, you can use IE on windows and Mozilla on Linux operating system.

HTML tags are somewhat like commands in programming languages. Tags are not themselves displayed, but tell the browser how to display the document's contents.

Every HTML tag is made up of a tag *name*, sometimes followed by an optional list of attributes, all of which appears between angle brackets < >. Nothing within the brackets will be displayed in the browser. The tag name is generally an abbreviation of the tag's function. Attributes are properties that extend or refine the tag's function. The name and attributes within a tag are not case sensitive. Tag attributes, if any, belong after the tag name, each separated by one or more spaces. Their order of appearance is not important. Most attributes take *values*, which follow an equal sign (=) after the attribute's name. Values are limited to 1024 characters in length and may be case sensitive. Sometimes the value needs to appear in quotation marks (double or single).

Most HTML tags are containers, meaning they have a beginning start tag and an end tag. An end tag contains the same name as the start tag, but it is preceded by a slash (/). Few tags do not have end tags.

Some HTML tags required to design simple web pages are given below

Tag	Description	Attributes	Example
<!-- ... -->	Allows one to insert a line of browser-invisible comments in the document		<!-- Starting my first web page --!>
<HTML> </HTML>	<HTML> tag tells the browser that this is start of the HTML and </HTML> marks its end.		<HTML> Hello world! </HTML>
<HEAD> </HEAD>	Every html page must have a header. < Head> tag defines the Head Segment of an html document		
<TITLE> </TITLE>	One of the most important parts of a header is title. Title is the small text that appears in title bar of viewer's browser.		<HEAD> <TITLE> My Web page </TITLE> </HEAD>
<BODY> </BODY>	Every web page needs a body in which one can enter web page content	<b>background=</b> designates a file to be displayed as background <b>bgcolor=</b> "#(hexadecimal color code)" sets the background color <b>text=</b> "#(hexadecimal color code)" sets the color of plain text. Text color default is black.	<BODY BGCOLOR="#00FF00" text="#FF0000"> Page with Green Color and red Text </BODY> Format of color number is RRGGBB, so if we write 00FF00 we mean (red=0, green=255, blue=0)
 	A single tag used to break lines	clear=all left right Breaks the text and resumes the next line after the specified margin is clear.	line   is broken
<p>	A single tag used to break text. Breaking text with the <p> tag adds vertical spacing		<p> break the line <p>adding extra space
<B> </B>	To make text appear bold		<B>This text will appear bold</B>
<U> </U>	To make text appear underlined		<U>This text will appear underlined</U>
<I> </I>	To make text appear italic		<B><I>This text is both Bold and italic</I>
<CENTER> </CENTER>	Centers enclosed text		<CENTER> Text is centered </CENTER>
<FONT> </FONT>	To change font which affects the style (color, typeface, and size) of the enclosed text.	color="#(hexadecimal color code)" sets the color. face=typeface (or list of typefaces) sets a typeface for the text ( if it is on the user's machine) size=value Sets the size of the	<FONT SIZE="5" FACE="ARIAL" COLOR="#00FF00"> How is this ? </FONT>

		type to an absolute value on a scale from 1 to 7 (3 is the default)	
<BIG> </BIG>	Sets the type one font size larger than the surrounding text		
<SMALL> </SMALL>	Sets the type one font size smaller than the surrounding text		
<BLINK> </BLINK>	Causes the contained text to flash on and off.		
<SUB> </SUB>	Formats enclosed text as subscript.		a<SUB> <SMALL> o </SUB> </SMALL>
<SUP> </SUP>	Formats enclosed text as superscript.		x<SUP> <SMALL> 2 </SUP> </SMALL>
<MARQUEE> </MARQUEE>	Creates a scrolling-text marquee area.	align=top middle bottom Aligns the marquee with the top, middle, or bottom of the neighbouring text line. behaviour=scroll slide alternate Specifies how the text should behave. Scroll is the default setting and means the text should start completely off one side, scroll all the way across and completely off, then start over again. Slide stops the scroll when the text touches the other margin. Alternate means bounce back and forth within the marquee. bgcolor="#rrggb" or color name Sets background color of marquee. direction=left right Defines the direction in which the text scrolls. height=number Defines the height in pixels of the marquee area. hspace=number Holds n pixels space clear to the left and right of the marquee.	<MARQUEE align=top behaviour =slide bgcolor="#00FF00" direction=right height=20 hspace =5 > scrolling all the way from one end to other </MARQUEE>
<IMG>	loads an inline image	src= " text" Provides the URL of the graphic file to be displayed alt="text" Provides alternate text if the image cannot be displayed. height=number Specifies the height of the image in pixels.	

		width= <i>number</i> Specifies the width of the image in pixels.	
--	--	---	--

An HTML document is divided into two major portions: the head and the body. The head contains information about the document, such as its title and “meta” information describing the contents. The body contains the actual contents of the document (the part that is displayed in the browser window).

A sample HTML document is given below

```

<!-- Starting my first web page assignment --!>
<HTML>
<HEAD>
<TITLE> My Web page </TITLE>
</HEAD>
<BODY BACKGROUND=" myimage.jpg" text="#FF0000">
The <FONT size=6 > Font size </FONT> can be changed <Br> as well as <FONT
color="#0000FF" > color of the text </Font> <BR> sometimes I prefer to change the <B> Style or
</B>underline <U> the text </U>
<MARQUEE align=bottom behaviour =scroll bgcolor="#00FF00" direction=left height=20
hspace =5 > Good Bye have a nice time </MARQUEE>
</BODY>
</HTML>

```



Create a background image called myimage.jpg by using any picture creating tool. Type the above sample html program in the text editor and view it through the browser. Modify it to include some blinking text.

