

# **The University of Lahore**



## **Computer Lab Manual**

*For*

## **Introduction to Information & Communication Technologies**

**(CS-09101)**

Course Instructor	Mr. Usman Asghar
Lab Instructor(s)	Mr. Usman Asghar
Section	F
Semester	Fall 2021

Department of Computer Science & Information Technology  
The University of Lahore, Lahore, Pakistan

## CONTENT

01. MS Word – Basics	03
02. MS Word – Formatting and Advanced	06
03. MS Excel – Basics	10
04. MS Excel – Conditional Formatting	16
05. MS Excel – Formulas	18
06. MS Excel – What If & Goal Seek Analysis	20
07. MS PowerPoint – Basics and Advanced	23
08. HTML5 and Lists	27
09. HTML5 – Nested List	32
10. HTML5 – Audio/Video	34
11. HTML5 – Tables	37
12. HTML5 – Forms	42
13. Introduction to Programming in C++	46
14. Variables, Constants & Data Types	52
15. Operators (Relational & Logical)	57
16. Conditional Statements	59

## LAB - 1

### MS Word – Basics

#### Objective:

After performing this lab, students should have knowledge of:

- ❖ Microsoft Word working environment.
- ❖ Modifying a Word document.
- ❖ How MS Word works.
- ❖ Text Editing & Text Formatting

#### Microsoft Word Working Environment:

Microsoft Word has evolved over the years as a typical word processing program into a document management powerhouse, you can use Word to create documents that combine text with graphics, sounds, or multimedia objects, and you can publish those documents on paper, on the World Wide Web, via e-mail, fax, or other electronic media.

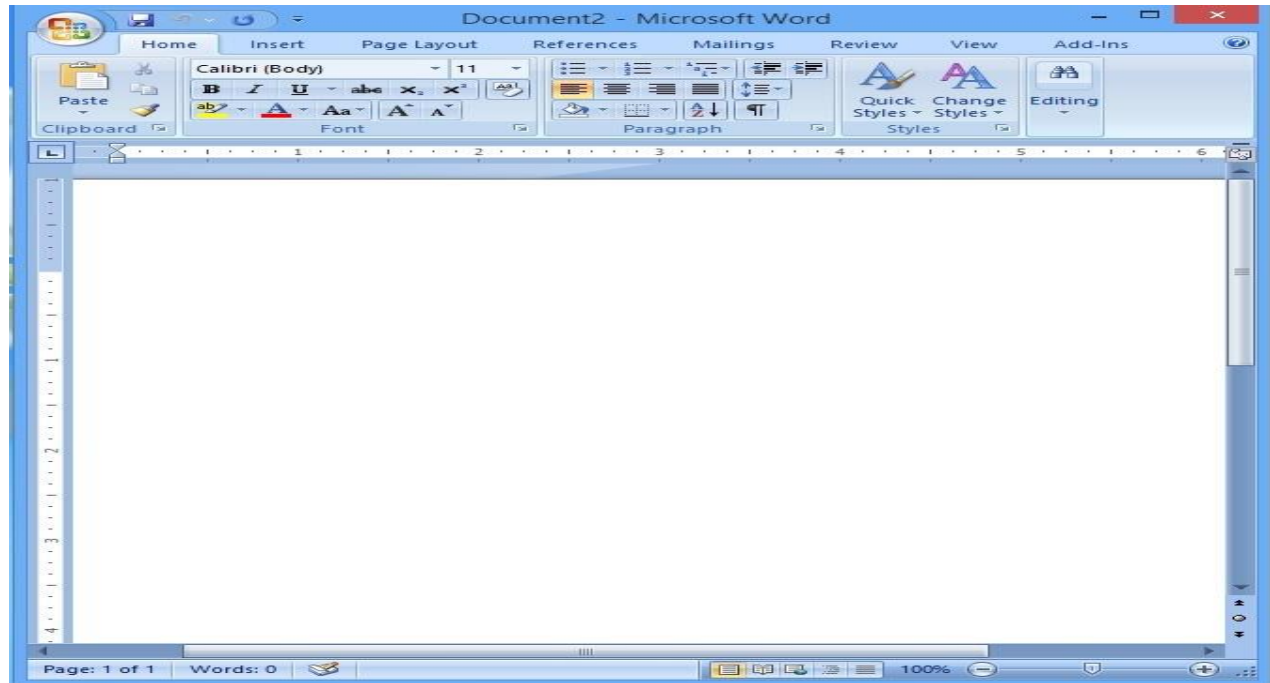
#### Getting Started:

#### Opening Microsoft Word:

You may have a shortcut to Word on your desktop, if so double click the icon and Word will open. If not follow the steps below:

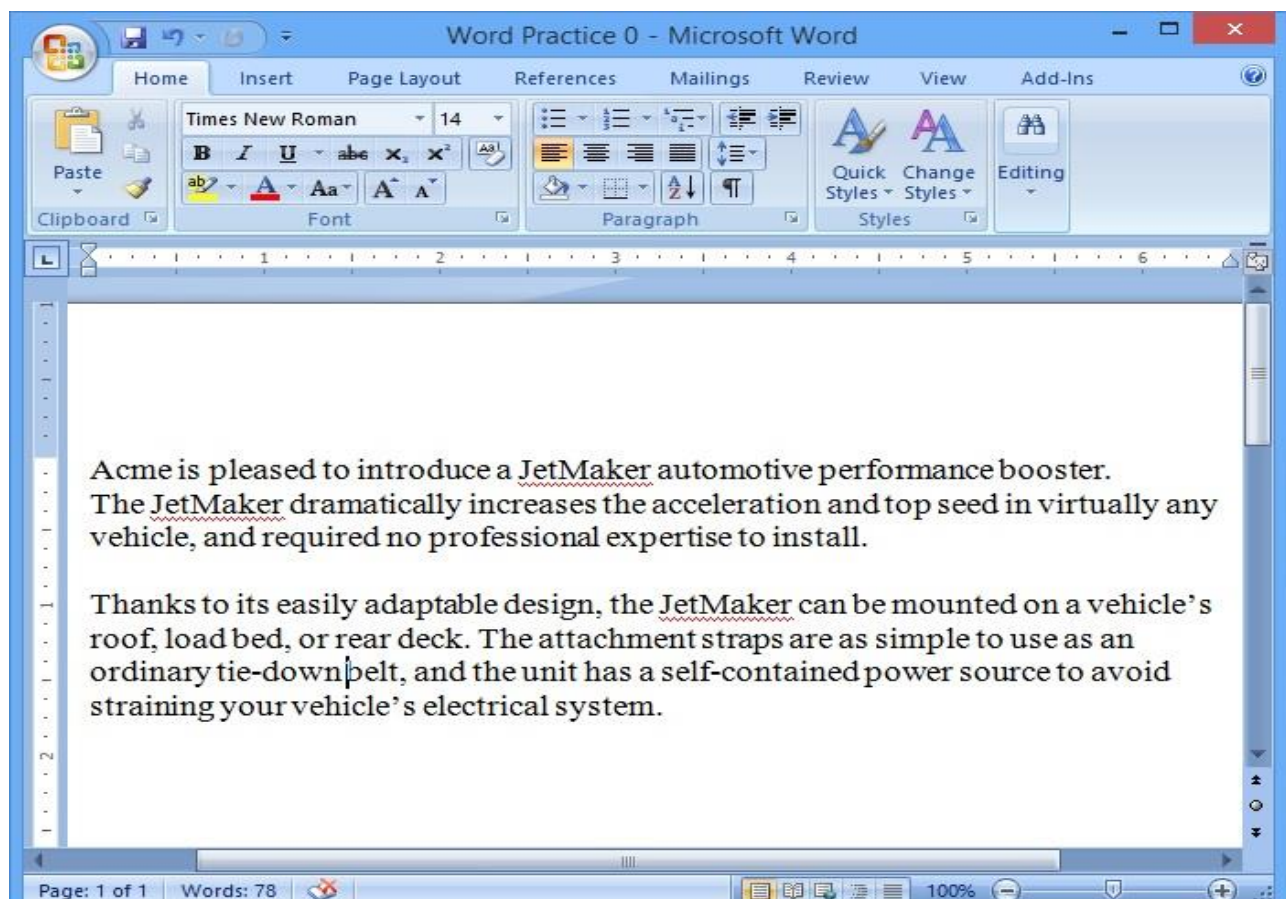
1. Click on the Start Button.
2. Highlight Programs.
3. Highlight Microsoft Office.
4. Click on Microsoft Word 2007.

A window like the following will appear:



### Lab Task 1:

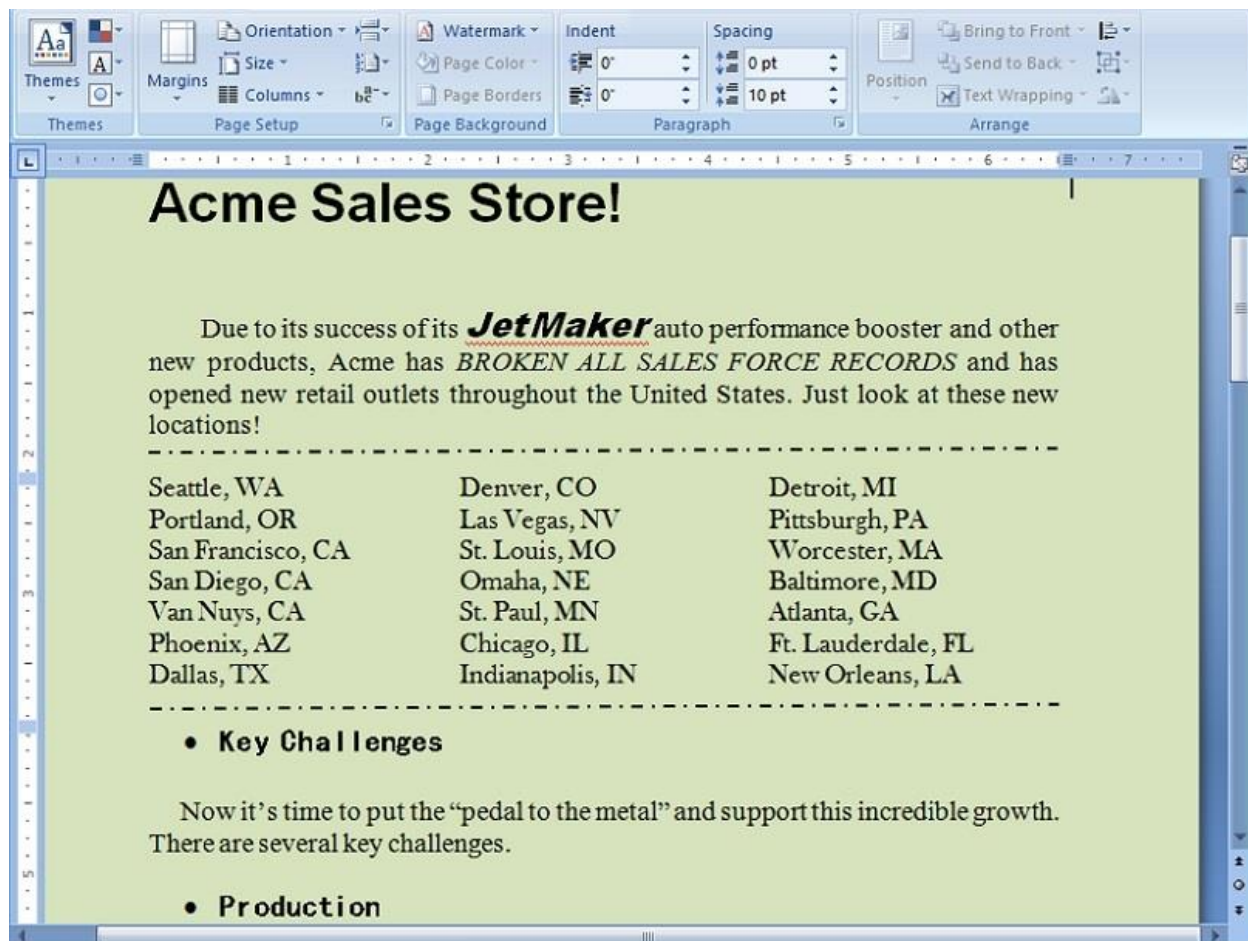
Create a simple draft.



## Lab Task 2:

Create a document given below, with following requirement:

- The document must have table.
- Merge Rows and Columns
- Add Bullets



## LAB - 2

### MS Word – Formatting

#### Bullets and Numbering:

Home → Paragraph → Bullets (Specify sign present specify sign)

- ❖ Templates
- ❖ Define new bullets
  - Symbol → Choose font style → ok
  - Picture → Import (for other picture) → ok
  - Font (For formatting)

Home → Paragraph → Numbering (Create outline with numbering style)

- ❖ Templates
- ❖ Define new numbering
  - More style
  - Font

#### Border and Shading:

Home → Paragraph → Border and Shading

- ❖ Border
  - Border Style
  - Border Line Style
  - Line Color
  - Line Width
- ❖ Page Border
  - Art
- Shading
  - Color

