lOMoARcPSD|1535302

CSC111

Intro to Information and Communication Technology

LAB MANUAL



# Department of Computing And Technology

# IQRA University

# Table of Contents

|  |  |  |  |
| --- | --- | --- | --- |
| **Sr. No.** | **Week** | **Description** | **Page No.** |
| 1 | -- | Course Insights& Outcomes | 3 |
| 2 | -- | Environment Required | 4 |
| 3 | 1 | Experiment 1: Introduction to MS Word | 5 |
| 4 | 2 | Experiment 2: Introduction to Microsoft Power Point | 13 |
| 5 | 3 | Experiment 3: Introduction to Microsoft Excel | 31 |
| 6 | 4 | Experiment 4: Pivot Tables in MS Excel | 61 |
| 7 | 5 | Experiment 5: Introduction to Flow Charts | 104 |
| 8 | 6 | Experiment 6: Flowcharts, Algorithms and Source code (Logic Building) | 111 |
| 9 | 7 | Experiment 7: Programs input ,Output | 118 |
| 10 | 8 | Experiment 8: Condition (IF ELSE) | 125 |
| 11 | 9 | Experiment 9: Programs using nested IF ELSE | 138 |
| 13 | 10 | Experiment 10: Loops | 143 |
| 14 | 11 | Experiment 11: Nested For loops | 160 |
| 15 | 12 | Experiment 12: Loops Practice | 168 |
| 16 | 13 | Experiment 13: Functions | 174 |
| 18 | 14 | Experiment 14:Arrays | 178 |
| 19 | -- | Appendix A: Lab Evaluation Criteria | 182 |

**Course Insights& Outcomes**

|  |  |
| --- | --- |
| **Course Code** | CSC102 |
| **Course Title** | Introduction to Information and Communication Technology |
| **Credit Hours** | 3+1 Lab |
| **Prerequisites by Course(s) and Topics** | None |
| **Course Description** | This is an introductory course in Computer Science designed for beginners. Apart from leading the participants through a whirlwind history of computing, the course also develops a feel for web programming through a series of lectures that help the students develop their own web page. Main objective of the course is to build an appreciation for the fundamental concepts in computing and to become familiar with popular PC productivity software. |
| **Course Objectives** | * + - 1. Understand basics of computing technology       2. Do number systems conversions and arithmetic       3. Have knowledge of types of software       4. Have knowledge of computing related technologies |
| **Assessment Instruments with Weights** (homework, quizzes, midterms, final, programming assignments, lab work, etc.) | * Minimum 4 Assignments and 4 quizzes * 1 midterm exam * 1 final exam * 1 project * Lab Work-25% |
| **Course Coordinator** | Dr. Naveed Ejaz/Dr.Sadaf Tanvir/Ms. Sumaira Shafiq |
| **URL (if any)** | All the course details can be found on <https://sites.google.com/site/sadaftanvir> |
| **Current Catalog Description** | -- |
| **Textbook** (or **Laboratory Manual** for Laboratory Courses) | Parker, Charles S., and Deborah Morley. Understanding Computers: Today and Tomorrow. Ft. Worth, TX: Dryden Press, 2020, 16th Edition. |
| **Reference Material** | Would be provided on student demand |
| **Course Goals** | Student should be able to understand basics of computing technology,do number systems conversions and arithmetic, have knowledge of types of software, have knowledge of computing related technologies |

**Environment Required**

|  |  |
| --- | --- |
| **Sr. No.** | **Description** |
| 1 | Hardware e.g., Workstations |
| 2 | Software e.g., Visual Studio  Code Block |

**LAB-01**

**INTRODUCTION TO MS WORD**

**Objectives**

**Introduction to MS-WORD**

**Description**

**Introduction**

Before windows launched MS-WORD as a part of MS-OFFICE.

**Starting ’MS-WORD’**

The following methods of starting and closing Word are applicable in Word 7 to onwards

Click on the ’Start’ button and select ’Programs’. Click on ’MS WORD’ option in ’Programs’.

**Closing ’MS-WORD’**

Click on the button with a cross sign.

**Creating a Document**

A file created in ’Word’ is known as Document. in order to use the capital characters while

writing, press the Caps Lock key. If you want to write only the first character Capital , press

the Shift key and the letter key simultaneously.

**Editing Text**

If you have made any mistakes in typing the text it becomes necessary to correct it. Some -times you may also want to change the text (insert or delete a word). Any correction or

change in text is called editing the text.

You can edit a text in three ways:

Using the delete key.

Using the backspace key.

Highlighting a word and using the delete key.

**Saving a Document**

1) Click on File in the menu bar.

2) Select the save option from menu to open the save dialog box. Then you can save a file in

the ’my documents’.

3) Type ’my first file’ in the ’file name’ box.

4) Click on the save button and the file is saved and the dialog box will close.

**Save Vs. Save As**

The first time you save a file there is no difference between the two.

Once you’ve saved a file:

– Choosing save automatically saves the current file to the hard drive.

– Choosing save as allows you to save the file using a different name and/or

**change the file’s location**.

Two files can’t have the same name and file type if they’re located in the same folder.

They can, however share the same name if they are in subfolders of one another.

**Font and font size**

The characters used in the text are called fonts. Times new roman, Arial etc. are some of the

fonts.

To change the font and the font size:

1) Highlight the text with the help of the mouse.

2) Click on the small arrow against the font name.

3) Select a new font from the list of font names and click on it.

4) The font of the highlighted text will be changed.

**Font style**

The manner or style in which text is displayed is called font style. There

are three font styles in word: BOLD, ITALICS and UNDERLINE. There

are three buttons in the menu with B, I and U on them.

**Cut Copy and Paste**

To cut and paste text:

1) Highlight the text

2) Click on the cut icon on the menu bar.

3) The highlighted text will be cut and placed in the clipboard.

4) Move the mouse cursor to that part of the document where you want to place the cut text.