Signature of the instructor

Date



**Set A**

1. Create an html page with 7 separate lines in different sizes. State size of each line in its text.
2. Create an html page with 7 separate lines in different colors. State color of each line in its text.
3. Create an html page with all the different text styles (bold, italic and underlined) and its combinations on separate lines. State style of each line in its text.
4. Create an html page containing the polynomial expression as follows  

$$a_0 + a_1x + a_2x^2 + a_3 x^3$$
5. Create an html page with red background with a message “warning” in large size blinking. Add scrolling text “read the message” below it.

Signature of the instructor

Date

**Set B**

1. Create an html page with following specifications
  - a. Title should be about myself
  - b. Color the background with pink color
  - c. Place your name at the top of the page in large text and centered
  - d. Add names of your family members each in a different size, color, style and typeface
  - e. Add scrolling text with a message of your choice
  - f. Add your image at the bottom
  
2. Create an html page with following specifications
  - a. Title should be about mycollege
  - b. Put the windows Logo image in the background
  - c. Place your College name at the top of the page in large text followed by address in smaller size
  - d. Add names of courses offered each in a different color, style and typeface
  - e. Add scrolling text with a message of your choice
  - f. Add college image at the bottom
  
3. Create an html page with following specifications
  - a. Title should be about myCity
  - b. Place your City name at the top of the page in large text and in blue color
  - c. Add names of landmarks in your city each in a different color, style and typeface
  - d. One of the landmark, your college name should be blinking
  - e. Add scrolling text with a message of your choice
  - f. Add some image at the bottom

Signature of the instructor

Date

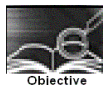
**Assignment Evaluation**

**Signature**

0: Not done <input type="text"/>	2: Late Complete <input type="text"/>	4: Complete <input type="text"/>
1: Incomplete <input type="text"/>	3: Needs improvement <input type="text"/>	5: Well Done <input type="text"/>

**Exercise 9**

**Start Date**



Objective  
HTML programming using lists, tables, frames and hyperlinks.



Reading  
You should read following topics before starting this exercise

1. Use of hyperlinks for navigating through pages
2. Use of lists , tables and frames



Ready Reference  
**Lists** : Lists are a great way to provide information in a structured and easy to read format.

There are two types of lists :

**1] Numbered List (Ordered List)**

An ordered list is used when sequence of list items is important.

**2] Bulleted List (Unordered List)**

An unordered list is a collection of related items that have no special order or sequence.

Tags used to create lists are given in the following table.



Tag	Description	Attributes	Example
<LI>	Specify the list item.		
<OL> </OL>	The <OL> tag formats the contents of an ordered list with numbers. The numbering starts at 1. It is incremented by one for each successive ordered list item tagged with <LI>	Type = a/A/i/l/1 Sets the numbering style to a,A,i,l,1 default 1 start = "A" Specifies the number or letter with which the list should start.	<body bgcolor= "pink"> <font face = "Arial" size= "6" color = "green"> <u> List of Cities.... </u> </font> <ol type = "A" start = "A"> <li> Mumbai <li> Pune <li> Nashik <li> Nagpur </ol> </body>
<UL> </UL>	<UL> tag defines the unordered list of items	Type = disc/square/circle Specifies the bullet type.	<body bgcolor= "sky blue" text ="yellow"> <font face = "Arial" size="6" color= "orange"> <i><u><b> List of Fruits </i></u></b> <ul type = "square"> <li> Apple <li> Pinapple <li> Mango <li> Guava </ul> </body>

**Tables :** A table is a two dimensional matrix, consisting of rows and columns. HTML tables are intended for displaying data in columns on a web page. Tables contains information such as text, images, forms, hyperlinks etc.

Tags used to create table are given in the following table.

Tag	Description	Attributes
<TABLE> </TABLE>	Create a table	Border=number Draws an outline around the table rows and cells of width equal to number. By default table have no borders number =0. Width=number Defines width of the table. Cellspacing=number Sets the amount of cell space between table cells. Default value is 2 Cellpadding=number Sets the amount of cell space, in number of pixels between the cellborder and its contents. Default is 2 Bgcolor="#rrggbb" sets background color of the table Bordercolor="#rrggbb" sets border color of the table align=left right center Aligns the table. The default alignment is left frame=void above below hsides lhs rhs vsides box border Tells the browser where to draw borders around the table
<TR> </TR>	Creates a row in the table	
<TH> </TH>	Cells are inserted in a	

	row of the table for heading	
<TD> </TD>	Data cells are inserted in a row of the table	

A sample HTML document for creating table is given below

```

<html>
<head>
</head>
<body>
<table border = 2 cellspacing = 4 cellpadding = 4 bordercolordark = "red"
bordercolorlight = "blue" align = "center">
<caption> List of Books </caption>
<tr>
<th rowspan = 2 align = "center"> Item No </th>
<th rowspan = 2 align = "center"> Item Name </th>
<th align = "center" colspan = 2> Price </th>
</tr>
<tr>
<th align = "center"> Rs. </th>
<th align = "center"> Paise </th>
</tr>
<tr>
<td align = "center"> 1 </td>
<td align = "center"> Programming in C++ </td>
<td align = "center"> 500 </td>
<td align = "center"> 50 </td>
</tr>
<tr>
<td align = "center"> 2 </td>
<td align = "center"> Programming in Java </td>
<td align = "center"> 345 </td>
<td align = "center"> 00 </td>
</tr>
</table>
</body>
</html>

```

**Hyperlinks** : Hyperlink is a specialized feature of HTML. Instead of clicking through sequentially organized pages, a hypertext user clicks specially highlighted text called 'hyperlink'. Hyperlinks are technically known as anchors. They are usually visible in blue underlines.

Tags used to add hyperlinks lists are given in the following table.