#### Page color:

Page Layout → Page color

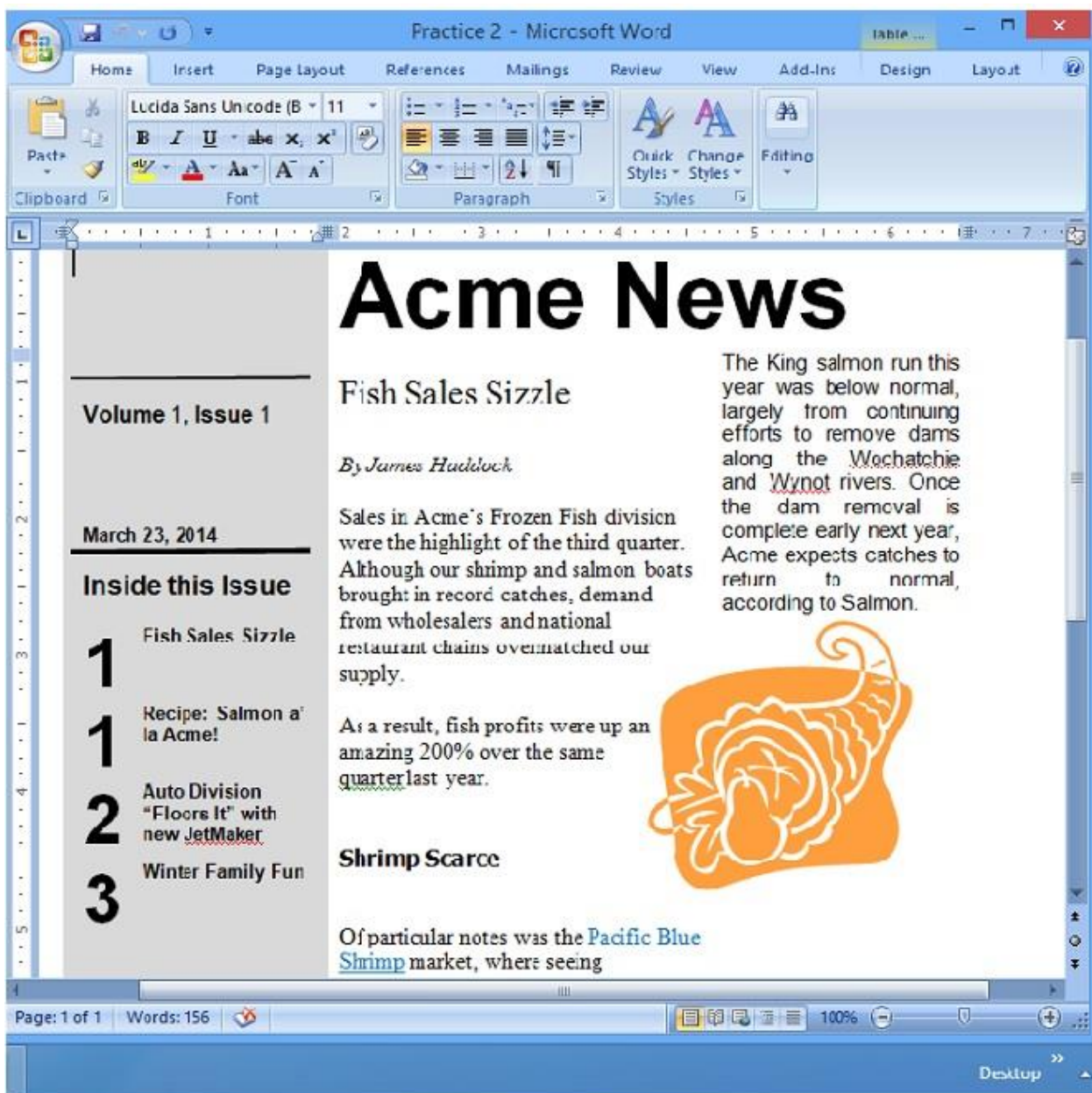
- Solid Color
- Custom Color
- More color



## Lab Task:

Create a document same as given below, with given below requirements:

- The document has three columns.
- The document must have a picture from Clip Art.
- The document must have different font families.
- The document has a date, which updates it automatically.
- The document must have some links to external web pages.



## MS Word – Advanced

### How to insert any object?

Insert → shape → Drag and Drop

3 types of node to manage Object

1. Blue Node (Re-Size)
2. Green Node (Rotate)
3. Yellow Node (Re-Shape)

*Select Object automatically show format menu*

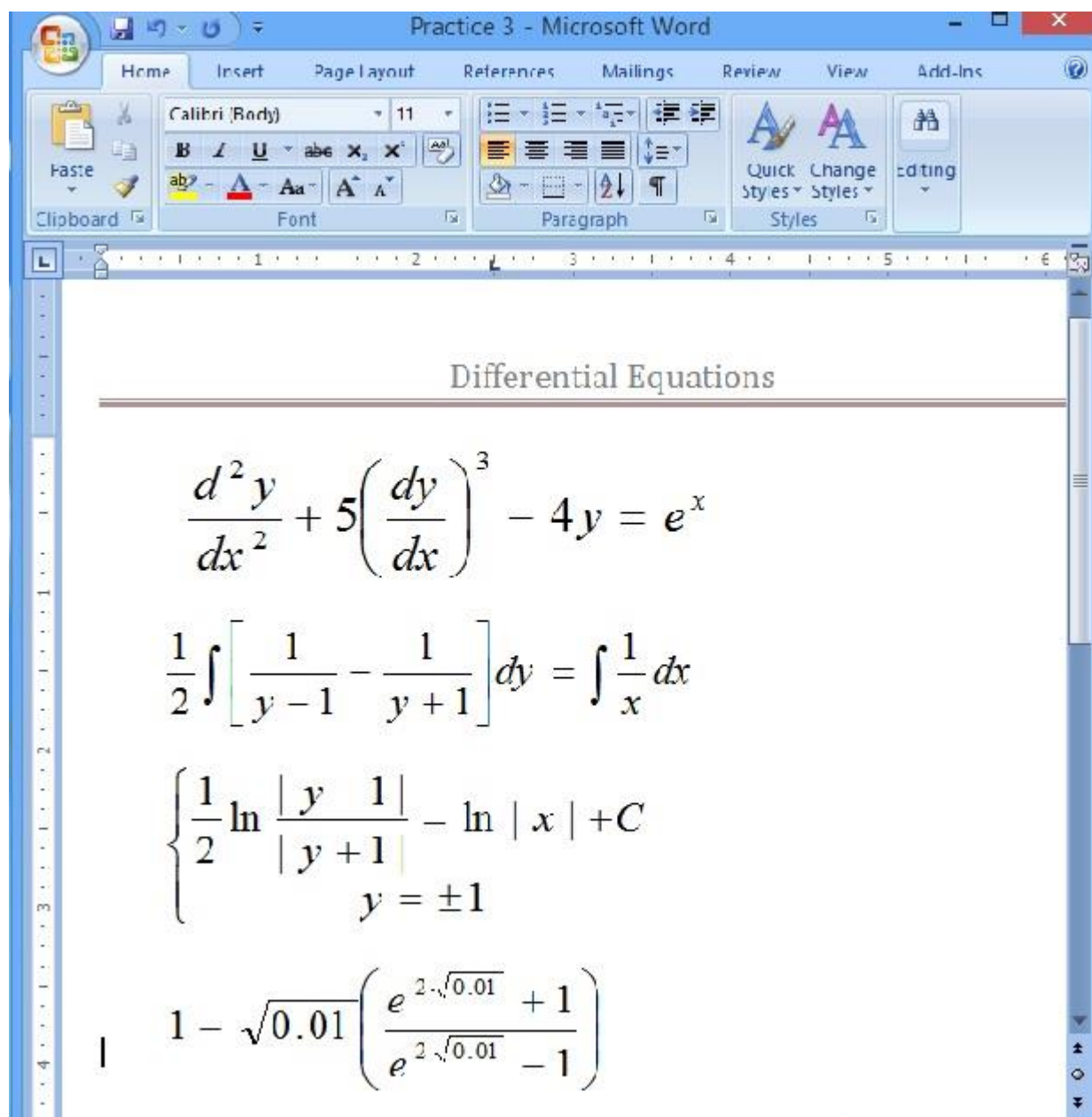
- ❖ Insert Shape
- ❖ Text box
- ❖ Style / Templates
- ❖ Fill Shape
  - Color
  - More color
  - Picture
  - Gradient
    - One color
    - Two color
    - Presets
  - Texture
  - Pattern lines
- ❖ Line Shape
  - Line Color
  - Line Width
  - Dash Style
- ❖ Change Shape
- ❖ Object Position



## Lab Task:

Create a document given below:

- Implement and write mathematical equations.
- Apply header and footer.



## LAB - 3

### MS Excel – Basics

#### Objectives

After performing this lab, students should have the knowledge of:

- ❖ Microsoft Excel's working environment
- ❖ Modifying an spreadsheet
- ❖ Working with Formulas
- ❖ Designing Charts
- ❖ What If Analysis
- ❖ Goal Seek Analysis

#### Microsoft Excel's Working Environment

Excel is a spreadsheet program in the Microsoft Office system. You can use Excel to create and format workbooks (a collection of spreadsheets) in order to analyze data and make more informed business decisions. Specifically, you can use Excel to track data, build models for analyzing data, write formulas to perform calculations on that data, pivot the data in numerous ways, and present data in a variety of professional looking charts.

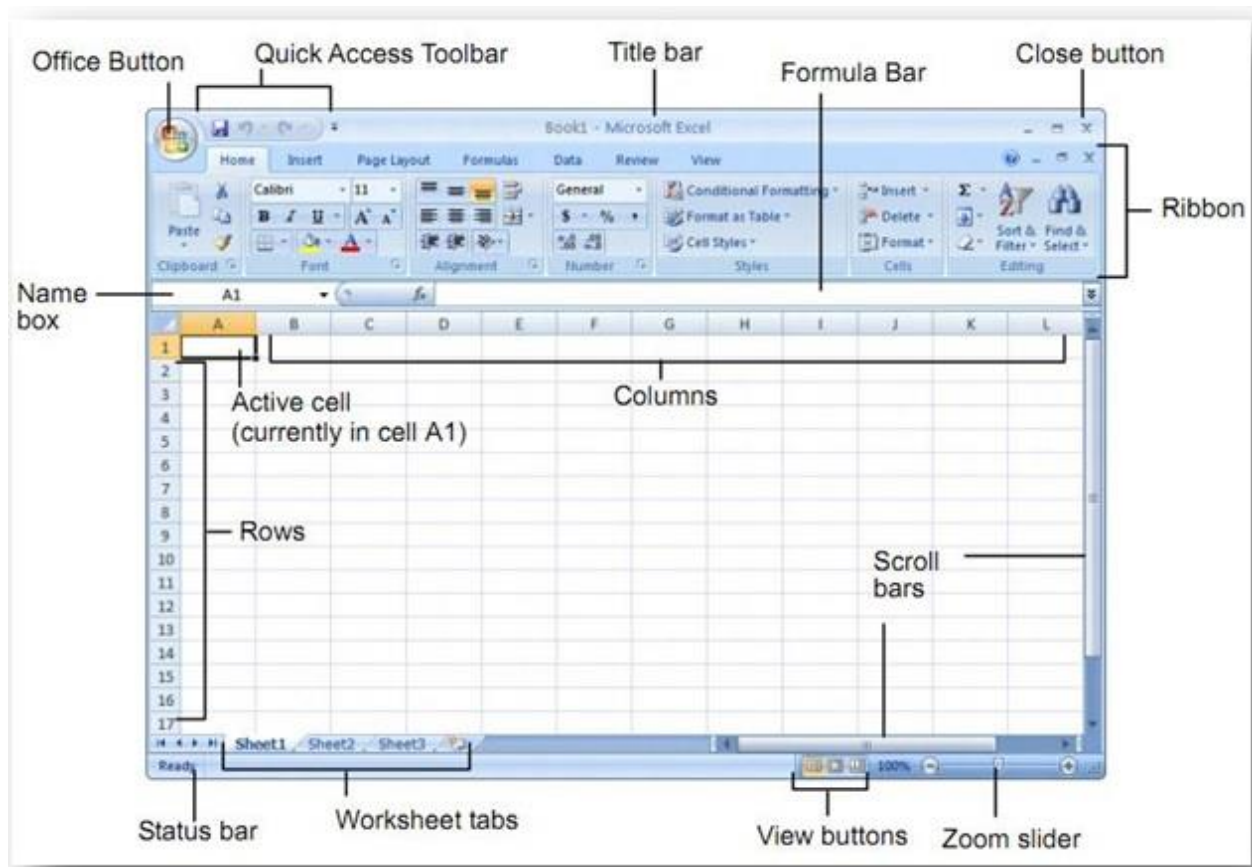
#### Getting Started:

##### Opening Microsoft Excel:

You may have a shortcut to Excel on your desktop, if so double click the icon and Excel will open. If not follow the steps below:

1. Click on the Start button
2. Highlight Programs
3. Highlight Microsoft Office
4. Click on Microsoft Excel 2007

A window like the following will appear:

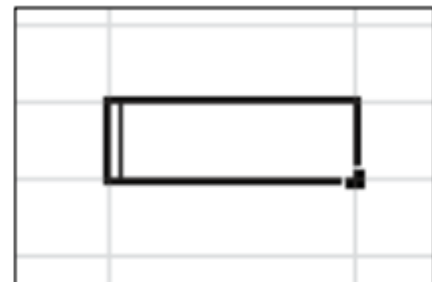


## Create a New Workbook

1. Click the **File** tab and then click **New**.
2. Under **Available Templates**, double click **Blank Workbook** or Click **Create**.

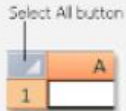
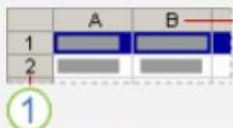
## Enter Data in Worksheet

1. Click the cell where you want to enter data.
2. Type the data in the cell.
3. Press enter or tab to move to the next cell



## Select Cells and Ranges

In order to complete more advanced processes in Excel you need to be able to highlight or select cells, rows and columns. There are a variety of ways to do this, see the table below to understand the options.

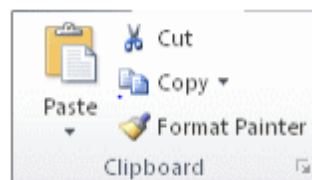
To select	Do this
A single cell	Click the cell, or press the arrow keys to move to the cell.
A range of cells	Click the first cell in the range, and then drag to the last cell, or hold down SHIFT while you press the arrow keys to extend the selection.
A large range of cells	Click the first cell in the range, and then hold down SHIFT while you click the last cell in the range. You can scroll to make the last cell visible.
All cells on a worksheet	 <p>Click the Select All button or press CTRL+A.</p>
Nonadjacent cells or cell ranges	<p>Select the first cell or range of cells, and then hold down CTRL while you select the other cells or ranges.</p> <p><b>NOTE:</b> You cannot cancel the selection of a cell or range of cells in a nonadjacent selection without canceling the entire selection.</p>
An entire row or column	 <p>Click the row or column heading.</p> <ul style="list-style-type: none"> <li>1 Row heading</li> <li>2 Column heading</li> </ul>
Adjacent rows or columns	Drag across the row or column headings. Or select the first row or column; then hold down SHIFT while you select the last row or column.
Nonadjacent rows or columns	Click the column or row heading of the first row or column in your selection; then hold down CTRL while you click the column or row headings of other rows or columns that you want to add to the selection.
Cells to the last used cell on the worksheet (lower-right corner)	Select the first cell, and then press CTRL+SHIFT+END to extend the selection of cells to the last used cell on the worksheet (lower-right corner).
Cells to the beginning of the worksheet	Select the first cell, and then press CTRL+SHIFT+HOME to extend the selection of cells to the beginning of the worksheet.
<b>NOTE:</b> To cancel a selection of cells, click any cell on the worksheet. This is not applicable to cells with formulas in it.	

## Modifying Spreadsheets

In order to create an understandable and professional document you will need to make adjustments to the cells, rows, columns and text. Use the following processes to assist when creating a spreadsheet.

## Cut, Copy, and Paste Data

You can use the Cut, Copy, and Paste commands in Microsoft Office Excel to move or copy entire cells or their contents. NOTE: Excel displays an animated moving border around cells that have been cut or copied. To cancel a moving border, press ESC.