5) Click on the paste icon on the menu bar.

6) The cut text will appear at that place.

Cut, copy and paste can also be done from the ’Edit’ menu.

**Paragraph format**

In the ’Word’ you can begin a new paragraph by pressing Enter key. If you press the Enter

key after typing a line, paragraph will be of that one line only.

Paragraph format is used to determine:

1) Font and font size

2) Tab

3) Borders

4) Alignment

5) Bullets

6) Background

**Word Menu**

The menu bars of the word are:

1) **File:** When you click on file ,you will see a drop-down menu

Click on New to create a new document. Open is used to open an existing document.

Click on Close to close a document.

Click on Save to save a document.

**Save As** is used to save a document with other name. Send to helps to e-mail a

document. Properties contain the details of a document.

2) **Edit:** This menu contains cut, copy, and paste. it also contains undo, typing and repeat

typing, find/replace commands.

3) **View:** When you click on View you will see a drop down menu and gives

Normal, print layout, zoom, toolbars, customize, standard and formatting.

4) **Insert:** This menu contains hyperlink , File and Object, picture, text box.

5) **Format:** This menu contains font, paragraph, bullets and numbering, borders shading.

6) **Tools:** Spelling and Grammar is the most important options of this menu.

7) **Table:** this menu is used to insert a table in a document.

8) **Window:** In word every file has its own window. You can open more than one window at a

time.

9) **Help:** This menu is useful to a new user of ’word’. All the information

About ’word’ is available under this option and a user can get answers to almost all his ques-tions.

**Template**

When you start ’word’ you open a file with the name ’Document 1’ on the Title bar. The first

blank page of this file opens with pre-set margin, font, font size.

These properties always remain the same because the default templates are predetermined

by ’word’.

The entire document prepared using this template will have identical properties.

**Style**

’Word’ provides a variety of easy options to format a document.

Style of a paragraph means to select the various properties which will determine the way a

paragraph will appear. It includes:

1) Selecting the font and size of a character.

2) Selecting the margin, tab, line spacing.

Click on the format menu and select style.

**Header and Footer**

Header represents the top most part of a page and Footer the bottom part of a page. Header

and Footer contain page number, date, name and logo of the company, File name. To insert

Header or Footer in a document click on the View menu and select Header and footer.

Besides Insert Auto Text, we can see the icons to:

1) Insert Page Number

2) Insert Number of pages

3) Format Page Number

4) Insert Date

5) Insert Time

6) Page Setup

7) Show/Hide Document Text

8) Same as previous

9) Switch between header and Footer

10) Show Previous

11) Show Next

12) Close header and Footer

**Table**

In the ’Table’ menu select Insert and click on the Table icon. You will be given the option to

fill in the number of rows and columns of the table.

If you want to add two rows:

1. Click on the insert menu and select Rows.

2. You have two options: Rows Above and Rows Below.

3. If you click on Rows Below, a row will be added below the row in which you have m oved

the cursor.

**Graphics**

The use of graphics in a document:

Click on the insert menu and select picture. We can see the following options:

1) Clip Art

2) From File

3) Auto shapes

4) WordArt

5) From Scanner

6) From Camera

7) Chart

**1) Clip Art:**

When you click on Clip Art you will see the different category and click on it to see it on the

screen.

For Examples:

Click on places you will see the Clip Art symbolizing various places.

Click on the Taj Mahal.

You will see an icon of Insert Picture. Click on it.

The picture of the Taj Mahal will be inserted in the document.

2) **From File:**

Pictured stored in form of files can also be inserted in the document. Click on Insert, select

pictures .Click on From File to open a Dialog box. Select a file from the place where it is

stored in the computer to insert the picture in the document. You can change the size of

picture, remove a picture or even place it. In the centre of the text by the following the same

procedure as in the clipart.

**Printing:**

A document typed in word has to be finally printed it is necessary to know whether the

printer is Dot matrix ,color ink jet or laser printer. When a document is ready for printing ,click

on print preview icon on the menu bar to see how the document will appear on the paper.

This view can be enlarged or reduced.

**Lab Tasks**

## Task 1 – Typing practice

If you are new to word processing, you will probably benefit from some typing practice and need to do this entire task. Otherwise you can move on to section 4 (Editing a document).

1. Enter the following text in a new document:

Jeremy Bentham

The philosopher and jurist Jeremy Bentham (1748-1832) was born at Houndsditch, London, on 15th February 1748. He proved to be something of a child prodigy: while still a toddler he was discovered sitting at his father’s desk reading a multi-volume history of England, and he began to study Latin at the age of three. At twelve, he was sent to Queen’s College Oxford, his father, a prosperous attorney, having decided that Jeremy would follow him into the law, and feeling quite sure that his brilliant son would one day be Lord Chancellor of England.

Bentham, however, soon became disillusioned with the law, especially after hearing the lectures of the leading authority of the day, Sir William Blackstone (1723-80). Instead of practising the law, he decided to write about it, and he spent his life criticising the existing law and suggesting ways for its improvement. His father’s death in 1792 left him financially independent, and for nearly forty years he lived quietly in Westminster, producing between ten and twenty sheets of manuscript a day, even when he was in his eighties.

1. Save the file with the name **jeremy.doc** on the R: drive in the training.dir\word\getting-started folder.
2. Close the file.

You are now ready to do Exercise 1.

## Task 2 – Moving around a document

1. Open the file **bentham.doc** on the R: drive in the training.dir\word\getting-started folder.
2. Use the **Page Down** key to find the paragraph beginning: *Even for those who have never read...*
3. From this position, using the keyboard, move one word at a time to the text **Bentham**.
4. Using the keyboard, move to the end of the document in one keystroke. What is the last word in the document?
5. Using the mouse, move to the heading **The Auto-Icon**.
6. Using the mouse, move to the start of the document. What keyboard shortcut could you have used to carry out this operation?
7. Using the **Go To** command (on the *Edit* menu), move to page 2. What is the keyboard shortcut for this command?