Tag	Description	Attributes	Example
<A> </A>	Add an anchor or hyperlink.	href= <i>url</i> Specifies the URL of the target page.	<BODY> <A HREF="http://www.yahoo.com">Click here to visit Yahoo</A> </BODY>

**Frames** : Using frames, one can divide the screen into multiple scrolling sections, each of which can display a different web page into it. It allows multiple HTML documents to be seen concurrently

Tags used to add frames are given in the following table.

Tag	Description	Attributes	Example
<FRAMESET> </FRAMESET>	Splits browser screen into frames.	Rows=number helps in dividing the browser screen into horizontal sections or frames. Cols=number divides the screen into vertical sections or frames. The number written in the rows and cols attribute can be given as absolute numbers or percentage value or an asterisk can be used to indicate the remaining space.	<frameset rows = "20%, 30%, *">
<FRAME> </FRAME>	used to define a single frame in a <frameset>	name= <i>text</i> Assigns a name to the frame noresize Prevents users from resizing the frame. src= <i>url</i> Specifies the location of the initial HTML file to be displayed by the frame. bordercolor=" <i>#rrggbb</i> " or <i>color name</i> Sets the color for frame's borders	<html> <frameset rows = "50%, *"> <frameset cols = "50%, *"> <frame src = "success.html" name = "frm1"> <frame src = welcome.html"> </frameset> <frame src = "failure.html"> </frameset> </html>



Self-Activity

1. Create an html program using the body given in the example for ordered list. Modify it to change the color of the item text to \_\_\_\_ and reduce the size of text one smaller than the heading.
2. Create an html program using the body given in the example for unordered list. Modify it to change the shape of the bullet to \_\_\_\_ and also reduce the size of bulleted items one smaller than the heading.
3. Type the sample HTML program using tables. Modify it to remove Rs and paise column and specify price as 500.50
4. Type the sample HTML program using frames. Create the required html files with appropriate messages. Modify it to change to a different frame structure.

Instructor should fill in the blanks with appropriate values.

Signature of the instructor

Date  /  /



Assessment - work

### Set A

1. Write the HTML code which generates the following output.
  - Coffee
  - Tea
    - Black Tea

- Green Tea
  - 1] Africa
  - 2] China
- Milk

2. Write the HTML code which generates the following output.

Country	Population (In Crores)	
	INDIA	1998
1999		90
2000		100
USA	1998	30
	1999	35
	2000	40
UK	1998	25
	1999	30
	2000	35

3. Divide the frame into different sections as shown below and add appropriate html files to each frame.

First Frame : Name and Address		
Second Frame Bulleted list of qualifications		Third Frame Links to Favourite sites
Fourth Frame Scrolling Message	Fifth Frame Blinking reminders	Sixth Frame image

Signature of the instructor

Date

### Set B

1. Create an html page with appropriate frames containing Heading and other information. Add a bulleted list of your favourite subjects. For each subject make a nested list that contains, teacher name, the start and end time. Add your photograph and message in a separate frame. Add link to teacher or college web site wherever teacher name appears.

2. Create an html page with appropriate frames containing Heading and other information. Add an ordered list of your educational qualifications. For each course make a nested list that contains, university or board name, the year and the percentage scored. Add link to university site where university name appears. Add your college photograph and message in a separate frame

Signature of the instructor

Date

### Assignment Evaluation

### Signature

0: Not done <input type="text"/>	2: Late Complete <input type="text"/>	4: Complete <input type="text"/>
1: Incomplete <input type="text"/>	3: Needs improvement <input type="text"/>	5: Well Done <input type="text"/>





2.

Choose your favourite ice cream flavour

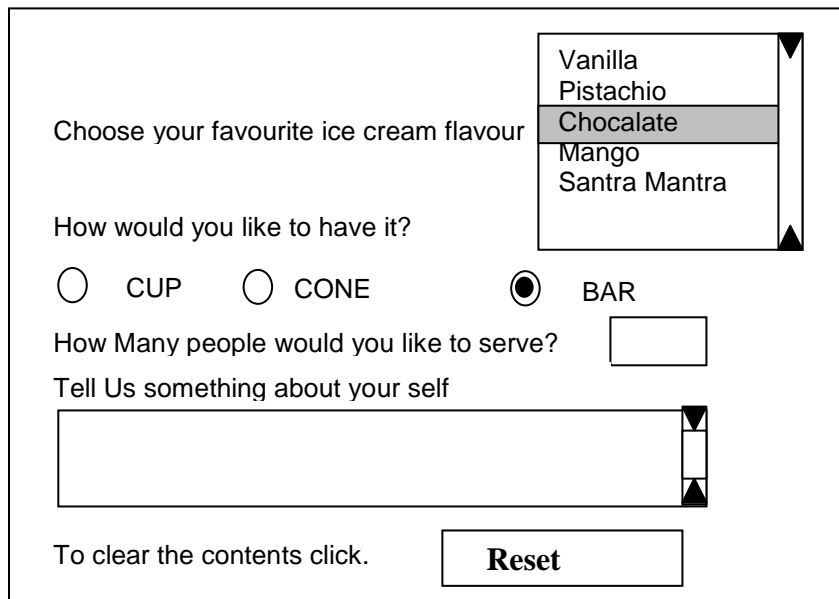
How would you like to have it?

CUP    CONE    BAR

How Many people would you like to serve?

Tell Us something about your self

To clear the contents click.



3.

