

F. Y. B. Sc.
(Computer Science)
Laboratory Course I & II
WorkBook

Name _____

College Name _____

Roll No. _____ Division _____

Academic Year _____

Written and Edited by

Dr. Shailaja C. Shirwaikar

Prof. Poonam Ponde

Prof. Reena Bharathi

Prof. Manisha Suryavanshi

Prof. Jeevan Limaye

Prof. Kalyani Ghanwat

Prof. Anagha Joshi

Prof. Vandana Babrekar

Prof. Manasi Keskar

Prof. Seema Purandare

Prof. Padmavathy M.

Prof. Anjali Sardesai

Prof. Parag Tamhankar

Prof. Rasika Rahalkar

Foreword

It gives me great pleasure to introduce this lab book for the F.Y.B.Sc (Computer Science) students. This being the first year of the course, a good foundation laid in this year will make it easier for students to grasp advanced concepts in the years to come. I congratulate the University of Pune and all faculty members who have taken the initiative and effort in bringing out this much needed book.

The importance of laboratory courses in the subject of Computer Science cannot be emphasized enough. It is imperative that the theory, which students learn in the classroom, is supported by appropriate and in-depth hands-on experience in the laboratory. Moreover, it has to be conducted in a regular and systematic manner for complete understanding of the subject. I am sure that this lab book will satisfy this need.

The laboratory assignments are designed in such a way that they consider the different learning pace of students and encourage original thinking. There is scope for variations in the questions posed thereby reducing the possibility of duplication of effort. The continuous assessment will prove beneficial not only to the students but also the institutions to gauge the performance of the student over the duration of the course. Most importantly, the collaborative wisdom of all the experts who have worked on this book will reach learners in the remotest locations.

I understand that this is the first collaborative effort of its kind. I am sure that it will be very successful and set a precedent for similar efforts in other subjects as well. I wish it all success.

Dr. Achyut S. Godbole

Table of contents

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

Lab course I	
Exercise 1	100
To demonstrate use of data types, simple operators (expressions)	
Exercise 2	103
To demonstrate decision making statements (if and if-else, nested structures)	
Exercise 3	106
To demonstrate decision making statements (switch case)	
Exercise 4	111
To demonstrate use of simple loops	
Exercise 5	116
To demonstrate use of nested loops	
Exercise 6	119
To demonstrate menu driven programs and use of standard library functions.	
Exercise 7	122
To demonstrate writing C programs in modular way (use of user defined functions)	
Exercise 8	125
To demonstrate recursive functions.	
Exercise 9	128
To demonstrate use of arrays (1-d arrays) and functions	
Exercise 10	132
To demonstrate use of multidimensional array(2-d arrays) and functions	
Exercise 11	136
To demonstrate use of pointers	
Exercise 12	140
To demonstrate concept of strings (strings and pointers)	
Exercise 13	144
To demonstrate array of strings.	
Exercise 14	146
To demonstrate use of bitwise operators.	
Exercise 15	149
To demonstrate structures (using array and functions)	
Exercise 16	152
To demonstrate nested structures and Unions	
Exercise 17	157
To demonstrate command line arguments and preprocessor directives.	
Exercise 18	160
To demonstrate file handling (text files)	
Exercise 19	164
To demonstrate file handling (binary files and random access to files)	
Exercise 20	167
Problem solving using C	
Appendix A- Guidelines for setting up the lab.....	172
References	180

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



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)	

Lab Course I		
Sr. No	Assignment Name	Grade
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.

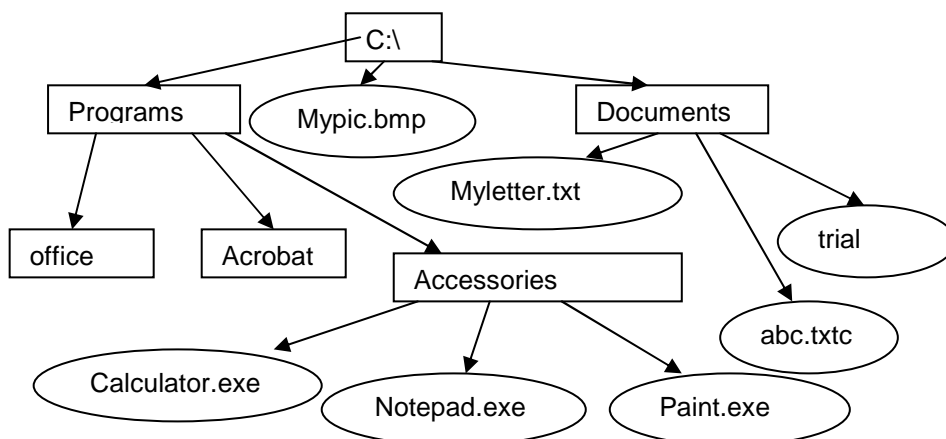


Operating system is an interface between the user and the computer hardware. MS-DOS (Microsoft's Disk Operating System) runs on any of the Intel 8088,80x86 or Pentium class CPU's on a Personal computer platform. The version of MS-DOS that runs on early IBM computers is called PC-DOS.

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

External Commands

ATTRIB	Used to change or display various file attributes.	<p>ATTRIB properties filename + sets</p> <p>- removes the set attributes</p> <p>Properties R : Read only</p> <p>A : Archive</p> <p>S : System file</p> <p>C:\> ATTRIB +R a.txt</p> <p>Makes the file a.txt Read-only</p> <p>C:\> ATTRIB +H C:\D2\D21\a.txt</p> <p>Makes file a.txt hidden file</p>
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.	<p>FORMAT [drive:] [/switches]</p> <p>Switches</p> <p>/I Formats double sided disk as a single sided disk</p> <p>/B leaves room for system files but system files are not copied</p> <p>/Q Quick formatting</p> <p>/S Transfer DOS system files to the formatted disk</p> <p>/U Prompts the user to add a volume label to the disk</p>
CHKDSK	Used to check the disk for errors and displays a status report.	<p>CHKDSK filename option</p> <p>Option</p> <p>/F Automatic correction of errors</p> <p>/V Displays a series of messages indicating the progress</p> <p>C:\> CHKDSK a:</p>
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



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

2: Late Complete

1: Incomplete

3: Needs improvement

Signature

4: Complete

5: Well Done

<input type="text"/>
<input type="text"/>

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

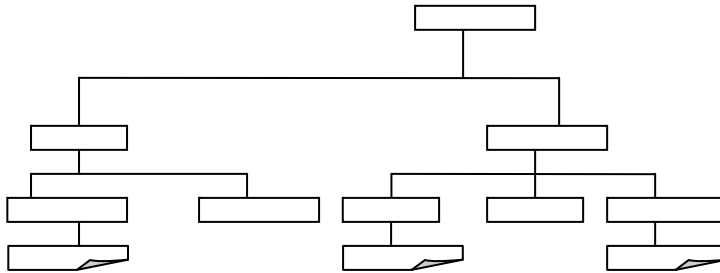
Signature of the instructor

Date



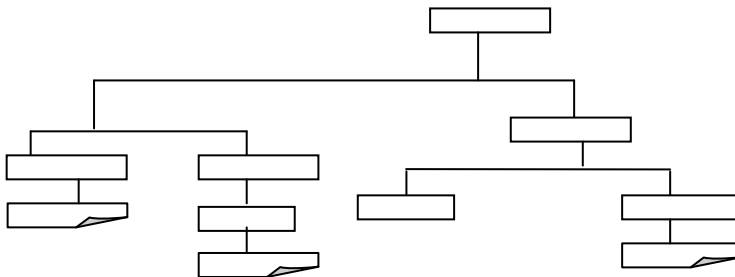
Set A

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

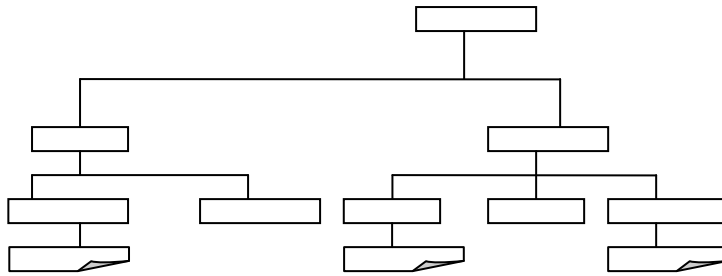
Date / /

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

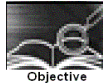
Signature

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.





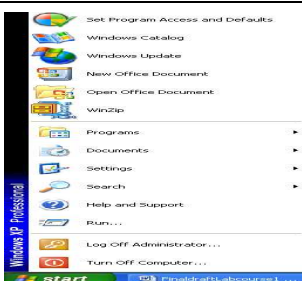







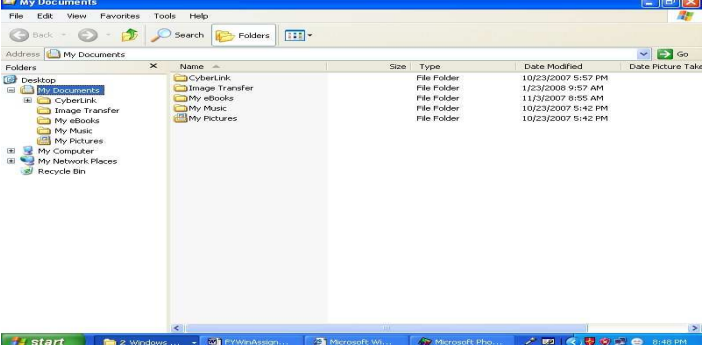
Microsoft has been making OS software utilizing graphical user interfaces since around 1985. Some of the earlier windows versions were Windows 3.1 (1990), Windows 95 (1995), Windows 98 (1998), Windows ME (2000), Windows 2000 (2000), and Windows XP (2001). Windows XP comes in two bundles Windows XP Professional 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:



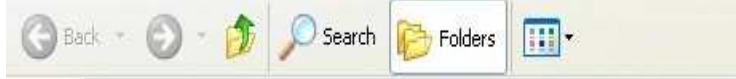


1. Assisting in starting and shutting down of a pc.
2. Controlling and handling the hardware, including RAM, I/O cards etc.
3. Providing a graphics user interface including various features.
4. Provides a platform for applications to execute like Word.
5. File Handling.
6. Provides an interface for various 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 "C:\My Documents\Hello World.txt"</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 - Hello World.txt. 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"> • .doc = Microsoft Word Document • .xls = Microsoft Excel Document • .ppt = Microsoft PowerPoint Presentation • .mdb = Microsoft Access Database • .bmp = Windows Bitmap Picture • .wav = Sound File • .html or .htm = Internet Document
<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>																														
<p>System Tray</p>	 <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>																														
<p>My Computer</p>	 <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>																														
<p>Recycle Bin</p>	 <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 Empty Recycle Bin.</p>																														
<p>My Documents</p>	 <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>																														
<p>Internet Explorer</p>	 <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>																														
<p>Window</p>	 <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 shows the following details:</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	 <p>Title bar shows the name of the Application we are in.</p>
Menu Bar	 <p>The menu bar contains the menus that will allow us access to all the operations that can be done with a file or folder</p>
Standard Bar	
Tool Bar	 <p>Optional. This bar includes the most commonly used buttons</p>
Minimize, Maximize, Restore, Close	 <p>These are the series of buttons which are present on the top right hand corner of every window.</p>

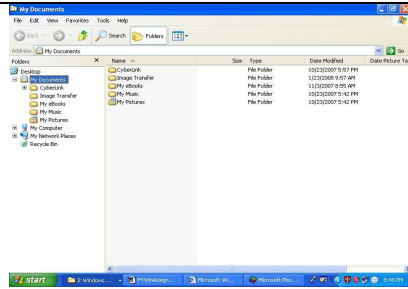
Windows Explorer

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 start button and selecting the explore menu option as shown above</p> <p>The other would be click start -> programs -> accessories -> windows explorer</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



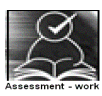
Self-Activity

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

/ /



Objective

Using basic Linux commands



Reading

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.



Ready Reference

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 > 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	\$cat < abc.txt Takes its input from abc.txt and the output by default is on console. The effect is same as \$cat tempfile
Output Redirection	>	It detaches the console from the standard output of command and attaches specific file	\$cat > file1 Takes its input from keyboard by default and writes the output to file1, effectively whatever typed at the keyboard goes into tempfile \$cat file1 abc.txt > file2 The contents of file1 and abc.txt will be concatenated and send to file2 \$cat file1 > /dev/lp0 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	\$cat file1 > file1 The file1 contents will be cleared \$cat file2 >> file2 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	\$ ls -l grep "abc" 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

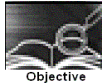
Signature

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

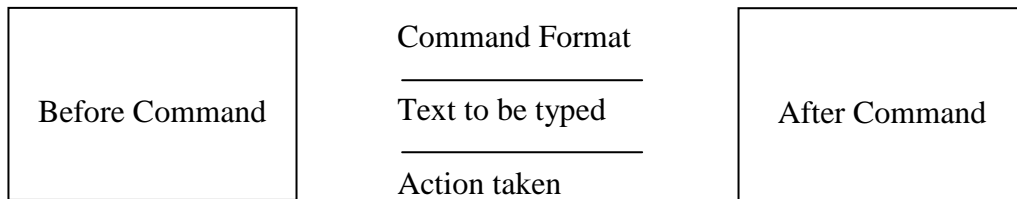
1. Three modes in which vi editor works
2. Commands in vi input mode for inserting, replacing, saving and quitting.
3. Commands in vi for deleting, paging and scrolling,
4. Undoing last editing instructions, search and replace



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
Using vi command			Delete and change		
*1.	vi file	Edit file	*16	dd	Delete line
*2.	vi -r file	Recover file from crash	*17	cc	Change line
Basic Cursor motions			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
Screen Control			*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	Word Commands		
9.	^L	Reprint page	*25.	w	Next word
Character input modes			*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	Generic commands <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
Search			Control Commands		
32.	/string/	Search for string	*42.	:w	Write file
33.	?string?	Reverse search for	*43.	:wq	Write file and quit

We are following the demo in the sequence shown below.



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

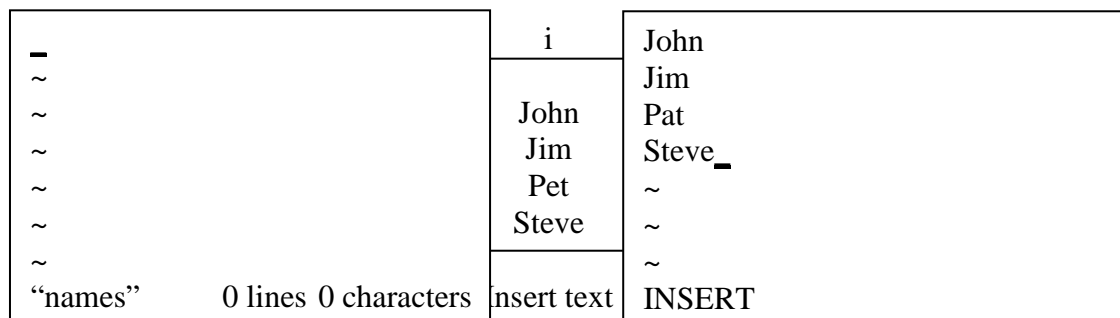
Getting more comfortable with the moving around the screen, we are now trying to add some text. To add the text into the file through vi editor you need to enter into the insert mode. The insert mode of the vi editor follows two scenarios.

Using the “i” (Insert) command

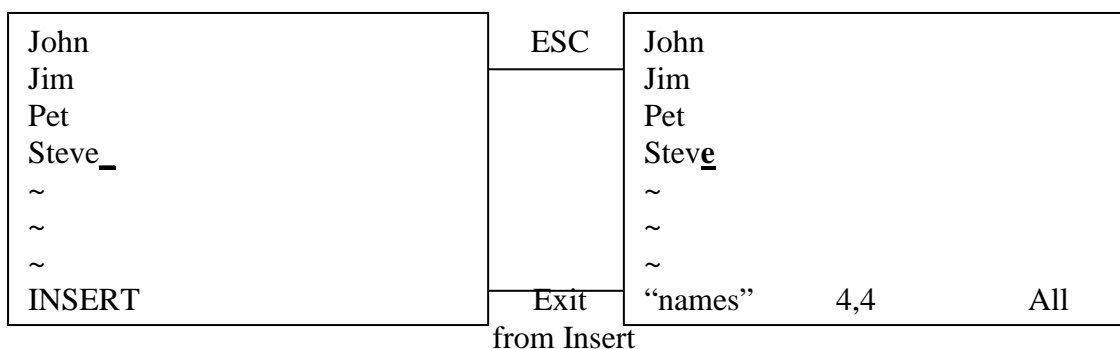
Using the “a” (append) command

Inserting a text using “i” (insert) command

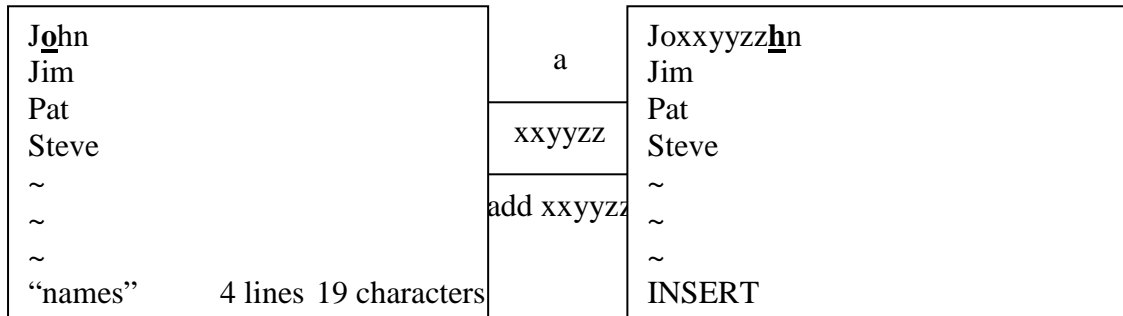
By pressing the “i” character you can have the insert mode of the vi editor. The characters typed by you are placed *before* the current character position. To come out from the insert mode, required to press an ESC key.



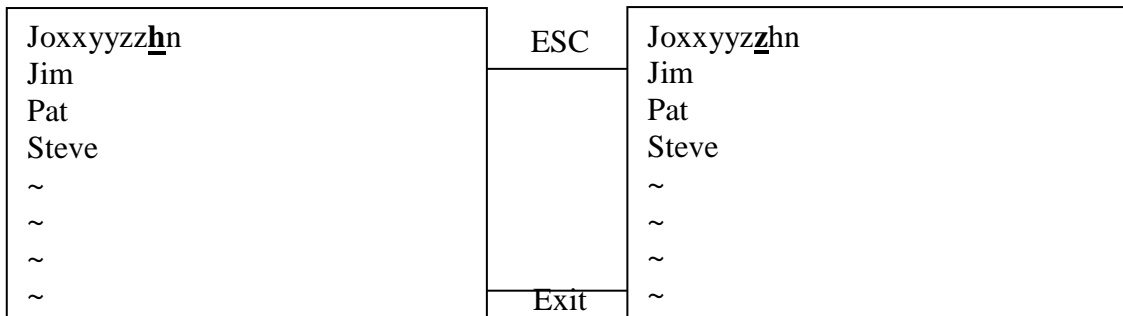
After the ESC is pressed, the cursor moves back to the last character inserted, just as with the “a” command.



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:

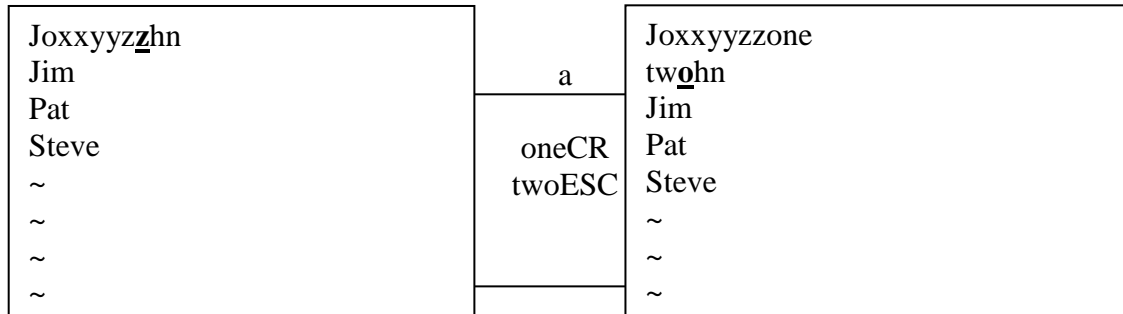


When you are done adding text, you press the ESC key. When you press ESC key, the cursor moves back to the last character you entered.



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.

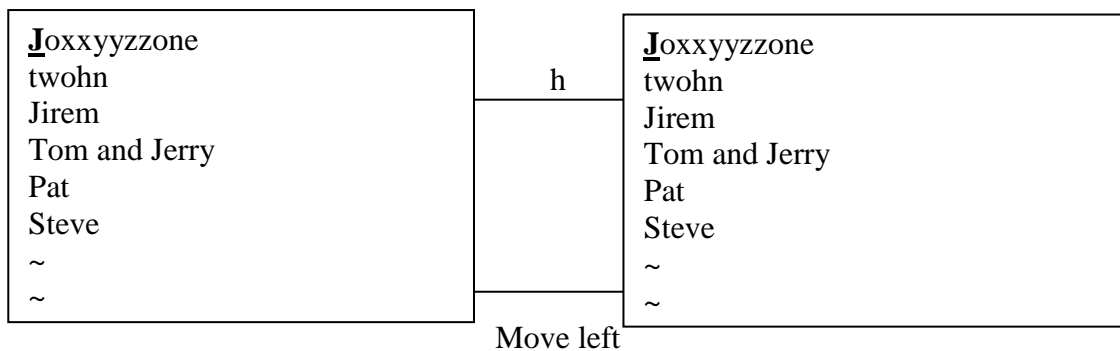
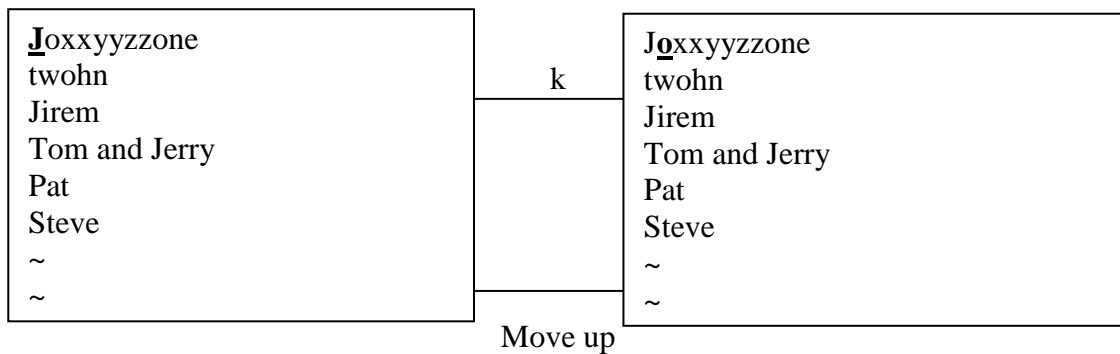
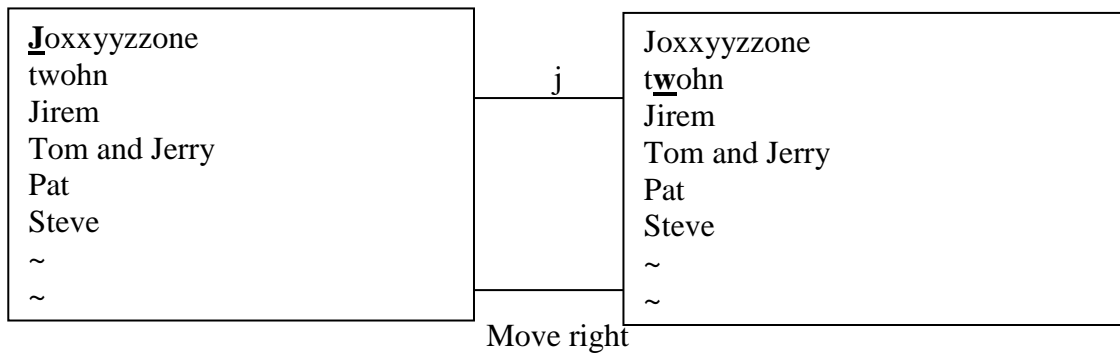
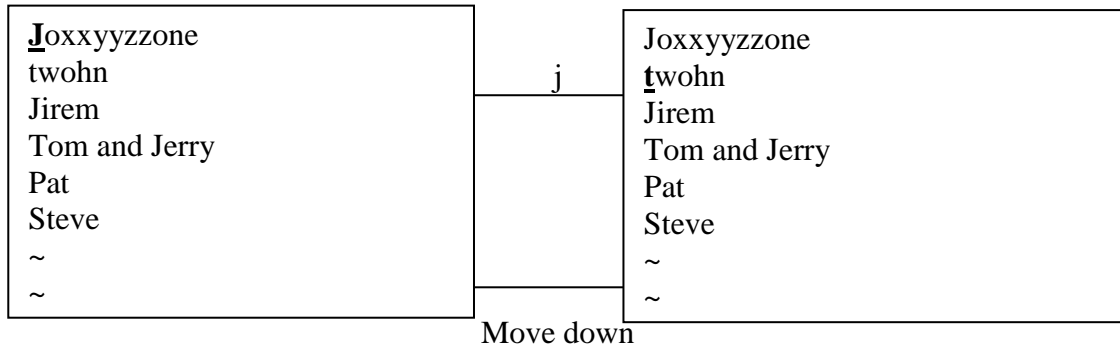
1. Perform the following changes to your file. Specify the command and the resulting text as a answer.