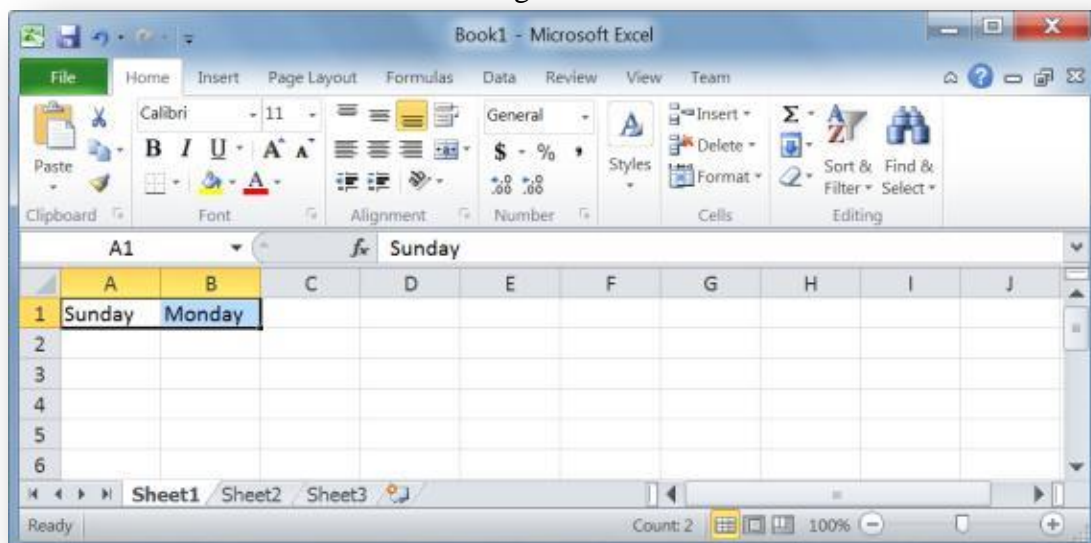
## Lab Task 1:

1. Enter the data “Sunday” into cell A1 and “Monday” into cell B1.
2. Type in “17/08” into cell E8.
3. Type in “2” into cell I8 and “4” into cell I9.

	A	B	C	D	E	F	G	H	I
1	Sunday	Monday							
2									
3									
4									
5									
6									
7									
8					17-Aug				2
9									4
10									
11									
12									

Your worksheet should now look like this:

Notice how Excel automatically detected that 17/08 was a date and converted it to 17-Aug. We will discuss formatting data later on in this lab. Now, we want to select both cells A1 and B1 together. To do this, click A1 and without releasing the mouse button, move the mouse over cell B1. Now there should be a rectangle around both cells as shown below



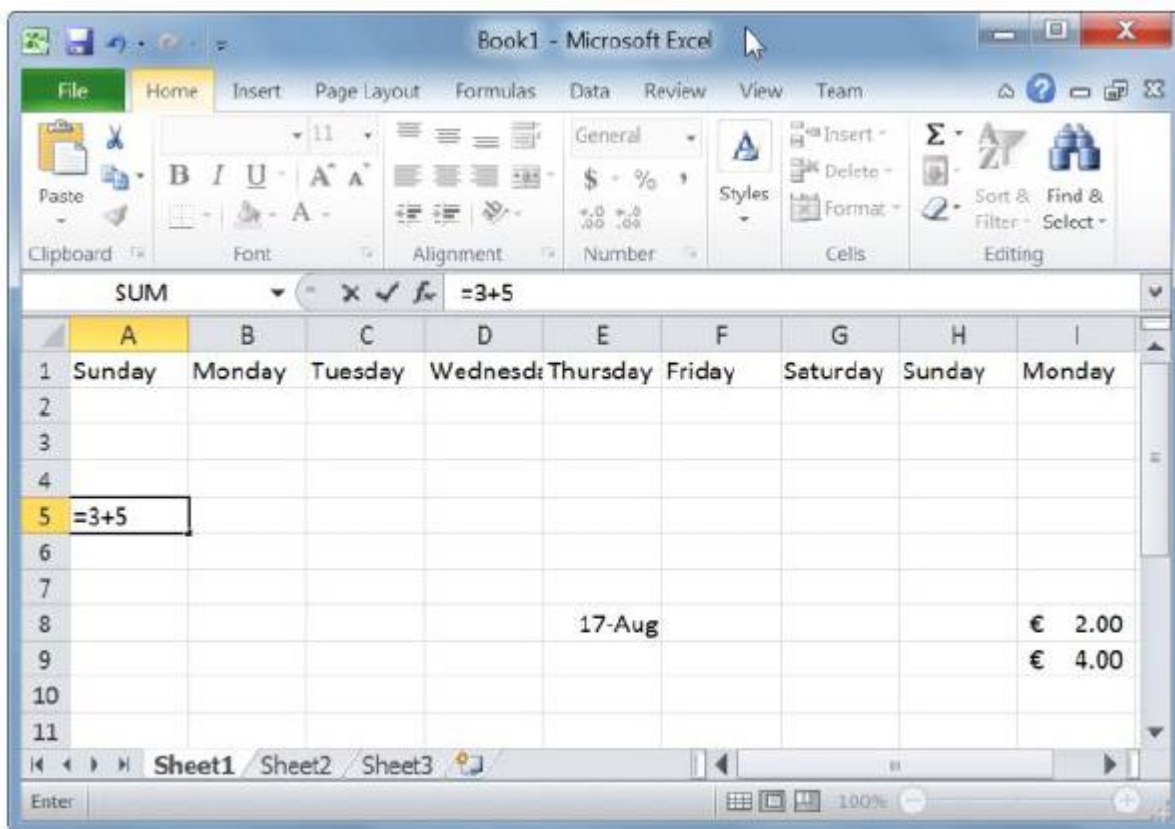


## Lab Task 2:

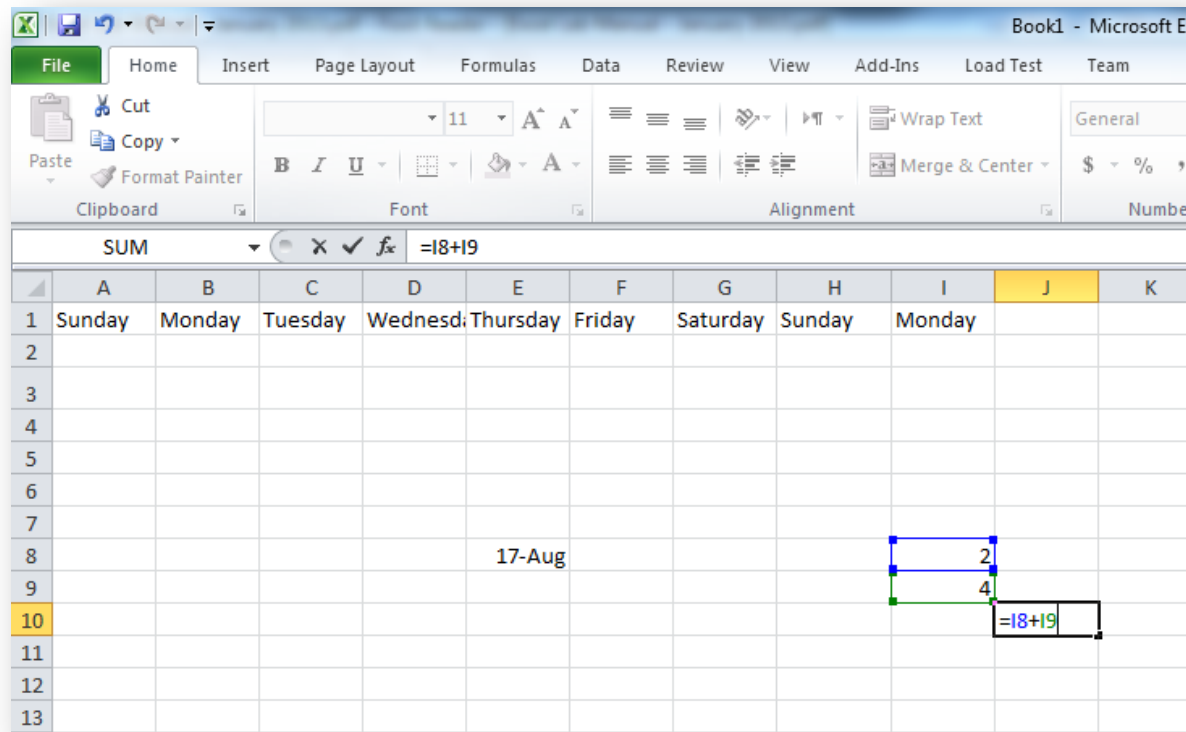
1. Auto-complete cells I8 and I9 all the way to I14.
2. Auto-complete cell E8 all the way to E12.

## Basic Calculations:

When working on a spreadsheet, you will almost definitely need to perform some calculations on the data you have. The first thing you need to remember about Excel calculations is that formulas always start with an = sign. Let us begin with a very simple calculation. Type “=3+5” into cell A5 as shown below. Press Enter. Excel automatically replaces the formula with the result of the equation.



Now let us calculate the sum of the numbers in I8 and I9. In cell J10, type “=I8+I9”. One

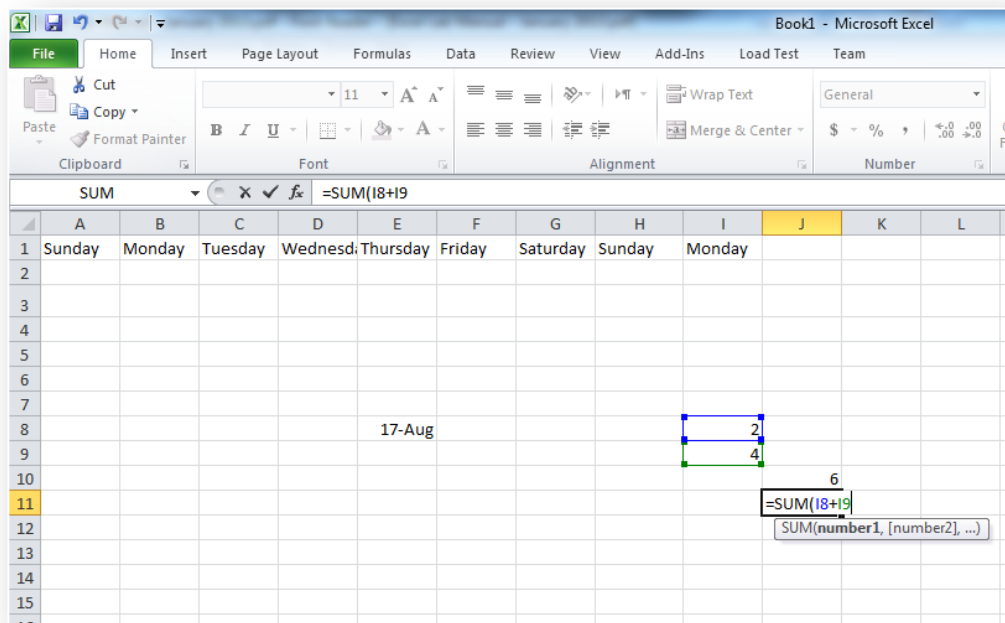


other option is to type in “=”, then select cell I8. After that, type in “+” and then select I9.

Pressing Enter will give you the result of the calculation.

Double clicking on the cell with the formula allows you to edit the formula.

Excel has built-in functions that make your life easier. One of them is the SUM function. In cell J11, type “=sum(“. Now select both cells I8 and I9.



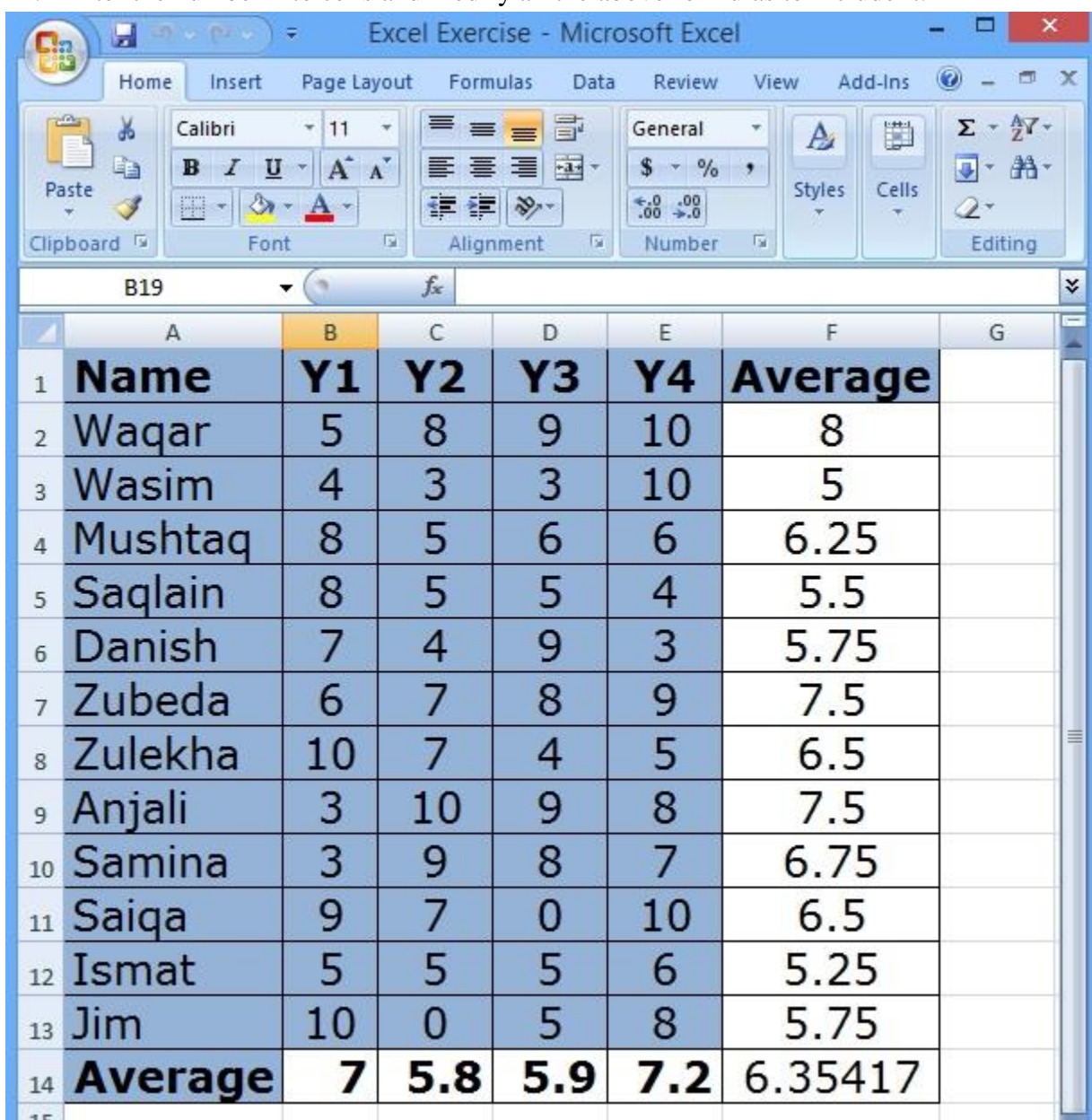


## LAB - 4

### MS Excel – Conditional Formatting

#### Lab Task 1:

1. Open Sheet 1 in your workbook.
2. Use auto-complete to fill in cells A1 to F13.
3. Calculate the following values for cells A1 to A8 using built-in Excel functions:  
a) Average
4. Enter the number into cells and modify all the above formulas to include it.