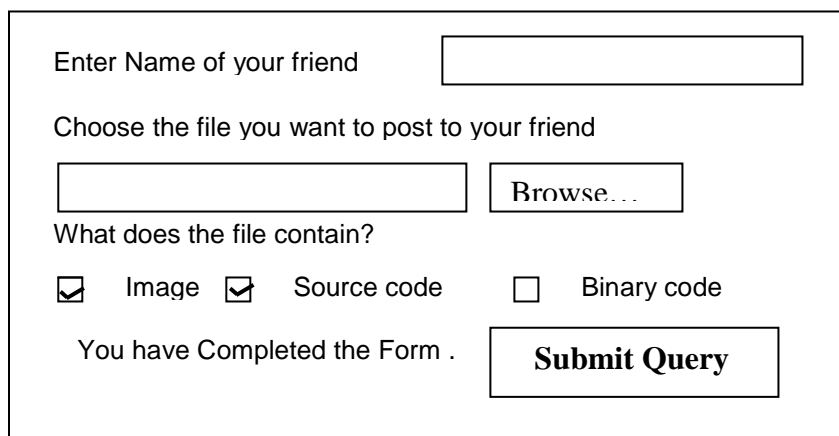
Enter Name of your friend

Choose the file you want to post to your friend

What does the file contain?

Image    Source code    Binary code

You have Completed the Form .



Signature of the instructor

Date

### Set B

1. Design an html form to take the information of a student registering for the course such as the name, address, gender, course (to be selected from a list of courses) etc. One should provide button to Submit as well as Reset the form contents.

2. Design an html form to take the information of a customer visiting a departmental store such as name, contact phone no, preferred days of purchasing, favourite item (to be selected from a list of items), suggestions etc. One should provide button to Submit as well as Reset the form contents.

3. Design an html form to take the information of a customer booking a travel plan such as name, address, contact phone no, gender, preferred season, location type (to be selected from a list) etc. One should provide button to Submit as well as Reset the form contents.

4. Design an html form to take the information of a article to be uploaded such as file path, author name , type (technical, literary, general), subject topic ( to be selected from a list) etc. One should provide button to Submit as well as Reset the form contents.

Signature of the instructor

Date

**Assignment Evaluation**

**Signature**

0: Not done

2: Late Complete

4: Complete

1: Incomplete

3: Needs improvement

5: Well Done



# **Lab Course I Section II**

**Exercise 11**

**Start Date** / /



To create simple tables , with only the primary key constraint ( as a table level constraint & as a field level constraint) (include all data types)



You should read following topics before starting this exercise

1. Designing relations into tables
2. Using DDL statements to create tables



A table is a database object that holds data. A table must have unique name, via which it can be referred. A table is made up of columns. Each column in the table must be given a unique name within that table. Each column will also have size a data-type and an optional constraint.

The data types permitted are

Data type	Syntax	Description	Example
Character data types	char(n)	It is fixed length character string of size n, default value 1 byte if n is not specified.	account_type char(6)
	varchar(n)	It is variable length character string with maximum size n.	employee_name varchar(50)
	Text	It is used to store large text data, no need to define a maximum	work_experience text
Numeric data types	Integer , int , serial	Serial is same as int, only that values are incremented automatically	Eno int Eno serial
	Numeric	A real number with P digits, S of them after decimal point.	Sal numeric(5,2) Sal numeric(n)
	Float	Real number	Weight Float
Date and time type	Date	Stores date information	Birthdate date
	Time	Stores time information	Birthtime time
	Timestamp	Stores a date & time	Birth timestamp
Boolean and Binary type	Boolean, bool	Stores only 2 values : true or false, 1 or 0, yes or no, y or n, t or f	Flag Boolean

Constraints can be defined as either of the following :

Name	Description	Example
Column level	When data constraint is defined only with respect to one column & hence defined after the column definition, it	Create tablename ( attribute1 datatype primary key , attribute2 datatype constraint constraint-name

	is called as column level constraint	,.....)
Table Level	When data constraint spans across multiple columns & hence defined after defining all the table columns when creating or altering a table structure, it is called as table level constraint	Create tablename ( attribute1 datatype , attribute2 datatype2 ,....., constraint pkey primary key(attribute1,attribute2))

**Syntax for table creation :**

Create tablename ( attribute list);

Attribute list : ( [ attribute name data type optional constraint] , ..... . )

**Primary key concept :**

Description	Properties	Example
A primary key is made up of one or more columns in a table, that uniquely identify each row in the table.	A column defined as a primary key, must conform to the following properties : a) The column cannot have NULL values. b) The data held across the column MUST be UNIQUE.	Create tablename ( attribute1 datatype primary key , attribute2 datatype ,.....) Create tablename ( attribute1 datatype , attribute2 datatype ,....., constraint pkey primary key(attribute1)) Create tablename ( attribute1 datatype, attribute2 datatype ,....., constraint constraint_name primarykey(attribute1,attribute2))



**Steps to Use DDL statements**

1. Login to linux server
2. Type the connection string to connect to database  
psql -h IP address of server -d database-name
3. Type in the DDL statement at the sql> prompt

1. Type \h and go through the commands listed.
2. Type \h command-name & read through the help data given for each command.

Type the following Create table Statements to create the tables . If the table creation is successful you get 'create table' as output.

Then Type \d <table name> and write the output

3. Create table emp (eno integer primary key, ename varchar[50] , salary float);
4. Create table books( id integer UNIQUE, title text NOT NULL, author\_id integer,sub\_id integer,CONSTRAINT books\_id\_pkey PRIMARY KEY(id));
5. Create table sales\_order(order\_no char[10] PRIMARY KEY, order\_date date, salesman\_no integer);
6. Create table client\_master(client\_no integer CONSTRAINT p\_client PRIMARY KEY, name varchar[50], addr text, bal\_due integer);
7. Create table inventory(inv\_no integer PRIMARY KEY,in\_stock Boolean);

8. create table sales\_order1(order\_no char[10], product\_no char[10], ,  
 qty\_ordered integer,product\_rate numeric(8,2),PRIMARY  
 KEY(order\_no,product\_no));

Signature of the instructor

Date  /  /



**Set A**

1.

Create a table with details as given below

Table Name		PLAYER	
Columns	Column Name	Column Data Type	Constraints
1	player_id	Integer	Primary key
2	Name	varchar (50)	
3	Birth_date	date	
4	Birth_place	varchar(100)	
Table level constraint			

2.