## Task 3 – Basic editing

The file **bentham.**doc in the r:\training.dir\word\getting-started folder is used in this task.

Correct the line spacing as you complete the tasks to maintain a line break between paragraphs.

1. Cut the heading *The Auto-Icon* and use the Clipboard task pane to paste it above the paragraph beginning: *At the end of the South Cloisters...* (the paragraph above).
2. In the fourth paragraph down from the heading The Auto-Icon, the one beginning: *Not surprisingly, this peculiar relic has...* Add the following sentence to the end of the paragraph:

In these cases the Auto-Icon invariably votes for the motion.

1. The last two paragraphs in the document are in the wrong order; reposition them so that the paragraph starting: *Many people have speculated as to exactly why Bentham...* is the last paragraph.
2. Copy the heading *Jeremy Bentham* at the top of the document and paste it to form a new heading above the paragraph beginning: *Bentham is often credited with being one of the founders of the University of London...* (the fifth paragraph down).
3. Now copy both of the headings in the document (The Auto Icon and Jeremy Bentham) at once using the **Ctrl** key to select them simultaneously. Open a new, blank document and paste the copied headings into it using the Clipboard task pane. Notice that all items which you have copied or pasted in the **bentham.doc** document are now available on the Clipboard task pane in the new document.
4. Close the new document without saving it, but save the **bentham.doc** file.

**You are now ready to do Exercise 2**.

## 

## Task 4 – More basic editing

The file **bentham.doc** in the r:\training.dir\word\getting-started folder is used again for this task.

1. Add the text and UCL to the Jeremy Bentham heading you have just copied so that the new heading reads *Jeremy Bentham and UCL*.
2. Now change this heading by overtyping the existing text (Jeremy Bentham and UCL) and replace it with the heading UCL and the Founding Father.
3. In the paragraph above, delete the sentence beginning: *Research into his work continues at UCL in the Bentham Project...* . (Remember that you can select the sentence and delete in one action.)
4. Undo all of the above steps in one go, i.e. back to the point where the subheading is returned to Jeremy Bentham and UCL.
5. Save the file with the same name – **bentham.doc**.

## Task 5– Find and replace

The file **bentham.doc** in the r:\training.dir\word\getting-started folder is used for this task.

1. The name Jeremy has been misspelled as **Jeremey** in several places in the text. Using **Find** **and** **Replace**, correct these mistakes.
2. The word **college** as it occurs in the text should be capitalised. Replace the word college with the correct capitalisation **College** throughout the text.
3. Find the word **defiant** and replace it with the word **pugnacious**.
4. Find the word **utilitarian** and replace it with the word **pragmatic** (take care not to replace the word utilitarianism).
5. Save the file with the same name – **bentham.doc**.

## Task 6 – Spelling

The file **bentham.doc** in the r:\training.dir\word\getting-started folder is used for this task.

1. Position the insertion point at the top of the document.
2. Spell-check the document. There are a number of spelling mistakes in the document. Correct the mistakes. Listed below are some of the errors that the spell-checker will identify:
3. The name Bentham is identified as not being in the dictionary. Choose **Ignore**.  
   The spell check will pick up the same name ‘Bentham’ again. Choose **Ignore** **All** to prevent this from happening or **Add** the word to the dictionary. The same problem occurs with the text **Bentham's**.
4. The word **contractarianism** has been identified as not in the dictionary, **Add** this word to it.
5. The word as has been repeated in the text, delete the second occurrence.
6. There are several names **Tonks**, **Flaxman**, **Wilkins**... which are identified as not being in the dictionary, choose to **Ignore** these names.
7. The word **nda** is identified. There are no suggestions for this word. Retype the text in the **Change****To** box – it should read **and**.
8. The word his has been repeated in the text. Delete the second occurrence.
9. Save the corrections and close the file.

## Task 7 – Character formatting

The file **bentham.doc** in the r:\training.dir\word\getting-started folder is again used for this task.

1. Select the whole document and change the document’s font to **Palatino** and point size to **12**.
2. Format the headings in the text (there should be three) so that they are bold and italic.
3. Format the credit at the end of the document so that it is italic.
4. In the third paragraph: *Even for those who have never...* select the quote at the end of the first sentence: *the greatest happiness of the greatest number*. Format the quote using the **AvantGarde** font and **italicise** the text.
5. Using the *Format* *Painter*, paste the format used above and apply it to the credit at the end of the text.
6. Change the format of the title **Jeremy** **Bentham** to the **Garamond** font, **16** pt, **bold** and remove the underlining.
7. Paste the above format to the other headings in the document. (Remember to double-click on the Format Painter button, when a format is to be copied more than once.)
8. Save the file with the same name: **bentham.doc**.

## Task 8 – Paragraph formatting

The file **bentham.doc** from the R: drive in the training.dir\word\getting-started folder is used in this task.

1. Select the whole document and, using a button on the toolbar, change the line spacing from single to 1.5 lines.
2. Fully justify the text using a button on the toolbar.
3. Move to the bottom of the document, leave a couple of empty lines and add the text::

For further information visit the UCL Web site.

1. Format the text to make it bold and italic.
2. Save the file with the same name: **bentham.doc**.

**You are now ready to try Exercise 10.**

## Task 9 – Views and page setup

1. Using the document **bentham.doc**, familiarise yourself with the different viewing modes by switching between them.
2. What view are you in if:

* the horizontal and vertical rulers are visible, page breaks are represented by areas of grey shading on the screen, and margins are represented by shading on the rulers and defined in white on the document page?
* the document occupies the full screen and there are no other screen elements?