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
~
~

```

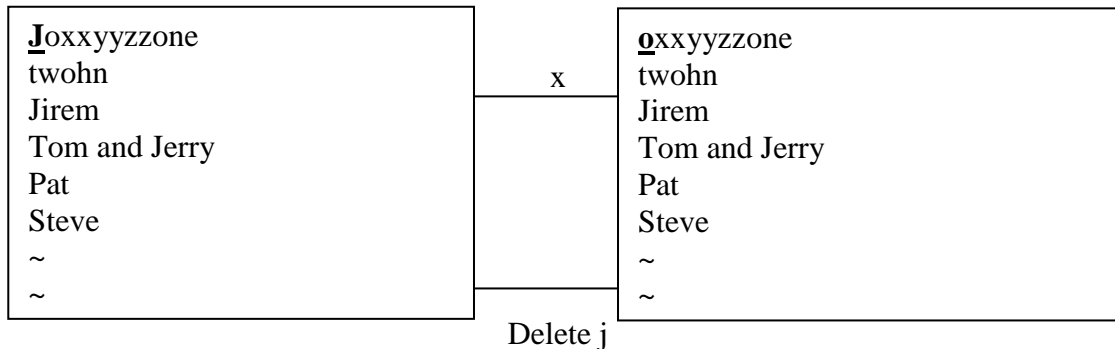
2. Perform the following operation with your file by specifying the command and the resulting text as answer.

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

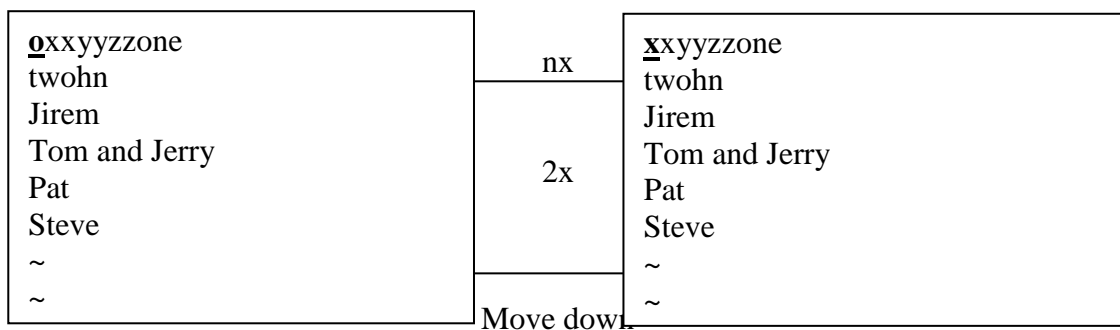
• **Deleting Text**

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

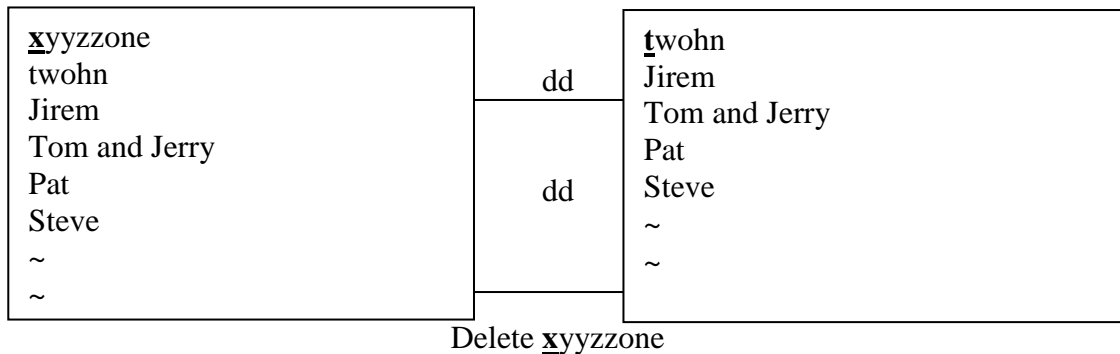
To delete one character, you use the “*x*” command. “*x*” deletes the character at the end current cursor position, moving the rest of the line left into the void created by the deleted character.



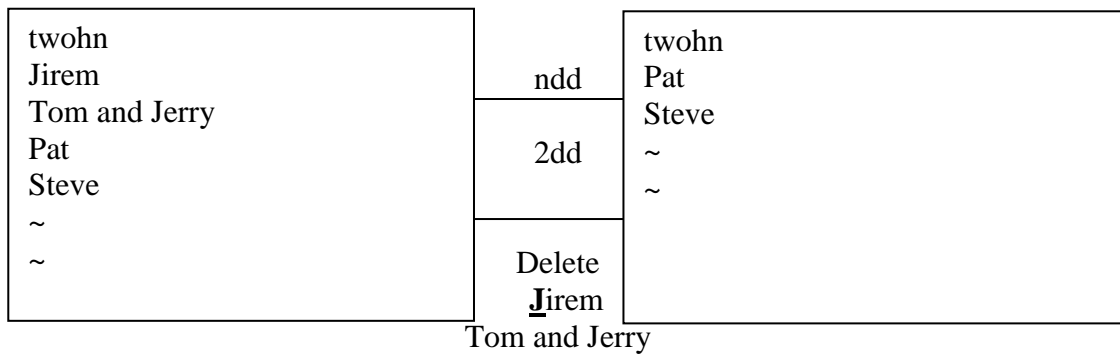
The “*x*” command can be preceded by a number to indicate how many characters you want to delete. You will get the “beep”, if you are trying to delete nonexistent characters.



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.

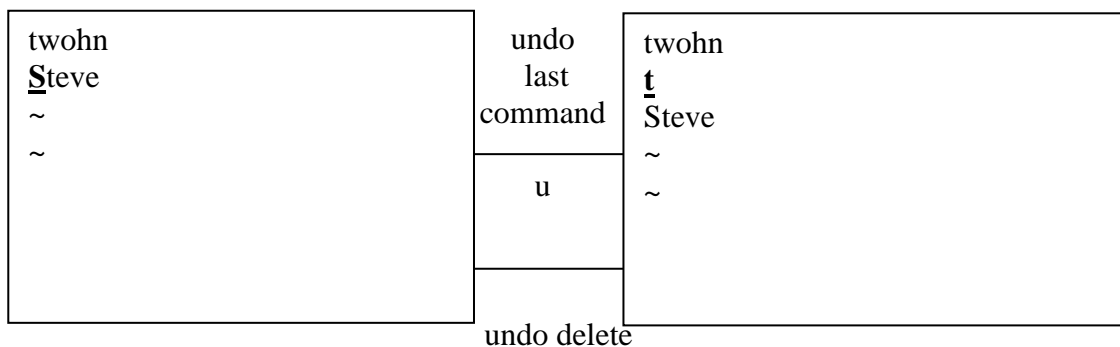


3. Perform the following operation by specifying the command and the resulting text as answer.

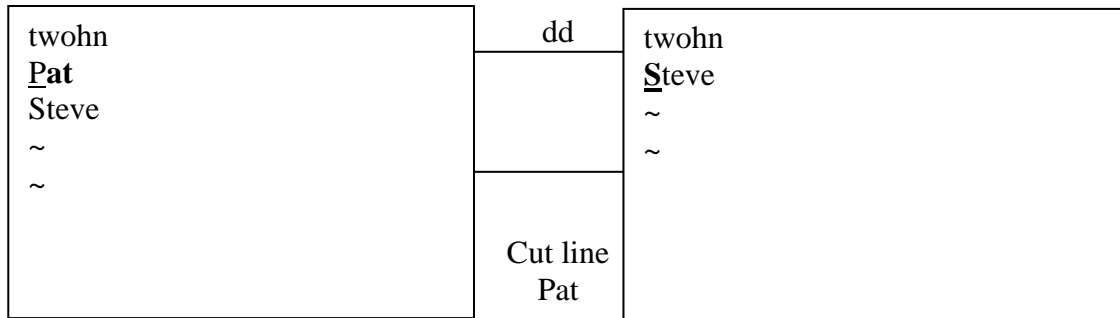
Action (from current cursor position)	Command typed	Result
Delete 2 characters		
Delete 3 characters from 3 rd line		
Delete 1 st line		
Delete 4 th line		

Miscellaneous Command

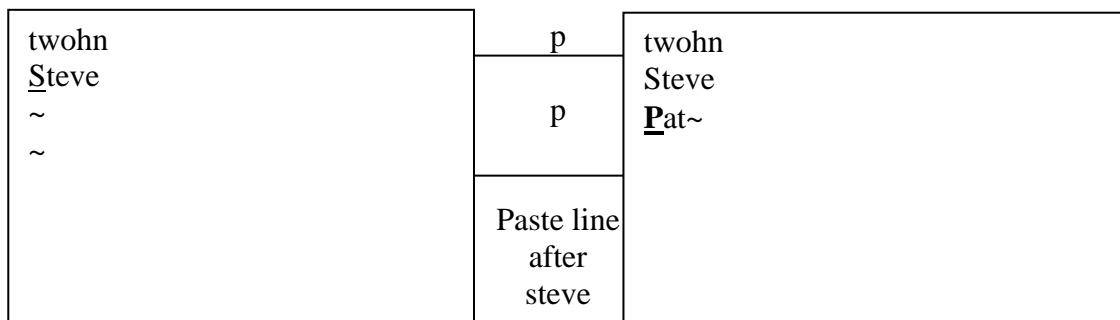
If we need to undo the activity, it is achieved by means of “u” command.



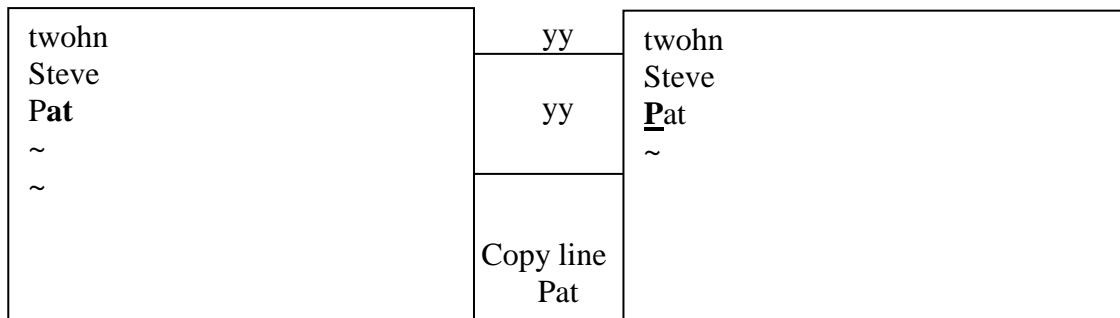
Cut the line(s) from the desired cursor position and paste those lines to the desired cursor position



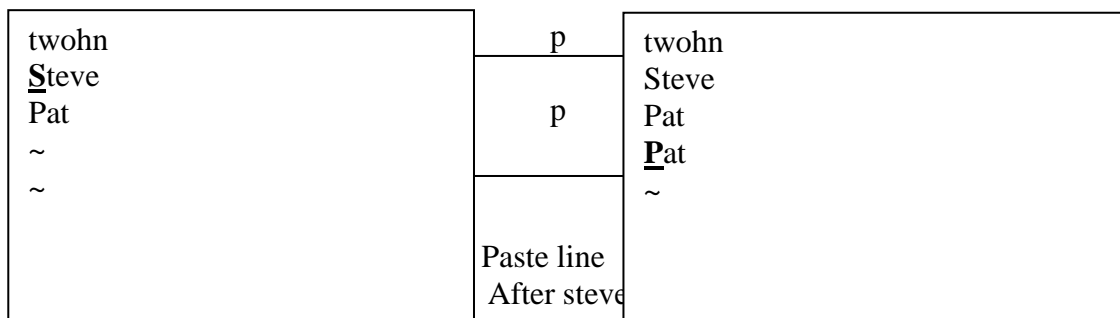
The “p” character is used to paste the line(s) before the desired cursor position.



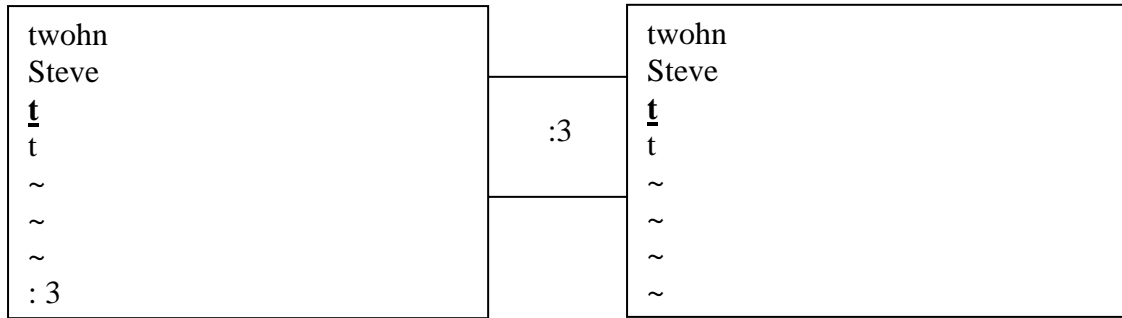
Copy the line(s) from the desired cursor position and paste those lines to the desired cursor position



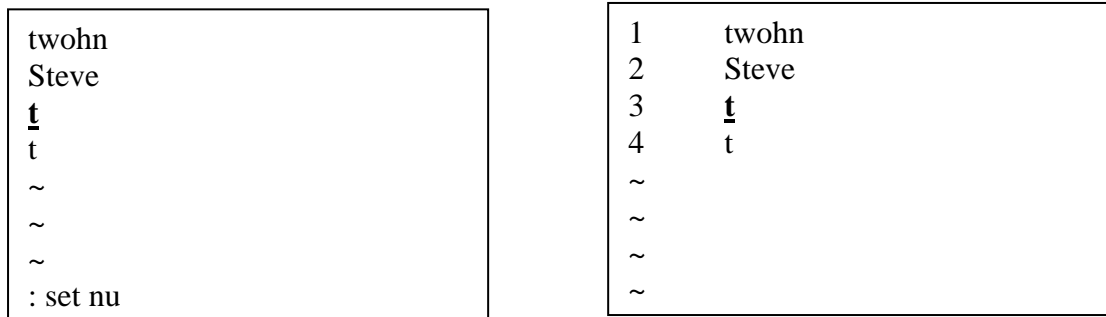
The “p” character is used to paste the line(s) before the desired cursor position.



: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.

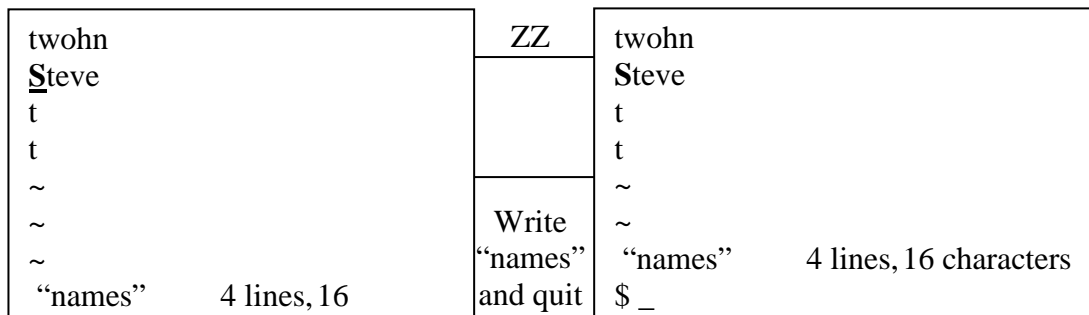


4. Perform the following operation by specifying the command and the resulting text as answer.

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		

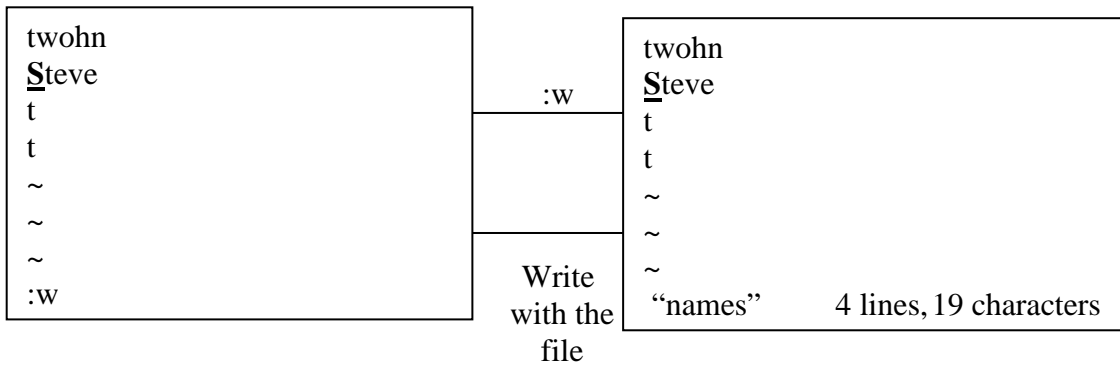
- **Control Commands**
- **Saving the file**

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.

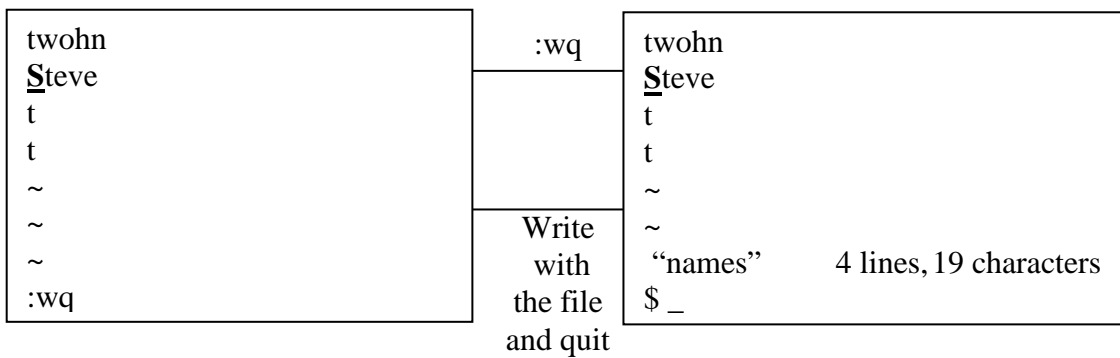


Usage of control commands through “:” prompt of vi editor.

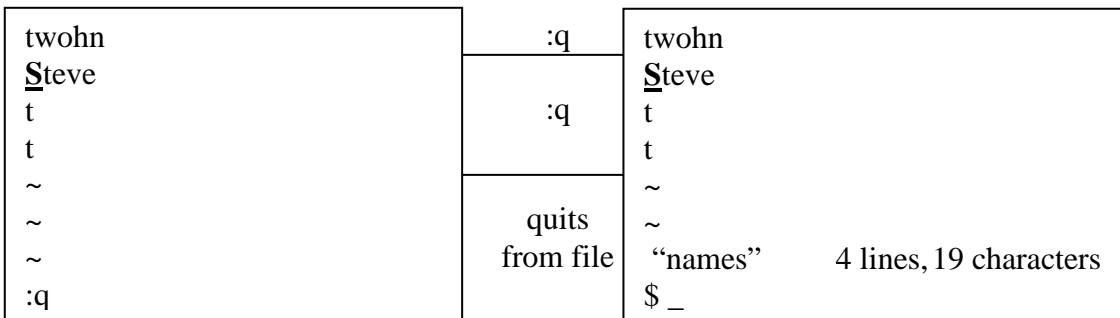
1. `:w` – command write the file without quitting vi editor.



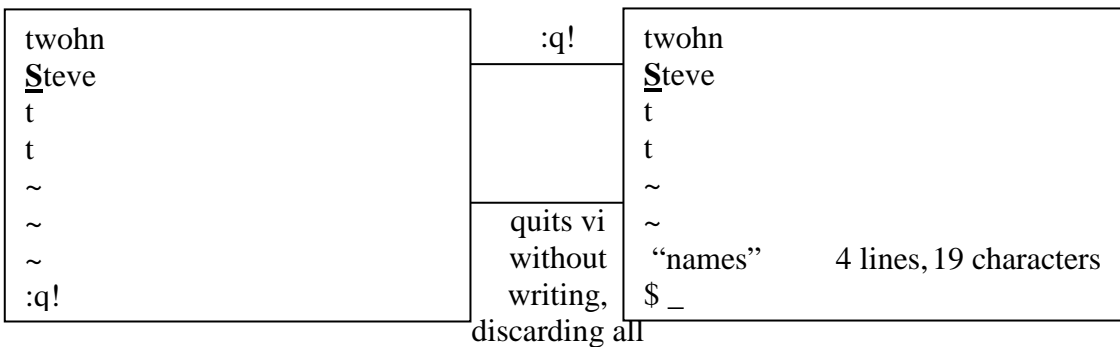
`:wq` – command write the file and put you back on to the shell prompt.



2. `:q` – command quits vi editor.



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



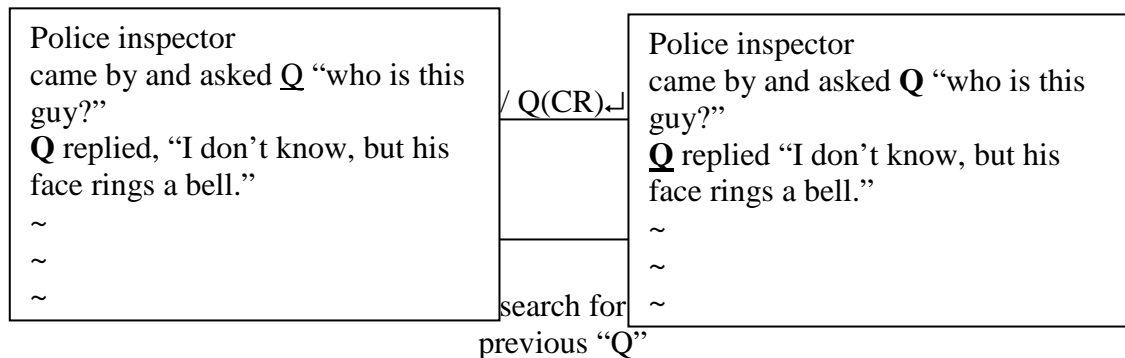
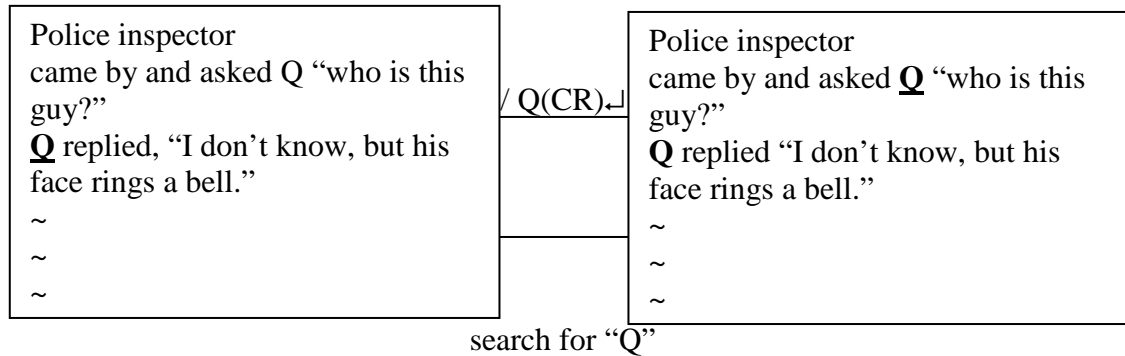
4. Perform the following operation by specifying the command and the resulting text as answer.

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?” Q replied “I don’t know, but his face rings a bell.” ~ ~ ~
	w	
	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?” Q replied “I don’t know, but his face rings a bell.” ~ ~ ~
	b	
	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?” Q replied “I don’t know, but his face rings a bell.” ~ ~ ~
	e	
	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</code> <code>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

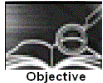
Signature

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

Exercise 8

Start Date

/ /



Objective

Creating simple HTML pages.



Reading

You should read following topics before starting this exercise

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



Ready Reference

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	background= designates a file to be displayed as background bgcolor= "#(hexadecimal color code)" sets the background color text= "#(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
 	To make text appear bold		This text will appear bold
<U> </U>	To make text appear underlined		<U>This text will appear underlined</U>
<I> </I>	To make text appear italic		<I>This text is both Bold and italic</I>
<CENTER> </CENTER>	Centers enclosed text		<CENTER> Text is centered </CENTER>
 	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	 How is this ?

		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.		
	Formats enclosed text as subscript.		a_{<SMALL> o} </SMALL>
	Formats enclosed text as superscript.		x^{<SMALL> 2} </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>
	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

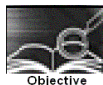
Assignment Evaluation

Signature

0: Not done <input style="width: 30px; height: 20px;" type="text"/>	2: Late Complete <input style="width: 30px; height: 20px;" type="text"/>	4: Complete <input style="width: 30px; height: 20px;" type="text"/>
1: Incomplete <input style="width: 30px; height: 20px;" type="text"/>	3: Needs improvement <input style="width: 30px; height: 20px;" type="text"/>	5: Well Done <input style="width: 30px; height: 20px;" 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
	Specify the list item.		
 	The 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 	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"> <u> List of Cities.... </u> <ol type = "A" start = "A"> Mumbai Pune Nashik Nagpur </body>
 	 tag defines the unordered list of items	Type = disc/square/circle Specifies the bullet type.	<body bgcolor= "sky blue" text ="yellow"> <i><u> List of Fruits </i></u> <ul type = "square"> Apple Pinapple Mango Guava </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> 	Add an anchor or hyperlink.	href= <i>url</i> Specifies the URL of the target page.	<BODY> Click here to visit Yahoo </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

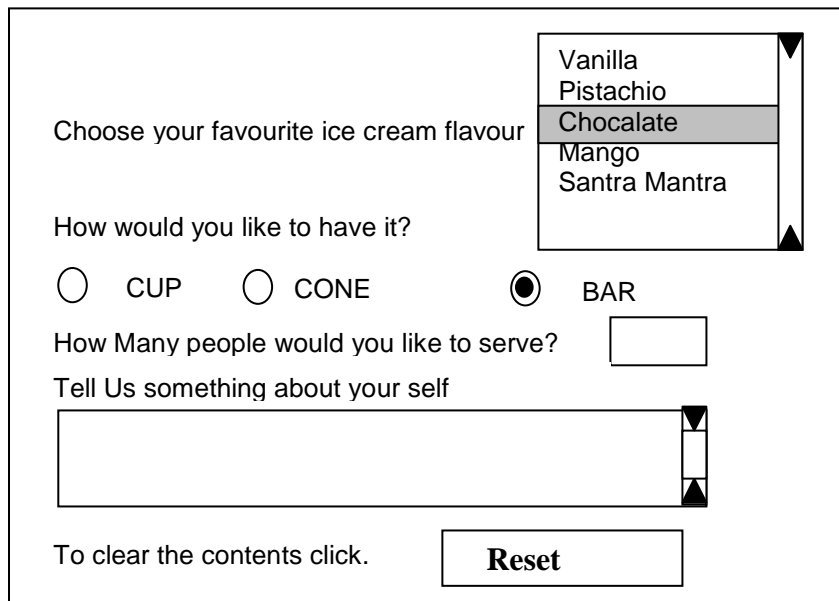
How would you like to have it?

CUP CONE BAR

How Many people would you like to serve?

Tell Us something about your self

To clear the contents click.



3.

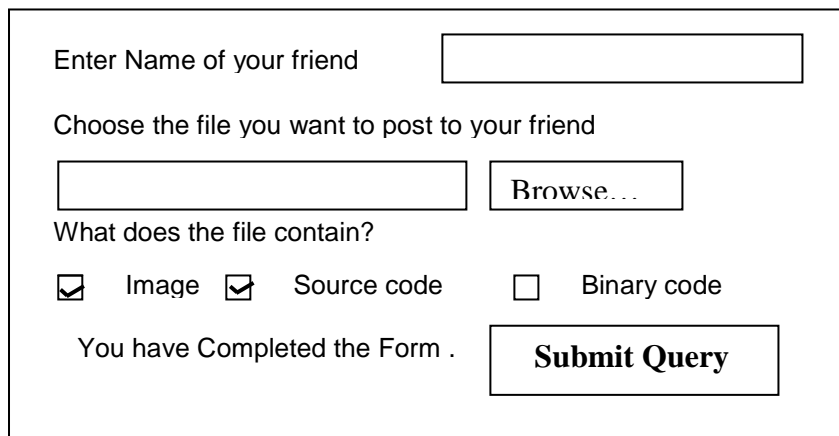
Enter Name of your friend

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));

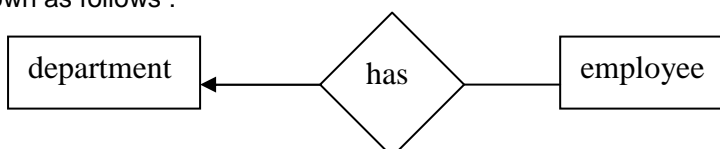
A multivalued, composite attribute	A composite attribute having multiple values , for each member of the entity set	A new relation is created, which will contain a place holder for each part of the composite attribute and the key attributes of the entity set that contains the composite multivalued attribute	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.

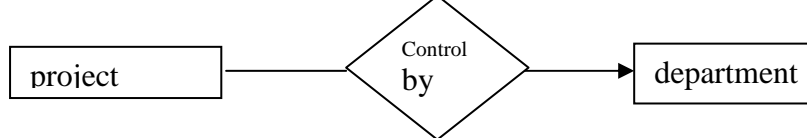
1. Consider two tables department & employee. One department can have one or more employees, but an employee belongs to exactly one department (1-m relation). It's pictorially shown as follows :



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);

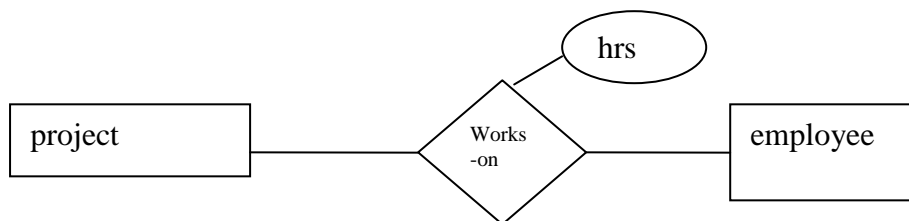
2. Consider the department table created above & another table called project. A project is controlled by exactly one department , but a department can control one or more projects(a m-1 relation). It's pictorially shown as follows :



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))

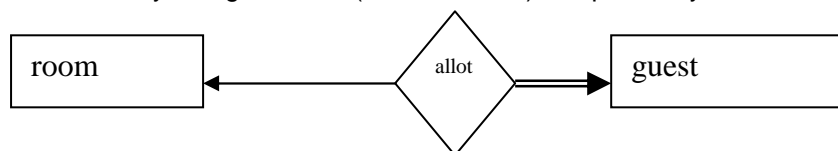
3. Consider the project & employee relations created above. An employee can work in one or more projects, and a project can have one or more employees working in it .(a m-m relation). It's shown pictorially as follows :



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 (2nd 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;

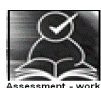


Consider the tables created in assignments 11,12,13,14. type and execute the below statements for these tables. Write the output of each statement & justify your answer

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

Set C

Create an appropriate set of tables in normalized form to keep some business information. Populate the tables with information for the business process. State the updations that can be done to the data in the table .Write and execute update / delete statements for the same. The names of tables & fields should be self-explanatory (i.e . their names should depict the kind of data being stored.)

Signature of the instructor

Date

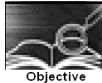
Assignment Evaluation

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 16

Start Date / /



To understand & get a Hands-on on Select statement



You should read following topics before starting this exercise

1. Creating relations as tables and inserting tuples as records into the table.
2. The use of select statement in extracting information from the relation.
3. Insert/update/delete with subquery.
4. Relationship types & their conversion to RDB.
5. Normal forms → 1NF, 2NF, 3NF.



The select statement :

Name	Description	Syntax	Example
Select statement	<p>Used to read a tuple, tuples, parts of a tuple from a relation in the database. Tuple means a record in an RDB & a relation means a table.</p> <p>The basic structure of a Select statement consists of 3 clauses :</p> <p>The select clause → corresponds to the projection operation in relational algebra. It is used to list the attributes desired in the query.</p> <p>The from clause → corresponds to the Cartesian product operation of RA. It lists the relations to be scanned in the evaluation of the expression.</p> <p>The where clause → corresponds to the selection operation of RA. It consists of a predicate involving the attributes of the relations that appear in the select clause.</p> <p>The other clauses are :</p> <p>Order by clause → causes the result of the</p>	<pre>select <attribute-list> from <relation-list> [where <condition> [group by <attribute list> [having <condition>] order by <attribute list>]];</pre>	<pre>Select * from emp; Select eno,name from emp; Select eno name from emp where sal > 4000 order by eno; Select sum(sal) from emp group by dno having sum(sal)> 100000;</pre>

	<p>query to appear in a sorted order.</p> <p>Group by clause → used to form groups of tuples, of the result. It is used when using aggregate functions.</p> <p>Having clause → Used with group by clause, to force a condition on the groups formed after applying group by clause, & selects only those groups in the output that satisfy the condition.</p> <p>The order of execution of the clauses is the same as given in the syntax.</p>		
--	--	--	--

The aggregate functions supported by SQL are as follows:

Name	Description	Usage	Example
Sum()	Gets the sum or total of the values of the specified attribute.	Sum(attribute-name)	Select sum(sal) from emp; Select dno, sum(sal) from emp group by dno;
Count()	Gives the count of members in the group	Count(attribute); Count(*)	Select count(*) from emp; Select count(*) from emp where sal > 5000;
Max()	Gives the maximum value for an attribute, from a group of members	Max(attribute)	Select max(sal) from emp; Select dno, max(sal) from emp group by dno having count(*) >10;
Min()	Gives the minimum value for an attribute, from a group of members	Min(attribute)	Select min(sal) from emp; Select dno, min(sal) from emp group by dno having min(sal) >100;
Avg()	Gives the average value for an attribute, from a group of members	Avg(attribute)	Select avg(sal) from emp; Select dno, avg(sal) from emp group by dno having count(*) >10;



Self-Activity