Create a table with details as given below

Table Name		Student	
Columns	Column Name	Column Data Type	Constraints
1	Roll_no	integer	
2	Class	varchar (20)	
3	Weight	numeric (6,2)	
4	Height	numeric (6,2)	
Table level constraint		Roll_no and class as primary key	

3.

Create a table with details as given below

Table Name		Project	
Columns	Column Name	Column Data Type	Constraints
1	project_id	integer	Primary key
2	Project_name	varchar (20)	
3	Project_description	text	
4	Status	boolean	
Table level constraint			

4.

Create a table with details as given below

Table Name		Donor	
Columns	Column Name	Column Data Type	Constraints
1	Donor_no	integer	Primary key
2	Donor_name	varchar (50)	
3	Blood_group	Char(6)	
4	Last_date	date	
Table level constraint			

**Set B**

Create table for the information given below by choosing appropriate data types and also specifying proper primary key constraint on fields which are underlined

1. Property (property\_id, property\_desc , area, rate, agri\_status )
2. Actor (actor\_id, Actor\_name, birth\_date )

3. Movie(movie-no, name, release-year )

4. Hospital(hno,hname,hcity)

Signature of the instructor

Date

**Set C**

Create table for the information given below by choosing appropriate data types and also specifying proper primary key constraint on fields which are underlined

1. Table \_\_\_\_\_ ( \_\_\_\_\_, \_\_\_\_\_, \_\_\_\_\_, \_\_\_\_\_,  
Primary key : \_\_\_\_\_

Instructor should fill in the blanks with appropriate values.

Signature of the instructor

Date

**Assignment Evaluation**

**Signature**

0: Not done

2: Late Complete

4: Complete

1: Incomplete

3: Needs improvement

5: Well Done

**Exercise 12**

**Start Date** / /



To create more than one table, with referential integrity constraint, PK constraint.



You should read following topics before starting this exercise

1. Referential Integrity constraints, relationship types (1-1,1-m,m-1,m-m)
2. Handling relationships while converting relations into tables in RDB, so that there are no redundancies.



The integrity constraints help us to enforce business rules on data in the database. Once an integrity constraint is specified for a table or a set of tables, all data in the table always conforms to the rule specified by the integrity constraint.

Referential integrity constraints designates a column or grouping of columns in a table called child table as a foreign key and establishes a relationship between that foreign key and specified primary key of another table called parent table.

The following is the list of constraints that can be defined for enforcing the referential integrity constraint.

Constraint	Use	Syntax and Example
Primary key	Designates a column or combination of columns as primary key	PRIMARY KEY (columnname[,columnname])
Foreign key	designates a column or grouping of columns as a foreign key with a table constraint	FOREIGN KEY (columnname[,columnname])
References	Identifies the primary key that is referenced by a foreign key. If only parent table name is specified it automatically references primary key of parent table	columnname datatype REFERENCES tablename[(columnname[,columnname])]
On delete Cascade	The referential integrity is maintained by automatically removing dependent foreign key values when primary key is deleted	columnname datatype REFERENCES tablename[(columnname)][ON DELETE CASCADE]
On update set null	If set, makes the foreign key column NULL, whenever there is a change in the primary key value of the referred table	Constraint name foreign key column- name references referred-table(column- name) on update set null

Rules to Handle relationships , attributes , enhanced E-R concepts during table creation :

Name	Description	Handling	Example	Create statement
One-to-one	A member from	The key	Room & guest.	Create table

	an entity set is connected to atmost one member from the other entity set & vice-versa	attribute from anyone entity set goes to the other entity set (may be the entity set that has full participation in relation) , as a foreign key.	Room no is foreign key in guest relation.guest has full participation in relation.	room( rno int primary key, desc char(30)); Create table guest(gno int, name varchar(20), rno int references room unique);
One-to-many, many-to-one	A member from the entity set on the one side is connected to one or more members from the other entity set, but a member from the entity set on the many side , is connected to atmost one member of the entity set on one side.	The key attribute of the entity set on one side is put as foreign key in the entity set on the many side.	Department & employee. Here department is on the one side & employee is on the many side.	Create table dept(dno int primary key, dname char(20)); Create table emp(eno int primary key, name char(30), dno int references dept);
May-to-many	A member from one entity set connected to one or more members of the other entity set & vice-versa	A new relation is created that will contain the key attributes of both the participating entity sets.	Student & subject . a student can opt for many subjects & a subject has many students opting for it.	Create table student(sno int primary key, name varchar(20)); Create table subject(sbno int primary key, name varchar(20)); Create table st-sub(sno int references student, sbno int references subject, constraint pkey primary key(sno,sbno));
A multivalued attribute	an attribute having multiple values for each member of the entity set	A new relation is created , which will contain a place holder for the multivalued attribute and the key attributes of the entity set that contains the multivalued attribute	An employee having multiple contact numbers, like home phone, mobile number, office number etc. hence the phone-no attribute in employee relation becomes a multivalued attribute.	Create table emp(eno int primary key, name char(30)); Create table emp-ph(eno int references emp, phno int , constraint pkey primary key(eno,phno));