1. Find out how a page break is represented on screen in **Normal** view.
2. Go to the *File* menu and click **Page** **Setup**. Change all the margins (top, bottom, left and right) to 3cm (1.18").
3. Set the *orientation* to **landscape** for the whole document.
4. View the document using *Page* *Layout* view.
5. Set the orientation back to **portrait** for the whole document.
6. Use the *Zoom* control to zoom in and out of the text to gain a different perspective of the document. What is the largest zoom and the smallest zoom available?
7. What zoom percentage is the screen displayed at if you select **Page Width** from the **Zoom** menu?
8. Close the document without saving it.

LAB NO. 2

Introduction to Microsoft PowerPoint

## Introduction:

Microsoft PowerPoint, part of Microsoft Office, creates and plays presentations. A presentation is something a speaker makes to an audience, typically using a computer and LCD projector to display material in a lecture hall or auditorium. PowerPoint works a lot like Microsoft Word, and the assumption here is that you are familiar with Word. A PowerPoint presentation is made up of "slides" that are individual frames or screens of information. To create a presentation, create the slides. A PowerPoint file (\*.ppt) is a collection of slides, typically for one and only one presentation, although files can be linked together to make up compound presentations.

# Objectives of This Lab:

* Creating and inserting new slides.
* Editing existing slides.
* Reordering existing slides.
* Learn how to make a simple power point presentation on a given topic.

# Time Division: (3 hours)

* Introduction to MS power point (lecture duration = 45 min)
* Practice tasks (2 hrs)

**Starting PowerPoint**



**Double click quickly** on the **PowerPoint 2007** icon on the **Windows desktop** (see **image** on **right**), or **click** the **Start** button in the **lower left corner** of the screen, and then **clickAllPrograms,** next move your cursor over **Microsoft Office,** then click **MicrosoftPowerPoint 2007**.

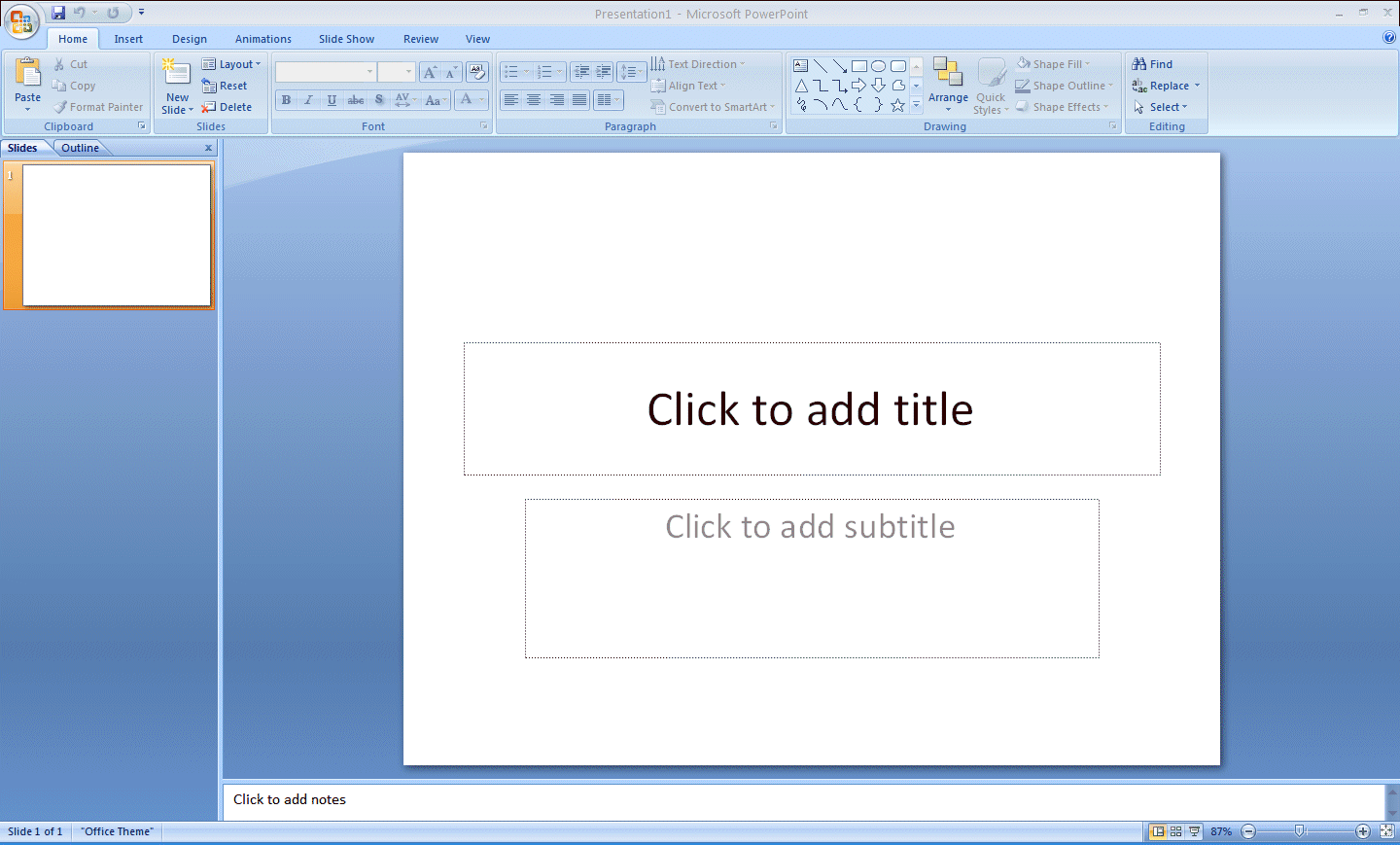
In this tutorial, whenever we indicate that you need to **click** a **mouse button**, it will mean to **click** the **left mouse button** – unless we indicate that you should click the right mouse button. So, always move the cursor over the “place” we indicate and “click left” unless we tell you otherwise.

In the image below you’ll immediately see that the **Menu Bar** has been **replaced** by **Tabs** and **Ribbons**. The **Tabs** and **Ribbons** are then **divided intoGroups**. We’ll be working with these new features in detail as we move through the tutorial.