As part of the self activity in exercise you have created a table employee with attributes empno, name, address, salary and deptno. You have also inserted atleast 10 records into the same.

To execute each query

type each query into the database prompt or

type queries in a file and cut and copy each query at the database prompt or

type queries in a file and type \i filename at SQL prompt. (all queries in the file will get executed one by one).

Execute following select queries & write the business task performed by each query.

1. Select * from emp;
2. Select empno, name from emp;
3. Select distinct deptno from emp;
4. Select * from emp where deptno = ____;
5. Select * from emp where address = 'pune' and sal > _____;
6. Select * from emp where address = 'pune and salary between _____ and _____;
7. Select * from emp where name like '---%'
8. Select * from emp where name like '%and%'
9. Select * from emp where salary is null;
10. Select * from emp order by eno;
11. Select * from emp order by deptno, eno desc;
12. Select deptno as department, sum(salary) as total from emp group by deptno order by deptno;
13. Select deptno as department , count(eno) as total_emp from emp group by deptno having count(eno) > _____ order by deptno;
14. select avg(salary) from emp;
15. select max(salary),deptno from emp group by deptno having max(sal) > _____;
16. select deptno, min(salary) from emp order by deptno;
17. update emp set salary = salary + 0.5*salary where deptno = (select deptno from department where dname = 'finance');
18. update emp set deptno = (select deptno from department where dname = 'finance')
Where deptno = (select deptno from department where dname = 'inventory');
19. insert into emp_backup(eno,ename) values(select eno,ename from emp);
20. delete from emp where deptno = (select deptno from department where dname='production');

Signature of the instructor

Date



Set A

Consider the relations Person (pnumber, pname, birthdate, income), Area(aname,area_type). An area can have one or more person living in it , but a person belongs to exactly one area. The attribute 'area_type' can have values as either urban or rural.

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

Assume appropriate data types for all the attributes. Add any new attributes as required, depending on the queries. Insert sufficient number of records in the relations / tables with appropriate values as suggested by some of the queries.

Write select queries for following business tasks and execute them.

1. List the names of all people living in '_____' area.
2. List details of all people whose names start with the alphabet '_' & contains maximum _ alphabets in it.
3. List the names of all people whose birthday falls in the month of _____.
4. Give the count of people who are born on '_____'
5. Give the count of people whose income is below _____.
6. List names of all people whose income is between _____ and _____;

7. List the names of people with average income
8. List the sum of incomes of people living in '_____'
9. List the names of the areas having people with maximum income (duplicate areas must be omitted in the result)
10. Give the count of people in each area
11. List the details of people living in '_____ ' and having income greater than _____;
12. List the details pf people, sorted by person number
13. List the details of people, sorted by area, person name
14. List the minimum income of people.
15. Transfer all people living in 'pune' to 'mumbai'.
16. delete information of all people staying in 'urban' area

Instructor should fill in the blanks with appropriate values.

Signature of the instructor

Date

Set B

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

Table name :

Field name	Data Type	Constraints

Insert sufficient number of records in the relations / tables with appropriate values as suggested by some of the queries.

Write & execute select queries for following business tasks

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

Instructor should fill in the blanks with appropriate values.

Signature of the instructor

Date

Set C

Create an appropriate table to keep some business information. Populate the table with information for the business process. State the business tasks that you need to perform to extract information. Write and execute queries for the same. The names of tables & fields should be self-explanatory (i.e. their names should depict the kind of data being stored.)

Signature of the instructor

Date

Assignment Evaluation

Signature

0: Not done <input style="width: 30px; height: 20px;" type="text"/>	2: Late Complete <input style="width: 30px; height: 20px;" type="text"/>	4: Complete <input style="width: 30px; height: 20px;" type="text"/>
1: Incomplete <input style="width: 30px; height: 20px;" type="text"/>	3: Needs improvement <input style="width: 30px; height: 20px;" type="text"/>	5: Well Done <input style="width: 30px; height: 20px;" type="text"/>

Exercise 17

Start Date / /



To understand & get a Hands-on on using set operations (union ,intersect and except) with select statement.



You should read following topics before starting this exercise

1. Relation algebra operation \cap , \cup and $-$.
2. SQL operations union, intersect & except



SQL Set operations :

Name	description	Syntax	Example
Union	Returns the union of two sets of values, eliminating duplicates.	<select query> Union <select query>	Select cname from depositor Union Select cname from borrower;
Union all	Returns the union of two sets of values, retaining all duplicates.	<select query> Union all <select query>	Select cname from depositor Union all Select cname from borrower;
Intersect	Returns the intersection of two sets of values, eliminating duplicates	<select query> intersect <select query>	Select cname from depositor intersect Select cname from borrower;
Intersect all	Returns the intersection of two sets of values, retaining duplicates	<select query> Intersect all <select query>	Select cname from depositor Intersect all Select cname from borrower;
except	Returns the difference between two set of values, i.e returns all values from set1 , not contained in set2 .eliminates duplicates	<select query> except <select query>	Select cname from depositor except Select cname from borrower;
Except all	Returns the difference between two set of values, i.e. returns all values from set1 , not contained in set2 .Retains all duplicates	<select query> Except all <select query>	Select cname from depositor Except all Select cname from borrower;

The relations participating in the SQL operations union, intersect & except must be compatible i.e. the following two conditions must hold :

- a)The relation r and s must be of the same arity. That is , they must have the same number of attributes.

b) The domains of the i^{th} attribute of r and the i^{th} attribute of s must be the same , for all i.

Consider the following relations, non-teaching, teaching, department.

One department can have one or more teaching & non-teaching staff, but a teaching or non-teaching staff belongs to exactly one department. Hence dno is a foreign key in the both the relations. Create these relations in your database .

Non-teaching (empno int primary key, name varchar(20), address varchar(20), salary int,dno references department)

Teaching(empno int primary key, name varchar(20), address varchar(20), salary int,dno references department)

Department(dno int primary key,dname)

- insert at least 10 records into both the relations.
- type the following select queries & write the output and the business task performed by each query

1. Select empno from non-teaching union select empno from teaching;
2. Select empno from non-teaching union all select empno from teaching;
3. Select name from non-teaching intersect select name from teaching;
4. Select name from non-teaching intersect all select name from teaching;
5. Select name from non-teaching except select name from teaching;
6. Select name from non-teaching except all select name from teaching;

Signature of the instructor

Date



Set A

Create the following relations, for an investment firm
emp(emp-id ,emp-name, address, bdate)

Investor(inv-name , inv-no, inv-date, inv-amt)

An employee may invest in one or more investments, hence he can be an investor.

But an investor need not be an employee of the firm.

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

Assume appropriate data types for the attributes. Add any new attributes , as required by the queries. Insert sufficient number of records in the relations / tables with appropriate values as suggested by some of the queries.

Write the following queries & execute them.

1. List the distinct names of customers who are either employees, or investors or both.
2. List the names of customers who are either employees , or investors or both.
3. List the names of employees who are also investors.
4. List the names of employees who are not investors.

Signature of the instructor

Date

Set B

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

Table name 1:

Field name	Data Type	Constraints

Table name 2:

Field name	Data Type	Constraints

Relationship → _____

Insert sufficient number of records in the relations / tables with appropriate values as suggested by some of the queries.

Write & execute queries for following business tasks

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

Instructor should fill in the blanks with appropriate values.

Signature of the instructor

Date

Set C

Create two compatible tables having similar set of attributes, to keep some business information. Populate the tables with information for the business process. State the business tasks that you need to perform on these tables involving information from both the tables. Write and execute queries for the same. The names of tables & fields should be self-explanatory (i.e. their names should depict the kind of data being stored.)

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"/>



To understand & get a Hands-on on nested queries & subqueries, that involves joining of tables.



You should read following topics before starting this exercise

1. Nesting of SQL queries and subqueries
2. SQL statements involving set membership, set comparisons and set cardinality operations.
3. Descriptive attributes & how they are handled while creating RDB.



A subquery is a select-from-where expression that is nested within another query.

Set membership	the 'in' & 'not in' connectivity tests for set membership & absence of set membership respectively.
Set comparison	the < some, > some, <= some, >= some, = some, <> some are the constructs allowed for comparison. = some is same as the 'in' connectivity. <> some is not the same as the 'not in' connectivity. Similarly sql also provides < all, >all, <=all, >= all, <> all comparisons. <>all is same as the 'not in' construct.
Set cardinality	The 'exists' construct returns the value true if the argument subquery is nonempty. We can test for the non-existence of tuples in a subquery by using the 'not exists' construct. The 'not exists' construct can also be used to simulate the set containment operation (the super set). We can write "relation A contains relation B" as "not exists (B except A)".

The complete Syntax of select statement containing connectivity or Comparison operators is as follows