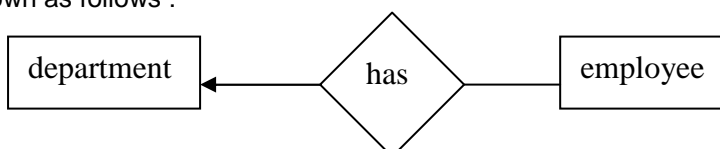
A multivalued, composite attribute	A composite attribute having multiple values , for each member of the entity set	A new relation is created, which will contain a place holder for each part of the composite attribute and the key attributes of the entity set that contains the composite multivalued attribute	An employee having multiple addresses , where each address is made up of a block no, street no, city, state. Hence the address attribute becomes a composite multivalued attribute.	Create table emp(eno int primary key, name char(30)); Create table emp-add(eno int references emp, addno int, hno int, street char(20), city char(20), constraint pkey primary key(eno,addno));
Generalization / specialization	The members of an entity set can be grouped into several subgroups, based on an attribute/s value/s. Each subgroup becomes an entity set. Depicts a parent-child type of relationship.	New relations for each subgroup , if the subgroups have its own attributes, other than the parent attributes. The parent entity set's key is added to each subgroup. If no specific attributes for each subgroup, then only the parent relation is created.	A person ( parent entity set) can be an employee, a student, a retired person. Here employee has its own set of attributes like company, salary etc. a student has its own set of attributes like college/ school, course etc. a retired person has its own set of attributes like hobby, pension etc. so we create a person relation , a student relation, an employee, a retire person. The student , employee, retired person entity sets will have the key of the person entity set added to it.	Create table person(ssno int primary key, name char(30)); Create table emp(ssno int references person, eno int, cname char(20), sal float, primary key(ssno)); Create table student(ssno int references person, class char(10), school varchar(50), primary key(ssno));



Self-Activity

You can type the following Create table Statements to create the tables satisfying referential integrity constraints. On table creation type \d <table name> and write the output.

1. Consider two tables department & employee. One department can have one or more employees, but an employee belongs to exactly one department ( 1-m relation). It's pictorially shown as follows :

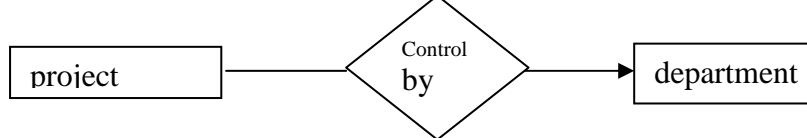




To handle the above relation, while creating the tables, 'deptno' is a foreign key in the employee table. The statement for creating the tables are as follows :

Create table department ( deptno integer primary key, deptname text);  
 Create table employee( empno integer primary key, ename varchar(50), salary float, dno integer references department(deptno) on delete cascade on update set null);

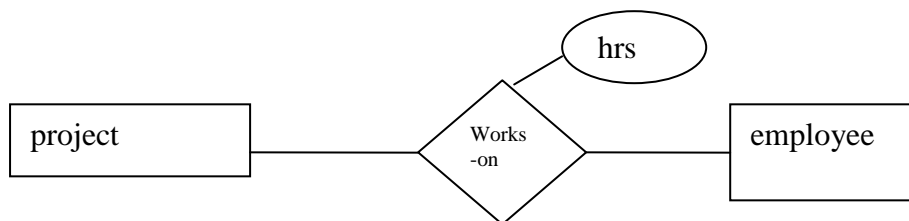
2. Consider the department table created above & another table called project. A project is controlled by exactly one department , but a department can control one or more projects( a m-1 relation). It's pictorially shown as follows :



To handle the above relationship, control-dno is a foreign key in project. The statement for creating the project table is as follows :

Create table project(pno int primary key, pname char(10), control-dno int, foreign key(control-dno) references department(dno))

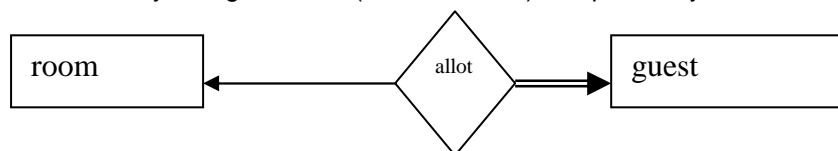
3. Consider the project & employee relations created above. An employee can work in one or more projects, and a project can have one or more employees working in it .( a m-m relation). It's shown pictorially as follows :



To handle the above relationship, we create a new table , works-on , as given below :

create table works(eno int references employee, pno int references project, hrs int, constraint pkey primary key(eno,pno))

4. Consider the relations guest and room. A guest is allocated exactly one room, and a room can contain exactly one guest in it. ( a 1-1 relation). It's pictorially shown as follows :



To handle the above relation, we add room-no as foreign key to guest, since a guest cannot be without being allocated to a room ( guest has full participation in relation). The statements for creating these relations are as follows

Create table room(room-no integer primary key , description char(20), charge integer);  
 Create table guest(guest-no integer primary key, name varchar(30), room-no references room unique);

Signature of the instructor

Date  /  /



### Set A

Create tables for the information given below by giving appropriate integrity constraints as specified.

1. Create the following tables :

Table Name		Property		
Columns	Column Name	Column Data Type	Constraints	
1	Pnumber	Integer	Primary key	
2	description	varchar (50)	Not null	
3	Area	Char(10)		

Table Name		Owner		
Columns	Column Name	Column Data Type	Constraints	
1	Owner-name	Varchar(50)	Primary key	
2	Address	varchar (50)		
3	Phoneno	Integer		

Relationship → A one-many relationship between owner & property. Define reference keys accordingly .

2. Create the following tables :

Table Name		Hospital		
Columns	Column Name	Column Data Type	Constraints	
1	Hno	Integer	Primary key	
2	Name	varchar (50)	Not null	
3	City	Char(10)		

Table Name		Doctor		
Columns	Column Name	Column Data Type	Constraints	
1	Dno	Integer	Primary key	
2	Dname	varchar (50)		
3	City	Char(10)		

Relationship → A many-many relationship between hospital & doctor.

3. Create the following tables :

Table Name		Patient		
Columns	Column Name	Column Data Type	Constraints	
1	pno	Integer	Primary key	
2	Name	varchar (50)	Not null	
3	Address	Varchar(50)		

Table Name		Bed		
Columns	Column Name	Column Data Type	Constraints	
1	Bedno	integer	Primary key	
2	Roomno	integer	Primary key	
3	description	Varchar(50)		

Relationship → a one-to-one relationship between patient & bed.