**Tabs**

**Groups**

**Ribbons**

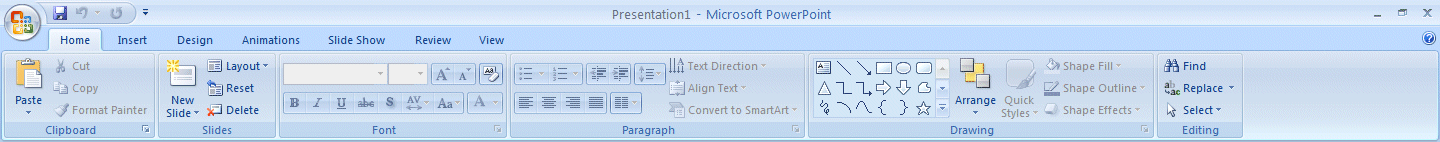


**Notice,** in **the image** on the **last page,** that the **screen** is “sort of” **divided** into **three sections**.

**Across the top** are the **Microsoft Office Button**, the **Quick Access Toolbar** and the Tabs, Ribbons and Groups (indicated on the last page).

**Microsoft Office Button**

**Quick Access Toolbar**

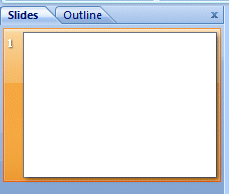


**Microsoft Office Button**

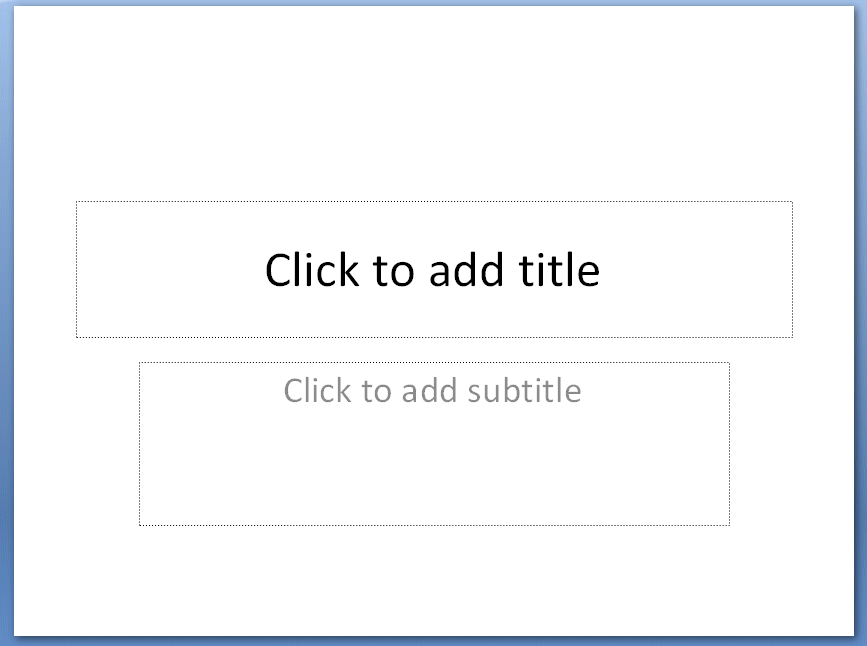
**Quick Access Toolbar**

PPT 2007 07

If you **refer** to the **Introduction to Microsoft 2007 Tutorial** you’ll find a **detailed explanation** of how to use these new features.



On the **left side** of the PowerPoint screen you’ll see an **area** that indicates **Outline** and **Slides** at the **top**. When you first open PowerPoint 2007 you’ll **notice** that the **Slides Tab** is “**white**.” This means that you’ll be able to **see a small version of each slide** as we create it.



**Beginning the presentation**

**Slide 1**

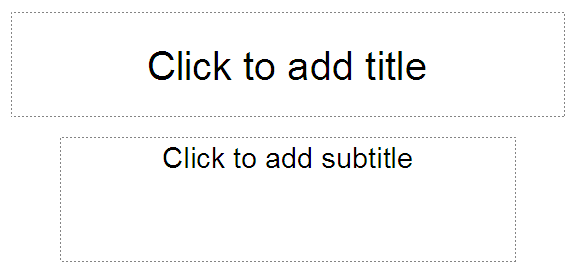
In PowerPoint 2007 a **Slide Layout** named **Title Slide always appears first**. PowerPoint “thinks” that you want to start your presentation with a title. So, logically, the Title Slide appears in the main section of the screen.

After you understand PowerPoint a bit more, you can choose any of the layouts you desire. We’ll show you how to do this as we proceed through the tutorial.

PPT 2007 09

You will notice, in the **lower left corner of the screen, Slide 1 of 1** is indicated.

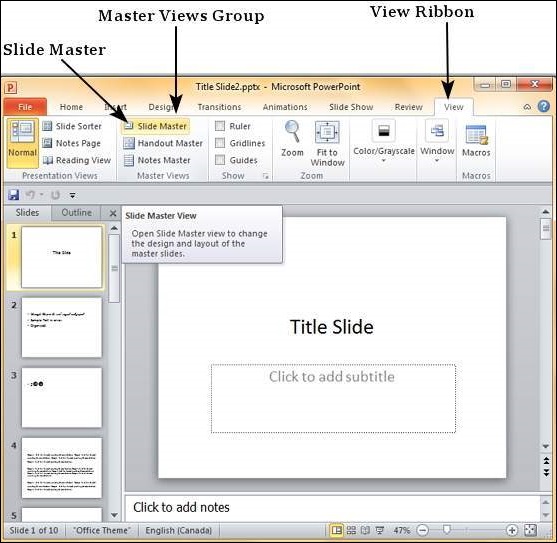
You will also see that your **screen looks like** the **image below**.