```
select <attribute-list> from <relation-list>
      where <connectivity / comparison > { sub-query };
```



Create the following relation in your database(primary keys underlined)

```
Employee(ename, street, city)
Works(ename, company-name, salary)
Company(company-name, city)
Manages(ename, manager-name )
```

An employee can work in one or more companies, a company can have one or more employees working in it. Hence the relation 'works' with key attributes as ename, company-name.

An employee manages one or more employees, but an employee is managed by exactly one employee (a recursive relationship), hence the relation 'manages' with key ename.

Insert sufficient number of records in the relations / tables with appropriate values as suggested by some of the queries.

Type the following queries , execute them and give the business task performed by each query

1. select ename from works w where salary >= all (select max(salary) from works);
2. select ename form works w where salary = (select max(salary) from works w1 where w1.company-name = w.company-name));
3. select manager-name from manages where manager-name in(select ename from works where company-name = "_____");
4. select manager-name from manages where manager-name not in(select ename from works where company-name = "_____");

5. select ename from works w where salary > some (select salary from works where company-name not in (select company-name from company where city = "_____"));
6. select ename from employee e where city = (select city from employee e1 , manages m where m.ename = e.ename and m.manager-name = e1.ename);
7. select * from employee where ename in (select manager-name from manages)
8. select city count(*) from employee group by city having count(*) >= all (select count(*) from employee group by city)
9. select ename from works w where salary <> all (select salary from works where ename <> w.ename);
10. select company-name, sum(salary) from works w group by company-name having sum(sal) >= all (select sum(sal) from works group by company-name)
11. select ename from employee e where city in('_____' , '_____');
12. select ename from employee e where city = (select city from company c, works w where w.ename = e.name and c.company-name = w.company-name);

Instructor should fill in the blanks with appropriate values.

Signature of the instructor

Date



Set A

Create the following relations :

Emp(eno,name,dno,salary)

Project(pno,pname,control-dno,budget)

Each employee can work on one or more projects, and a project can have many employees working in it. The number of hours worked on each project , by an employee also needs to be stored.

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

Assume appropriate data types for the attributes. Add any new attributes , new relations as required by the queries.

Insert sufficient number of records in the relations / tables with appropriate values as suggested by some of the queries.

Write the queries for following business tasks & execute them.

1. list the names of departments that controls projects whose budget is greater than ____.
2. list the names of projects, controlled by department No __, whose budget is greater than atleast one project controlled by department No ____.
3. list the details of the projects with second maximum budget
4. list the details of the projects with third maximum budget.
5. list the names of employees, working on some projects that employee number __ is working.
6. list the names of employees who do not work on any project that employee number __ works on
7. list the names of employees who do not work on any project controlled by '_____' department
8. list the names of projects along with the controlling department name, for those projects which has atleast __ employees working on it.
9. list the names of employees who is worked for more than 10 hrs on atleast one project controlled by '_____' dept.
10. list the names of employees , who are males , and earning the maximum salary in their department.
11. list the names of employees who work in the same department as '_____ '.
12. list the names of employees who do not live in _____ or _____.

Instructor should fill in the blanks with appropriate values.

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

Table name :

Field name	Data Type	Constraints

Relationship → _____

Insert sufficient number of records in the relations / tables with appropriate values as suggested by some of the queries.

Write & execute queries for following business tasks. (business tasks should be using set operations & should be similar to the ones given in set A)

- a)
- b)
- c)
- d)

Instructor should fill in the blanks with appropriate values.

Signature of the instructor

Date

Set C

Create an appropriate set of tables to keep some business information. Populate the tables with information for the business process. State the business tasks that involve set of operations that you need to perform to extract information. Write and execute queries for the same. The names of tables & fields should be self-explanatory (i.e. their names should depict the kind of data being stored.)

Signature of the instructor

Date

Assignment Evaluation

Signature

0: Not done <input style="width: 30px; height: 20px;" type="text"/>	2: Late Complete <input style="width: 30px; height: 20px;" type="text"/>	4: Complete <input style="width: 30px; height: 20px;" type="text"/>
1: Incomplete <input style="width: 30px; height: 20px;" type="text"/>	3: Needs improvement <input style="width: 30px; height: 20px;" type="text"/>	5: Well Done <input style="width: 30px; height: 20px;" type="text"/>

Exercise 19

Start Date / /



To understand & get a Hands-on on nested queries & subqueries, that involves joining of tables, to demonstrate set cardinality.



You should read following topics before starting this exercise

1. Nesting of SQL queries and subqueries
2. SQL statements involving set membership, set comparisons and set cardinality operations.



SQL includes a feature for testing whether a subquery has any tuples in its result, using the following clauses .:

Name	Description	Syntax	Example
Exists	The 'exists' construct returns the value true if the argument subquery is nonempty	select <attribute-list> from <relation-list> where <exists> { sub-query} ;	Select cname from borrower b where exists(select * from depositor where dname = b.cname);
Not exists	We can test for the non-existence of tuples in a subquery by using the 'not exists' construct. The 'not exists' construct can also be used to simulate the set containment operation (the super set). We can write "relation A contains relation B" as "not exists (B except A)"	select <attribute-list> from <relation-list> where <not exists> { sub-query};	Select cname from borrower b where not exists(select * from depositor where dname = b.cname);



Consider the table you have prepared as part of self activity of exercise 18, Type the following queries , execute them and give the business task performed by each query

1. Select company-name from company c where not exists (select city from company where company-name = "_____" except (select city from company where company-name = c.company-name));
2. Select ename from employee e where exists (select manager-name from manages where manager-name = e.ename group by manager-name having count(*) >3);
3. Select company-name from company c where not exists (select city from company where company-name = c.company-name except (select city from company where company-name = "_____"));
4. Select ename from employee e where exists (select city from employee where city = e.city and ename <> e.ename group by city having count(*) > 5)

5. Select company-name from company c where not exists (select company-name from company where city = c.city and company-name <> c.company-name)

Instructor should fill in the blanks with appropriate values.

Signature of the instructor

Date



Set A

Consider the table you have prepared as part of Assessment work set A of exercise 18, Type the following queries, execute them and give the business task performed by each query

1. List the names of employees who work in all the projects that “_____” works on.
2. List the names of employees who work on only some projects that “_____” works on
3. List the names of the departments that have atleast one project under them .(write using ‘exists ‘ clause)
4. List the names of employees who do not work on “sales” project (write using ‘not exists’) clause
5. List the names of employees who work only on those projects that are controlled by their department .
6. List the names of employees who do not work on any projects that are controlled by their department

Instructor should fill in the blanks with appropriate values.

Signature of the instructor

Date

Set B

1. Design 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 → _____

Insert sufficient number of records in the relations / tables with appropriate values as suggested by some of the queries.

Write & execute queries for following business tasks. (business tasks should be using set cardinality operations & should be similar to the ones given in set A)

- a)
- b)

- c)
- d)
- e)
- f)

Instructor should fill in the blanks with appropriate values.

Signature of the instructor

Date

Set C

Create an appropriate set of tables to keep some business information. Populate the tables with information for the business process. State the business tasks that involve set cardinality operations that you need to perform to extract information. Write and execute queries for the same. The names of tables & fields should be self-explanatory (i.e. their names should depict the kind of data being stored.)

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 20

Start Date

/ /



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)



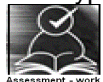
You should read following topics before starting this exercise

1. All the assignments from 11 to 19



Steps in solving a case study:

1. Read through the given case study carefully.
 2. Create the given relations in the database. The database thus created should be in 3NF. (no data duplication, appropriate handling of the relationships)
 3. Insert sufficient number of records in the relations / tables.
 4. Create a new file with all the select queries in it.
1. To execute each query
Cut & copy each query into the database prompt OR
Type \i filename at SQL prompt. (all queries in the file will get executed , one by one.



- 1 Consider the following case study :

A 4-wheeler rental company needs to develop a database to store the following information : the information about the cars , like the registration number, the chassis number, the type of the vehicle (car, jeep, SUV etc). the vehicles may have one or more luxurious features like AC, Stereo, tape, DVD player etc).

The company also needs to maintain the information about its drivers like driver licence no, name, address , age etc.

A car is driven by different drivers on different days , a driver may drive different cars on different days . The company also needs information regarding the different places to which the car had been driven down, the names of drivers who have driven it to these places alongwith the name of customers who had booked the car to that place. The information of the different destinations to which the cars from this company can be driven down, also needs to be stored.

Regarding customers, a customer can book more than one car to a place. The customers are allowed to book multiple cars to different places , in a single booking transaction. The name, address , no of passengers travelling in the car, the destination ,the rental cost etc needs to be stored.

The following constraints are to be defined for the vehicles, drivers, destination places :

- a) the vehicle make should be after the year 2000.
- b) only vehicles of maruti, Tata are used by the company
- c) drivers should be above 20 years of age
- d) drivers should be staying in "pune" city
- e) the destination places should be within 500km radius from Pune.

Design the relational database for the above company , so that the following queries can be answered :

1. List the names of drivers who have driven a car to "Mumbai"
2. List the name of customers who have booked a "SUV" to "satara"
3. List the names of customers who have booked cars to pune or Mumbai or Lonavla
4. List the details of cars that have never driven down to "Mumbai"
5. List the details of the place to which maximum number of customers have driven down.
6. List the details of the driver who have driven all the vehicles of the company.

7. List the names of the drivers who have driven atleast two cars to "Mumbai"
8. List the names of drivers who have also driven some vehicles to "Mumbai"
9. List the details of customers who have booked more than two vehicles to "solapur"
10. List the names of customers who have booked maximum number of vehicles

2.Consider the following case study

An insurance agent sells policies to clients. Each policy is of a particular type like vehicle insurance, life insurance, accident insurance etc, and there can be many policies of a particular type. Each policy will have many monthly premiums, and each premium is associated to only one policy. Assume appropriate attributes for agents, policy , premiums, policy-types.

The following constraints have to be defined on the relations

- a. The policy types can be only accident, life , vehicle.
- b. The agents can be only from pune, Mumbai, Chennai
- c. The policy amount should be greater than 20000
- d. The policy-sale-date should be greater than the policy-intro-date.

Design the relational database for the above , so that the following queries can be answered:

1. List the names of agents living in ' _____ '
2. List the names of policy holders , who have bought policies from the agent 'joshi'
3. List the names of policyholders, who have bought more than two policies from 'joshi'
4. List the names of agents , who have sold policies to only customers who live in their own City.
5. List the names of agents who have sold atleast two policies.
6. List the names of cities, which has the maximum number of agents.
7. List the names of customers who have bought the maximum number of policies.
8. List the details of all premiums , paid f
or the policy number _____
9. Update all policy amount to _____ , for all policies bought by customers from _____ city.
10. Delete all policies , bought from 'joshi'
11. _____
12. _____
13. _____
14. _____
15. _____

Instructor should fill in the blanks with appropriate values.

3.Consider the following case study

A movie studio wants to develop a database to manage their office information , related to movies, actors, directors, producers.

The following facts are relevant

- a. Each actor has acted in one or more movies
- b. Each director has directed many movies.
- c. Each producer has produced many movies.
- d Each movie is directed by one and only one director, but can be produced by more than one producers.
- e. Each movie has one or more actors acting in it, in different roles.
- f. Each actor & director has several addresses. Each address is made up of a house-no, street, city, state.

The following constraints are defined on the relations.

- a. Each movie can have a maximum budget of 10 lakhs
- b. Each actor can chare a maximum of Rs. 10 lakhs for a movie.
- c. The roles that an actor can act in a movie can be any of the following : villan, hero, heroine, support.

Design the relational database for the above , so that the following queries can be answered:

1. List the names of movies in which _____ has acted.
2. List the names of actors who have acted in at least one movie, in which shahrukh has acted.
3. List the names of actors who have acted in every movie in which _____ has acted.

4. List the names of actors who have acted as a 'villan' in every movie, in which the actor has acted
5. List the names of movies with the highest budget
6. List the names of movies with the second highest budget
7. List the names of actors who have acted in the maximum number of movies.
8. _____
9. _____
10. _____
11. Update the address of producer _____. set the city to _____
12. Delete information of all actors who have an address in pune.
13. _____
14. _____
- 15 List the names of movies , produced by more than one producer.

Instructor should fill in the blanks with appropriate values.

4. Consider the following case study :

A music company wants to go in for automation of their requirements. They want to develop a database for maintaining the information of their music albums, singers, musicians, instruments.

The following facts are relevant :

- a. each album is produced by many musicians, a musician can produce many albums
- b. a singer can sing for many albums, but an album consists of songs of only one singer.
- c. a musician can play many instruments, an instrument can be played by many musicians.

The following constraints are to be placed on the relations

- a. each musician is paid a minimum of 50000 Rs. for each album
- b. all singers are from either pune, Mumbai or Chennai
- c. each instrument cost is maximum 10000

Design the relational database for the above , so that the following queries can be answered:

1. _____
2. List the names of musicians who have played guitar for the album _____
3. list the names of musicians who palsy at least one instrument same as the one "joshi" plays.
4. List the names of albums , in which " _____ " has sung.
5. _____
6. _____
7. List the names of albums released in 1998
8. List the names of albums that have more than two instruments being played in it
9. Delete all information of singers who have not sung in any album
10. Delete all information of musicians , who have worked in the album " _____ "

Instructor should fill in the blanks with appropriate values.

5 Consider the following case study

A housing society needs to manage the administrative information related to the society.

The society is made up of different types of flats like 2BHK, 1BHK, 3BHK. Each type has a well defined square-feet area . The outright sale rate & the rental value of the flat depends on the type of the flat. Each flat has a single owner. Each owner can have one or more flats in his name. The name, address , phone etc of the owner need to be maintained. For each flat, its type, the floor no, any internal specifications needs to be maintained.

The society also contains a club-house, which is rented out to flat owners , at a nominal rate for conducting various functions / programmes. Society would like to print reports like number of functions held in the club-house during a month / period etc.

Every month maintenance amount is collected from the owners of the flats. Society needs to maintain this finance information, like how much amount collected for a month, whether any defaulters for a month, sending reminders to the defaulters etc.

The expenditure information includes money spent on maintenance of the society like paying the sweepers, cleaners of the common area of the society, any emergency expense, salaries of the security etc.

Every month the society would like to print a report of expenditure versus collection.

Design the relational database for the above , so that the following queries can be answered:

1. List the flats of 2bhk type.
2. List the 3bhk flats that are currently vacant.
3. List the functions held in clubhouse during the month of “ _____ ”
4. List the names of owners , who have never conducted any functions in the clubhouse.
5. List the payment defaulters for the month of “april”
6. List the total expenditure for the month of _____
7. List the month with the least expenditure.
8. Transfer the flat in the name of _____ to _____
9. _____
10. List the names of owners , who own both a 2bhk and a _____
11. _____
12. _____

Instructor should fill in the blanks with appropriate values.

NB : More small case studies can be designed by the instructors , so that there can be maximum variation in work assigned to each student in a batch.

The case studies must cover almost all types of entities, attributes & relationships.

The queries on the case studies , must be similar to the ones done in assignments 15 to 19.

Signature of the instructor

Date

Assignment Evaluation

Signature

0: Not done <input style="width: 40px; height: 20px;" type="text"/>	2: Late Complete <input style="width: 40px; height: 20px;" type="text"/>	4: Complete <input style="width: 40px; height: 20px;" type="text"/>
1: Incomplete <input style="width: 40px; height: 20px;" type="text"/>	3: Needs improvement <input style="width: 40px; height: 20px;" type="text"/>	5: Well Done <input style="width: 40px; height: 20px;" type="text"/>

Lab Course I

Exercise 1

Start Date

/ /



To demonstrate the use of data types, simple operators and expressions



You should read following topics before starting this exercise

1. Different basic data types in C and rules of declaring variables in C
2. Different operators and operator symbols in C
3. How to construct expressions in C, operator precedence
4. Problem solving steps- writing algorithms and flowcharts



1. Data type Table

Data	Data Format	C Data Type	C Variable declaration	Input Statement	Output statement
quantity month credit- card number	Numeric	int Short int long int	int quantity; short month; long ccno;	scanf("%d",&quantity); scanf("%d",&month); scanf("%ld", &ccno);	printf("The quantity is %d", quantity); printf("The credit card number is %ld, ccno);
price π	real	float double	float price; const double pi=3.141593;	scanf("%f",&price);	printf("The price is %5.2f", price);
grade	character		char grade;	scanf("%c",&grade)	printf("The grade is %c",grade);

2. Expression Examples

Expression	C expression
Increment by a 3	a = a + 3
Decrement b by 1	b = b-1 or b--
$2a^2 + 5b/2$	2*a*a + 5*b/2
$7/13(x-5)$	(float)7/13*(x-5)
5% of 56	(float)5/100*56
n is between 12 to 70	n>=12 && n<=70
$\pi^2 h$	Pi*r*r*h
n is not divisible by 7	n % 7 != 0
n is even	n%2== 0
ch is an alphabet	ch>='A' && ch<='Z' ch>='a' && ch<='z'

Note: The operators in the above expressions will be executed according to precedence and associativity rules of operators.

3. Sample program- to calculate and print simple interest after accepting principal sum, number of years and rate of interest.

Program development steps

Step 1 : Writing the Algorithm	Step 2 : Draw the flowchart	Step 3 : Writing Program
<ol style="list-style-type: none"> 1. Start 2. Accept principal sum, rate of interest and number of years 3. Compute Simple interest 4. Output Simple Interest 5. Stop 	<pre> graph TD Start([start]) --> Read[/Read ,principal sum, rate and no of years/] Read --> Compute[Compute Simple interest] Compute --> Print[/Print Simple Interest/] Print --> Stop([stop]) </pre>	<pre> /* Program to calculate simple interest */ #include <stdio.h> main() { /* variable declarations */ float amount, rateOfInterest, simpleInterest; int noOfYears; /* prompting and accepting input */ printf("Give the Principal Sum"); scanf("%f",&amount); printf("Give the Rate of Interest"); scanf("%f",&rateOfInterest); printf("Give the Number of years"); scanf("%d",&noOfYears); /* Compute the simple Interest*/ simpleInterest=amount*noOfYears*rateOfInterest / 100; /* Print the result*/ printf("The simple Interest on amount %7.2f for %d years at the rate %4.2f is %6.2f", amount, noOfYears, rateOfInterest, simpleInterest); } </pre>



1. Type the sample program given above. Execute it for the different values as given below and fill the last column from the output given by the program. Follow the following guidelines
 - a. At \$ prompt type vi followed by filename. The filename should have .c as extension for example
\$vi pnr.c
 - b. Type the sample program given above using vi commands and save it
Compile the program using cc compiler available in Linux
\$cc pnr.c
It will give errors if any or it will give back the \$ prompt if there are no errors
A executable file a.out is created by the compiler in current directory. The program can be executed by typing name of the file as follows giving the path.
\$./a.out
Alternatively the executable file can be given name by using -o option while compiling as follows
\$cc pnr.c -o pnrexec
\$./pnrexec
The executable file by specified name will be created. Note that you have to specify the path of pnrexec as ./pnrexec , i. e., pnrexec in current (. Stands for current directory) directory otherwise it looks for program by that name in the path specified for executable programs

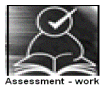
Sr. No	Principal sum	No of years	Rate of interest	Simple Interest
1	2000	3	_____	
2	4500	_____	4.5	
3	_____	6	8.3	

2. If you have not typed the program correctly, i.e., if there are syntactical errors in the program, compiler will pinpoint the errors committed and are called compile-time errors. C compiler gives line no along with error messages when it detects grammatical or syntactical errors in the program. These messages are not so straightforward and you may find it difficult to identify the error. You may miss a semicolon at the end of a statement and the compiler points out error in the next statement. You may miss just a closing ‘*/’ of a comment and it will show errors in several statements following it.
- Another type of error which is quite common is the run-time or execution error. You are able to compile the program successfully but you get run-time messages or garbage output when you execute the program.
- Modify the above program to introduce the following changes, compile, write the error messages along with line numbers, remove the error execute and indicate the type of error whether it was compile-time or execution time error.

Modified line	Error messages and line numbers	Type of error
/* Program to calculate simple interest		
int noofYears;		
scanf("%f",&amount)		
scanf("%f", amount);		
scanf("%d", noOfYears);		

Signature of the instructor

Date / /



Set A . Apply all the three program development steps for the following examples.

1. Accept dimensions of a cylinder and print the surface area and volume (Hint: surface area = $2\pi r^2 + 2\pi rh$, volume = $\pi r^2 h$)
2. Accept temperatures in Fahrenheit (F) and print it in Celsius(C) and Kelvin (K) (Hint: $C=5/9(F-32)$, $K = C + 273.15$)
3. Accept initial velocity (u), acceleration (a) and time (t). Print the final velocity (v) and the distance (s) travelled. (Hint: $v = u + at$, $s = u + at^2$)
4. Accept inner and outer radius of a ring and print the perimeter and area of the ring (Hint: perimeter = $2\pi(a+b)$, area = $\pi(a^2-b^2)$)
5. Accept two numbers and print arithmetic and harmonic mean of the two numbers (Hint: AM = $(a+b)/2$, HM = $ab/(a+b)$)
6. Accept three dimensions length (l), breadth(b) and height(h) of a cuboid and print surface area and volume (Hint : surface area= $2(lb+lh+bh)$, volume = lbh)
7. Accept a character from the keyboard and display its previous and next character in order. Ex. If the character entered is ‘d’, display “The previous character is c”, “The next character is e”.
8. Accept a character from the user and display its ASCII value.

Signature of the instructor

Date / /

Set B . Apply all the three program development steps for the following examples.

1. Accept the x and y coordinates of two points and compute the distance between the two points.
2. Accept two integers from the user and interchange them. Display the interchanged numbers.
3. A cashier has currency notes of denomination 1, 5 and 10. Accept the amount to be withdrawn from the user and print the total number of currency notes of each denomination the cashier will have to give.

Signature of the instructor

Date

Set C. Write a program to solve the following problems

1. Consider a room having one door and two windows both of the same size. Accept dimensions of the room, door and window. Print the area to be painted (interior walls) and area to be whitewashed (roof).
2. The basic salary of an employee is decided at the time of employment, which may be different for different employees. Apart from basic, employee gets 10% of basic as house rent, 30% of basic as dearness allowance. A professional tax of 5% of basic is deducted from salary. Accept the employee id and basic salary for an employee and output the take home salary of the employee.

Signature of the instructor

Date

Assignment Evaluation

Signature

0: Not done <input style="width: 40px; height: 20px;" type="text"/>	2: Late Complete <input style="width: 40px; height: 20px;" type="text"/>	4: Complete <input style="width: 40px; height: 20px;" type="text"/>
1: Incomplete <input style="width: 40px; height: 20px;" type="text"/>	3: Needs improvement <input style="width: 40px; height: 20px;" type="text"/>	5: Well Done <input style="width: 40px; height: 20px;" type="text"/>

Exercise 2

Start Date / /



To demonstrate use of decision making statements such as if and if-else.



You should read following topics before starting this exercise

1. Different types of decision-making statements available in C.
2. Syntax for these statements.



During problem solving, we come across situations when we have to choose one of the alternative paths depending upon the result of some condition. Condition is an expression evaluating to true or false. This is known as the Branching or decision-making statement. Several forms of If and else constructs are used in C to support decision-making.

- 1) if statements
- 2) if – else
- 3) Nested if

Note: If there are more than one statement in the if or else part, they have to be enclosed in { } braces

Sr. No	Statement Syntax	Flowchart	Example
1.	<p>if statement</p> <pre>if (condition) { statement; }</pre>		<pre>if(n > 0) printf("Number is positive");</pre>
2.	<p>if - else statement</p> <pre>if (condition) { statement; } else { statement; }</pre>		<pre>if(n % 2 == 0) printf("Even"); else printf("Odd");</pre>

3.	<p>Nested if</p> <pre> if (condition) { if (condition) { statement;} else { statement;} } else { if (condition) { statement; } else { statement; } } </pre>	<pre> graph TD Start(()) --> D1{a >= b} D1 -- True --> AMax[a is max] D1 -- False --> D2{b >= c} D2 -- True --> BMax[b is max] D2 -- False --> D3{c >= a} D3 -- True --> CMax[c is max] D3 -- False --> AMax AMax --> End(()) BMax --> End CMax --> End </pre>	<pre> If (a >= b) { if (a >= c) printf(" %d is maximum",a); else printf(" %d is maximum",c); } else { if (b >= c) printf(" %d is maximum",b); else printf(" %d is maximum",c); } } </pre>
----	--	---	---

4. Sample program- to check whether a number is within range.

Step 1: Writing the Algorithm	Step 2 : Draw the flowchart	Step 3 : Writing Program
<ol style="list-style-type: none"> 1. Start 2. Accept the number 3. Check if number is within range 4. if true print "Number is within range" otherwise print "number is out of range". 5. Stop 	<pre> graph TD Start((start)) --> Read[/Read number/] Read --> D{If(n in range)} D -- True --> W[/Number is within range/] D -- False --> O[/Number is out of range/] W --> Stop((stop)) O --> Stop </pre>	<pre> /* Program to check range */ #include <stdio.h> main () { /* variable declarations */ int n; int llimit=50, ulimit = 100; /* prompting and accepting input */ printf("Enter the number"); scanf("%d",&n); if(n>=llimit && n <= ulimit) printf("Number is within range"); else printf("Number is out of range"); } </pre>



1. Execute the following program for five different values and fill in the adjoining table

<pre>main() { int n; printf("Enter no."); scanf("%d",&n); if(n%___==0) printf("divisible"); else printf("not divisible"); }</pre>	n	output

2. Type the above sample program 4 and execute it for the following values.

n	Output message
50	
100	
65	

3. Using the sample code 3 above write the complete program to find the maximum of three numbers and execute it for different set of values.

Instructor should fill in the blanks with appropriate values.

Signature of the instructor

Date



Set A: Apply all the three program development steps for the following examples.

- Write a program to accept an integer and check if it is even or odd.
- Write a program to accept three numbers and check whether the first is between the other two numbers. Ex: Input 20 10 30. Output: 20 is between 10 and 30
- Accept a character as input and check whether the character is a digit. (Check if it is in the range '0' to '9' both inclusive)
- Write a program to accept a number and check if it is divisible by 5 and 7.
- Write a program, which accepts annual basic salary of an employee and calculates and displays the Income tax as per the following rules.

Basic: < 1,50,000	Tax = 0
1,50,000 to 3,00,000	Tax = 20%
> 3,00,000	Tax = 30%
- Accept a lowercase character from the user and check whether the character is a vowel or consonant. (Hint: a,e,i,o,u are vowels)

Signature of the instructor

Date

Set B: Apply all the three program development steps for the following examples.

1. Write a program to check whether given character is a digit or a character in lowercase or uppercase alphabet. (Hint ASCII value of digit is between 48 to 58 and Lowercase characters have ASCII values in the range of 97 to 122, uppercase is between 65 and 90)
2. Accept the time as hour, minute and seconds and check whether the time is valid. (Hint: $0 \leq \text{hour} < 24$, $0 \leq \text{minute} < 60$, $0 \leq \text{second} < 60$)
3. Accept any year as input through the keyboard. Write a program to check whether the year is a leap year or not. (Hint leap year is divisible by 4 and not by 100 or divisible by 400)
4. Accept three sides of triangle as input, and print whether the triangle is valid or not. (Hint: The triangle is valid if the sum of each of the two sides is greater than the third side).
5. Accept the x and y coordinate of a point and find the quadrant in which the point lies.
6. Write a program to calculate the roots of a quadratic equation. Consider all possible cases.
7. Accept the cost price and selling price from the keyboard. Find out if the seller has made a profit or loss and display how much profit or loss has been made.

Signature of the instructor

Date

Set C: Write programs to solve the following problems

1. Write a program to accept marks for three subjects and find the total marks secured, average and also display the class obtained. (Class I – above __%, class II – __% to __%, pass class – __% to __% and fail otherwise)
2. Write a program to accept quantity and rate for three items, compute the total sales amount, Also compute and print the discount as follows: (amount > ____ – 20% discount, amount between ____ to ____ -- 15% discount, amount between – ____ to ____ -- 8 % discount)
3. A library charges a fine for every book returned late. Accept the number of days the member is late, compute and print the fine as follows:(less than five days Rs. ____ fine, for 6 to 10 days Rs. ____ fine and above 10 days Rs. ____ fine)

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 3

Start Date / /



To demonstrate decision making statements (switch case)



You should read following topics before starting this exercise

1. Different types of decision-making statements available in C.
2. Syntax for switch case statements.



The control statement that allows us to make a decision from the number of choices is called a switch-case statement. It is a multi-way decision making statement.

1. Usage of switch statement

Statement Syntax	Flowchart	Example
<pre> switch(expression) { case value1: block1; break; case value2: block2; break; . . . default : default block; break; } </pre>	<pre> graph TD Start([start]) --> Case1{case 1} Case1 -- True --> Block1[Block 1] Case1 -- False --> Case2{case 2} Case2 -- True --> Block2[Block 2] Case2 -- False --> Case3{case 3} Case3 -- True --> Block3[Block 3] Case3 -- False --> Case4{case 4} Case4 -- True --> Block4[Block 4] Case4 -- False --> Default[Default Block] Block1 --> Stop([stop]) Block2 --> Stop Block3 --> Stop Block4 --> Stop Default --> Stop </pre>	<pre> switch (color) { case 'r' : case 'R' : printf ("RED"); break; case 'g' : case 'G' : printf ("GREEN"); break; case 'b' : case 'B' : printf ("BLUE"); break; default : printf ("INVALID COLOR"); } </pre>

2. The switch statement is used in writing menu driven programs where a menu displays several options and the user gives the choice by typing a character or number. A Sample program to display the selected option from a menu is given below.

Step 1: Writing the Algorithm	Step 2: Draw the flowchart	Step 3: Writing Program
1. Start 2. Display the menu options 3. Read choice 4. Execute statement block depending on choice 5. Stop		<pre> /* Program using switch case and menu */ #include <stdio.h> main() { /* variable declarations */ int choice; /* Displaying the Menu */ printf("\n 1. Option 1\n"); printf(" 2. Option 2\n"); printf(" 3. Option 3\n"); printf("Enter your choice"); scanf("%d",&choice); switch(choice) { case 1: printf("Option 1 is selected"); break; case 2: printf("Option 2 is selected"); break; case 3: printf("Option 3 is selected"); break; default: printf("Invalid choice"); } } </pre>



1. Write the program that accepts a char-type variable called color and displays appropriate message using the sample code 1 above. Execute the program for various character values and fill in the following table. Modify the program to include all rainbow colours

Input character	Output Message
V	
I	
B	
G	
R	

Signature of the instructor

Date / /



Set A: Apply all the three program development steps for the following examples.

1. Accept a single digit from the user and display it in words. For example, if digit entered is 9, display Nine.
2. Write a program, which accepts two integers and an operator as a character (+ - * /), performs the corresponding operation and displays the result.
3. Accept two numbers in variables x and y from the user and perform the following operations

Options	Actions
1. Equality	Check if x is equal to y
2. Less Than	Check if x is less than y
3. Quotient and Remainder	Divide x by y and display the quotient and remainder
4. Range	Accept a number and check if it lies between x and y (both inclusive)
5. Swap	Interchange x and y

Signature of the instructor

Date / /

Set B: Apply all the three program development steps for the following examples.

1. Accept radius from the user and write a program having menu with the following options and corresponding actions

Options	Actions
1. Area of Circle	Compute area of circle and print
2. Circumference of Circle	Compute Circumference of circle and print
3. Volume of Sphere	Compute Volume of Sphere and print

2. Write a program having a menu with the following options and corresponding actions

Options	Actions
1. Area of square	Accept length ,Compute area of square and print
2. Area of Rectangle	Accept length and breadth, Compute area of rectangle and print
3. Area of triangle	Accept base and height , Compute area of triangle and print

Signature of the instructor

Date / /

Set C: Write a program to solve the following problems

1. Accept the three positive integers for date from the user (day, month and year) and check whether the date is valid or invalid. Run your program for the following dates and fill the table. (Hint: For valid date $1 \leq \text{month} \leq 12, 1 \leq \text{day} \leq \text{no-of-days}$ where no-of-days is 30 in case of months 4, 6, 9 and 11. 31 in case of months 1, 3, 5, 7, 8, 10 and 12. In case of month 2 no-of-days is 28 or 29 depending on year is leap or not)

Date	Output
12-10-1984	
32-10-1920	
10-13-1984	
29-2-1984	
29-2-2003	
29-2-1900	

2. Write a program having menu that has three options - add, subtract or multiply two fractions. The two fractions and the options are taken as input and the result is displayed as output. Each fraction is read as two integers, numerator and denominator.

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 4

Start Date



To demonstrate use of simple loops.



You should read following topics before starting this exercise

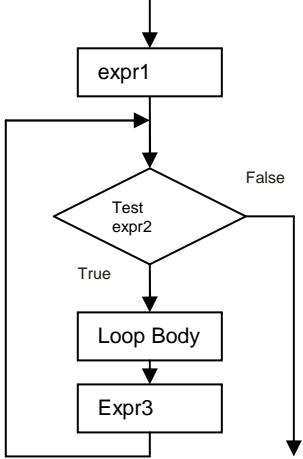
1. Different types of loop structures in C.
2. Syntax and usage of these statements.



We need to perform certain actions repeatedly for a fixed number of times or till some condition holds true. These repetitive operations are done using loop control statements. The types of loop structures supported in C are

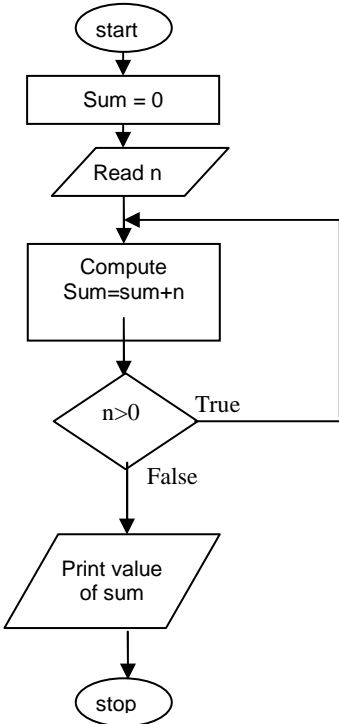
1. while statement
2. do-while statement
3. for statement

Sr. No	Statement Syntax	Flowchart	Example
1.	<p>while statement</p> <pre>while (condition) { statement 1; statement 2; . . }</pre>		<pre>/* accept a number*/ scanf("%d", &n); /* if not a single digit */ while (n > 9) { /* remove last digit n = n /10; }</pre>
2.	<p>do-while statement</p> <pre>do { statement 1; statement 2; . . } while (condition);</pre>		<pre>/*initialize sum*/ sum =0; do { /* Get a number */ printf(" give number"); scanf("%d",&n); /* add number to sum*/ sum=sum+n; } while (n>0); printf ("sum is %d", sum);</pre>

3.	for statement for(expr1; expr2; expr3) { statement 1 . . } expr1 = initialization expression expr2 = loop condition expr3 = alteration expression which alters the loop variable	 <pre> graph TD Start(()) --> Expr1[expr1] Expr1 --> Test{Test expr2} Test -- True --> LoopBody[Loop Body] LoopBody --> Expr3[Expr3] Expr3 --> Test Test -- False --> Exit(()) </pre>	/* display first 10 multiples of 2 */ for(i=1; i <= 10; i++) { printf ("2 X %d = %d\n", i, 2*i); }
----	---	--	---

Note: Usually the for loop is used when the statements have to be executed for a fixed number of times. The while loop is used when the statements have to be executed as long as some condition is true and the do-while loop is used when we want to execute statements at least once (example: menu driven programs)

3. Sample program- to print sum of 1+2+3+.....n.

Step 1: Writing the Algorithm	Step 2: Draw the flowchart	Step 3: Writing Program
1. Start 2. Initialize sum to 0. 3. Accept n. 4. Compute sum=sum+n 5. Decrement n by 1 6. if n > 0 go to step 4 7. Display value of sum. 8. Stop	 <pre> graph TD Start([start]) --> Sum0[Sum = 0] Sum0 --> ReadN[/Read n/] ReadN --> Compute[Compute Sum=sum+n] Compute --> Ngt0{n>0} Ngt0 -- True --> Compute Ngt0 -- False --> Print[/Print value of sum/] Print --> Stop([stop]) </pre>	/* Program to calculate sum of numbers */ #include <stdio.h> main() { /* variable declarations */ int sum = 0, n; printf("enter the value of n : "); scanf("%d",&n); while (n>0) { sum = sum + n; n--; } printf("\n The sum of numbers is %d", sum); }

4. Sample program- To read characters till EOF (Ctrl+Z) and count the total number of characters entered.

Step 1 : Writing the Algorithm	Step 2 : Draw the flowchart	Step 3 : Writing Program
<ol style="list-style-type: none"> 1. Start 2. Initialize count to 0. 3. Accept ch. 4. If ch !=EOF Count = count +1 Else Go to step 6 5. Go to step 3 7. Display value of sum. 8. Stop 	<pre> graph TD Start([start]) --> Init[count = 0] Init --> Read[/Read ch/] Read --> Decision{Ch=EOF} Decision -- True --> Increment[Count = count+1] Increment --> Read Decision -- False --> Print[/Print count/] Print --> Stop([stop]) </pre>	<pre> /* Program to count number of characters */ #include <stdio.h> main() { char ch; int count=0; while((ch=getchar())!=EOF) count++; printf("Total characters = %d", count); } </pre>



1. Write a program that accepts a number and prints its first digit. Refer sample code 1 given above. Execute the program for different values.
2. Write a program that accepts numbers continuously as long as the number is positive and prints the sum of the numbers read. Refer sample code 2 given above. Execute the program for different values.
3. Write a program to accept n and display its multiplication table. Refer to sample code 3 given above.
4. Type the sample program to print sum of first n numbers and execute the program for different values of n.
5. Write a program to accept characters till the user enters EOF and count number of times 'a' is entered. Refer to sample program 5 given above.

Signature of the instructor

Date



Assessment - work

Set A . Apply all the three program development steps for the following examples.

1. Write a program to accept an integer n and display all even numbers upto n.
2. Accept two integers x and y and calculate the sum of all integers between x and y (both inclusive)
3. Write a program to accept two integers x and n and compute x^n
4. Write a program to accept an integer and check if it is prime or not.
5. Write a program to accept an integer and count the number of digits in the number.
6. Write a program to accept an integer and reverse the number. Example: Input: 546, Output 645.
7. Write a program to accept a character, an integer n and display the next n characters.

Signature of the instructor

Date / /

Set B. Apply all the three program development steps for the following examples.

1. Write a program to display the first n Fibonacci numbers. (1 1 2 3 5)
2. Write a program to accept real number x and integer n and calculate the sum of first n terms of the series $x+ 3x+5x+7x+...$
3. Write a program to accept real number x and integer n and calculate the sum of first n terms of the series $\frac{1}{x} + \frac{2}{x^2} + \frac{3}{x^3} + \dots$
4. Write a program to accept characters till the user enters EOF and count number of alphabets and digits entered. Refer to sample program 5 given above.
5. Write a program, which accepts a number n and displays each digit in words. Example: 6702 Output = Six-Seven-Zero-Two. (Hint: Reverse the number and use a switch statement)

Signature of the instructor

Date / /

Set C. Write C programs to solve the following problems

1. Write a program to accept characters from the user till the user enters * and count the number of characters, words and lines entered by the user. (Hint: Use a flag to count words. Consider delimiters like $\backslash n \backslash t$, ; . and space for counting words)
2. Write a program which accepts a number and checks if the number is a palindrome (Hint number = reverse of number)
Example: number = 3472 Output: It is not a palindrome
number = 262, Output : It is a palindrome
3. A train leaves station A at 4.00 a.m and travels at 80kmph. After every 30 minutes, it reaches a station where it halts for 10 minutes. It reaches its final destination B at 1.00 p.m. Display a table showing its arrival and departure time at every intermediate station. Also calculate the total distance between A and B.

4. A task takes $4\frac{1}{2}$ hours to complete. Two workers, A and B start working on it and take turns alternately. A works for 25 minutes at a time and is paid Rs 50, B works for 75 minutes at a time and is paid Rs. 150. Display the total number of turns taken by A and B, calculate their total amounts and also the total cost of the task.

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 5

Start Date / /



To demonstrate use of nested loops



In the previous exercise, you used while, do-while and for loops. You should read following topics before starting this exercise

1. Different types of loop structures in C.
2. Syntax for these statements.
3. Usage of each loop structure



Nested loop means a loop that is contained within another loop. Nesting can be done upto any levels. However the inner loop has to be completely enclosed in the outer loop. No overlapping of loops is allowed.

Sr. No	Format	Sample Program
1.	<p>Nested for loop</p> <pre> for(exp1; exp2 ; exp3) { for(exp11; exp12 ; exp13) { } } </pre>	<pre> /* Program to display triangle of numbers*/ #include <stdio.h> void main() { int n , line_number , number; printf("How many lines: "); scanf("%d",&n); for(line_number =1 ;line_number <=n; line_number++) { for(number = 1; number <= line_number; number++) printf ("%dt", number); printf ("\n"); } } </pre>
2.	<p>Nested while loop / do while loop</p> <pre> while(condition1) { while(condition2) { } } do { while(condition1) { } } </pre>	<pre> /* Program to calculate sum of digits till sum is a single digit number */ #include <stdio.h> void main() { int n , sum; printf("Give any number "); scanf("%d",&n); do { sum =0; printf("%d --->",n); while (n>0) { sum +=n%10; </pre>

<pre>..... } while (condition2);</pre>	<pre>n= n/10; } n=sum; } while(n >9); printf (“ %d” , n); }</pre>
--	--

Note: It is possible to nest any loop within another. For example, we can have a for loop inside a while or do while or a while loop inside a for.



1. The Sample program 1 displays n lines of the following triangle. Type the program and execute it for different values of n.