Signature of the instructor

Date  /  /

**Set B**

Create table for the information given below by choosing appropriate data types and integrity constraints as specified.

1. Table \_\_\_\_\_ ( \_\_\_\_\_, \_\_\_\_\_, \_\_\_\_\_, \_\_\_\_\_  
\_\_\_\_\_ ( \_\_\_\_\_, \_\_\_\_\_, \_\_\_\_\_, \_\_\_\_\_  
Constraints: \_\_\_\_\_, \_\_\_\_\_

2. Table \_\_\_\_\_ ( \_\_\_\_\_, \_\_\_\_\_, \_\_\_\_\_, \_\_\_\_\_  
\_\_\_\_\_ ( \_\_\_\_\_, \_\_\_\_\_, \_\_\_\_\_, \_\_\_\_\_  
Constraints: \_\_\_\_\_, \_\_\_\_\_

Relationship → \_\_\_\_\_

Instructor should fill in the blanks with appropriate values.

Signature of the instructor

Date

**Assignment Evaluation**

**Signature**

0: Not done	<input type="text"/>	2: Late Complete	<input type="text"/>	4: Complete	<input type="text"/>
1: Incomplete	<input type="text"/>	3: Needs improvement	<input type="text"/>	5: Well Done	<input type="text"/>

**Exercise 13**

**Start Date**



To create one or more tables with Check constraint , Unique constraint, Not null constraint , in addition to the first two constraints (PK & FK)



You should read following topics before starting this exercise

1. Different integrity constraints
2. Specification of different integrity constraints , in create table statement .



The following is the list of additional integrity constraints.

Constraint	Use	Syntax and Example
NULL	Specifies that the column can contain null values	CREATE TABLE client_master (client_no text NOT NULL , name text NOT NULL, addr text NULL , bal_due integer);
NOT NULL	Specifies that the column can not contain null values	CREATE TABLE client_master (client_no text NOT NULL , name text NOT NULL, addr text NOT NULL , bal_due integer);
UNIQUE	Forces the column to have unique values	CREATE TABLE client_master (client_no text UNIQUE, name text ,addr text, bal_due integer);
CHECK	Specifies a condition that each row in the table must satisfy. Condition is specified as a logical expression that evaluates either TRUE or FALSE.	CREATE TABLE client_master (client_no text CHECK(client_no like 'C%'), name text CHECK(name=upper(name)), addr text);



1. create client master by using any one DDL statement given above. On table creation type \d <table name> and write the output

Signature of the instructor

Date



**Set A**

1.

Create a table with details as given below

Table Name		Machine	
Columns	Column Name	Column Data Type	Constraints
1	Machine_id	integer	Primary key
2	Machine_name	varchar (50)	NOT NULL, uppercase
3	Machine_type	varchar(10)	Type in ( 'drilling', 'milling', 'lathe', 'turning', 'grinding')
4	Machine_price	float	Greater than zero
5	Machine_cost	float	
Table level constraint		Machine_cost less than Machine_price	

2.

Create a table with details as given below

Table Name		Employee	
Columns	Column Name	Column Data Type	Constraints
1	Employee_id	integer	Primary key
2	Employee_name	varchar (50)	NOT NULL, uppercase
3	Employee_desg	varchar(10)	Designation in ( 'Manager', 'staff', 'worker')
4	Employee_sal	float	Greater than zero
5	Employee_uid	text	Unique
Table level constraint		Employee_uid not equal to Employee_id	

Signature of the instructor

Date

**Set B**

1.

Create a table with details as given below

Table Name			
Columns	Column Name	Column Data Type	Constraints
1			
2			
3			
4			
5			
Table level constraint			

Instructor should fill in the blanks with appropriate values.

Signature of the instructor

Date

**Assignment Evaluation**

**Signature**

0: Not done

2: Late Complete

4: Complete

1: Incomplete

3: Needs improvement

5: Well Done

**Exercise 14**

**Start Date** / /



To drop a table from the database, to alter the schema of a table in the Database.



You should read following topics before starting this exercise  
1 Read through the drop, Alter DDL statement



The Drop & Alter DDL statements :

Name	Description	Syntax	Example
Drop	Deletes an object (table/view/constraint) schema from the database	Drop object-type object-name;	Drop table employee; Drop table sales; Drop constraint pkey;
Alter	<p>ALTER TABLE command of SQL is used to modify the structure of the table It can be used for following purposes</p> <p>a) adding new column b) modifying existing columns c) add an integrity constraint d) To redefine a column</p> <p>Restrictions on the alter table are that, the following tasks cannot be performed with this clause</p> <p>a) Change the name of the column b) Drop a column c) Decrease the size of a column if table data exists</p>	<p>Alter table tablename Add ( new columnname datatype(size), new columnname datatype(size)...);</p> <p>Alter table tablename modify (columnname new datatype(new size),..);</p>	<p>Alter table emp( add primary key (eno)); alter table student( add constraint city- chk check city in (‘pune’, ‘mumbai’));</p> <p>alter table student modify (city varchar(100));</p>



Create the table given below . Assume appropriate data types for attributes.  
Modify the table, as per the alter statements given below, type \d <table name>  
and write the output.