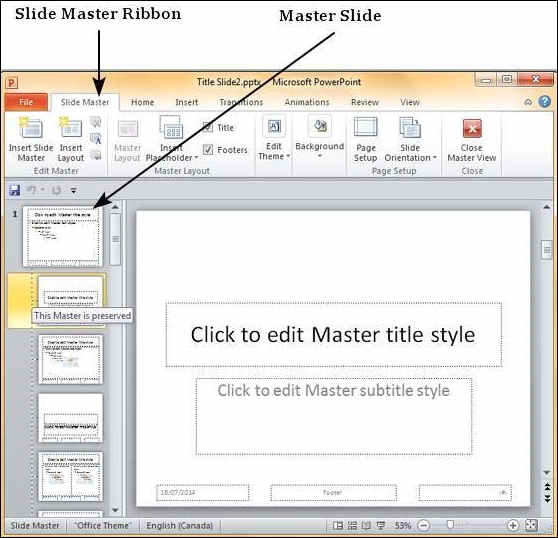
**Slide master**: Slide master is simple way of applying changes to the entire slide. Every presentation has at least one slide master, but you can have more than one. Using features like copy/ paste and format painting you can make changes to different sections within the presentation to make them look similar, however, if you want to use a theme and background throughout the slide, using the slide master is a simpler and more graceful way of approaching it.

Given below are the steps to customize your slide master.

**Step 1** − Go to the **Master Views** group under the **View**ribbon.

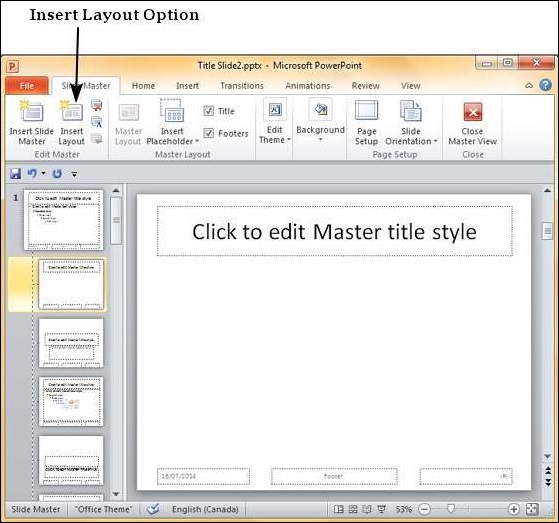


**Step 2** − Click on **Slide Master** to open the **Slide Master**Ribbon. The top most slide in the left sidebar is the **Master** slide. All the slides within this master template will follow the settings you add on this master slide.

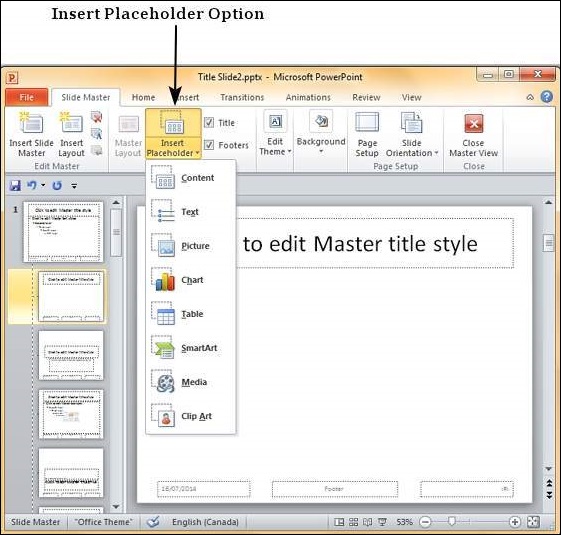


**Step 3** − You can make changes to the master slide in terms of the theme, design, font properties, position and size of the title and other content using the remaining ribbons which are still accessible.

**Step 4** − While PowerPoint provides some default slide layouts, you can create your own layouts by clicking on the "**Insert Layout**" in the Edit Master section of the Slide Master ribbon.

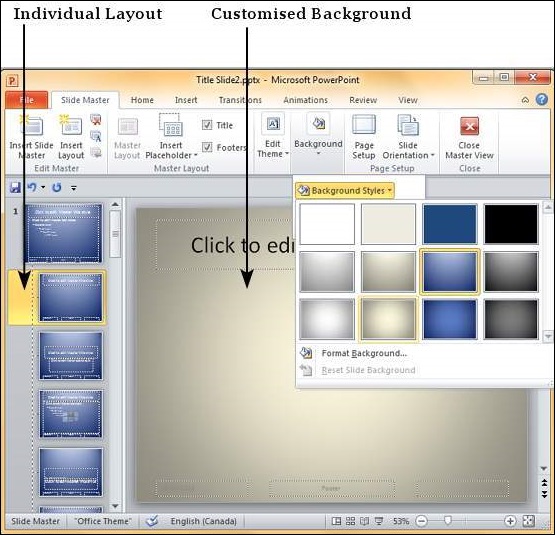


**Step 5** − You can add content placeholders to the slide layouts using the "**Insert Placeholder**" in the Master Layout group under the Slide Master ribbon. Under the Placeholder dropdown, you can either create a generic content placeholder or specify the kind of content you want in that placeholder.



**Step 6** − You can apply different themes, background and page setup settings to all the slides from the master slide

**Step 7** − You can also customize individual slide layouts to be different from the master slide using the menu options available with the layouts.



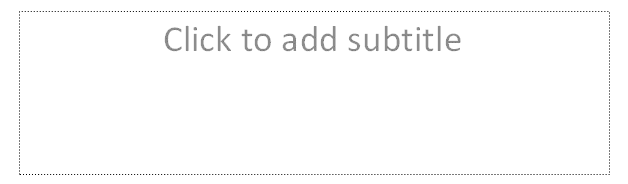
Now we’ll have some fun and create a PowerPoint 2007 presentation on how to make a Peanut Butter and Jelly sandwich.

Place your cursor in the “**Click to add title**” box and **Click the** left mouse button. Your text box, after you click, will look similar to the one below.