```

1
1 2
1 2 3
1 2 3 4
```

2. Modify the sample program 1 to display n lines of the Floyd’s triangle as follows (here n=4).

```

1
2 3
4 5 6
7 8 9 10
```

3. The sample program 2 computes the sum of digits of a number and the process is repeated till the number reduces to a single digit number. Type the program and execute it for different values of n and give the output

Input number	Output
6534	
67	
8	
567	

Signature of the instructor

Date



Set A . Write C programs for the following problems.

- Write a program to display all prime numbers between ____ and ____.
- Write a program to display multiplication tables from ____ to ____ having n multiples each. The output should be displayed in a tabular format. For example, the multiplication tables of 2 to 9 having 10 multiples each is shown below.

```

2 × 1 = 2          3 × 1 = 3 .....9 × 1 = 9
2 × 2 = 4          3 × 2 = 6.....9 × 2 = 18
.....
2 × 10 = 20       3 × 10 = 30.....9 × 10 = 90
```

3. Modify the sample program 1 to display n lines as follows (here n=4).

```

A      B      C      D
E      F      G
H      I
J
```

Signature of the instructor

Date / /

Set B. Write C programs for the following problems.

1. Write a program to display all Armstrong numbers between 1 and 500. (An Armstrong number is a number such that the sum of cube of digits = number itself Ex. $153 = 1*1*1 + 5*5*5 + 3*3*3$)
2. Accept characters till the user enters EOF and count the number of lines entered. Also display the length of the longest line. (Hint: A line ends when the character is \n)
3. Display all perfect numbers below 500. [A perfect number is a number, such that the sum of its factors is equal to the number itself]. Example: 6 (1 + 2 + 3), 28 (1+2+4+7+14)

Signature of the instructor

Date / /

Set C. Write C programs to solve the following problems

1. A company has four branches, one in each zone: North, South, East and West. For each of these branches, it collects sales information once every quarter (four months) and calculates the average sales for each zone. Write a program that accepts sales details for each quarter in the four branches and calculate the average sales of each branch.
2. A polynomial in one variable is of the form $a_0 + a_1x + a_2x^2 + \dots$. For example, $6 - 9x + 2x^5$. Write a program to calculate the value of a polynomial. Accept the number of terms n , the value of x, and n+1 coefficients.
3. The temperature of a city varies according to seasons. There are four seasons – spring, summer, Monsoon and winter. The temperature ranges are: Spring (15-25 degrees), Summer (25-40 degrees), Monsoon (20-35 degrees), Winter (5-20 degrees). Accept weekly temperatures (12 weeks per season) for each season, check if they are in the correct range and calculate the average temperature for each season.

Signature of the instructor

Date / /

Assignment Evaluation

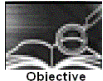
Signature

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 6

Start Date

/ /



To demonstrate menu driven programs and use of standard library functions



You should read following topics before starting this exercise

1. Use of switch statement to create menus as in exercise 3
2. Use of while and do while loops as in exercise 4



A function is a named sub-module of a program, which performs a specific, well-defined task. It can accept information from the calling function in the form of arguments and return only 1 value. C provides a rich library of standard functions. We will explore some such function libraries and use these functions in our programs.

ctype.h : contains function prototypes for performing various operations on characters. Some commonly used functions are listed below.

Function Name	Purpose	Example
isalpha	Check whether a character is a alphabet	if (isalpha(ch))
isalnum	Check whether a character is alphanumeric	if (isalnum(ch))
isdigit	Check whether a character is a digit	if (isdigit(ch))
isspace	Check whether a character is a space	if (isspace(ch))
ispunct	Check whether a character is a punctuation symbol	if (ispunct(ch))
isupper	Check whether a character is uppercase alphabet	if (isupper(ch))
islower	Check whether a character is lowercase alphabet	if (islower(ch))
toupper	Converts a character to uppercase	ch = toupper(ch)
tolower	Converts a character to lowercase	ch = tolower(ch)

math.h : This contains function prototypes for performing various mathematical operations on numeric data. Some commonly used functions are listed below.

Function Name	Purpose	Example
cos	cosine	$a^2 + b^2 - 2ab \cos(\text{abangle})$
exp(double x)	exponential function, computes e^x	exp(x)
log	natural logarithm	$c = \log(x)$
log10	base-10 logarithm	$y = \log_{10}(x)$
pow(x,y)	compute a value taken to an exponent, x^y	$y = 3^{\text{pow}(x, 10)}$
sin	sine	$z = \sin(x) / x$
sqrt	square root	$\text{delta} = \sqrt{b^2 - 4ac}$

Note: If you want to use any of the above functions you must include the library for example

```
#include <ctype.h>
```

```
#include <math.h>
```

In case of math library , it needs to be linked to your program. You have to compile the program as follows

```
$ cc filename -lm
```

A program that does multiple tasks, provides a menu from which user can choose the appropriate task to be performed. The menu should appear again when the task is completed so that the user can choose another task. This process continues till the user decides to quit. A menu driven program can be written using a combination of do-while loop containing a switch statement. One of the options provided in a menu driven program is to exit the program.

Statement Syntax	Flowchart	Example
<pre>do { display menu; accept choice; switch(choice) { case value1: block1; break; case value2: block2; break; . . . default : default block; } }while(choice != exit);</pre>		<pre>ch = getchar(); do { printf("\n 1: ISUPPER "); printf("\n 2: ISLOWER "); printf("\n 3: ISDIGIT "); printf("\n 4: EXIT"); printf("Enter your choice :"); scanf("%d", &choice); switch (choice) { case 1: if(isupper(ch)) printf("Uppercase"); break; case 2: if(islower(ch)) printf("Lowercase"); break; case 3: if(isdigit(ch)) printf("Digit"); break; } }while (choice!=4);</pre>



1. Write a menu driven program to perform the following operations on a character type variable.
 - i. Check if it is an alphabet
 - ii. Check if it is a digit.
 - iii. Check if it is lowercase.
 - iv. Check if it is uppercase.
 - v. Convert it to uppercase.
 - vi. Convert it to lowercase.

Refer to the sample code given above and use standard functions from ctype.h



Set A . Write C programs for the following problems

1. Write a program, which accepts a character from the user and checks if it is an alphabet, digit or punctuation symbol. If it is an alphabet, check if it is uppercase or lowercase and then change the case.

2. Write a menu driven program to perform the following operations till the user selects Exit. Accept appropriate data for each option. Use standard library functions from math.h
 i. Sine ii. Cosine iii. log iv. e^x v. Square Root vi. Exit

3. Accept two complex numbers from the user (real part, imaginary part). Write a menu driven program to perform the following operations till the user selects Exit.
 i. ADD ii. SUBTRACT iii. MULTIPLY iv. EXIT

Signature of the instructor

Date / /

Set B . Write C programs for the following problems

1. Accept x and y coordinates of two points and write a menu driven program to perform the following operations till the user selects Exit.
 i. Distance between points.
 ii. Slope of line between the points.
 iii. Check whether they lie in the same quadrant.
 iv. EXIT
 (Hint: Use formula $m = (y_2 - y_1) / (x_2 - x_1)$ to calculate slope of line.)

2. Write a simple menu driven program for a shop, which sells the following items:
 The user selects items using a menu. For every item selected, ask the quantity. If the quantity of any item is more than 10, give a discount of ____%. When the user selects Exit, display the total amount.

Item	Price

Instructor should fill in the blanks with appropriate values.

Signature of the instructor

Date / /

Set C . Write C programs for the following problems

1. Write a program to calculate the total price for a picnic lunch that a user is purchasing for her group of friends. She is first asked to enter a budget for the lunch. She has the option of buying apples, cake, and bread. Set the price per kg of apples, price per cake, and price per loaf of bread in constant variables. Use a menu to ask the user what item and how much of each item she would like to purchase. Keep calculating the total of the items purchased. After purchase of an item, display the remaining amount. Exit the menu if the total has exceeded the budget. In addition, provide an option that allows the user to exit the purchasing loop at any time.

Signature of the instructor

Date / /

Assignment Evaluation

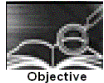
Signature

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

Exercise 7

Start Date

/ /



To demonstrate writing C programs in modular way (use of user defined functions)



You should read following topics before starting this exercise

1. Declaring and Defining a function
2. Function call
3. Passing parameters to a function
4. Function returning a value



You have already used standard library functions. C allows to write and use user defined functions. Every program has a function named main. In main you can call some functions which in turn can call other functions.

The following table gives the syntax required to write and use functions

Sr. No	Actions involving functions	Syntax	Example
1.	Function declaration	returntype function(type arg1, type arg2 ...);	void display(); int sum(int x, int y);
2.	Function definition	returntype function(type arg1, type arg2 ...) { /* statements*/ }	float calcarea (float r) { float area = Pi *r*r ; return area; }
3.	Function call	function(arguments); variable = function(arguments);	display(); ans = calcarea(radius);

1. Sample code

The program given below calculates the area of a circle using a function and uses this function to calculate the area of a cylinder using another function.

```
/* Program to calculate area of circle and cylinder using function */
```

```
#include <stdio.h>
void main()
{
    float areacircle (float r);
    float areacylinder(float r, int h);
    float area, r;
    printf("\n Enter Radius: ");
    scanf("%f",&r);

    area=areacircle(r);
    printf("\n Area of circle =%6.2f", area);

    printf("\n Enter Height: ");
```



```

scanf("%d",&h);
area=areacylinder(r,h);
printf("\n Area of cylinder =%6.2f", area);
}

float areacircle (float r)
{
    const float pi=3.142;
    return(pi * r*r );
}
float areacylinder (float r, int h)
{
    return 2*areacircle(r)*h;
}

```

2. Sample code

The function `iswhitespace` returns 1 if its character parameter is a space, tab or newline character. The program accepts characters till the user enters EOF and counts the number of white spaces.