	A	B	C	D	E	F	G
1	<b>Name</b>	<b>Y1</b>	<b>Y2</b>	<b>Y3</b>	<b>Y4</b>	<b>Average</b>	
2	Waqar	5	8	9	10	8	
3	Wasim	4	3	3	10	5	
4	Mushtaq	8	5	6	6	6.25	
5	Saqlain	8	5	5	4	5.5	
6	Danish	7	4	9	3	5.75	
7	Zubeda	6	7	8	9	7.5	
8	Zulekha	10	7	4	5	6.5	
9	Anjali	3	10	9	8	7.5	
10	Samina	3	9	8	7	6.75	
11	Saiqa	9	7	0	10	6.5	
12	Ismat	5	5	5	6	5.25	
13	Jim	10	0	5	8	5.75	
14	<b>Average</b>	<b>7</b>	<b>5.8</b>	<b>5.9</b>	<b>7.2</b>	6.35417	
15							

## Lab Task 2:

1. Open Sheet 2 in your workbook.
2. Use auto-complete to fill in cells A1 to F13.
3. Calculate the following values for cells A1 to F13 using built-in Excel functions:
  - a) Average
4. Sort the Average in ascending order.
5. Apply conditional formatting on the data.

	A	B	C	D	E	F	G
1	<b>Name</b>	<b>Y1</b>	<b>Y2</b>	<b>Y3</b>	<b>Y4</b>	<b>Average</b>	
2	Wasim	4	3	3	10	5	
3	Ismat	5	5	5	6	5.25	
4	Saqlain	8	5	5	4	5.5	
5	Danish	7	4	9	3	5.75	
6	Jim	10	0	5	8	5.75	
7	Mushtaq	8	5	6	6	6.25	
8	Zulekha	10	7	4	5	6.5	
9	Saiqa	9	7	0	10	6.5	
10	Samina	3	9	8	7	6.75	
11	Zubeda	6	7	8	9	7.5	
12	Anjali	3	10	9	8	7.5	
13	Waqar	5	8	9	10	8	
14	<b>Average</b>	<b>6.5</b>	<b>5.8</b>	<b>5.9</b>	<b>7.2</b>	<b>6.35417</b>	

## LAB - 5

### MS Excel – Formulas

#### Lab Task 1:

1. Open Sheet 2 in your workbook.
2. Use auto-complete to fill in cells A1 to D13.
3. Calculate the following values for cells A1 to D13 using built-in Excel functions:
  - a) Sum
  - b) CountIF
4. Enter the numbers in the cell and modify all the above formulas to include it.
5. Calculate the sum count the students w.r.t. age, and count them w.r.t. their GPA.

The screenshot shows an Excel worksheet titled "Excel Exercise - Microsoft Excel". The worksheet contains a table of student data and two summary tables. The student data table has columns for Student ID, Gender, Age, and GPA. The summary tables show the number of students by age and by GPA range.

Student ID	Gender	Age	GPA
101	m	14	3.5
102	m	15	3.35
103	f	15	3.25
104	m	13	1.5
105	f	15	2.5
106	m	14	1.9
107	f	16	2.5
108	m	15	3.75
109	f	14	3.5
110	m	13	1

No. of Students by Age	
13 Years	2
14 Years	3
15 Years	4
16 Years	1
Total	10

No. of Students by GPA	
3.5 - 4.0	3
3.0 - 3.5	2
2.0 - 3.0	2
1.0 - 2.0	3
0.0 - 1.0	0
Total	10



## Lab Task 2:

1. Open Sheet 3 in your workbook.
2. Calculate the following values for cells A1 to A8 using built-in Excel functions:
  - a) StDev
  - b) Average
  - c) Median
  - d) Maximum
  - e) Minimum
  - f) Mode
  - g) Count
  - h) AveDev
3. Enter the numbers in the cell and modify all the above formulas to include it.

The screenshot shows an Excel worksheet titled 'Excel Exercise - Microsoft Excel'. The worksheet contains a table of student data and a summary of statistical calculations.

Student ID	Gender	Age	GPA
101	m	14	3.5
102	m	15	3.35
103	f	15	3.25
104	m	13	1.5
105	f	15	2.5
106	m	14	1.9
107	f	16	2.5
108	m	15	3.75
109	f	14	3.5
110	m	13	1

Average GPA by Gender	
m	2.5
f	2.9375
Average	2.71875

Statistical Calculations	
Stdev	0.951679801
Average	2.675
Median	2.875
Max	3.75
Min	1
Mode	3.5
Count	10
Avedev	0.795

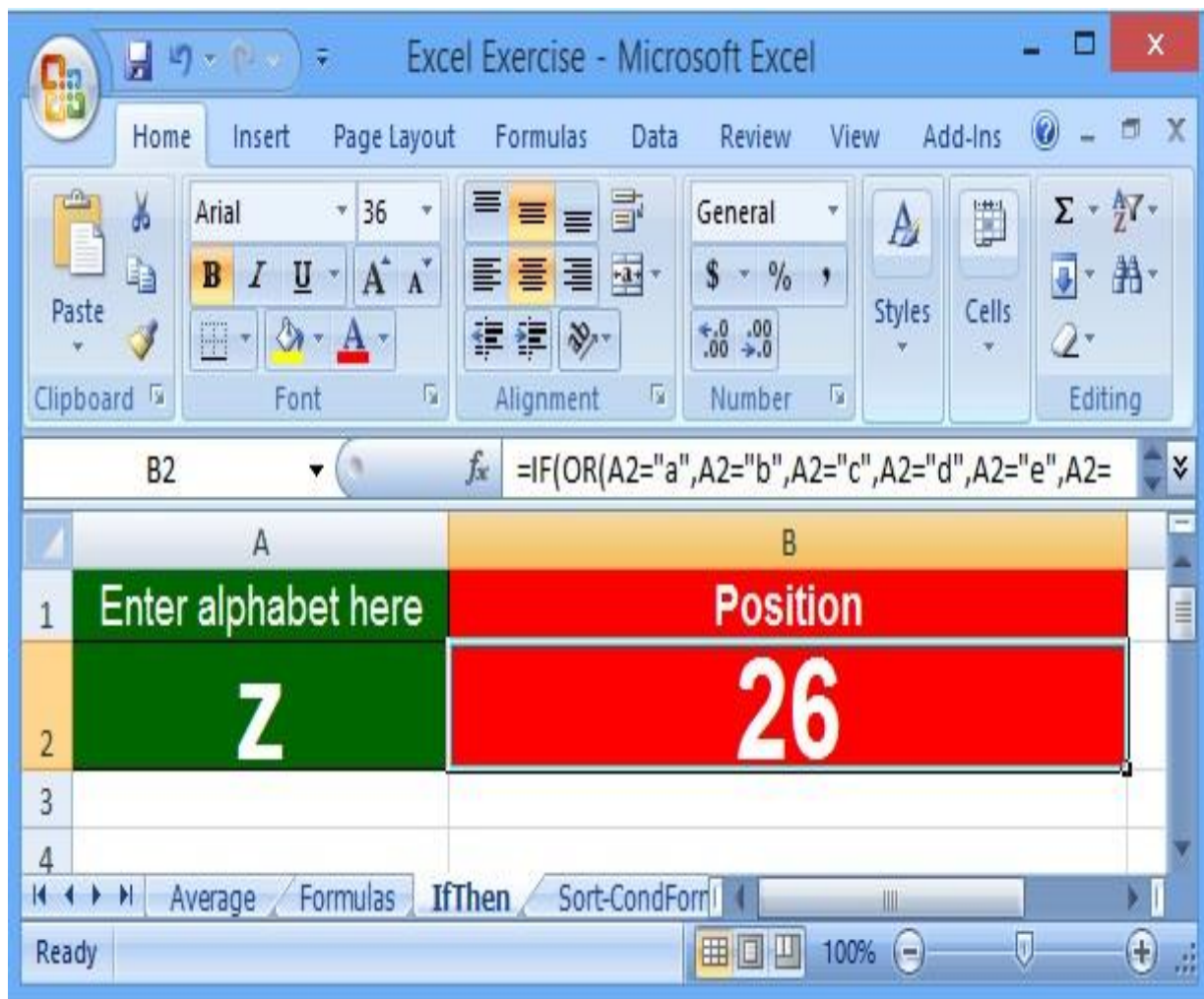
The status bar at the bottom shows: Average: 40.85833333, Count: 45, Sum: 1225.75.

## LAB - 6

### MS Excel – What-If & Goal Seek Analysis

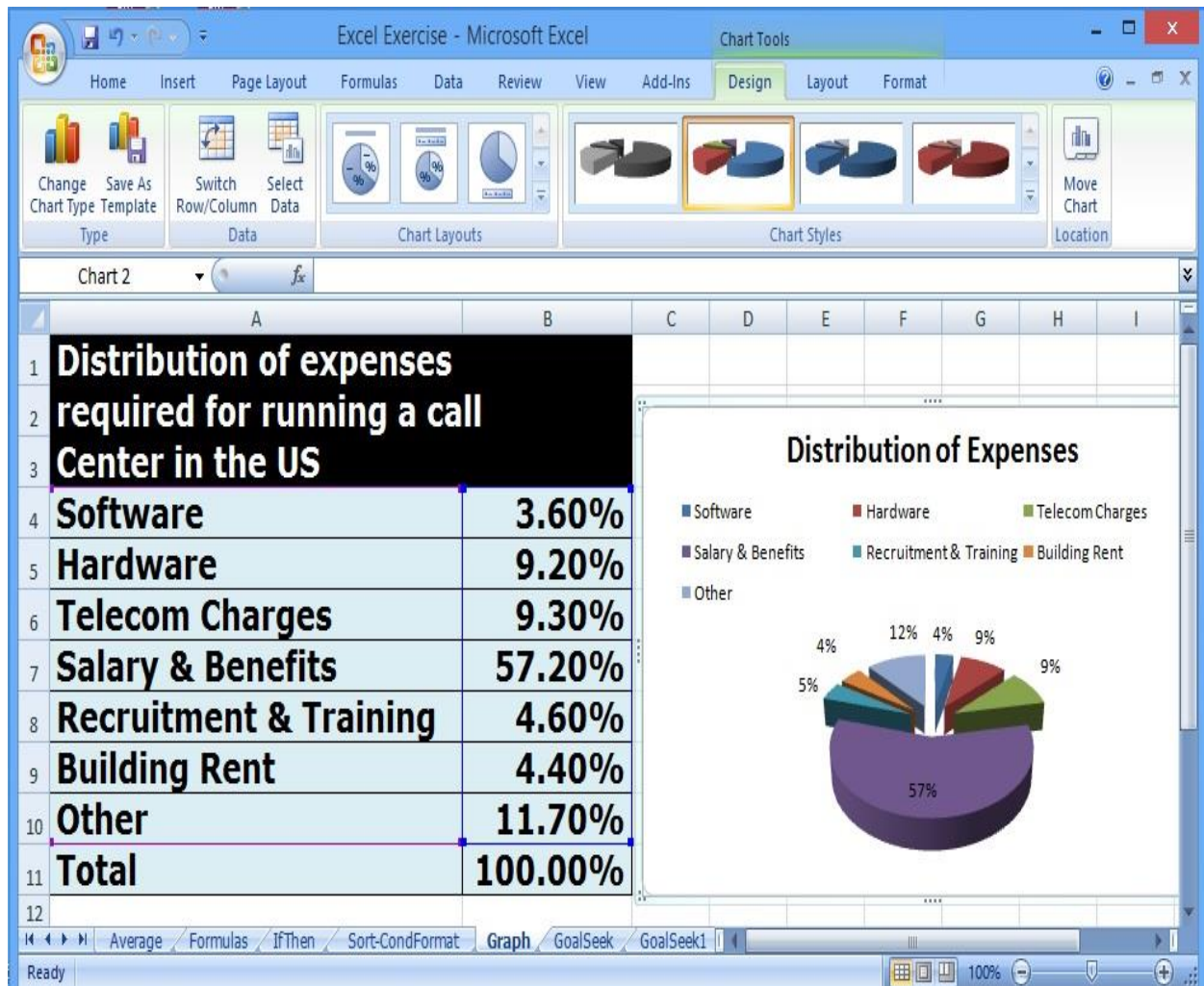
#### Lab Task 1:

1. Open Sheet 1 in your workbook.
2. Make a program which tells the position of alphabets of English.



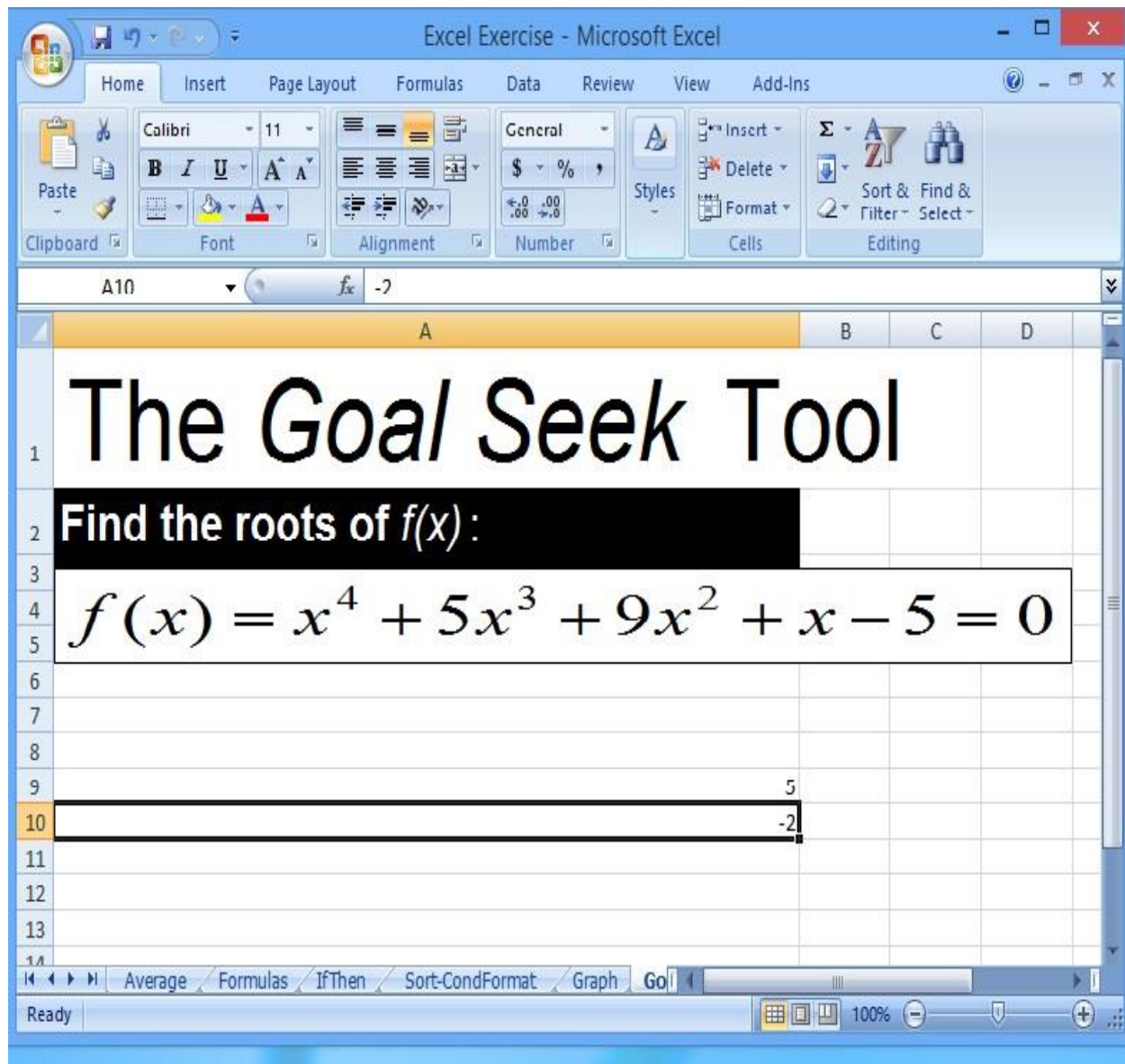
## Lab Task 2:

1. Open Sheet 1 in your workbook.
2. Use auto-complete to fill in cells A1 to B11.
3. Create a Pie-Chart from this data.



### Lab Task 3:

1. Open Sheet 2 in your workbook.
2. Use auto-complete to fill in cells A1 to B11.
3. Apply Goal Seek Analysis to find roots of a equation.





## LAB - 7

### MS PowerPoint - Basics

#### Microsoft PowerPoint Working Environment

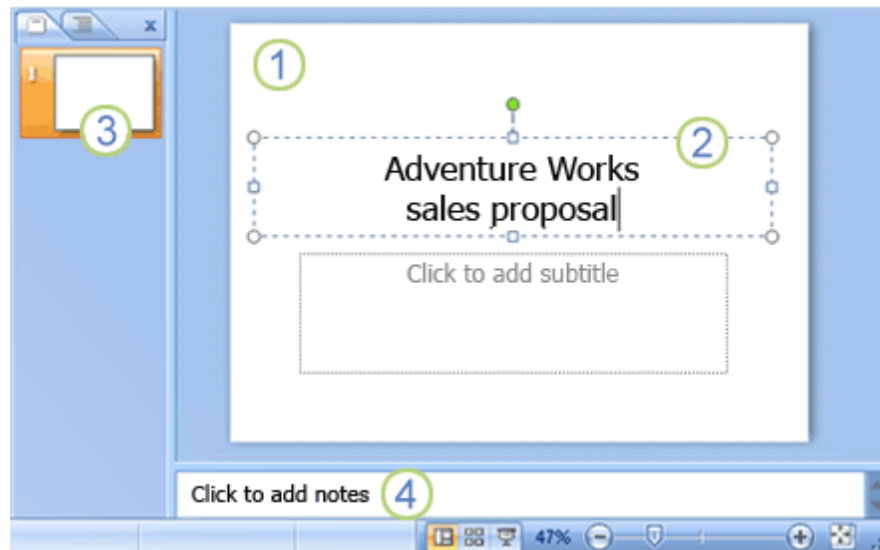
PowerPoint is a system in the Microsoft Office Suite that enables you to present information in office meetings, lectures and seminars to create maximum impact in a minimal amount of time. PowerPoint presentations can amplify your message, accelerate the information being absorbed and assist with comprehension enabling faster decision making.

#### Getting Started:

#### New PowerPoint Document:

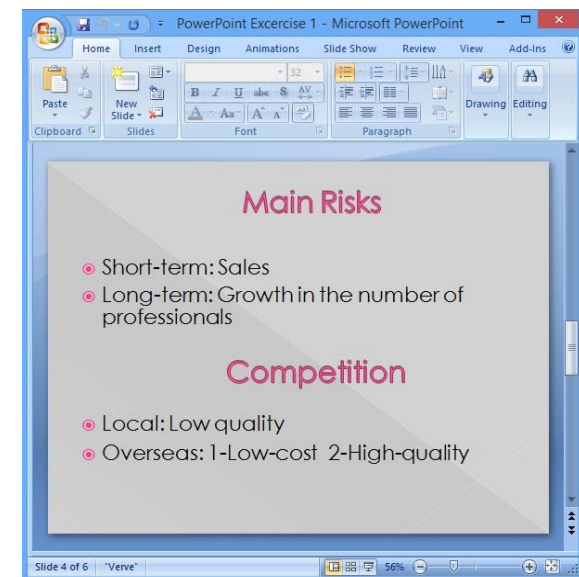
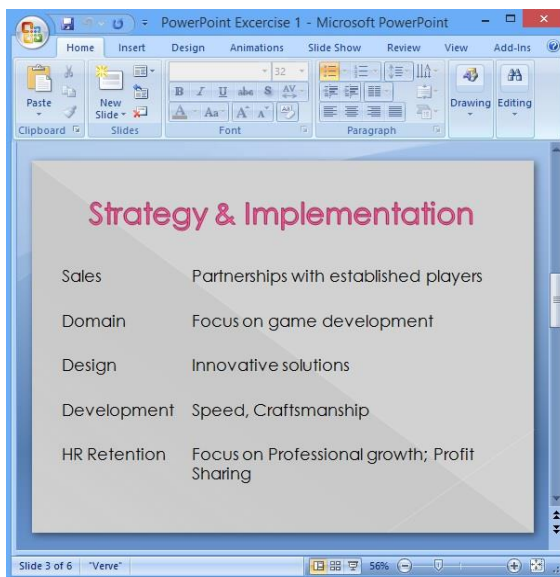
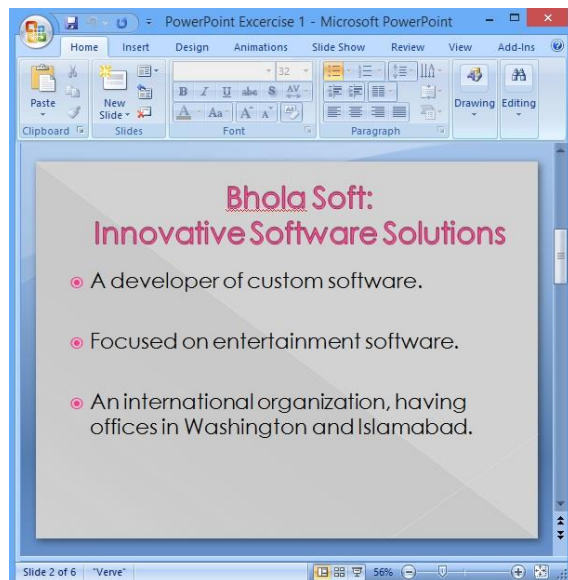
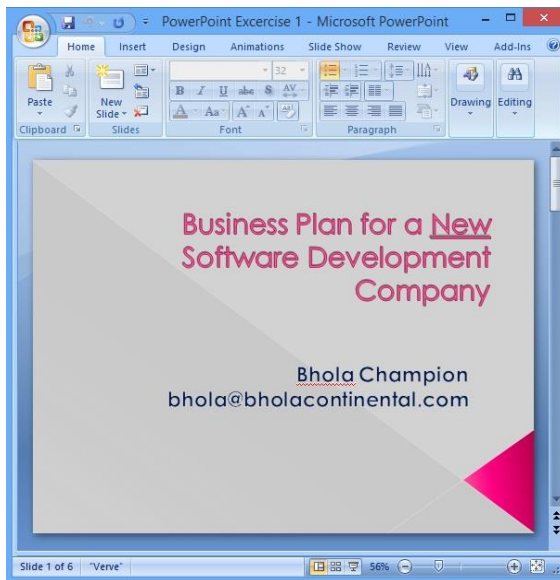
When you first open PowerPoint you will see what's called the **Normal** view.

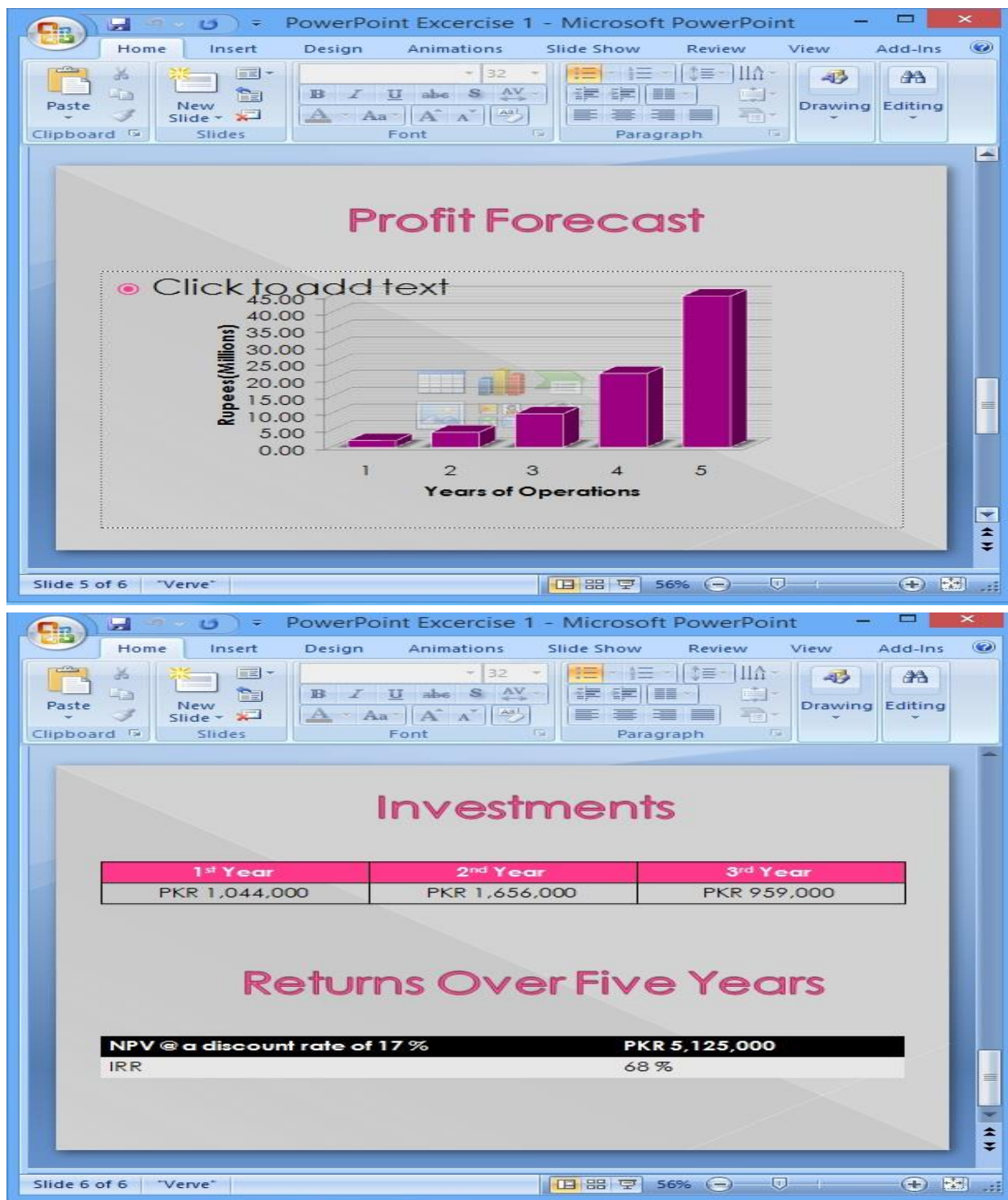
1. The **slide pane** is the big area in the middle. This is the area you will work in to create your slides.
2. On each slide, you will see various boxes with the dotted borders which are called **placeholders**. This is where you type your text. Placeholders can be customized to different sizes and can contain pictures, charts, and other non-text items.
3. On the left of the screen are **thumbnail** versions of the slides in your presentation; the slide you're working will be highlighted.
4. The bottom area is the **notes pane**, this is where you type speaker notes that you can refer to when you present. You can also print speaker notes to use when presenting a slide show.



## Lab Task 1:

Create a Business Plan for a New Software Development Company.





## MS PowerPoint - Advanced

### Lab Task 1:

Create Animated Presentations.

**TRAVELING SALESMAN PROBLEM**

- A salesman needs to **visit** each of the  $n$  cities.

Is there a **route** that takes the **salesman** through every **city** and back to starting **city A** at a cost of **\$520**?

**Example**

$$125_{10} = ?_2$$

2	125	1
2	62	0
2	31	1
2	15	1
2	7	1
2	3	1
2	1	1
	0	1

$$125_{10} = 1111101_2$$

Info contained in the form

Browser

User's Computer

Server-Side Script

Web Server

Acknowledgement

Message to the receiver's eMail address

**SOLUTION**

- State transition diagram with outputs for this system.

```

graph LR
    S0((S0)) -- "0/'A'" --> S0
    S0 -- "2/'I'" --> S2((S2))
    S2 -- "0/'A'" --> S0
    S0 -- "1/'B'" --> S1((S1))
    S1 -- "1/'B'" --> S1
    S1 -- "2/'I'" --> S2
  
```

## LAB – 08

### HTML5

**HTML5** is the newest version of Hyper Text Markup Language. The first web browser was introduced in 1993 and the name was **MOSAIC**. The development of **MOSAIC** was at the **NCSA** (National Center for Supercomputing Applications). Later it was discontinued to development on 7th of January 1997. Still the people were using the nonstandard version of **HTML**.

The standard version came into existence in 1995, when **HTML 2.0** was announced. Later after two years **HTML 3.0** and after two years **HTML 4.01** was announced. And still we are using the milestone of **HTML 4.01**.

#### Comment Tag

**HTML5 Comment** tag is used to understand the code easily. It is used for creating comments in the **HTML scripts**. Comments aren't displayed in the browser, but from the programmer's point view to understand code easily, the **comment tag** is more sufficient. It is also used if, we want to keep the script but don't want it to be executed in the browser.