To insert the text in this formatted text box, we simply enter (**type-in**) the title: **How to Make a Great PBJ –** go ahead and type this text in the box**.**

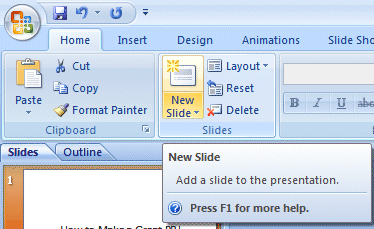
Now**, Click** in the second box “**Click to add sub-title”** and **type**:

**A Gourmet Recipe (**tap the **Enter** key**)**

**From (**tap the **Enter** key**)**

**Your Name (type in your name like Janie Schwark or Greg Butler)**

**New Slide Button**

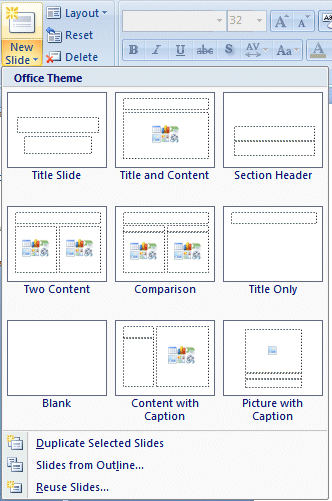
Now it’s time to create the next slide in your presentation. To do this, we’ll need to **find** the **New Slide** button.

At the top left of the screen, in the **Home Tab** you will see a **New Slide “button”** which looks like the **image** on the **right.**

PPT 2007 13When you **move** your **cursor arrow over** the button you will **see** a **Microsoft Help Text box** appear that says **New Slide**.

Now this is a **bit tricky**….. **Look** carefully at the **lower right corner** of the **New Slide button** and you will see a **down arrow**.

**Click-on the down arrow** to create your **next** slide.

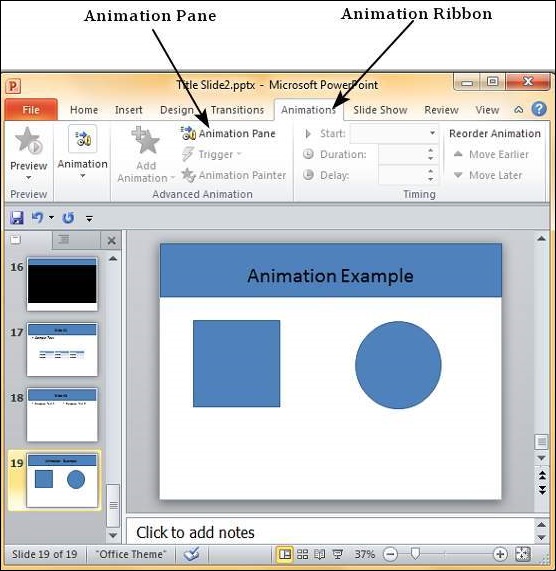
If you **accidentally click** the **button**, and a new slide appears, **don’t worry**, we’ll show you how to change to the slide format you desire later in the tutorial.

When you **click** the **arrow** an **image** similar to the one on the **right** will appear. We’ll use the **Title and Content** slide for our second slide. C**lick** this choice. In the similar manner keep on adding new slides and add contents and pictures as per your choice.

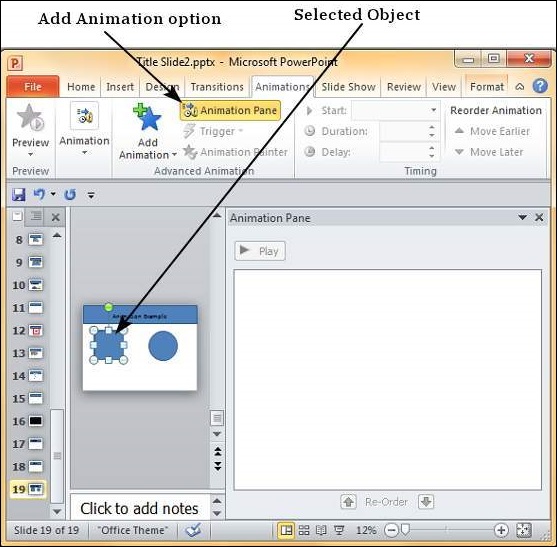
**Animation**: PowerPoint offers animation support which can be used effectively to add some motion in a monotonous presentation and make it more interesting. Animation can be applied to any object on the slide and the motions can the automated, timed or trigger.

The following steps will help you add and preview animations in the slide.

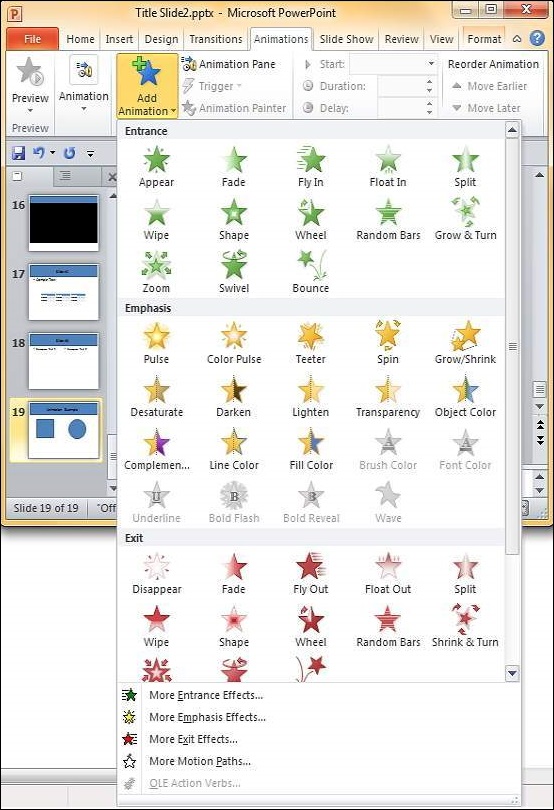
**Step 1** − Go to the **Animation** ribbon and click on the **Animation Pane** to display the animation sidebar.