```

/* Program to count whitespaces using function */

#include <stdio.h>
void main()
{
    int iswhitespace (char ch);
    char ch;
    int count=0;

    printf("\n Enter the characters. Type CTRL +Z to terminate: ");
    while((ch=getchar())!=EOF)
        if(iswhitespace(ch))
            count++;
    printf("\n The total number of white spaces =%d", count);
}
int iswhitespace (char ch)
{
    switch(ch)
    {
        case ' ':
        case '\t':
        case '\n': return 1;
        default : return 0;
    }
}

```



Self-Activity

1. Type the program given in sample code 1 above and execute the program. Comment function declarations and note down the type of error and the error messages received. Add another function to calculate the volume of sphere and display it.

2. Type the program given in sample code 2 above and execute the program. Comment function declaration and note down the type of error and the error messages received. Modify the function such that it returns 1 if the character is a vowel. Also count the total number of vowels entered.



Set A . Write C programs for the following problems

1. Write a function isEven, which accepts an integer as parameter and returns 1 if the number is even, and 0 otherwise. Use this function in main to accept n numbers and ckeck if they are even or odd.

2. Write a function, which accepts a character and integer n as parameter and displays the next n characters.

3. Write a function, which accepts a character and integer n as parameter and displays the next n characters.

Signature of the instructor

Date / /

Set B . Write C programs for the following problems

1. Write a function isPrime, which accepts an integer as parameter and returns 1 if the number is prime and 0 otherwise. Use this function in main to display the first 10 prime numbers.

2. Write a function that accepts a character as parameter and returns 1 if it is an alphabet, 2 if it is a digit and 3 if it is a special symbol. In main, accept characters till the user enters EOF and use the function to count the total number of alphabets, digits and special symbols entered.

3. Write a function power, which calculates x^y . Write another function, which calculates n! Using for loop. Use these functions to calculate the sum of first n terms of the Taylor series:

$$\sin(x) = x - \frac{x^3}{3!} + \frac{x^5}{5!} + \dots$$

Signature of the instructor

Date / /

Set C . Write C programs for the following problems

1. Write a menu driven program to perform the following operations using the Taylor series. Accept suitable data for each option. Write separate functions for each option.

i. e^x
$$e^x = \sum_{n=0}^{\infty} \frac{x^n}{n!} = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \dots \text{ for all } x$$

ii. $\sin(x)$
$$\sin x = \sum_{n=0}^{\infty} \frac{(-1)^n}{(2n + 1)!} x^{2n+1} = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \dots \text{ for all } x$$

iii. $\cos(x)$
$$\cos x = \sum_{n=0}^{\infty} \frac{(-1)^n}{(2n)!} x^{2n} = 1 - \frac{x^2}{2!} + \frac{x^4}{4!} - \dots \text{ for all } x$$

Define separate functions to calculate x^y and n! and use them in each function.

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 / /



To demonstrate Recursion.



You should read the following topics before starting this exercise

1. Recursive definition
2. Declaring and defining a function
3. How to call a function
4. How to pass parameters to a function



Recursion is a process by which a function calls itself either directly or indirectly. The points to be remembered while writing recursive functions

- i. Each time the function is called recursively it must be closer to the solution.
- ii. There must be some terminating condition, which will stop recursion.
- iii. Usually the function contains an if –else branching statement where one branch makes recursive call while other branch has non-recursive terminating condition

Expressions having recursive definitions can be easily converted into recursive functions

Sr. No	Recursive definition	Recursive Function	Sample program
1.	<p>The recursive definition for factorial is given below:</p> $n! = 1 \quad \text{if } n = 0 \text{ or } 1$ $= n * (n-1)! \text{ if } n > 1$	<pre>long int factorial (int n) { If (n==0) (n==1) /* terminating condition */ return(1); else return(n* factorial(n- 1)); /* recursive call */ }</pre>	<pre>#include <stdio.h> main() { int num; /* function declaration */ long int factorial(int n); printf("\n enter the number:"); scanf("%d",&num); printf("\n The factorial of %d is %d",num,factorial(num)); } /* function code*/</pre>
2.	<p>The recursive definition for nCr (no of combinations of r objects out of n objects) is as follows</p> $nCn = 1$ $nC0 = 1$ $nCr = n-1Cr + nCr-1$	<pre>long int nCr(int n, int r) { if(n==r r==0) /* terminating condition */ return(1); else return (nCr(n-1,r) + nCr(n, r-1)); /* recursive call */ }</pre>	<pre>#include <stdio.h> /* function code*/ main() { int n, r; printf("\n enter the total number of objects:"); scanf("%d",&n); printf("\n enter the number of objects to be selected"); scanf("%d",&r); printf("\n The value %dC%d is %ld",n, r, nCr(n,r)); }</pre>



1. Write the sample program 1 given above and execute the program. Modify the program to define a global integer variable count and increment it in factorial function. Add a printf statement in main function for variable count. Execute the program for different values and fill in the following table.

Sr. No.	num	factorial	Count
1.	0		
2	1		
3	5		
4	___		
5	___		

2. Write the sample program 2 given above and execute the program for different values of n and r. Modify the program to define a global integer variable count and increment it in nCr function. Add a print statement in main function for variable count. Execute the program for different values and fill in the following table

Sr. No.	n	r	nCr	Count
1.	5	0		
2	5	5		
3	5	2		
4	5	___		
5	___	___		

Instructor should fill in the blanks with appropriate values.

Signature of the instructor

Date



Set A . Write C programs for the following problems

1. Write a recursive C function to calculate the sum of digits of a number. Use this function in main to accept a number and print sum of its digits.

2. Write a recursive C function to calculate the GCD of two numbers. Use this function in main. The GCD is calculated as :

$$\begin{aligned} \text{gcd}(a,b) &= a && \text{if } b = 0 \\ &= \text{gcd}(b, a \text{ mod } b) && \text{otherwise} \end{aligned}$$

3. Write a recursive function for the following recursive definition. Use this function in main to display the first 10 numbers of the following series

$$\begin{aligned} a_n &= 3 && \text{if } n = 1 \text{ or } 2 \\ &= 2 * a_{n-1} + 3 * a_{n-2} && \text{if } n > 2 \end{aligned}$$

4. Write a recursive C function to calculate x^y . (Do not use standard library function)

Signature of the instructor

Date

Set B . Write C programs for the following problems

1. Write a recursive function to calculate the nth Fibonacci number. Use this function in main to display the first n Fibonacci numbers. The recursive definition of nth Fibonacci number is as follows:

$$\begin{aligned} \text{fib}(n) &= 1 && \text{if } n = 1 \text{ or } 2 \\ &= \text{fib}(n-2) + \text{fib}(n-1) && \text{if } n > 2 \end{aligned}$$

2. Write a recursive function to calculate the sum of digits of a number till you get a single digit number. Example: 961 -> 16 -> 5. (Note: Do not use a loop)

3. Write a recursive C function to print the digits of a number in reverse order. Use this function in main to accept a number and print the digits in reverse order separated by tab.

Example 3456

6 5 4 3

(Hint: Recursiveprint(n) = print n if n is single digit number
 = print n % 10 + tab + Recursiveprint(n/10)

Signature of the instructor

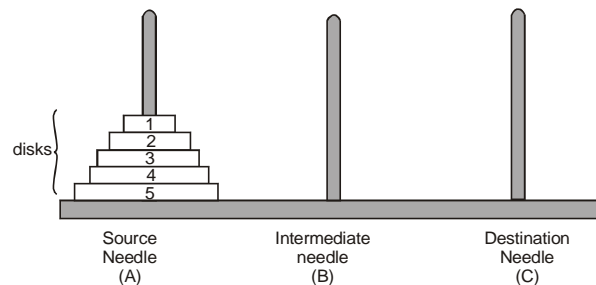
Date / /

Set C . Write C programs for the following problems

1. The "Towers of Hanoi" problem: The objective is to move a set of disks arranged in increasing sizes from top to bottom from the source pole to a destination pole such that they are in the same order as before using only one intermediate pole subject to the condition that

- Only one disk can be moved at a time
- A bigger disk cannot be placed on a smaller disk.

Write a recursive function which displays all the steps to move n disks from A to C.



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 / /



To demonstrate use of 1-D arrays and functions.



You should read the following topics before starting this exercise

1. What are arrays and how to declare an array?
2. How to enter data in to array and access the elements of an array.
3. How to initialize an array and how to check the bounds of an array?
4. How to pass an array to a function



An array is a collection of data items of the same data type referred to by a common name. Each element of the array is accessed by an index or subscript. Hence, it is also called a subscripted variable.

Actions involving arrays	syntax	Example
Declaration of array	<i>data-type array_name[size];</i>	int temperature[10]; float pressure[20];
Initialization of array	<i>data-type array_name[]={element1, element2,, element n};</i> <i>data-type array_name[size]={element-1, element-2,, element-size};</i> You cannot give more number of initial values than the array size. If you specify less values, the remaining will be initialized to 0.	int marks[]={45,57,87,20,90}; marks[3] refers to the fourth element which equals 20 int count[3]={4,2,9}; count[2] is the last element 9 while 4 is count[0]
Accessing elements of an array	The array index begins from 0 (zero) To access an array element, we need to refer to it as array_name[index].	Value = marks[3]; This refers to the 4 th element in the array
Entering data into an array.		for (i=0; i<=9; i++) scanf("%d", &marks[i]);
Printing the data from an array		for(i=0; i<=9; i++) printf("%d", marks[i]);
Arrays and function	We can pass an array to a function using two methods. 1. Pass the array element by element 2. Pass the entire array to the function	<i>/* Passing the whole array*/</i> void modify(int a[5]) { int i; for(i=0; i<5 ; i++) a[i] = i; }

Sample program to find the largest element of an array

```
/* Program to find largest number from array */
#include<stdio.h>
int main()
{
    int arr[20]; int n;
    void accept(int a[20], int n);
    void display(int a[20], int n);
    int maximum(int a[20], int n);

    printf("How many numbers :");
    scanf("%d", &n);
    accept(arr,n);
    display(arr,n);
    printf("The maximum is :%d" , maximum(arr,n));
}

void accept(int a[20], int n)
{
    int i;
    for(i=0; i<n ; i++)
        scanf("%d", &a[i]);
}

void display(int a[20], int n)
{
    int i;
    for(i=0; i<n ; i++)
        printf("%d\t", a[i]);
}

int maximum(int a[20], int n)
{
    int i, max = a[0];

    for(i=1; i<n ; i++)
        if(a[i] > max)
            max = a[i];
    return max;
}
```



1. Write a program to accept n numbers in an array and display the largest and smallest number. Using these values, calculate the range of elements in the array. Refer to the sample code given above and make appropriate modifications.

2. Write a program to accept n numbers in an array and calculate the average. Refer to the sample code given above and make appropriate modifications.

Signature of the instructor

Date



Set A. Write programs to solve the following problems

1. Write a program to accept n numbers in the range of 1 to 25 and count the frequency of occurrence of each number.
2. Write a function for Linear Search, which accepts an array of n elements and a key as parameters and returns the position of key in the array and -1 if the key is not found. Accept n numbers from the user, store them in an array. Accept the key to be searched and search it using this function. Display appropriate messages.
3. Write a function, which accepts an integer array and an integer as parameters and counts the occurrences of the number in the array.
4. Write a program to accept n numbers and store all prime numbers in an array called prime. Display this array.

Signature of the instructor

Date

Set B. Write programs to solve the following problems

1. Write a program to accept n numbers from the user and store them in an array such that the elements are in the sorted order. Display the array. Write separate functions to accept and display the array. (Hint: Insert every number in its correct position in the array)
2. Write a function to sort an array of n integers using Bubble sort method. Accept n numbers from the user, store them in an array and sort them using this function. Display the sorted array.
3. Write a program to accept a decimal number and convert it to binary, octal and hexadecimal. Write separate functions.
4. Write a program to find the union and intersection of the two sets of integers (store it in two arrays).
5. Write a program to remove all duplicate elements from an array.

Signature of the instructor

Date

Set C. Write programs to solve the following problems

1. Write a program to merge two sorted arrays into a third array such that the third array is also in the sorted order.

a1	10	25	90					
a2	9	16	22	26	10			
					0			
a3	9	10	16	22	25	26	90	100

2. Write a program to accept characters from the user till the user enters EOF and calculate the frequency count of every alphabet. Display the alphabets and their count.

Input: THIS IS A SAMPLE INPUT

Output:

Character	Count
T	2
H	1
I	3
.....	

3. Write a recursive function for Binary Search, which accepts an array of n elements and a key as parameters and returns the position of key in the array and -1 if the key is not found. Accept n numbers from the user, store them in an array and sort the array. Accept the key to be searched and search it using this function. Display appropriate messages

Signature of the instructor

Date

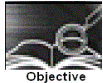
Assignment Evaluation

Signature

0: Not done <input style="width: 40px; height: 20px;" type="text"/>	2: Late Complete <input style="width: 40px; height: 20px;" type="text"/>	4: Complete <input style="width: 40px; height: 20px;" type="text"/>
1: Incomplete <input style="width: 40px; height: 20px;" type="text"/>	3: Needs improvement <input style="width: 40px; height: 20px;" type="text"/>	5: Well Done <input style="width: 40px; height: 20px;" type="text"/>

Exercise 10

Start Date / /



To demonstrate use of 2-D arrays and functions.



You should read the following topics before starting this exercise

1. How to declare and initialize two-dimensional array
2. Accessing elements
3. Usage of two dimensional arrays



Actions involving 2-D arrays	syntax	Example
Declaration of 2-D array	<i>data-type array_name[size][size];</i>	int mat[10][10]; float sales[4][10];
Initialization of 2-D array	<i>data-type array_name[rows][cols]={ {elements of row 0},{ elements of row 1},.....}; data-type array_name[][cols]={element1, element2,, element size};</i>	int num[][2] = {12, 34, 23, 45, 56, 45}; int num[3][2] = { {1,2}, {3,4}, {5,6}}; int num[3][2] = { 1,2,3,4, 5,6};
Accessing elements of 2-D array	Accessing elements of an two-dimensional array - in general, the array element is referred as array_name[index1][index2] where <i>index1</i> is the row location of and <i>index2</i> is the column location of an element in the array.	int m[3][2]; m is declared as a two dimensional array (matrix) having 3 rows (numbered 0 to 2) and 2 columns (numbered 0 to 1). The first element is m[0][0] and the last is m[2][1]. value = m[1][1];
Entering data into a 2-D array.		int mat[4][3]; for (i=0; i<4; i++) /* outer loop for rows */ for (j=0;j<3; j++) /* inner loop for columns */ scanf("%d", &mat[i][j]);
Printing the data from a 2-D array		for (i=0; i<4; i++) /* outer loop for rows */ { for (j=0;j<3; j++) /* inner loop for columns */ printf("%d\t", mat[i][j]); printf("\n"); }

Sample program to accept, display and print the sum of elements of each row of a matrix.

```
/* Program to calculate sum of rows of a matrix*/

#include<stdio.h>
int main()
{
    int mat[10][10], m, n;
    void display(int a[10][10], int m, int n);
    void accept(int a[10][10], int m, int n);
    void sumofrows(int a[10][10], int m, int n);

    printf("How many rows and columns? ");
    scanf("%d%d",&m, &n);

    printf("Enter the matrix elements .:");
    accept(mat, m, n);
    printf("\n The matrix is :\n");
    display(mat, m, n);
    sumofrows(mat,m,n);
}

void accept(int a[10][10], int m, int n)
{
    int i,j;
    for (i=0; i<m; i++) /* outer loop for rows */
        for (j=0;j<n; j++) /* inner loop for columns */
            scanf("%d", &a[i][j]);
}

void display(int a[10][10], int m, int n)
{
    int i,j;
    printf("\nThe elements of %d by %d matrix are\n", m, n);
    for (i=0; i<m; i++) /* outer loop for rows */
    {
        for (j=0;j<n; j++) /* inner loop for columns */
            printf("%d\t", a[i][j]);
        printf("\n");
    }
}

void somofrows(int a[10][10], int m, int n)
{
    int i,j, sum;
    for (i=0; i<m; i++) /* outer loop for rows */
    { sum=0
        for (j=0;j<n; j++) /* inner loop for columns */
            sum= sum+a[i][j];
        printf("Sum of elements of row %d = %d", i, sum);
    }
}
```



1. Write a program to accept, display and print the sum of elements of each row and sum of elements of each column of a matrix. Refer to sample code given above.

Signature of the instructor

Date / /



Set A . Write C programs for the following problems.

1. Write a program to accept a matrix A of size mXn and store its transpose in matrix B. Display matrix B. Write separate functions.
2. Write a program to add and multiply two matrices. Write separate functions to accept, display, add and multiply the matrices. Perform necessary checks before adding and multiplying the matrices.

Signature of the instructor

Date

Set B . Write C programs for the following problems.

1. Write a menu driven program to perform the following operations on a square matrix. Write separate functions for each option.
 - i) Check if the matrix is symmetric.
 - ii) Display the trace of the matrix (sum of diagonal elements).
 - iii) Check if the matrix is an upper triangular matrix.
2. Write a menu driven program to perform the following operations on a square matrix. Write separate functions for each option.
 - i) Check if the matrix is a lower triangular matrix.
 - ii) Check if it is an identity matrix.

3. Write a program to accept an mXn matrix and display an m+1 X n+1 matrix such that the m+1th row contains the sum of all elements of corresponding row and the n+1th column contains the sum of elements of the corresponding column.

Example:

A				B			
1	2	3		1	2	3	6
4	5	6		4	5	6	15
7	8	9		7	8	9	24
				12	15	18	45

Signature of the instructor

Date

Set C. Write programs to solve the following problems

1. Pascal's triangle is a geometric arrangement of the binomial coefficients in a triangle. It is named after Blaise Pascal. Write a program to display n lines of the triangle.

```

      1
     1 1
    1 2 1
   1 3 3 1
  1 4 6 4 1
 1 5 10 10 5 1
1 6 15 20 15 6 1

```

2. A magic square of order n is an arrangement of n^2 numbers, in a square, such that the n numbers in all rows, all columns, and both diagonals sum to the same constant. A normal magic square contains the integers from 1 to n^2 . The magic constant of a magic square depends on n and is $M(n) = (n^3+n)/2$. For $n=3,4,5$, the constants are 15, 34, 65 resp. Write a program to generate and display a magic square of order n .

2	7	6	→	15
9	5	1	→	15
4	3	8	→	15
15	↓	↓	↓	↓
	15	15	15	15

Signature of the instructor

Date

Assignment Evaluation

Signature

0: Not done	<input style="width: 40px; height: 20px;" type="text"/>	2: Late Complete	<input style="width: 40px; height: 20px;" type="text"/>	4: Complete	<input style="width: 40px; height: 20px;" type="text"/>
1: Incomplete	<input style="width: 40px; height: 20px;" type="text"/>	3: Needs improvement	<input style="width: 40px; height: 20px;" type="text"/>	5: Well Done	<input style="width: 40px; height: 20px;" type="text"/>

Exercise 11

Start Date / /



To demonstrate use of pointers in C.



You should read the following topics before starting this exercise

1. What is a pointer?
2. How to declare and initialize pointers.
3. '*' and '&' operators.
5. Pointer to a pointer.
6. Relationship between array and pointer.
7. Pointer to array and Array of pointers.
8. Dynamic memory allocation (malloc, calloc, realloc, free).



A Pointer is a variable that stores the memory address of another variable

Actions involving Pointers	syntax	Example
Declaration of pointers	<code>data_type * pointer_name</code>	<code>int *p1,*p2; float *temp1;</code>
Initialization of pointers	<code>pointer =&variable p1=&n;</code>	<code>int a, *p= &a;</code>
Pointer Arithmetic	The C language allow arithmetic operations to be performed on pointers: Increment, Decrement, Addition, Subtraction When a pointer is incremented (or decremented) by 1, it increments by sizeof(datatype). For example, an integer pointer increments by sizeof(int).	
Pointers and Functions	We can pass the address of a variable to a function. The function can accept this address in a pointer and use the pointer to access the variable's value.	
Arrays And Pointers	An array name is a pointer to the first element in the array. It holds the base address of the array. Every array expression is converted to pointer expression as follows: <code>a[i]</code> is same as <code>*(a+i)</code>	<code>int n; *n , *(n + 0)</code> represents 0 th element <code>n[j] , *(n+ j) , *(j + n) , j [n] :</code> represent the value of the j th element of array n
Pointer To Pointer		<code>int a; int * p; int **q; p = &a;</code>

<p>To allocate memory dynamically</p>	<p>The functions used are : malloc, calloc, realloc ptr = (cast-type *) malloc (byte-size) ; Allocates a block of contiguous bytes. If the space in heap is not sufficient to satisfy request, allocation fails, returns NULL. ptr1 = (cast-type *) calloc (byte-size); Similar to malloc, but initializes the memory block allocated to 0. ptr = realloc (ptr, new size); To increase / decrease memory size.</p>	<pre>q = *p ; int * p,*p1; p = (int *) malloc(10 * sizeof(int)); p1 = (int *) calloc(10, sizeof(int)); p1=realloc(p1,20*sizeof(int));</pre>
---------------------------------------	--	---

1. Sample program

```
/* Program to swap two numbers*/
main()
{
    int a = 10, b = 20;
    void swap1( int x, int y);
    void swap2( int *ptr1, int *ptr2);

    printf("\nBefore swapping : a=%d, b=%d", a,b);
    swap1(a, b);
    printf("\nAfter swapping by swap1 : a=%d, b=%d", a,b);
    swap2(&a, &b);
    printf("\nAfter swapping by swap2 : a=%d, b=%d", a,b);
}

void swap1( int x, int y)
{
    int temp;
    temp = x;
    x = y;
    y = temp;
}

void swap2( int *ptr1, int *ptr2)
{
    int temp;
    temp = *ptr1;
    *ptr1 = *ptr2;
    *ptr2 = temp;
}
```

2. Sample program

```
/* Program to allocate memory for n integers dynamically*/
#include <stdlib.h>
void main()
{
    int *p, n,i;
    printf("How many elements :");
    scanf("%d",&n);

    p = (int *)malloc(n*sizeof(int));
    /* Accepting data */
    for(i=0; i<n;i++)
        scanf("%d",p+i);

    /* Displaying data */
    for(i=0; i<n;i++)
        printf("%d\t",*(p+i));
}
```



Self-Activity

1. Type the sample program 1 given above, execute it and write the output.
2. Sample program 2 allocates memory dynamically for n integers and accepts and displays the values. Type the sample program 2 given above, execute it. Modify the program to allocate memory such that the allocated memory is initialized to 0.



Assessment - work

Set A . Write C programs for the following problems.

1. Write a function which takes hours, minutes and seconds as parameters and an integer s and increments the time by s seconds. Accept time and seconds in main and Display the new time in main using the above function.
2. Write a program to display the elements of an array containing n integers in the reverse order using a pointer to the array.
3. Write a program to allocate memory dynamically for n integers such that the memory is initialized to 0. Accept the data from the user and find the range of the data elements.

Signature of the instructor

Date

Set B . Write C programs for the following problems.

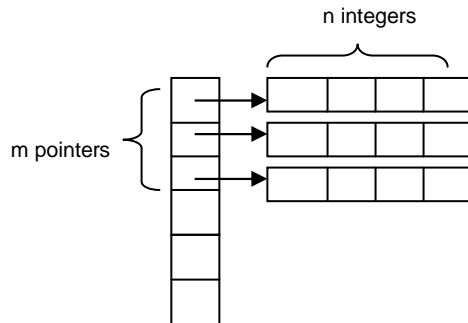
1. Accept n integers in array A. Pass this array and two counter variables to a function which will set the first counter to the total number of even values in the array and the other to the total number of odd values. Display these counts in main. (Hint: Pass the addresses of the counters to the function)
2. Write a function which accepts a number and three flags as parameters. If the number is even, set the first flag to 1. If the number is prime, set the second flag to 1. If the number is divisible by 3 or 7, set the third flag to 1. In main, accept an integer and use this function to check if it is even, prime and divisible by 3 or 7. (Hint : pass the addresses of flags to the function)

Signature of the instructor

Date

Set C. Write programs to solve the following problems

1. Accept the number of rows (m) and columns (n) for a matrix and dynamically allocate memory for the matrix. Accept and display the matrix using pointers. Hint: Use an array of pointers.



2. There are 5 students numbered 1 to 5. Each student appears for different number of subjects in an exam. Accept the number of subjects for each student and then accept the marks for each subject. For each student, calculate the percentage and display. (Hint: Use array of 5 pointers and use dynamic memory allocation)

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 12

Start Date

/ /



Objective

To demonstrate strings in C.



Reading

You should read the following topics before starting this exercise

1. String literals
2. Declaration and definition of string variables
3. The NULL character
4. Accepting and displaying strings
5. String handling functions



Ready Reference

A string is an array of characters terminated by a special character called NULL character(\0). Each character is stored in 1 byte as its ASCII code.

Actions Involving strings	Explanation	Example
Declaring Strings		<code>char message[80];</code>
Initializing Strings		<code>char message[] = { 'H', 'e', 'l', 'l', 'o', '\0' }; char message [] = "Hello";</code>
Accepting Strings	scanf and gets can be used to accept strings	<code>char name[20], address[50]; printf("\n Enter your name :); scanf("%s", name); printf("\n Enter your address :); gets(address);</code>
Displaying Strings	printf and puts can be used to display strings.	<code>printf("\n The name is %s:", name); printf("\n The address is :"); puts(address);</code>
String functions	All string operations are performed using functions in "string.h". Some of the most commonly used functions are a. strlen – Returns the number of characters in the string (excluding \0) b. strcpy – Copies one string to another c. strcmp – Compares two strings. Returns 0 (equal), +ve (first string > second), -ve (first string < second). It is case sensitive d. strcmpi – Same as strcmp but ignores case e. strcat – Concatenates the second string to the first.	<code>#include <string.h> main() { char str1[50], str2[50],str3[100]; printf("\n Give the first string:"); gets(str1); printf("\n Give the second string string:"); gets(str2); if (strlen(str1) == strlen(str2)) {strcpy(str3, str1); strcat(str3, str2); puts(str3); } else puts(str1); }</code>

	Returns the concatenated string. f. <code>strrev</code> – Reverses a string and returns the reversed string. g. <code>strupr</code> – Converts a string to uppercase. h. <code>strlwr</code> - Converts a string to lowercase	
--	--	--

Sample program :

The following program demonstrates how to pass two strings to a user defined function and copy one string to other using pointers

```

void string_copy (char *t,char *s)
{
    while (*s !='\0')      /* while source string does not end */
    {
        *t=*s;
        s++;
        t++;
    }
    *t ='\0'; /*      terminate target string */
}

void main()
{
    char str1[20], str2[20];
    printf("Enter a string :");
    gets(str1);
    string_copy(str2, str1);
    printf("The copied string is :");
    puts(str2);
}

```



1. Write a program to accept two strings `str1` and `str2`. Compare them. If they are equal, display their length. If `str1 < str2`, concatenate `str1` and the reversed `str2` into `str3`. If `str1 > str2`, convert `str1` to uppercase and `str2` to lowercase and display. Refer sample code for string functions above.

2. Type the sample program above and execute it. Modify the program to copy the characters after reversing the case. (Hint: First check the case of the character and then reverse it)

Signature of the instructor

Date / /



Set A . Write C programs for the following problems.

1. Write a menu driven program to perform the following operations on strings using standard library functions:

- | | | | |
|---------|-----------|---------------|------------|
| Length | Copy | Concatenation | Compare |
| Reverse | Uppercase | Lowercase | Check case |

2. Write a program that will accept a string and character to search. The program will call a function, which will search for the occurrence position of the character in the

string and return its position. Function should return -1 if the character is not found in the string.

3. A palindrome is a string that reads the same-forward and reverse. *Example:* "madam" is a Palindrome. Write a function which accepts a string and returns 1 if the string is a palindrome and 0 otherwise. Use this function in main.

4. For the following standard functions, write corresponding user defined functions and write a menu driven program to use them. strcat, strcmp, strrev,strupr

5. Write a program which accepts a sentence from the user and alters it as follows: Every space is replaced by *, case of all alphabets is reversed, digits are replaced by ?

Signature of the instructor

Date

Set B . Write C programs for the following problems.

1. Write a menu driven program which performs the following operations on strings. Write a separate function for each option. Use pointers

- i. Check if one string is a substring of another.
- ii. Count number of occurrences of a character in the string.
- iii. Replace all occurrences of a character by another.

2. Write a program in C to reverse each word in a sentence.

3. Write a function which displays a string in the reverse order. (Use recursion)

Signature of the instructor

Date

Set C. Write programs to solve the following problems

1. Write a program that accepts a sentence and returns the sentence with all the extra spaces trimmed off. (In a sentence, words need to be separated by only one space; if any two words are separated by more than one space, remove extra spaces)

2. Write a program that accepts a string and displays it in the shape of a kite. Example: "abcd" will be displayed as :

```
aa
abab
abcabc
abcdabcd
abcabc
abab
aa
```

3. Write a program that accepts a string and generates all its permutations. For example: ABC, ACB, BAC, BCA, CAB, CBA

4. Write a program to display a histogram of the frequencies of different characters in a sentence. Note: The histogram can be displayed as horizontal bars constructed using * character. Example: this is a single string

t * *
h *
i * * * *
s * * * *
a * *
n * *
g * *
l *
e *
r *

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 demonstrate array of Strings.



You should read the following topics before starting this exercise

1. How to declare and initialize strings.
2. String handling functions
3. How to create and access an array of strings.
4. Dynamic memory allocation



An array of strings is a two dimensional array of characters. It can be treated as a 1-D array such that each array element is a string.

Actions Involving array of strings	Explanation	Example
Declaring String array	char array[size1][size2];	char cities[4][10]
Initializing String array		char cities[4][10] = { "Pune", "Mumbai", "Delhi", "Chennai"};

Sample program-

The following program illustrates how to accept 'n' names , store them in an array of strings and search for a specific name.