#### Syntax for COMMENT TAG

<!-- -->

#### Example

```
<!DOCTYPE html>
<html>

  <head>
    <title>
      Page title will go here
    </title>
  </head>

  <body>
    <!-- Start Writing Here. -->
    This is test page
  </body>

</html>
```



## Lab Task 1:

On your page, include some or all of the following information:

- On your page there must be at least two hyper links.
- Address of your e-mail.
- The link of the web page of web programming course.
- The link on an image.
- Add META tag to redirect page to this page  
<http://www.sites.google.com/site/cs1117webprog>

**Fatima Ahmed**

**Lecturer of Computer Science**

[The University of Lahore.](#)

Defence Road Off Raiwind Road, Bhobatain Chowk, Lahore - Pakistan.

[fatima.ahmed@cs.uol.edu.pk](mailto:fatima.ahmed@cs.uol.edu.pk)

I teach [web programming](#) course.



Click here to view my profile.

**Code:**

```
<!DOCTYPE html>
<html>
<head>
    <title>Welcome to Yasir's Web Page</title>
    <meta http-equiv="refresh" content="5; url=http://sites.google.com/site/cs1113webprog"
/>
</head>

<body>
<h1><font color="green">Fatima Ahmed</font></h1>
<b><font color="red">Lecturer of Computer Science</font></b><br/>
<a href="www.uol.edu.pk" target="_blank">The University of Lahore.</a><br/>
Defence Road Off Raiwind Road, Bhobatain Chowk, Lahore - Pakistan.<br/>
<br/>
<a href="mailto: fatima.ahmed@cs.uol.edu.pk">fatima.ahmed@cs.uol.edu.pk</a><br/><br/>
I teach <a href="www.sites.google.com/site/cs1113webprog">web programming</a>
course.<br/><br/>

Click here <a href="www.facebook.com/xyz" target="_blank"></a> to view my profile.
</body>
</html>
```



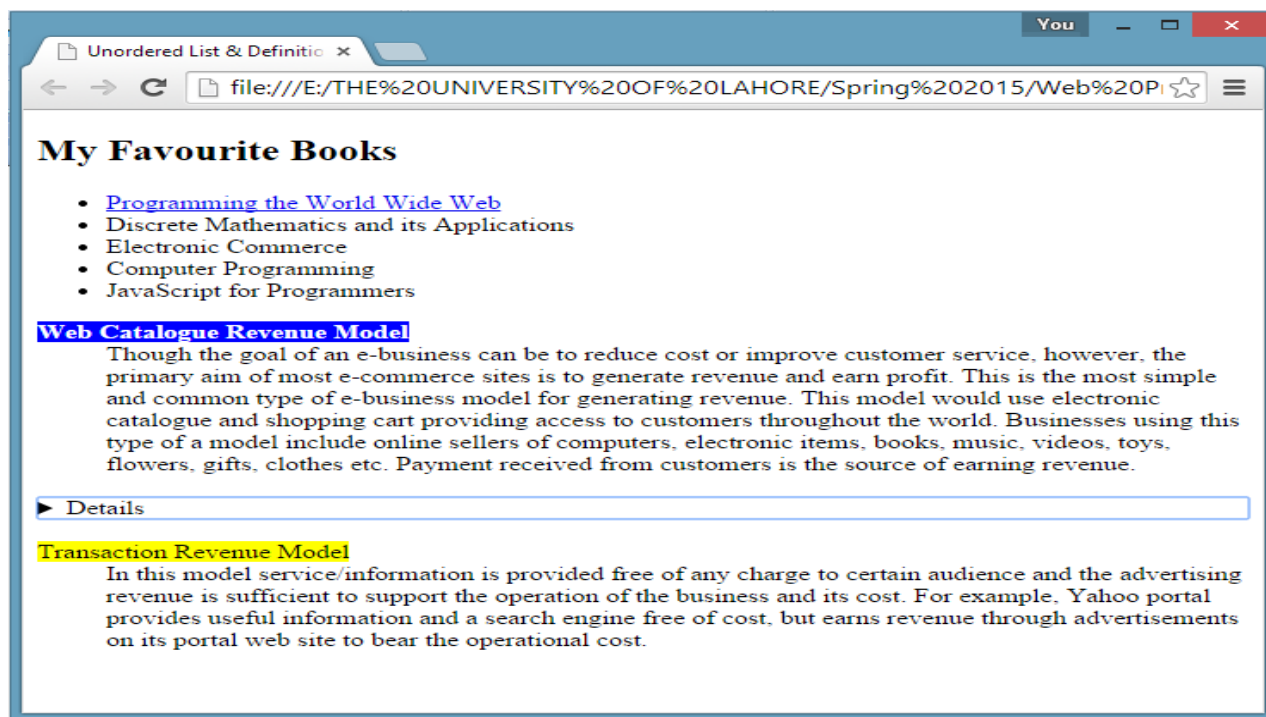
## HTML5 – List

### Lab Task 1:

On your page, include some or all of the following information:

- Create a simple un-order and definition list given below:
- Add hyper links on text.
- Highlight the definition list terms by using mark tag.
- Style the mark tag with different background color and foreground color.
- Apply <details> tag that is introduced by HTML5.

You are going to match the output given below:



**Code:**

```
<!DOCTYPE html>
<html>
<head>
    <title>Unordered List & Definition List</title>
</head>

<body>
<h2>My Favourite Books</h2>
<ul>
    <li><a href="www.w3schools.com" target="_blank">Programming the World Wide
Web</a></li>
    <li>Discrete Mathematics and its Applications</li>
    <li>Electronic Commerce</li>
    <li>Computer Programming</li>
    <li>JavaScript for Programmers</li>
</ul>
<dl>
    <dt><mark style="background-color: blue; color: white;"><b>Web Catalogue Revenue
Model</b></mark></dt>
    <dd>Though the goal of an e-business can be to reduce cost or improve customer service,
however, the primary aim of most e-commerce sites is to generate revenue and earn profit. This
is the most simple and common type of e-business model for generating revenue. This model
would use electronic catalogue and shopping cart providing access to customers throughout the
world. Businesses using this type of a model include online sellers of computers, electronic
items, books, music, videos, toys, flowers, gifts, clothes etc. Payment received from customers is
the source of earning revenue.</dd>
    <br/>
    <details>
    <dt>Advertising Revenue Model</dt>
    <dd>In this model service/information is provided free of any charge to certain audience
and the advertising revenue is sufficient to support the operation of the business and its cost. For
example, Yahoo portal provides useful information and a search engine free of cost, but earns
revenue through advertisements on its portal web site to bear the operational cost.</dd>
    </details>
    <br/>
    <dt><mark>Transaction Revenue Model</mark></dt>
    <dd>In this model service/information is provided free of any charge to certain audience
and the advertising revenue is sufficient to support the operation of the business and its cost. For
example, Yahoo portal provides useful information and a search engine free of cost, but earns
revenue through advertisements on its portal web site to bear the operational cost.</dd>
</dl>
</body>
</html>
```

## LAB – 09

### HTML5 – Nested List

#### Lab Task:

On your page, include some or all of the following information:

- Create an ordered list.
- Create a nested list given below:

(You may be tempted to spend all 50 of your minutes on this part of the lab. But try to keep the page somewhat short so that you can attempt some of the other exercises.)

#### Courses Taught

- Web Programming
- Discrete Structures
- Electronic Commerce
- Database Systems
- Introduction to Computing

#### Nested List Example

- Web Programming
  - Autumn 2014
    - September - January
    - February - June
  - Spring 2015
- Discrete Structures
  - Summer 2014
- Electronic Commerce
- Database Systems
- Introduction to Computing

**Code:**

```
<!DOCTYPE html>
<html>
<head>
    <title>Ordered List & Nested List</title>
</head>

<body>
<h1>Courses Taught</h1>
<ol type="a">
    <li>Web Programming</li>
    <li>Discrete Structures</li>
    <li>Electronic Commerce</li>
    <li>Database Systems</li>
    <li>Introduction to Computing</li>
</ol>
<h1>Nested List Example</h1>
<ol type="I">
    <li>Web Programming</li>
        <ul type="square">
            <li>Autumn 2014</li>
                <ol type="I" start="3">
                    <li>September - January</li>
                    <li>February - June</li>
                </ol>
            <li>Spring 2015</li>
        </ul>
    <li>Discrete Structures</li>
        <ol type="i">
            <li>Summer 2014</li>
        </ol>
    <li>Electronic Commerce</li>
    <li>Database Systems</li>
    <li>Introduction to Computing</li>
</ol>

</body>
</html>
```

## LAB – 10

### HTML5 – Audio/Video

#### Lab Task 1:

Create a page named `audio.html`. On your page, include some or all of the following information:

- Embed an Audio file and add feature of autoplay and continuous play.



#### Code:

```
<!DOCTYPE html>

<html lang='en'>
<head>
    <meta charset="UTF-8" />
    <title>Audio File</title>
</head>

<body>
<h1>
    HTML5 Audio
</h1>

<p> Here's a song: </p>
<audio controls autoplay loop>
    <source src="media/song.m4a" type="audio/x-aac" />
```

```
<source src="media/song.mp3" type="audio/mpeg" />
<source src="media/song.ogg" type="audio/ogg" />
<p> Your browser does not support the HTML5 audio feature. </p>
</audio>

</body>
</html>
```

## Lab Task 2:

Create a page named `video.html`. On your page, include some or all of the following information:

- Embed a Video file and add feature of autoplay and continuous play.

## HTML5 Video

Here's a Video:



**Code:**

```
<!DOCTYPE html>

<html lang='en'>
<head>
    <meta charset="UTF-8" />
    <title>Video File</title>
</head>

<body>
<h1>
    HTML5 Video
</h1>

<p> Here's a Video: </p>
<video width="660" height="380" controls>

    <source src="media/video.ogv" type="video/ogg" />
    <source src="media/video.webm" type="video/webm" />
    <source src="media/video.m4v" type="video/mp4" />
    <p> Your browser does not support the HTML5 video feature. </p>
</video>

</body>
</html>
```



## LAB – 11

### HTML5 – Tables

#### Lab Task 1:

On your page, include some or all of the following information:

- Create a simple table which has 5 rows and 4 columns.
- Apply cellpadding and cellspacing attributes.

## HTML Tables

Country	Ranking	Population	Per Capita Income
Denmark	1	5.5 million	\$ 34,600
Switzerland	2	7.5 million	\$ 32,300
Austria	3	8.2 million	\$ 32,700
Bhama	5	303,800	\$ 20,200
Bhutan	8	2.3 million	\$ 1,600

#### Code:

```
<!DOCTYPE html>
<html>
<head>
  <title>Tables</title>
</head>

<body>

<h1>HTML Tables</h1>
<table border="2" cellpadding="5" cellspacing="1" bgcolor="#87CEFA">
```

```
<caption><strong>Nation Information</strong></caption>
<tr>
  <th>Country</th>
  <th>Ranking</th>
  <th>Population</th>
  <th>Per Capita Income</th>
</tr>
<tr align="center">
  <td>Denmark</td>
  <td>1</td>
  <td>5.5 million</td>
  <td>$ 34,600</td>
</tr>
<tr align="center">
  <td>Switzerland</td>
  <td>2</td>
  <td>7.5 million</td>
  <td>$ 32,300</td>
</tr>
<tr align="center">
  <td>Austria</td>
  <td>3</td>
  <td>8.2 million</td>
  <td>$ 32,700</td>
</tr>
<tr align="center">
  <td>Bhamas</td>
  <td>5</td>
  <td>303,800</td>
  <td>$ 20,200</td>
</tr>
<tr align="center">
  <td>Bhutan</td>
  <td>8</td>
  <td>2.3 million</td>
  <td>$ 1,600</td>
</tr>

</table>

</body>
</html>
```

## Lab Task 2:

On your page, include some or all of the following information:

- Create another table same as given below.
- Apply rowspan and colspan attributes to achieve desired result.

Income Plan					
Dream Technologies					
Year	Quarter	Expenses		Income	
		Quetta	Dubai	Quetta	Dubai
2001	1	1,900	8,650	9,000	7,780
	2	2,230	8,650	8,500	8,670
	3	4,000	8,650	9,900	9,870
	4	2,200	8,650	9,800	9,900
2002	1	7,780	8,650	7,780	9,000
	2	8,670	8,650	8,670	8,500
	3	9,870	8,650	9,870	9,900
	4	9,900	8,650	9,900	9,800

## Code:

```
<!DOCTYPE html>
<html>
<head>
  <title>Tables</title>
</head>

<body>
```

```
<h1>Income Plan</h1>
<table border="2" cellpadding="5" cellspacing="1" bgcolor="#FFD700">
<caption><strong>Dream Technologies</strong></caption>
<tr>
    <th rowspan="2">Year</th>
    <th rowspan="2">Quarter</th>
    <th colspan="2">Expenses</th>
<!--    <th>Expenses</th> -->
    <th colspan="2">Income</th>
<!--    <th>Income</th> -->
</tr>
<tr align="center">
<!--    <td>Year</td> -->
<!--    <td>Quarter</td> -->
        <td>Quetta</td>
        <td>Dubai</td>
        <td>Quetta</td>
        <td>Dubai</td>
</tr>
<tr align="center">
    <td rowspan="4">2001</td>
    <td>1</td>
    <td>1,900</td>
    <td>8,650</td>
    <td>9,000</td>
    <td>7,780</td>
</tr>
<tr align="center">
<!--    <td>2001</td> -->
        <td>2</td>
        <td>2,230</td>
        <td>8,650</td>
        <td>8,500</td>
        <td>8,670</td>
</tr>
<tr align="center">
<!--    <td>2001</td> -->
        <td>3</td>
        <td>4,000</td>
        <td>8,650</td>
        <td>9,900</td>
        <td>9,870</td>
</tr>
```

```
<tr align="center">
<!-- <td>2001</td> -->
    <td>4</td>
    <td>2,200</td>
    <td>8,650</td>
    <td>9,800</td>
    <td>9,900</td>
</tr>
<tr align="center">
    <td rowspan="4">2002</td>
    <td>1</td>
    <td>7,780</td>
    <td>8,650</td>
    <td>7,780</td>
    <td>9,000</td>
</tr>
<tr align="center">
<!-- <td>2002</td> -->
    <td>2</td>
    <td>8,670</td>
    <td>8,650</td>
    <td>8,670</td>
    <td>8,500</td>
</tr>
<tr align="center">
<!-- <td>2002</td> -->
    <td>3</td>
    <td>9,870</td>
    <td>8,650</td>
    <td>9,870</td>
    <td>9,900</td>
</tr>
<tr align="center">
<!-- <td>2002</td> -->
    <td>4</td>
    <td>9,900</td>
    <td>8,650</td>
    <td>9,900</td>
    <td>9,800</td>
</tr>
</table>

</body>
</html>
```