Supplier\_master( supplier\_no, supplier\_name,city,phone-no,amount)

1. Alter table supplier\_master  
add primary key (supplier\_no);
2. Alter table supplier\_master add constraint city-check check city in(‘pune’, ‘mumbai’, ‘calcutta’);
- 3.alter table supplier\_master drop phone-no;

4. alter table supplier\_master modify (supplier\_name varchar(50));
5. alter table supplier\_master drop constraint city-check;
6. drop table supplier\_master;

Signature of the instructor

Date



**Set A**

1. Remove employee table from your database. Create table employee( eno, ename, sal). Add designation column in the employee table with values restricted to a set of values.
2. Remove student table from your database. Create table student( student\_no, sname, date\_of\_birth). Add new column into student relation named address as a text data type with NOT NULL integrity constraint and a column phone of data type integer.
3. Consider the project relation created in the assignment 12. Add a constraint that the project name should always start with the letter 'R'
4. Consider the relation hospital created in assignment 12. Add a column hbudget of type int , with the constraint that budget of any hospital should always > 50000.

Signature of the instructor

Date

**Set B**

1. Remove \_\_\_\_\_ table from your database. Create table \_\_\_\_\_( \_\_\_\_\_, \_\_\_\_\_, \_\_\_\_\_). Add \_\_\_\_\_

Instructor should fill in the blanks with appropriate values.

Signature of the instructor

Date

**Assignment Evaluation**

**Signature**

0: Not done <input type="text"/>	2: Late Complete <input type="text"/>	4: Complete <input type="text"/>
1: Incomplete <input type="text"/>	3: Needs improvement <input type="text"/>	5: Well Done <input type="text"/>

**Exercise 15**

**Start Date** / /



To insert / update / delete records using tables created in previous Assignments. ( use simple forms of insert / update / delete statements)



You should read following topics before starting this exercise

1. Read through the insert , update, delete statement.
2. Go through the variations in syntaxes of the above statements.
3. Go through some examples of these statements
4. Go through the relationship types & conversion of relations to tables in RDB.
5. Normal forms → 1NF, 2NF, 3NF



The insert / update / delete statements

Name	Description	Syntax	Example
Insert	<p>The insert statement is used to insert tuples or records into a created table or a relation.</p> <p>We specify a list of comma-separated column values, which must be in the same order as the columns in the table.</p> <p>To insert character data we must enclose it in single quotes('). If a single quote is part of the string value to be inserted , then precede it with a backslash(\).</p> <p>When we don't have values for every column in the table, or the data given in insert is not in the same column order as in the table, then we specify the column names also alongwith the values (2<sup>nd</sup> syntax)</p>	<p>INSERT INTO tablename VALUES (list of column values);</p> <p>INSERT INTO tablename(list of column names) VALUES (list of column values corresponding to the column names );</p>	<p>Insert into emp values(1,'joshi',4000,'pune');</p> <p>Insert into emp(eno,city) values(2,'mumbai');</p>



Update	The UPDATE command is used to change or modify data values in a table. To specify update of several columns at the same time, we simply specify them as a comma-separated list	UPDATE tablename  SET columnname = value where condition;	Update emp set sal = sal +0.5*sal;  Update emp set sal = sal+1000 where eno =1;
Delete	The DELETE statement is used to remove data from tables.	DELETE FROM table name where condition;	Delete from emp ;  Delete from emp where eno = 1;

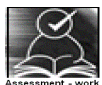


Consider the tables created in assignments 11,12,13,14. type and execute the below statements for these tables. Write the output of each statement & justify your answer

1. INSERT INTO sales\_order(s\_order\_no,s\_order\_date,client\_no)  
VALUES ('A2100', now() , 'C40014');
2. INSERT INTO client\_master values('A2100','NAVEEN','Natraj apt','pune\_nagar road','pune',40014);
3. insert into client\_master values ('A2100','NAVEEN',NULL,'pune\_nagar road','pune',40014);
4. UPDATE emp SET netsal= net\_sal\_basic\_sal\*0.15;
5. UPDATE student  
SET name='SONALI',address='Jayraj apartment'  
WHERE stud\_id=104 ;
6. DELETE from emp;
- 7.DELETE from emp  
WHERE net\_sal <1000;

Signature of the instructor

Date

 /  / 


### Set A

1. Create the following tables ( primary keys are underlined.).  
Property(pno,description, area)  
Owner(oname,address,phone)

An owner can have one or more properties, but a property belongs to exactly one owner . Create the relations accordingly ,so that the relationship is handled properly and the relations are in normalized form (3NF).

- a) Insert two records into owner table.
- b) insert 2 property records for each owner .
- c) Update phone no of "Mr. Nene" to 9890278008
- d) Delete all properties from "pune" owned by " Mr. Joshi"

- 2 . Create the following tables ( primary keys are underlined.).  
Emp(eno,ename,designation,sal)  
Dept(dno,dname,dloc)

There exists a one-to-many relationship between emp & dept.

Create the Relations accordingly, so that the relationship is handled properly and the relations are in normalized form (3NF).

- a) Insert 5 records into department table
- b) Insert 2 employee records for each department.
- c) increase salary of "managers" by 15%;
- d) delete all employees of department 30;
- e) delete all employees who are working as a "clerk"
- f) change location of department 20 to 'KOLKATA'

3 . Create the following tables ( primary keys are underlined.).

Sales\_order(s\_order\_no,s\_order\_date)

Client(client\_no, name, address)

A client can give one or more sales\_orders , but a sales\_order belongs to exactly one client. Create the Relations accordingly, so that the relationship is handled properly and the relations are in normalized form (3NF).

- a) insert 2 client records into client table
- b) insert 3 sales records for each client
- c) change order date of client\_no 'C004' to 12/4/08
- d) delete all sale records having order date before 10th feb. 08
- e) delete the record of client "joshi"

Signature of the instructor

Date

**Set B**

1. Design a set of tables with the following constraints. Add any new attributes , as required by the queries.

Table name :

Field name	Data Type	Constraints

Table name :

Field name	Data Type	Constraints

Relationship → \_\_\_\_\_

Write & execute insert/ update / delete statements for following business tasks

- a)
- b)
- c)
- d)
- e)
- f)

Instructor should fill in the blanks with appropriate values.

Signature of the instructor

Date