```
/* Program to search for name from array */
#include <stdio.h>
void main( )
{
    char list[10][20]; /*list is an array of 10 strings */
    char name[20];
    int i,n;
    printf("\n How many names ?:");
    scanf("%d", &n);
    for (i=0;i<n; i++)
    {
        printf("\n Enter name %d,");
        scanf("%s", list[i]);
    }
    printf("\n The names in the list are : \n");
    for (i=0; i<n; i++)
        printf("%s", list[i]);
    printf("\n Enter the name to be searched ");
    scanf("%s", name);
    for (i=0; i<n; i++)
        if(strcmp(list[i],name)==0)
        {
            printf("Match found at position %d", i);
            break;
        }
    if(i==n)
        printf("Name is not found in the list");
}
```



Self-Activity

1. Type the above sample program and execute the same for different inputs.

Signature of the instructor

Date / /



Assessment - work

Set A . Write C programs for the following problems.

1. Write a program that accepts n words and outputs them in dictionary order.
2. Write a program that accepts n strings and displays the longest string.
3. Write a program that accepts a sentence and splits the sentence into words. Sort each word and reconstruct the sentence.

Input – this is a string Output – hist is a ginrst

Signature of the instructor

Date / /

Set B . Write C programs for the following problems.

1. Write a function, which displays a given number in words.
For Example: 129 One Hundred Twenty Nine
 2019 Two Thousand Nineteen

2. Define two constant arrays of strings, one containing country names (ex: India, France etc) and the other containing their capitals. (ex: Delhi, Paris etc). Note that country names and capital names have a one-one correspondence. Accept a country name from the user and display its capital. Example: Input: India , Output: Delhi.

Signature of the instructor

Date / /

Set C. Write programs to solve the following problems

1. Create a mini dictionary of your own. Each entry in the dictionary contains three parts (word, its meaning, similar word). The entries are stored in the sorted order of words. Write a menu driven program, which performs the following operations.
 - i. Add a new word (Insert new word and its details in the correct position)
 - ii. Dictionary look-up
 - iii. Find similar word
 - iv. Delete word
 - v. Display All words starting with a specific alphabet (along with their meaning).

(Hint: Use 2-D array of strings having n rows and 3 columns)

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 14

Start Date

/ /



Objective

Assignment to demonstrate bitwise operators.



Reading

You should read the following topics before starting this exercise

1. Bitwise operators and their usage (&, |, ^, ~, <<, >>)



Ready Reference

1. **Bitwise operators:** C provides 6 operators to perform operations on bits. These operators operate on integer and character but not the float and double. Ones complement operator (~) is unary while the others are binary.

Operator	Purpose	Example
~	One's complement	~a : Complements each bit of variable a
>>	Right shift	a=a>>1; Shifts bits of a one position to the right
<<	Left Shift	a=a<<n; Shifts bits of a n positions to the left
&	Bitwise AND	a = b&c; performs bitwise AND on b and c a = a&0xFF00; Masks the lower order 8 bits of a
	Bitwise OR	a = a b; performs bitwise OR on b and c
^	Bitwise XOR	x = x^y; y=x^y; x=x^y; Swaps x and y by performing bitwise XOR.

Sample code: The following function accepts an integer argument and displays it in binary format. It uses shift operator and AND masking.

```
void displaybits(unsigned int n)
{
    unsigned int mask = 32768;
    /*set MSB of mask to 1 */
    while (mask>0)
    {
        if((n&mask)==0)
            printf("0");
        else
            printf("1");
        mask = mask >>1; /* shift mask right */
    }
}
```



Self-Activity

1. Write a program to accept n integers and display them in binary. Use the function given above.

Signature of the instructor

Date

/ /



Set A . Write C programs for the following problems.

1. Write a program to accept 2 integers and perform bitwise AND, OR, XOR and Complement. Display the inputs and results in binary format. Use the function in the above exercise.
2. Write a program to swap two variables without using a temporary variable. (Hint: Use XOR)
3. Write a program which accepts two integers x and y and performs $x \ll y$ and $x \gg y$. Display the result in binary.

Signature of the instructor

Date

Set B . Write C programs for the following problems.

1. Write functions to calculate the size of an integer, character, long and short integer using bitwise operators. Store their declaration in file "myfunctions.h" and their definitions in file "myfunctions.c". Include these files in your program and use these functions to display the size of each.
2. Write a program to perform the following operations on an unsigned integer using bitwise operators and display the result in hexadecimal format.
 - i. Swap the ____ and ____ nibble (4 bits)
 - ii. Remove the lower order nibbles from the number.
For example: Input: A3F1 Output 00A3
 - iii. Reverse the nibbles
For example: Input: A3F1 Output 1F3A
3. Write a program which accepts an integer and checks whether it is a power of 2.

Signature of the instructor

Date

Set C. Write programs to solve the following problems

1. Write a program to add, subtract, multiply and divide two integers using bitwise operators.
2. Packing and Unpacking Data: A date consists of three parts : day, month, year. To store this information, we would require 3 integers. However, day and month can take only limited values. Hence, we can store all three in a single integer variable by packing bits together. If we are using the dd-mm-yy format, the date will be stored in memory as an unsigned integer (16 bits) in the following format. Year (Bits 15-9), Month (bits 8 – 5), Day (Bits 4 - 0).

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
y	y	y	y	y	y	y	m	m	m	m	d	d	d	d	d
hour							Month				day				

Accept day, month and year from the user and pack them into a single unsigned int. Unpack and display them in the binary format. (Hint: for packing, use: $512 * \text{year} + 32 * \text{month} + \text{day}$)

The output should be:

Enter the date, month and year –dd mm yy :

31 12 89

Packed date = 1011001110011111

Day = 31

0000000000011111

Month = 12
 0000000000001100
 Year = 89
 000000001011001

3. Packing and Unpacking Data: Time consists of three parts : hours, minutes, seconds. To store this information, we would require 3 integers. However, all these three variable take only limited values. Hence, we can store all three in a single integer variable by packing bits together. Time being 0 to 23 hours, it will require maximum 5 bits, minutes being 0 to 59 will require 6 bits. The two together take up 11 bits. The remaining 5 bits cannot store seconds which are also in the range 0 to 59 hence we store double seconds which are in the range 0 to 29

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
h	h	h	h	h	m	m	m	m	m	m	ds	ds	ds	ds	ds
hour					Minutes					Double seconds					

Accept hour, minute and double seconds from the user and pack them into a single unsigned int. Unpack and display them in the binary format.

The output should be:

Enter the hour, minutes and double seconds –hh mm ss :

07 12 20

Packed date = 0011100110010100

Hour = 07

0000000000000111

Minutes = 12

0000000000001100

Double seconds = 20

000000000010100

Signature of the instructor

Date

Assignment Evaluation

Signature

0: Not done

2: Late Complete

4: Complete

1: Incomplete

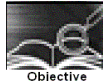
3: Needs improvement

5: Well Done

Exercise 15

Start Date

/ /



Structures in C



You should read the following topics before starting this exercise

1. Concept of structure
2. Declaring a structure
3. Accessing structure members
4. Array of structures
5. Pointer to a structure.
6. Passing structures to functions



A structure is a composition of variables possibly of different data types, grouped together under a single name. Each variable within the structure is called a 'member'.

Operations performed	Syntax / Description	Example
Declaring a structure	<pre>struct structure-name { type member-1 ; type member-2; . . type member-n ; };</pre>	<pre>struct student { char name[20]; int rollno; int marks; };</pre>
Creating structure variables	struct structurename variable;	struct student stud1;
Accessing structure members	variable.member	stud1.name stud1.rollno stud1.marks
initializing a structure variable	the initialization values have to be given in {} and in order	struct student stud1 = {"ABCD",10,95};
Pointer to a structure	struct structure-name * pointer-name;	struct student *ptr; ptr = &stud1;
Accessing members using Pointer	pointer-name -> member-name;	ptr->name; ptr->rollno;
Array of structures	struct structure-name array-name[size];	struct student stud[10];
passing Structures to Functions	return-type function-name (struct structure-name variable);	void display(struct student s);
pass an array of structures to a function	return-type function-name (struct structure-name array[size]);	void display(struct student stud[10]);

Sample Code:

```
/* Program for student structure */

#include<stdio.h>
struct student
{
    char name[20];
    int rollno;
    int marks[3];
    float perc;
};
void main( )
{
    int i, sum j;
    struct student s[10];
    printf("\n Enter the details of the 10 students \n");
    for (i=0;i<10;i++)
    {
        printf("\n Enter the name and roll number \n");
        scanf("%s%d",s[i].name, &s[i].rollno);
        printf("\n Enter marks for three subjects:");
        sum = 0 ;
        for { j=0;j<3;j++)
        {
            scanf("%d",&s[i].marks[j]);
            sum = sum + s[i].marks[j];
        }
        s[i].perc = (float)sum/3;
    }
    /* Display details of students */
    printf("\n\n Name \t Roll no\t Percentage");
    printf("\n=====");
    for(i=0;i<10;i++)
    {
        printf("\n%s\t%d\t%f",s[i].name,s[i].rollno,s[i].perc);
    }
}
```



Self-Activity

1. The program in Sample code 1 demonstrates an array of structures of the type student. Type the above program and run it. Modify the program to display the details of the student having the highest percentage.

Signature of the instructor

Date / /



Assessment - work

Set A . Write C programs for the following problems.

1. Create a structure student (roll number, name, marks of 3 subjects, percentage). Accept details of n students and write a menu driven program to perform the following operations. Write separate functions for the different options.

- i) Search
- ii) Modify
- iii) Display all student details
- iv) Display all student having percentage > _____
- v) Display student having maximum percentage

2. Create a structure employee (id, name, salary). Accept details of n employees and write a menu driven program to perform the following operations. Write separate functions for the different options.

- i) Search by name
- ii) Search by id
- iii) Display all
- iv) Display all employees having salary > _____
- v) Display employee having maximum salary

Instructor should fill in the blanks with appropriate values.

Signature of the instructor

Date

Set B . Write C programs for the following problems.

1. Create a structure having the following fields:

Structure name: _____

Fields: _____, _____, _____, _____, _____, _____

Accept details of n variables of the above structure and write a menu driven program to perform the following operations. Write separate functions for the different options.

- i) _____
- ii) _____
- iii) _____
- iv) _____

2. Create a structure Fraction (numerator, denominator). Accept details of n fractions and write a menu driven program to perform the following operations. Write separate functions for the different options. Use dynamic memory allocation. Note: While accepting fractions, store the fractions in the reduced form.

- i) Display the largest fraction
- ii) Display the smallest fraction
- iii) Sort fractions
- iv) Display all

Signature of the instructor

Date

Set C. Write programs to solve the following problems

1. Accept book details of 'n' books viz, book title, author, publisher and cost. Assign an accession numbers to each book in increasing order. (Use dynamic memory allocation). Write a menu driven program for the following options.

- i. Books of a specific author
- ii. Books by a specific publisher
- iii. All books having cost >= _____ .
- iv. Information about a particular book (accept the title)
- v. All books.

2. The government of a state wants to collect census information for each city and store the following information : city name, population of the city, literacy percentage. After collecting data for all cities in the state, the government wants to view the data according to :

- i. Literacy level
- ii. Population
- iii. Details of a specific city.

Write a C program using structures and dynamic memory allocation.

Signature of the instructor

Date

Assignment Evaluation

Signature

0: Not done <input style="width: 40px; height: 20px;" type="text"/>	2: Late Complete <input style="width: 40px; height: 20px;" type="text"/>	4: Complete <input style="width: 40px; height: 20px;" type="text"/>
1: Incomplete <input style="width: 40px; height: 20px;" type="text"/>	3: Needs improvement <input style="width: 40px; height: 20px;" type="text"/>	5: Well Done <input style="width: 40px; height: 20px;" type="text"/>

Exercise 16

Start Date / /



Nested Structures and Unions



You should read the following topics before starting this exercise

1. Creating and accessing structures
2. Array of structures
3. Dynamic memory allocation
4. Structure within a structure
5. Creating and accessing unions



Nested structures: The individual members of a structure can be other structures as well. This is called nesting of structures.

Operations performed	Syntax	Example
Creating a nested structure	<pre>struct structure1 { ... struct structure2 { ... } variable; ... }; Method 2 struct structure2 { ... }; struct structure1 { ... struct structure2 variable; ... };</pre>	<pre>struct student { int rollno; char name[20]; struct date { int dd, mm, yy; } bdate, admdate; }; struct date { int dd, mm, yy; }; struct student { int rollno; char name[20]; struct date bdate, admdate; };</pre>
Accessing nested structure members	nested structure members can be accessed using the (.) operator repeatedly.	stud1.bdate.dd, stud1.bdate.mm
self referential structure	A structure containing a pointer to the same structure	<pre>struct node { int info; struct node *next;</pre>

		};
Unions	A union is a variable that contains multiple members of possibly different data types grouped together under a single name. However, only one of the members can be used at a time. They occupy the same memory area.	union u { char a; int b; };

Sample Code 1:

Example: The following structure is for a library book with the following details : id, title, publisher, code (1 – Text book, 2 – Magazine, 3 – Reference book). If the code is 1, store no-of-copies. If code = 2, store the issue month name. If code = 3, store edition number. Also store the cost.

```

/* Program for demonstrating structure and union */

struct library_book
{
    int id;
    char title[80],publisher[20] ;
    int code;
    union u
    {
        int no_of_copies;
        char month[10];
        int edition;
    }info;
    int cost;
};

void main( )
{
    struct library_book book1;
    printf("\n Enter the details of the book \n");

    printf("\n Enter the id, title and publisher \n");
    scanf("%d%s%s",&book1.id, book1.title, book1.publisher);
    printf("\n Enter the code: 1-Text Book, 2-Magazine, 3-Reference");
    scanf("%d",book1.code);
    switch(book1.code)
    {
        case 1: printf("Enter the number of copies :");
                scanf("%d",&book1.info.no_of_copies);
                break;
        case 2:  printf("Enter the issue month name :");
                scanf("%s",book1.info.month);
                break;
        case 3:  printf("Enter the edition number:");
                scanf("%d",&book1.info.edition);
                break;
    }
    printf("Enter the cost :");
    scanf("%d",&book1.cost);

    /*      Display details of book */
    printf("\n id = %d", book1.id);

```

```

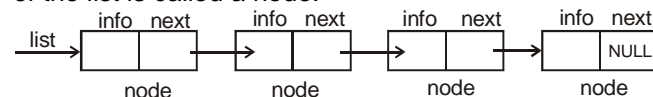
printf("\n Title = %s", book1.title);
printf("\n Publisher = %s", book1.publisher);
switch(book1.code)
{
    case 1:    printf("Copies = %d:", book1.info.no_of_copies);
               break;
    case 2:    printf("Issue month name = %s",book1.info.month);
               break;
    case 3:    printf("Edition number =%d:",book1.info.edition);
               break;
}
printf("\n Cost = %d", book1.cost);
}

```

Sample Code 2:

A linked list is a collection of data elements which are linked to one another by using pointers i.e. the every node stores the address of the next node. The advantage of using a linked list over an array is that it is easy to insert and delete elements from the list.

To create a linked list, we have to use a self referential structure (See table above). Each element of the list is called a node.



To create a node, we have to allocate memory dynamically. The following program creates 5 nodes , stores the numbers 1...5 in them and displays the data.

```

/* Program to create a linked list of 5 nodes */

#include <stdio.h>
struct node
{
    int info;
    struct node *next;
};
struct node *list = NULL; /* list is a pointer to the linked list */

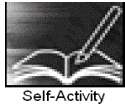
void createlist()
{
    struct node *temp, *p;
    int i;
    for(i=1;i<=5;i++)
    {
        p=(struct node *)malloc(sizeof(struct node)); /* create a node */
        p->info = i;
        p->next=NULL;
        if(list == NULL)
            list=temp=p; /* list points to the first node */
        else
        {
            temp->next=p; /* link new node to the last */
            temp=p;
        }
    }
}
void displaylist()
{
    struct node *temp;

```



```
for(temp=list; temp!=NULL; temp=temp->next) /* use a temporary pointer */
    printf("%d \t", temp->info);
}

void main( )
{
    createlist();
    displaylist();
}
```

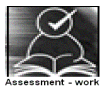


1. The sample code 1 given above demonstrates how we can create a variable of the above structure and accept and display details of 1 book. Type the program and execute it. Modify the program to accept and display details of n books.

2. The sample code 2 given above demonstrates how we can create a linked list and traverse the list. Type the program and execute it. Modify the displaylist function to display only the even numbers from the list.

Signature of the instructor

Date



Set A . Write C programs for the following problems.

1. Modify the sample program 1 above to accept details for n books and write a menu driven program for the following:

- i) Display all text books
- ii) Search Text Book according to Title
- iii) Find the total cost of all books (Hint: Use no_of_copies).

2. Modify the sample program 1 to accept details for n books and write a menu driven program for the following:

- i) Display all magazines
- ii) Display magazine details for specific month.
- iii) Find the “costliest” magazine.

3. Modify the sample program 1 to accept details for n books and write a menu driven program for the following:

- i) Display all reference books
- ii) Find the total number of reference books
- iii) Display the edition of a specific reference book.

Signature of the instructor

Date

Set B. Write programs to solve the following problems

1. Create a structure named _____ having the following fields:

Field name	Description

Write a menu driven program to perform the following operations :

i) _____ ii) _____ iii) _____ iv) _____ v) _____

2. Write a program to create a linked list of n nodes and accept data from the user for each node. Display the list. Accept a number from the user and search for the element in the list.

Signature of the instructor

Date / /

Set C. Write programs to solve the following problems

1. A shop sells electronic items. Each item has an id, company name, code (1-TV, 2-Mobile phones, 3-Camera) and cost. The following additional details are stored for each item.

- TV - size, type (CRT-1 / LCD- 2 / Plasma-3)
- Mobile Phone - type (GSM – 1 / CDMA – 2) , model number.
- Camera – resolution, model number.

The shop wants to maintain a list of all items and perform the following operations for each of the item types:

- i) Display all
- ii) Search for specific item
- iii) Sort according to cost

2. Write a program to create a linked list of n nodes and accept data from the user for each node. Write a menu driven program to perform the following operations:

- i) Display the list
- ii) Search for specific number
- iii) Display the element after _____
- iv) Find the maximum / minimum

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 17

Start Date / /



Assignment to demonstrate command line arguments and preprocessor directives.



You should read the following topics before starting this exercise

1. Passing arguments from the command line to main
2. Accessing command line arguments
3. File inclusion, macro substitution and conditional compilation directives.
4. Argumented and Nested macros



Preprocessor directives	They begin with a # which must be the first non-space character on the line. They do not end with a semicolon.	
Macro Substitution Directive	# define MACRO value	# define PI 3.142
Argumented macro	# define MACRO(argument) value	# define SQR(x) x*x #define LARGER(x,y) ((x)>(y)?(x):(y))
Nested macro	one macro using another	#define CUBE(x) (SQUARE(x)*(x))
File Inclusion directive	#include <filename> #include "filename"	#include <stdio.h>
Conditional Compilation directive	# if, # else, # elif, # endif #ifdef	#ifdef PI #undef PI #endif
Command Line Arguments	int argc - argument counter char *argv[]-argument vector	void main(int argc, char *argv[]) { printf("There are %d arguments in all", argc); for (i=0; i<argc; i++) printf("Argument %d =%s",i,argv[i]); }
To run a program using command line arguments	Compile the program using cc Execute the program using a.out followed by command line arguments	Example: a.out ABC 20 Here, ABC and 20 are the two command line arguments which are stored in the form of strings. To use 20 as an integer, use function atoi . Example: int num = atoi(argv[2]);

Sample Code 1

```
/* Program for argumented macros */
#define INRANGE(m) ( m >= 1 && m<=12)
#define NEGATIVE(m) (m<0)
#define ISLOWER(c) (c>='a'&&c<='z')
#define ISUPPER(c) (c>='A'&&c<='Z')
#define ISALPHA(c) (ISUPPER(c)||ISLOWER(c))
#define ISDIGIT(c) (c>='0'&&c<='9')

void main()
{
    int m; char c;
    printf("Enter an integer corresponding to the month");
    scanf("%d",&m);
    if(NEGATIVE(m))
        printf("Enter a positive number");
    else
        if(INRANGE(m))
            printf("You Entered a valid month");

    printf("Enter a character :");
    c=getchar();
    if(ISAPLHA(c))
        printf("You entered an alphabet");
    else
        if(ISDIGIT(c))
            printf("You Entered a digit");
}
```

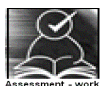


1. Write a program to display all command line arguments passed to main in the reverse order. Hint: See table above.

2. Sample code 1 above demonstrates the use of argumented and nested macros. Type the program and execute it.

Signature of the instructor

Date / /



Set A . Write C programs for the following problems.

1. Write a program to accept three integers as command line arguments and find the minimum, maximum and average of the three. Display error message if invalid number of arguments are entered.

2. Write a program which accepts a string and two characters as command line arguments and replace all occurrences of the first character by the second.

3. Define a macro EQUALINT which compares two parameters x and y and gives 1 if equal and 0 otherwise. Use this macro to accept pairs of integers from the user. Calculate the sum of digits of both and continue till the user enters a pair whose sum of digits is not equal.

4. Define a macro EQUALSTR which compares two strings x and y and gives 1 if equal and 0 otherwise. Use this macro to accept two strings from the user and check if they are equal.

Signature of the instructor

Date

Set B . Write C programs for the following problems.

1. Write a program to accept two strings as command line arguments and display the union and intersection of the strings. If the user enters invalid number of arguments, display appropriate message.

2. Write a program which accepts a string and an integer (0 or 1) as command line arguments. If the integer entered is 0, sort the string alphabetically in the ascending order and if it is 1, sort it in the descending order. If the user enters invalid number of arguments, display appropriate message. (Hint – use atoi)

Signature of the instructor

Date

Set C . Write C programs for the following problems.

1. Create a header file “mymacros.h” which defines the following macros.
 i. SQR(x) ii. CUBE(x) - nested iii. GREATER2(x,y) iv. GREATER3 (x,y,z) – nested
 v. FLAG (value = 1) (which may or may not be defined)

Include this file in your program. Write a menu driven program to use macros SQR, CUBE, GREATER2 and GREATER3. Your program should run the first two macros if the macro called FLAG has been defined. If it is not defined, execute the other two macros. Run the program twice – with FLAG defined and with FLAG not defined.

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"/>



To demonstrate text files using C



You should read the following topics before starting this exercise

1. Concept of streams
2. Declaring a file pointer
3. Opening and closing a file
4. Reading and Writing to a text file
5. Command line arguments



Operations performed	Syntax	Example
Declaring File pointer	FILE * pointer;	FILE *fp;
Opening a File	fopen("filename",mode); where mode = "r", "w", "a", "r+", "w+", "a+"	fp=fopen("a.txt", "r");
Checking for successful open	if (pointer==NULL)	if(fp==NULL) exit(0);
Checking for end of file	feof	if(feof(fp)) printf("File has ended");
Closing a File	fclose(pointer); fcloseall();	fclose(fp);
Character I/O	fgetc, fscanf fputc, fprintf	ch=fgetc(fp); fscanf(fp, "%c",&ch); fputc(fp,ch);
String I/O	fgets, fscanf fputs, fprintf	fgets(fp,str,80); fscanf(fp, "%s",str);
Reading and writing formatted data	fscanf fprintf	fscanf(fp, "%d%s",&num,str); fprintf(fp, "%d\t%s\n", num, str);
Random access to files	ftell, fseek, rewind	fseek(fp,0,SEEK_END); /* end of file*/ long int size = ftell(fp);

Sample Code 1

The following program reads the contents of file named a.txt and displays its contents on the screen with the case of each character reversed.