## LAB – 12

### HTML5 - Forms

#### Lab Task 1:

On your page, include some or all of the following information:

- Create forms same as given below:

**Continental Hotel Reservation Form**

<b>Name</b>	<input type="text" value="Enter your name"/>
<b>eMail</b>	<input type="text" value="info@continental.pk"/>
<b>Mobile No</b>	<input type="text" value="Enter your cell number"/>
<b>Arrival Time</b>	<input type="text" value="Enter your arrival time"/>
<b>Departure</b>	<input type="text"/> <input type="text"/>
<b>Room Type</b>	<input type="radio"/> Single <input type="radio"/> Double <input type="radio"/> Presidential Suite
<b>Preferences</b>	<input type="checkbox"/> Non-Smoking <input type="checkbox"/> Internet <input type="checkbox"/> Firm Mattress
<b>Mode of Payment</b>	<input type="checkbox"/> Cash <input type="checkbox"/> Check <input type="checkbox"/> Credit Card <input type="text" value="Select Card"/>
<input type="button" value="Send Data"/>	

**Code:**

```
<!DOCTYPE html>
<html>
<head>
    <title>Continental Hotel Reservation Form</title>
</head>

<body bgcolor="#E0E0E0">
<form name="hotel" method="post" action="serversideScriptURL">
<table border="2" cellpadding="5" bgcolor="#CCE5FF">
<caption><h3><font face="Trebuchet MS" color="red">Continental Hotel Reservation
Form</h3></caption>
<tr>
    <td align="right"><b>Name</b></td>
    <td><input type="text" name="username" placeholder="Enter your name" size="30"
required /></td>
</tr>
<tr>
    <td align="right"><b>eMail</b></td>
    <td><input type="email" name="useremail" value="info@continental.pk" size="30"
readonly/></td>
</tr>
<tr>
    <td align="right"><b>Mobile No</b></td>
    <td><input type="text" name="mobile" placeholder="Enter your cell number"
size="30"/></td>
</tr>
<tr>
    <td align="right"><b>Arrival Time</b></td>
    <td><input type="datetime-local" name="arrival" placeholder="Enter your arrival time"
/></td>
</tr>
<tr>
    <td align="right"><b>Departure</b></td>
    <td>
        <input type="time" name="depart_time" title="Departure Time" />
        <input type="date" name="depart_date" title="Departure Date" />
    </td>
</tr>
<tr>
    <td align="right"><b>Room Type</b></td>
    <td>
        <input type="radio" name="room_type" value="Single" />Single
        <input type="radio" name="room_type" value="Double" />Double<br />
    </td>
</tr>
</table>
</form>
</body>
</html>
```



```

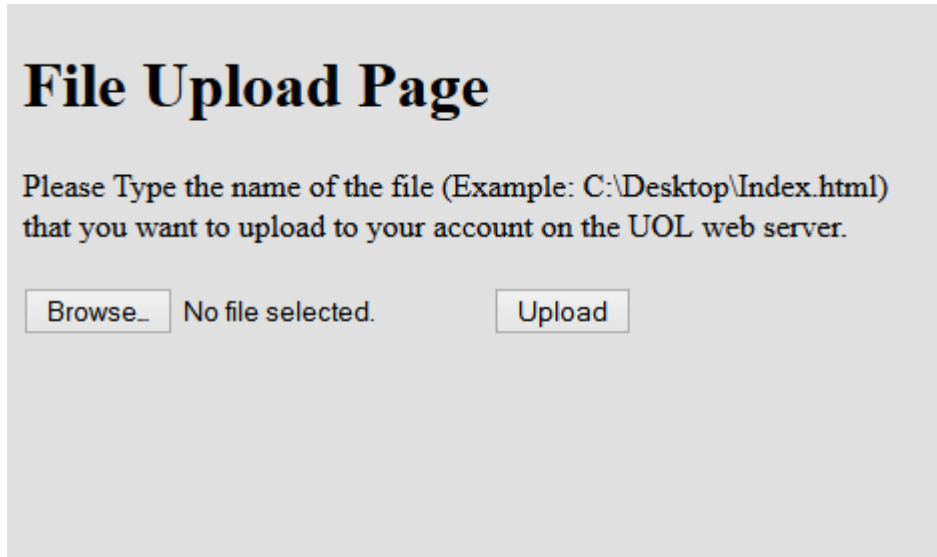
Suite
    <input type="radio" name="room_type" value="Presidential Suite" />Presidential
</td>
</tr>
<tr>
    <td align="right"><b>Preferences</b></td>
    <td>
        <input type="checkbox" name="non_smoking" value="Non-Smoking" />Non-
Smoking<br/>
        <input type="checkbox" name="internet" value="Internet" />Internet<br/>
        <input type="checkbox" name="firm" value="Firm Mattress" />Firm Mattress
    </td>
</tr>
<tr>
    <td align="right"><b>Mode of Payment</b></td>
    <td>
        <input type="checkbox" name="cash" value="Cash" />Cash
        <input type="checkbox" name="check" value="Check" />Check<br/>
        <input type="checkbox" name="credit" value="Credit Card" />Credit Card
        <select name="bank">
            <option value="-1">Select Card</option>
            <option value="master">Master</option>
            <option value="visa">Visa</option>
        </select>
    </td>
</tr>
<tr align="center">
    <td colspan="2">
        <input type="submit" name="submit" value="Send Data" />
    </td>
</tr>
</table>
</form>
</body>
</html>

```

## Lab Task 2:

On your page, include some or all of the following information:

- Create forms same as given below:



### Code:

```
<!DOCTYPE html>
<html>
<head>
    <title>File Upload Form</title>
</head>

<body bgcolor="#E0E0E0">

<h1>File Upload Page</h1>

<form name="upload" method="post" action="serversideScriptURL" enctype="multipart/form-
data">
Please Type the name of the file (Example: C:\Desktop\Index.html) that you want to upload to
your account on the UOL web server.<br/><br/>
<input type="file" name="cv" /><input type="submit" name="submit" value="Upload" />

</form>
</body>
</html>
```

## LAB – 13

### Introduction to Programming in C++

The objective of this exercise is to understand the programming basic concepts.

#### Introduction to Programming in C++

#### Introduction

##### **What is a programming language?**

C++ is immensely popular, particularly for applications that require speed and/or access to some low-level features. C++ is high-level languages; other high-level languages you might have heard of are Java, C and Pascal FORTRAN. As you might infer from the name “high-level language,” there are also low-level languages, sometimes referred to as machine language or assembly language. Loosely-speaking, computers can only execute programs written in low-level languages. Thus, programs written in a high-level language have to be translated before they can run. This translation takes some time, which is a small disadvantage of high-level languages. But the advantages are enormous. First, it is much easier to program in a high-level language; by “easier” I mean that the program takes less time to write, it’s shorter and easier to read, and it’s more likely to be correct. Secondly, high-level languages are portable, meaning that they can run on different kinds of computers with few or no modifications. Low-level programs can only run on one kind of computer and have to be rewritten to run on another. Due to these advantages, almost all programs are written in high-level languages. Low-level languages are only used for a few special applications.

##### **Why Use a Language Like C++?**

At its core, a computer is just a processor with some memory, capable of running tiny instructions like “store 5 in memory location 23459.” Why would we express a program as a text file in a programming language, instead of writing processor instructions?

The advantages:

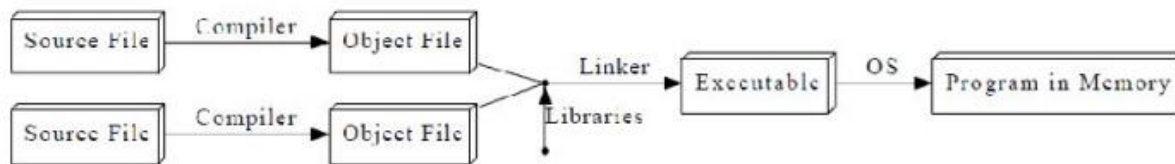
1. Conciseness: programming languages allow us to express common sequences of commands more concisely. C++ provides some especially powerful short hands.
2. Maintainability: modifying code is easier when it entails just a few text edits, instead of rearranging hundreds of processor instructions.
3. Portability: different processors make different instructions available. Programs written as text can be translated into instructions for many different processors; one of C++’s strengths is that it can be used to write programs for nearly any processor.

##### **The Compilation Process**

A program goes from text files (or source files) to processor instructions as follows:

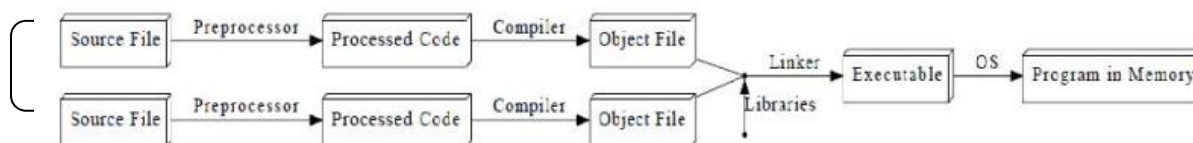
A compiler is a program that reads a high-level program and translates it all at once, before executing any of the commands. Often you compile the program as a separate step, and then

execute the compiled code later. In this case, the high-level program is called the source code, and the translated program is called the object code or the executable.



Object files are intermediate files that represent an incomplete copy of the program: each source file only expresses a piece of the program, so when it is compiled into an object file, the object file has some markers indicating which missing pieces it depends on. The linker takes those object files and the compiled libraries of predefined code that they rely on, fills in all the gaps, and spits out the final program, which can then be run by the operating system (OS).

In C++, all these steps are performed ahead of time, before you start running a program. In some languages, they are done during the execution process, which takes time. This is one of the reasons C++ code runs far faster than code in many more recent languages. C++ actually adds an extra step to the compilation process: the code is run through a preprocessor, which applies some modifications to the source code, before being fed to the compiler. Thus, the modified diagram is:



## First Program in C++

### Source Code

```
//my first program in C++
#include <iostream>
using namespace std;
void main ()
{
    cout << "Hello World!";
}
```

### Output

Hello World!

### // my first program in C++

This is a comment line. All lines beginning with two slash signs (//) are considered comments and do not have any effect on the behavior of the program. The programmer can use them to include short explanations or observations within the source code itself.

### **#include <iostream>**

Lines beginning with a hash sign (#) are directives for the preprocessor. They are not regular code lines with expressions but indications for the compiler's preprocessor. In this case the directive `#include<iostream>` tells the preprocessor to include the `iostream` standard file. This

specific file (`iostream`) includes the declarations of the basic standard input-output library in C++, and it is included because its functionality is going to be used later in the program.

### **Using namespace std;**

All the elements of the standard C++ library are declared within what is called a namespace, the namespace with the name `std`. So in order to access its functionality we declare with this expression that we will be using these entities. This line is very frequent in C++ programs that use the standard library.

### **Void main ()**

The main function is the point by where all C++ programs start their execution, independently of its location within the source code. It does not matter whether there are other functions with other names defined before or after it – the instructions contained within this function's definition will always be the first ones to be executed in any C++ program. For that same reason, it is essential that all C++ programs have a main function. The word `main` is followed in the code by a pair of parentheses `()`. That is because it is a function declaration: In C++, what differentiates a function declaration from other types of expressions are these parentheses that follow its name. Optionally, these parentheses may enclose a list of parameters within them. Right after these parentheses we can find the body of the main function enclosed in braces `{ }`. What is contained within these braces is what the function does when it is executed.

### **cout << "Hello World!";**

This line is a C++ statement. This statement performs the only action that generates a visible effect in our first program. `Cout` represents the standard output stream in C++, and the meaning of the entire statement is to insert a sequence of characters (in this case the `Hello World` sequence of characters) into the standard output stream (which usually is the screen). `cout` is declared in the `iostream` standard file within the `std` namespace, so that's why we needed to include that specific file and to declare that we were going to use this specific namespace earlier in our code.

Notice that the statement ends with a semicolon character `(;)`. This character is used to mark the end of the statement and in fact it must be included at the end of all expression statements in all C++ programs (one of the most common syntax errors is indeed to forget to include some semicolon after a statement).

### **Source Code**

```
//my second program in C++
#include <iostream>
using namespace std;
voidmain ()
{
```

### **Output**

```
20
100
0
1
```

```
cout << 10+10<<endl;  
cout<<10*10<<endl;  
cout<<10-10<<endl;  
cout<<10/10<<endl;}
```

You can also do mathematical operations using cout statements like above. You can also do all the mathematical operation using single cout statement like this:

```
cout<<10/10<<endl<<10*10<<endl<<10+10<<endl<<10-10<<endl;
```

You can perform all the mathematical operation in one single line by inserting << output operators.

**For Formula below, you have to include the header file: math.h (#include<math.h>)**

Absolute: **cout<<abs (-8) <<"\n";** means that it will give modulus of value given i.e. +ive value

Exponent: **cout<<exp (6) <<"\n";**

Power: **cout<<pow (2, 9) <<"\n";** means that 2 raise to power 9 i.e.  $2^9$ .

Square Root: **cout<<sqrt (16) <<"\n";**

### **Logarithm & Trigonometric Functions:**

Natural Algorithm: **cout<< log (3) <<"\n";** Gives the natural logarithm of 3.

Sine: **cout<< sin (45) <<"\n";** Gives the Sine value of 45 degrees in radians.

Cosine: **cout<<cos (45) <<"\n";** Gives the cosine value of 45 degrees in radians.

Tangent: **cout<<sin (45) <<"\n";** Gives the Tangent value of 45 degrees in radians.

Inverse of Sine: **cout<<asin (30) <<"\n";** Gives the inverse sine of 30 degrees in radians.

Inverse of Cosine: **cout<<acos (30) <<"\n";** Gives the inverse cosine of 30 degrees in radians.

Inverse of Tangent: **cout<<atan (30) <<"\n";** Gives the inverse tangent of 30 degrees in radians.

Hyperbolic Sine: **cout<<sinh (60) <<"\n";** Gives the hyperbolic sine of 60 degrees in radians.

Hyperbolic Cosine: **cout<<cosh (60) <<"\n";** Gives the hyperbolic cosine of 60 degrees in radians.

Hyperbolic Tangent: **cout<<tanh (60) <<"\n";** Gives the hyperbolic tangent of 60 degrees in radians.

### **Tokens**

Tokens are the minimal chunk of program that has meaning to the compiler –the smallest meaningful symbols in the language. Our code displays all 6 kinds of tokens:



Token type	Description/Purpose	Examples
Keywords	Words with special meaning to the compiler	int, double, for, auto
Identifiers	Names of things that are not built into the language	cout, std, x, myFunction
Literals	Basic constant values whose value is specified directly in the source code	"Hello, world!", 24.3, 0, 'c'
Operators	Mathematical or logical operations	+, -, &&, %, <<
Punctuation/Separators	Punctuation defining the structure of a program	{ } ( ) , ;
Whitespace	Spaces of various sorts; ignored by the compiler	Spaces, tabs, newlines, comments

**High-Level Language:** A programming language like C++ that is designed to be easy for humans to read and write.

**Low-Level Language:** A programming language that is designed to be easy for a computer to execute also called “machine language” or “assembly language.”