```

/* Program reverse case of characters in a file */

#include <stdio.h>
#include <ctype.h>
void main()
{
    FILE * fp;
    fp = fopen("a.txt", "r");
    if(fp==NULL)
    {
        printf("File opening error");
    }
}
    
```

```

        exit(0);
    }
    while( !feof(fp))
    {
        ch = fgetc(fp);
        if(isupper(ch))
            putchar(tolower(ch));
        else
            if(islower(ch))
                putchar(toupper(ch));
            else
                putchar(ch);
    }
    fclose(fp);
}

```

Sample Code 2

The following program displays the size of a file. The filename is passed as command line argument.

```

/* Program to display size of a file */
#include <stdio.h>
void main(int argc, char *argv[])
{
    FILE * fp;
    long int size;
    fp = fopen(argv[1], "r");
    if(fp==NULL)
    {
        printf("File opening error");
        exit(0);
    }
    fseek(fp, 0, SEEK_END); /* move pointer to end of file */
    size = ftell(fp);
    printf("The file size = %ld bytes", size);
    fclose(fp);
}

```

Sample Code 3

The following program writes data (name, roll number) to a file named student.txt , reads the written data and displays it on screen.

```

#include <stdio.h>
void main()
{
    FILE * fp;
    char str[20]; int num;
    fp = fopen("student.txt", "w+");
    if(fp==NULL)
    {
        printf("File opening error");
        exit(0);
    }
    fprintf(fp,"%s\t%d\n", "ABC", 1000);
    fprintf(fp,"%s\t%d\n", "DEF", 2000);
    fprintf(fp,"%s\t%d\n", "XYZ", 3000);
}

```

```

rewind(fp);
while( !feof(fp))
{
    fscanf(fp,"%s%d", str, &num);
    printf("%s\t%d\n", str, num);
}
fclose(fp);
}

```



1. Create a file named a.txt using the vi editor. Type the sample program 1 given above and execute the program. Modify the program to accept a character from the user and count the total number of times character occurs in the file.
2. Type the sample program 2 above and execute it. Modify the program to display the last n characters from the file.
3. Type the sample program 3 above and execute it. Modify the program to accept details of n students and write them to the file. Read the file and display the contents in an appropriate manner.

Signature of the instructor

Date



Set A . Write C programs for the following problems.

1. Write a program to accept two filenames as command line arguments. Copy the contents of the first file to the second such that the case of all alphabets is reversed.
2. Write a program to accept a filename as command line argument and count the number of words, lines and characters in the file.
3. Write a program to accept details of n students (roll number, name, percentage) and write it to a file named "student.txt". Accept roll number from the user and search the student in the file. Also display the student details having the highest percentage.

Signature of the instructor

Date

Set B. Write programs to solve the following problems

1. A file named numbers.txt has a set of integers. Write a C program to read this file and convert the integers into words and write the integer and the words in another file named numwords.txt.

Example:

numbers.txt	numwords.txt
11	Eleven
261	Two hundred Sixty One
9	Nine

2. Write a program which accepts a filename and an integer as command line arguments and encrypts the file using the key. (Use any encryption algorithm)

Signature of the instructor

Date

Set C . Write C programs for the following problems.

1. A text file contains lines of text. Write a program which removes all extra spaces from the file.
2. Write a menu driven program for a simple text editor to perform the following operations on a file, which contains lines of text.
 - i. Display the file
 - ii. Copy m lines from position n to p
 - iii. Delete m lines from position p
 - iv. Modify the nth line
 - v. Add n lines
3. Write a program which reads the contents of a C program and replaces all macros occurring in the program with its value. Assume only simple substitution macros (ex: #define FALSE 0).

Signature of the instructor

Date

Assignment Evaluation

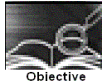
Signature

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 19

Start Date

/ /



To demonstrate binary file handling using C.



You should read the following topics before starting this exercise

1. Concept of streams
2. Declaring a file pointer
3. Opening and closing files
4. File opening modes
5. Random access to files
6. Command line arguments



In binary files, information is written in the form of binary . All data is written and read with no interpretation and separation i.e. there are no special characters to mark end of line and end of file.

I/O operations on binary files

Reading from a binary file	fread(address,size-of-element,number of elements,pointer);	fread (&num,sizeof(int),1,fp); fread (&emp,sizeof(emp),1,fp); fread(arr,sizeof(int),10,fp);
Writing to a binary file	fwrite(address,size-of-element,number of elements,pointer);	fwrite (&num,sizeof(int),1,fp); fwrite (&emp,sizeof(emp),1,fp);

Sample Code

```
/* Program to demonstrate binary file */  
  
struct employee  
{  
    char name[20];  
    float sal;  
};  
main( )  
{  
    FILE *fp;  
    struct employee e;  
    int i;  
    if((fp=fopen ("employee.in","wb"))==NULL)  
        {  
            printf("Error opening file");  
            exit( );  
        }  
  
    for(i=0;i<5;i++)  
    {  
        printf("\n Enter the name and salary");  
        scanf("%s%f",e.name,&e.sal);  
        fwrite(&e,sizeof(e),1,fp);  
    }  
}
```

```

fclose(fp);

fp=fopen("employee.in","rb"); /*      reopen file */
if(fp==NULL)
{      fprintf(stderr, "Error opening file);
exit( );
}
for(i=0;i<5;i++)
{
      fread(&e,sizeof(e),1,fp);
      printf("\n Name = %s Salary = %f",e.name,e.sal);
}
fclose(fp);
}

```



SelfActivity

1. Type program given above, writes data of 5 employees to a binary file and then reads the file. Modify the program to search an employee by name.

Signature of the instructor

Date / /



Assessment - work

Set A . Write C programs for the following problems.

1. Create a structure student (roll number, name, percentage) Write a menu driven program to perform the following operations on a binary file- "student.dat". Write separate functions for the different options.

1. Add a student (Note: Students should be assigned roll numbers consecutively)
2. Search Student
 - a. according to name
 - b. according to roll number
3. Display all students

2. Create a structure student (roll number, name, percentage) Write a menu driven program to perform the following operations on a binary file- "student.dat". Write separate functions for the different options.

1. Add a student (Note: Students will be assigned roll numbers consecutively)
2. Modify details
 - a. according to name
 - b. according to roll number
3. Display all students

3. Create a structure student (roll number, name, percentage). Write a menu driven program to perform the following operations on a binary file- "student.dat". Write separate functions for the different options.

1. Add a student (Note: Students will be assigned roll numbers consecutively)
2. Delete student
 - a. according to name
 - b. according to roll number
3. Display all students

Signature of the instructor

Date

Set B . Write C programs for the following problems.

1. Create two binary files such that they contain roll numbers, names and percentages. The percentages are in ascending orders. Merge these two into the third file such that the third file still remains sorted on percentage. Accept the three filenames as command line arguments.

2. Create a structure having the following fields:

Structure name: _____

Fields: _____

Store information for n variables of the above structure in a binary file. Write a menu driven program to perform the following operations Write separate functions for the different options.

i) _____ ii) _____ iii) _____ iv) _____

Signature of the instructor

Date

Set C . Write C programs for the following problems.

1. Create a binary file which contains details of student projects namely roll number, project name, project guide. The first line of the file contains an integer indicating the total number of students. When the program starts, read all these details into an array and perform the following menu driven operations. When the user selects Exit from the menu, store these details back into the file.

1. Add 2. Delete 3. Search 2. Modify 3. Display all 4. Exit

Signature of the instructor

Date

Assignment Evaluation

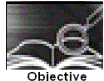
Signature

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 20

Start Date

/ /



Problem Solving Assignment



Write C programs for the following problems.

1. The calendar problem

Display a calendar for a particular year. If month-number is supplied, only that month is displayed.

2. Large number multiplication problem

Write a program that will multiply two (2) N digit positive integers, where N may be arbitrarily large. Your program should output the product(s).

Input format: One N digit positive integer per line in input and output files.

Sample Input:

```
65656432310964579864321356898765432243578987876654
94454
```

Sample Output:

```
6201512657499848426504609444515990137135009720901476916
```

3. Room IDentification Problem

A company has just finished construction of their new pentagon shaped office building. However identifying the location of a room is a problem.

Here is how the room numbering scheme works:

Room numbers will be between 100 and 99999 inclusive. The ones digit (rightmost) tells which side of the pentagon the room is on. The next one after that, the tens digit, gives the hall number in which the room is (see table). The digits after that give which floor the room is on.

0 , 1	Hall 1
2 , 3	Hall 2
4 , 5	Hall 3
6 , 7	Hall 4
8 , 9	Hall 5

The least significant digit (right-most) tells whether the room is on the courtyard or outside edge of the hall. If it is even, the room faces the courtyard, if it is odd, it faces the outside.

Input Format: The input will consist a list of room numbers not longer than 5 digits. The input ends with -1.

Output Format: You will print the Room, Floor, Hall, and Side that each room number represents.

Room r is on Floor f in Hall h facing {*courtyard*|*outside*}

Example Input:

111
1322
455
512
-1

Example Output:

Room 111 is on Floor 1 in Hall 1 facing outside
Room 1322 is on Floor 13 in Hall 2 facing courtyard
Room 455 is on Floor 4 in Hall 3 facing outside
Room 512 is on Floor 5 in Hall 1 facing courtyard

4.The Anagram Problem

An anagram is a pair of words or sentences that contain the same number of the same letters. Examples include Dormitory whose anagram is Dirty Room. You will write a program to recognize whether a pair of words or sentences are anagrams.

Input: The input accept two strings and check whether they are anagrams.

Output: If a pair of strings tested are anagrams of each other, print "An Anagram," otherwise print, "Not An Anagram."

Example Input 1:

dormitory
dirtyroom

Output:

An Anagram

Example Input 2:

eleven plus two
twelve plus one

Output:

An Anagram

Example Input 3:

thisisntananagram
andthatissuchashame

Output:

Not An Anagram

5. The Secret Word problem :

You have determined that the enemy is using the following mechanism to encode secret words. You believe that the first letters of each word in enemy messages form secret words. Only the first letters of consecutive words are used to form the secret words. Further, a sentence may contain other words before and/or after the actual words that make up the secret word. Messages always contain a single space between words.

Write a program that takes a secret word and a message as input and determines if the message contains the secret word. The program should not be case-sensitive and should ignore punctuation.

Input Format: The first line of input consists of the secret word. The second line contains the sentence to check.

Output Format: The output will be "Secret word found" if the secret word is found in the sentence, otherwise, output "Secret word not found."

Input:
year
the yellow elephant ate raw bananas

Output:
Secret word found

Input:
you
you will often fail unless you try harder

Output:
Secret word not found

6. The Credit Card Verification Problem

You are provided with a credit card number with the length varying from 13 digits to 16 digits. Each digit of the credit card is weighted by either 2 or 1. The credit card number must be zero-filled on the left to create a sixteen digit number, and then the pattern starts with 2, alternating with a 1. If the number multiplied by the weight results in a 2-digit number, each digit is added to the sum. The final sum with the check digit should be a multiple of 10.

Example:

5	4	9	9	0	0	1	1	0	0	1	2	0	0	3	4	
2	1	2	1	2	1	2	1	2	1	2	1	2	1	2	1	
1+0	+4	+1+8	+9	+0	+0	+2	+1	+0	+0	+2	+2	+0	+0	+6	+4	=40

$40 \bmod 10 = 0$. If the last digit did not result in a number divisible by 10, the credit card number is invalid.

Input Format: The user will provide you with a credit card number without spaces.

Output Format: The program will return "Valid" or "Invalid" depending on the success or failure of the check digit.

Input:
5499001100120034

Output:
Valid

Input:
5499001100120036

Output:
Invalid

7. The library problem:

Write a program to solve the following problem: A library wants a program that will calculate the due date for a book on the basis of the issue date. The no. of days for which the book is issued is decided by the librarian at the time of issuing the book. For e.g. If the librarian enters the current date as 14-01-2000 and the no of days in which the book is due as 15, then your program should calculate the due date and give the output as 29-01-2000. Your program should accept the

current date (day, month, year) and the number of issue days as input and generate the due date as output.

8. The Comment Removal problem

Write a C program which reads the contents of a file containing a C program and removes all comments from the program.

9. The Histogram problem

Write a C program which reads the contents of a text file and generates a histogram of frequencies of all alphabets in the file. Use * to draw the histogram bars.

10. The Cryptarithmic puzzle

“Cryptarithmic” puzzles are puzzles in which one gets problems like these

```
hello
+ there
-----
world
```

and is asked to assign digits to each letter so that the resulting addition is correct. Each digit from 0 to 9 must be used at most once, and the leading digits may not be 0. In the above cases, for example, we can get the solutions

```
56442
+ 15606
-----
72048
```

Write a program to find a solution to cryptarithmic problems for which the input consists of triples of strings each containing up to 128 lower-case letters and the output is in the form given in the sample below.

For the input
hello there world

Produce the output

```
hello      56442
+ there    + 15606
-----
world      72048
```

11. The Compression problem

Write a program which compresses a text file such that consecutive occurrences of specific character are replaced by the character followed by a digit indicating the number of times the character occurs. Replace only if the character occurs 3 or more times consecutively. For example, if the input text is “aath1111yy6666kkk baabbbbdg”, the output should be “aath14yy65k3 4baab4dg”. Write a decompression program which reads a compressed file and generates the original text.

12. The 4 queens problem

This problem is to place 4 queens on a 4X4 chessboard such that no two queens can attack. i.e. No two queens are on the same row, same column or diagonal. Write a program to generate all possible valid placements. One possible solution is shown below.

	Q		
			Q
Q			
		Q	

The output is a set of column numbers { c1, c2, c3, c4} such that c_j is the column number in which Queen j is placed (in row j). For the above example, the output is {2,4,1,3}. Extend your program for n queens.

Signature of the instructor

Date

Assignment Evaluation

Signature

0: Not done

2: Late Complete

4: Complete

1: Incomplete

3: Needs improvement

5: Well Done

Appendix A

1. Configuring The NFS Server

Here are the steps to configure the NFS server in this scenario:

1. Edit the `/etc/exports` file to allow NFS mounts of the `/home` directory with read/write access.

```
/home *(rw,sync)
```

2. Let NFS read the `/etc/exports` file for the new entry, and make `/home` available to the network with the `exportfs` command.

```
[root@bigboy tmp]# exportfs -a
```

3. Make sure the required `nfslock`, and `portmap` daemons are both running and configured to start after the next reboot.

```
[root@bigboy tmp]# chkconfig nfslock on
```

```
[root@bigboy tmp]# chkconfig nfs on
```

```
[root@bigboy tmp]# chkconfig portmap on
```

```
[root@bigboy tmp]# service portmap start
```

```
[root@bigboy tmp]# service nfslock start
```

```
[root@bigboy tmp]# service nfs start
```

After configuring the NFS server, we have to configure its clients, This will be covered next.

2. Configuring The NFS Client

You also need to configure the NFS clients to mount their `/home` directories on the NFS server.

1. Make sure the required `netfs`, `nfslock`, and `portmap` daemons are running and configured to start after the next reboot.

```
[root@smallfry tmp]# chkconfig nfslock on
```

```
[root@smallfry tmp]# chkconfig netfs on
```

```
[root@smallfry tmp]# chkconfig portmap on
```

```
[root@smallfry tmp]# service portmap start
```

```
[root@smallfry tmp]# service netfs start
```

```
[root@smallfry tmp]# service nfslock start
```

2. Keep a copy of the old `/home` directory, and create a new directory `/home` on which you'll mount the NFS server's directory.

```
[root@smallfry tmp]# mv /home /home.save
```

```
[root@smallfry tmp]# mkdir /home
```

3. Make sure you can mount bigboy's `/home` directory on the new `/home` directory you just created. Unmount it once everything looks correct.

```
[root@smallfry tmp]# mount 192.168.1.100:/home /home/
```

```
[root@smallfry tmp]# ls /home
```

```
[root@smallfry tmp]# umount /home
```

4. Start configuring `autofs` automounting. Edit your `/etc/auto.master` file to refer to file `/etc/auto.home` for mounting information whenever the `/home` directory is accessed. After five minutes, `autofs` unmounts the directory.

```
#/etc/auto.master
```

```
/home /etc/auto.home --timeout 600
```

5. Edit file `/etc/auto.home` to do the NFS mount whenever the `/home` directory is accessed. If the line is too long to view on your screen, you can add a `\` character at the end to continue on the next line.

```
#/etc/auto.home
```

```
* -fstype=nfs,soft,intr,rsize=8192,wsiz=8192,nosuid,tcp \
```

```
192.168.1.100:/home:&
```

6. Start `autofs` and make sure it starts after the next reboot with the `chkconfig` command.

```
[root@smallfry tmp]# chkconfig autofs on
```

```
[root@smallfry tmp]# service autofs restart
```

3. Configuring The NIS Server

1. Edit Your `/etc/sysconfig/network` File

You need to add the NIS domain you wish to use in the `/etc/sysconfig/network` file. For the school, call the domain `NISNETWORK`.

```
#/etc/sysconfig/network
NISDOMAIN="NISNETWORK"
```

2. Edit Your `/etc/yp.conf` File

NIS servers also have to be NIS clients themselves, so you'll have to edit the NIS client configuration file `/etc/yp.conf` to list the domain's NIS server as being the server itself or `localhost`.

```
#/etc/yp.conf - ybind configuration file
ypserver 127.0.0.1
```

4. Start The Key NIS Server Related Daemons

Start the necessary NIS daemons in the `/etc/init.d` directory and use the `chkconfig` command to ensure they start after the next reboot.

```
[root@bigboy tmp]# service portmap start
[root@bigboy tmp]# service yppasswdd start
[root@bigboy tmp]# service ypserv start
[root@bigboy tmp]# chkconfig portmap on
[root@bigboy tmp]# chkconfig yppasswdd on
[root@bigboy tmp]# chkconfig ypserv on
```

5. Initialize Your NIS Domain

Now that you have decided on the name of the NIS domain, you'll have to use the `ypinit` command to create the associated authentication files for the domain. You will be prompted for the name of the NIS server, which in this case is `bigboy`.

With this procedure, all nonprivileged accounts are automatically accessible via NIS.

```
[root@bigboy tmp]# /usr/lib/yp/ypinit -m
```

At this point, we have to construct a list of the hosts which will run NIS servers. `bigboy` is in the list of NIS server hosts. Please continue to add the names for the other hosts, one per line. When you are done with the list, type a `<control D>`.

```
next host to add: domainname
```

```
next host to add:
```

The current list of NIS servers looks like this:

```
domainname
```

```
Is this correct? [y/n: y] y
```

```
We need a few minutes to build the databases...
```

6. Start The `ybind` and `ypxfrd` Daemons

You can now start the `ybind` and the `ypxfrd` daemons because the NIS domain files have been created.

```
[root@bigboy tmp]# service ybind start
Listening for an NIS domain server.
[root@bigboy tmp]# service ypxfrd start
[root@bigboy tmp]# chkconfig ybind on
[root@bigboy tmp]# chkconfig ypxfrd on
```

7. Configuring The NIS Client

Now that the NIS server is configured, it's time to configure the NIS clients. There are a number of related configuration files that you need to edit to get it to work. Take a look at the procedure.

1. Run `authconfig`

The `authconfig` or the `authconfig-tui` program automatically configures your NIS files after prompting you for the IP address and domain of the NIS server.

```
[root@smallfry tmp]# authconfig-tui
```

Once finished, it should create an `/etc/yp.conf` file that defines, amongst other things, the IP address of the NIS server for a particular domain. It also edits the `/etc/sysconfig/network` file to define the NIS domain to which the NIS client belongs.

2. Start The NIS Client Related Daemons

Start the `ypbind` NIS client, and `portmap` daemons in the `/etc/init.d` directory and use the `chkconfig` command to ensure they start after the next reboot. Remember to use the `rpcinfo` command to ensure they are running correctly.

```
[root@smallfry tmp]# service portmap start
```

```
[root@smallfry tmp]# service ypbind start
```

8. Adding NIS Users

```
[root@bigboy tmp]# useradd -g users nisuser
```

```
[root@bigboy tmp]# passwd nisuser
```

```
[root@bigboy tmp]# cd /var/yp
```

```
[root@bigboy yp]# make
```

9. Initializing Database (Postgresql)

1. Steps to Initialize and configure Database

```
# chown postgres /var/lib/pgsql/data
```

```
# su - postgres
```

```
# initdb -D /var/lib/pgsql/data
```

```
# chkconfig postgresql on
```

```
# service postgresql start
```

2. Configuration Files

In `pg_hba.conf` file we have to define

```
* max_connection allowed
```

```
* port
```

`postgresql.conf` -This file contents Client Authentication Configuration. In this file we have to define Database name and database owner

References

1. Forouzan B. and Gilbert R, "Structured Programming approach using C", 2nd Edition , Thomson learning Publications
2. Brian W. Kernighan and Dennis M. Ritchie, "The C Programming Language", Second Edition, Prentice Hall, Englewood Cliffs, NJ,
3. Herbert Schildt, "The Complete Reference – C", Fourth Edition, Osborne Publications
4. Ramez Elmasri and S. Navathe, "Fundamentals of Database Systems", 4th Edition, Pearson Education
5. Abraham Silberschatz, Henry F. Korth and S. Sudarshan, "Database System Concepts", 5th Edition. McGraw-Hill
6. Raghu Ramakrishnan and Johannes Gehrke , "Database Management Systems" , McGraw-Hill
7. Sumitabha Das, " UNIX Concepts and Applications" Tata Mcgraw Hill
8. Practical PostgreSQL, O'Reilly Publications
9. MS-DOS Manual