**Portability:** A property of a program that can run on more than one kind of computer

**Formal Language:** Any of the languages people have designed for specific purposes, like representing mathematical ideas or computer programs. All programming languages are formal languages.

**Natural Language:** Any of the languages people speak that have evolved naturally.

**Compile:** To translate a program in a high-level language into a low-level language, all at once, in preparation for later execution.

**Source Code:** A program in a high-level language, before being compiled.

**Object Code:** The output of the compiler, after translating the program.

**Executable:** Another name for object code that is ready to be executed.

**Algorithm:** A general process for solving a category of problems.

**Bug:** An error in a program.

**Syntax:** The structure of a program.

**Syntax Error:** An error in a program that makes it impossible to parse (and therefore impossible to compile).

**Run-Time Error:** An error in a program that makes it fails at run-time.

**Logical Error:** An error in a program that makes it does something other than what the programmer intended.

**Debugging:** The process of finding and removing any of the three kinds of errors.

## LAB – 14

### Variables, Constants & Data Types

The objective of this exercise is to understand variables and their data types use in programming.

#### Variables, Constants & Data Types

##### Introduction

##### CPU & Memory:

A computer acts on some data according to the instructions specified in a program and produces the output. This computation is done by Central Processing Unit (CPU) of the computer. A CPU of computer performs arithmetic operations such as addition, subtraction, multiplication and division and also performs logical operation on the data such as comparing two numbers.

**Memory** of a computer is the area where data and programs are stored. A computer memory is made up of registers and cells which are capable of holding information in the form of binary digits 0 and 1 (bits). This is the way information is stored in a memory just like we memorize numbers using 0 to 9. Collection of 8 bits is called a **byte**. Normally the CPU does not access memory by individual bits. Instead, it accesses data in a collection of bits, typically 8 bits, 16 bit, 32 bit or 64 bit. The amount of bits on which it can operate simultaneously is known as the word length of the computer.

When we say that Pentium 4 is a 32 bit machine, it means that it simultaneously operates on 32 bit of data. Data is stored in the memory at physical memory locations. These locations are known as the memory address. It is very difficult for humans to remember the memory addresses in numbers like 46735, so instead we name a memory address by a **variable**.

##### Variables

“A variable is a named location that stores a value. “When you create a new variable, you have to declare what type it is and then the name of that variable.

##### Syntax:

Type of Variable                      Name of Variable;

##### **Declaring a variable in C++:**

char a;

The type of a variable determines what kind of values it can store. A char variable can contain 1 characters, and it should come as no surprise that int variables can store integers.

To create an integer variable, the syntax is:-

```
int a;
```

The statement below show that how to declare two variables in one instruction.

```
int b, c;
```

## **Data Types**

Every variable in C++ can store a value. However, the type of value which the variable can store has to be specified by the programmer. C++ supports the following inbuilt data types: **int** (to store integer values), **float** (to store decimal values) and **char** (to store characters), **bool** (to store Boolean value either 0 or 1) and etc.

### **A. Integer:**

Integer (int) variables are used to store integer values like 34, 68704 etc. To declare a variable of type integer, type keyword int followed by the name of the variable. You can give any name to a variable but there are certain constraints, they are specified in Identifiers section. For example, the statement

**int sum;** declares that sum is a variable of type int. You can assign a value to it now or later. In order to assign values along with declaration use the **assignment operator (=)**.

```
int sum = 25;
```

Assigns value 25 to variable sum. There are three types of integer variables in C++, **short**, **int** and **long** int. All of them store values of type integer but the size of values they can store increases from short to long int. This is because of the size occupied by them in memory of the computer. The size which they can take is dependent on type of computer and varies. More is the size, more the value they can hold. For example, int variable has 2 or 4 bytes reserved in memory so it can hold  $2^{32} = 65536$  values. Variables can be signed or unsigned depending they store only positive values or both positive and negative values. And short, variable has 2 bytes. Long int variable has 4 bytes.

### **B. Float:**

To store decimal values, you use float data type. Floating point data types comes in three sizes, namely float, double and long double. The difference is in the length of value and amount of precision which they can take and it increases from float to long double. For example, statement

```
float average = 2.34;
```

Declares a variable average which is of type float and has the initial value 2.34

### **C. Character:**

A variable of type char stores one character. Its size of a variable of type char is typically 1 byte. The statement

```
char name = 'c';
```

Declares a variable of type char (can hold characters) and has the initial values as character c.  
Note that value has to be under single quotes.

#### **D. Boolean:**

A variable of type bool can store either value true or false and is mostly used in comparisons and logical operations. When converted to integer, true is any non-zero value and false corresponds to zero.

C++ imposes fairly strict rules on how you can name your variables:

- variable names must begin with a letter
- variable names are "case-sensitive" (i.e., the variable "MYNUMBER" which is different from the variable "mYnUmBeR")
- variable names can't have spaces
- variable names can't have special characters (typographic symbols)

Name	Description	Size*	Range*
char	Character or small integer.	1byte	signed: -128 to 127 unsigned: 0 to 255
short int (short)	Short Integer.	2bytes	signed: -32768 to 32767 unsigned: 0 to 65535
int	Integer.	4bytes	signed: -2147483648 to 2147483647 unsigned: 0 to 4294967295
long int (long)	Long integer.	4bytes	signed: -2147483648 to 2147483647 unsigned: 0 to 4294967295
bool	Boolean value. It can take one of two values: true or false.	1byte	true or false
float	Floating point number.	4bytes	+/- 3.4e +/- 38 (~7 digits)
double	Double precision floating point number.	8bytes	+/- 1.7e +/- 308 (~15 digits)
long double	Long double precision floating point number.	8bytes	+/- 1.7e +/- 308 (~15 digits)
wchar_t	Wide character.	2 or 4 bytes	1 wide character

#### **Assignment Operator (=)**

Now that we have created some variables, we would like to store values in them. We do that with an assignment statement.

Char a; \\Declaring a variable of character type.

a = 'H'; \\Assigning a character value to the character variable.

- When you declare a variable, you create a named storage location.
- When you make an assignment to a variable, you give it a value.

#### **Declaration of Variable of Data Type (Integer)**

\\Assigning a Fixed Value to Variable

```
#include <iostream.h>
```

```
void main()
```

```
{  
    int length;  
    length = 7;  
    cout<<"The Length is: ";  
    cout<<length;  
    cout<<"\n\n";  
}
```

### **Assign the Value to Variable through Keyboard**

Remember to take value from keyboard, we use cin with input operator >>.

```
#include <iostream.h>
```

```
void main()
```

```
{  
    int length;  
    cin>>length;  
    cout<<"The Length is: ";  
    cout<<length;  
    cout<<"\n\n";  
}
```

### **Calculate the Area of the Box**

```
#include <iostream.h>
```

```
void main()
```

```
{  
    int length;  
    int width;  
    length=7;  
    width=5;  
    cout<<"The area is: ";  
    cout<<length*width;  
    cout<<"\n\n";  
}
```

### **Program should take the value of length and width from User**

```
#include <iostream.h>
```

```
void main()
```

```
{  
    int length, width, area; //you can declare the variable in one line.  
    cin>>length;  
    cin>>width;  
    area = length * width;  
    cout<<"The area is: ";  
    cout<<area;  
    cout<<"\n\n";}
```

**You can take the value from User in one line as Well.**



```
#include <iostream.h>
void main()
{
    int length, width, area; //you can declare the variable in one line.
    cin>>length>>width;
    area = length * width;
    cout<<"The area is: ";
    cout<<area;
    cout<<"\n\n";
}
```

**What is the difference of float data type from integer data type?**

```
#include <iostream.h>
void main()
{
    int a, b, c;
    a=10;
    b=3;
    c=a/b;
    cout<<"Answer is: ";
    cout<<c;
    cout<<"\n\n";
}
```

The output of this program is 3, but actually  $10/3 = 3.33333$ , but whenever fraction comes the int variable always takes the lower value like  $10/6=1.66667$  but the int variable will not convert it to 2 it will take the lower value i.e. 1 and excludes the fraction.

**Variable can be declared with float.**

```
#include <iostream.h>
void main()
{
    float a, b, c;
    a=10;
    b=3;
    c=a/b;
    cout<<"Answer is: ";
    cout<<c;
    cout<<"\n\n";
}
```

## LAB – 15

### Operators (Relational & Logical)

The objective of this exercise is to get familiar with operators in programming.

#### Operators (Relational & Logical)

##### Introduction

##### Control Structures

Control structures are portions of program code that contain statements within them and, depending on the circumstances, execute these statements in a certain way. There are typically two kinds: conditionals and loops.

#### **I. Conditionals**

In order for a program to change its behavior depending on the input, there must be a way to test that input. Conditionals allow the program to check the values of variables and to execute (or not execute) certain statements. C++ has if and switch-case conditional structures.

#### **II. Operators**

An operator is a symbol that tells the compiler to perform a specific mathematical or logical manipulation. C++ has four general classes of operators: arithmetic, bitwise, relational, and logical.

C++ also has several additional operators that handle certain special situations.

##### **1. Arithmetic**

C++ defines the following arithmetic operators:

Operator	Meaning
+	Addition
-	Subtraction (also unary minus)
*	Multiplication
/	Division
%	Modulus
++	Increment
--	Decrement

The operators +, -, \*, and / all work the same way in C++ as they do in algebra. These can be applied to any built-in numeric data type. They can also be applied to values of type char.

## 2. Increment & Decrement (++ and --)

The increment operator adds 1 to its operand, and the decrement operator subtracts 1. Therefore,

$x = x + 1$ ; is the same as  $x++$ ;

$x = x - 1$ ; is the same as  $--x$ ;

Both the increment and decrement operators can either precede (prefix) or follow (postfix) the operand.

For example,

$x = x + 1$ ; can be written as  $++x$ ; // **prefix form** or as  $x++$ ; // **postfix form**.

In this example, there is no difference whether the increment is applied as a prefix or a postfix.

The precedence of the arithmetic operators is shown here:

Highest	$++$ $--$
	$-$ (unary minus)
	$*$ $/$ $\%$
Lowest	$+$ $-$

## 3. Relational & Logical Operators

Relational refers to the relationships that values can have with one another, and logical refers to the ways in which true and false values can be connected together. Since the relational operators produce true or false results, they often work with the logical operators. In C++, not equal to is represented by  $!=$  and equal to is represented by the double equal sign,  $==$ .

Relational Operators	
Operator	Meaning
$>$	Greater than
$>=$	Greater than or equal to
$<$	Less than
$<=$	Less than or equal to
$==$	Equal to
$!=$	Not equal to
Logical Operators	
Operator	Meaning
$\&\&$	AND
$\ \ $	OR
$!$	NOT

The following table shows the relative precedence of the relational and logical operators.

Highest	$!$
	$>$ $>=$ $<$ $<=$
	$==$ $!=$
	$\&\&$
Lowest	$\ \ $

## LAB – 16

### Conditional Statements

The objective of this exercise is to get familiar with conditional statements in programming.

#### Conditional Loops

##### If Statement

The if keyword is used to execute a statement or block only if a condition is fulfilled.

##### Syntax

```
if (condition)
    statement;
```

Where condition is the expression that is being evaluated. If this condition is true, statement is executed. If it is false, statement is ignored (not executed) and the program continues right after this conditional structure. For example, the following code fragment prints x is 100 only if the value stored in the x variable is indeed 100:

```
if ( x == 100 )
    cout<<"x is 100";
```

If we want more than a single statement to be executed in case that the condition is true we can specify a block using braces { }:

```
if (x == 100)
{
    cout << "x is ";
    cout << x;
}
```

In code below, only that if statement will be executed which condition will be true.

```
int a, b;
cin>>a>>b;
if(a == b) cout<<"a is equal to b"<<endl;
if(a != b) cout<<"a is not equal to b"<<endl;
if(a > b) cout<<"a is greater than b"<<endl;
if(a < b) cout<<"a is less than b"<<endl;
```

### **If-Else Statement**

We can additionally specify what we want to happen if the condition is not fulfilled by using the keyword else. Its form used in conjunction with if is:

#### **Syntax**

```
if (condition)
{
    Statements;
}
else
{
    Statements;
}
```

The following program demonstrates the if by playing a simple version of the “guess the magic number” game. The program generates a random number, prompts for your guess, and prints the message **\*\*Right \*\*** if you guess the magic number. This program also introduces another C++ library function, called `rand ( )`, which returns a randomly selected integer value. It requires the `<cstdlib>`.

```
#include<iostream.h>
#include<cstdlib.h>
void main( )
{
    int magic; // magic number
    int guess; // user's guess
    magic = rand(); // get a random number
    cout<<"enter your guess: ";
    cin>>guess;
    if (guess==magic) cout<<" ** RIGHT ** ";
    else cout<<" .... SORRY YOU ARE WRONG .... ";
}
```

### **Nested If's Statement**

A nested if is an if statement that is the target of another if or else. Nested ifs are very common in programming. The main thing to remember about nested ifs in C++ is that an else statement always refers to the nearest if statement that is within the same block as the else and not already associated with an else. Here is an example:

```
#include<iostream.h>
#include<cstdlib.h>
void main( )
{
    int magic, guess;
```

```
magic = rand(); // get a random number
cout<<"enter your guess: ";
cin>>guess;
if (guess==magic) cout<<" ** right ** ";
else
{
cout<<" . . . sorry you are wrong . . . ";
If (guess > magic) cout<<" YOUR GUESS IS TOO HIGH .\n ";
else cout<<" YOUR GUESS IS TOO LOW. \n ";
}
}
```

### **If-Else-If Ladder**

A common programming construct that is based upon nested ifs is the if-else-if ladder, also referred to as the if-else-if staircase. It looks like this:

#### **Syntax**

```
if (condition)
    statement;
else if (condition)
    statement;
else if (condition)
    statement;
else
    statement;
```

The conditional expressions are evaluated from the top downward. As soon as a true condition is found, the statement associated with it is executed, and the rest of the ladder is bypassed. If none of the conditions is true, then the final else statement will be executed. The final else often acts as a default condition; that is, if all other conditional tests fail, then the last else statement is performed. If there is no final else and all other conditions are false, then no action will take place.

The following program demonstrates the if-else-if ladder:

```
#include<iostream.h>
void main( )
{
    int x=10;
    if(x==1) cout<<"x is one. \n";
    else if (x==2) cout<<"x is two. \n";
    else if(x==3) cout<<"x is three. \n";
    else if(x==4) cout<<"x is four. \n";
    else cout<<" x is not between 1 and 4.\n";
}
```