**Step 2** − Select one of the objects in the slide and click on the **Add Animation** menu option.

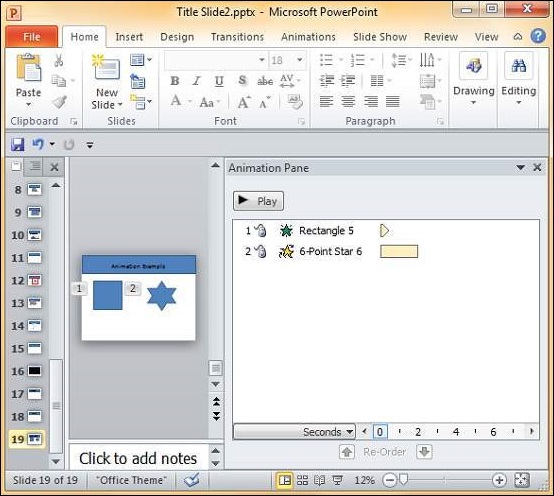


**Step 3** − Choose from one of the Animation options.

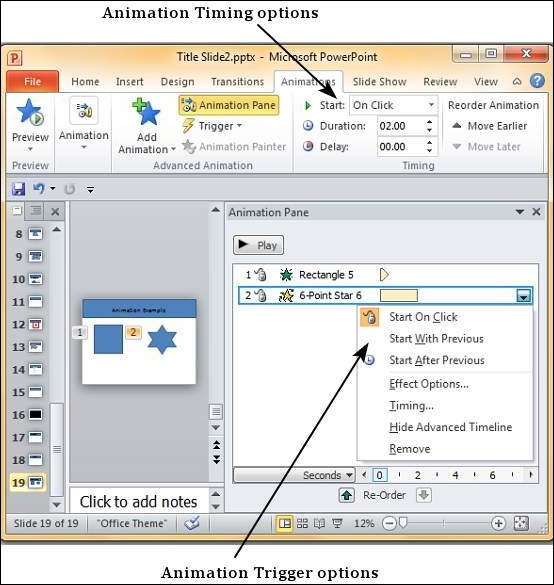


* **Entrance** will cause the object to appear in the screen.
* **Emphasis** will cause the object to emphasis without appearing or leaving the screen.
* **Exit** will cause the object to disappear from the screen.

**Step 4** − Once you add the animation for an object, it will show up in the **Animation pane**.



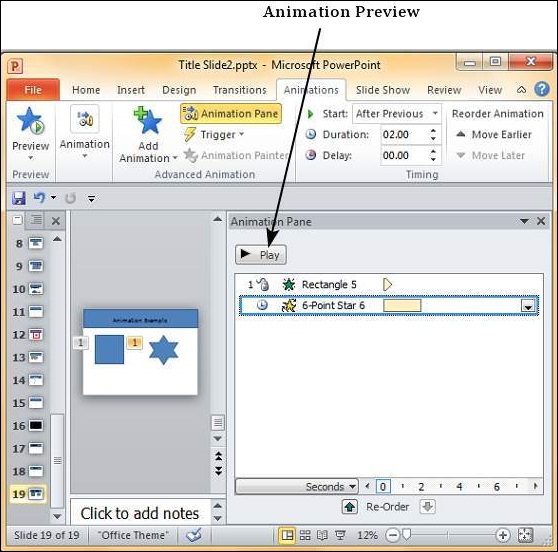
**Step 5** − By default, all the animations are initiated by a click, but you can change this. To change the trigger, right-click on the animation object on the pane and choose an alternate trigger.



* **Start On Click** will cause the animation to start when you click the mouse.
* **Start With Previous** will cause the animation to begin with the previous animation; if this is the first object, it will begin as soon as you reach the slide during the slide show.
* **Start After Previous** will cause the animation to begin after the previous animation ends.

**Step 6** − From the timing section, you can also manipulate the animation timings.

**Step 7** − To preview the animation settings, just click **Play** on the animation pane.



# OVERALL STEPS:

1. **Start with a template or theme.** A template or theme will dress up your presentation with pre-made layouts and color schemes. Choose one by clicking the Design tab.
2. **Browse through the slides from the template and see what you like.** On the sidebar at left, you can click different template or theme slides to see what they look like. Here are a few things you can do with the different slides:

* Duplicate slides. Right click on a slide, and select Duplicate Slide.
* Delete slides. Right click on a slide and select Delete Slide, or click Home on the top bar, then Delete Slide.
* Change the layout of your slides. You can select slides that have more or less text boxes, photo boxes, or any other template items you want. Select a slide, right click, and hover over Layout. Or, click Home on the top bar, and then the drop menu next to Layout.
* Add slides. If you want your new slide to look exactly like the one before it, right click that slide and select New. Or, if you want a new slide with a completely different layout, click Home, then the drop menu under New Slide.
* Organize your slides. You can drag and drop slides in the sidebar to the left to reorder them.

1. **Start adding content.** Here are some tips for putting information into your PowerPoint presentation:

Use short, concise words to guide your audience, and let yourself do the detailed explaining. Keywords show that you know your subject when you go more in-depth during your powerpoint. For example, use "Fire Kiln" as a keyword in a PowerPoint, but explain the process during the actual presentation.

Think bullet points. Don't use full sentences in your PowerPoint presentations unless it's absolutely necessary.

Don't be afraid to spread information across multiple slides. It's better than overcrowding a PowerPoint!

1. **Add elements.** To insert anything (text, charts, etc), simply click on the designated box to activate it and start inserting away.

Inserting pictures and graphs is a good idea to make your presentation more visually engaging. Break up your text!

Use color effectively in your PowerPoint. Have a theme of colors and be consistent when using them to highlight key points. This makes your presentation look more professional.

1. **Test run your presentation.** Click Slide Show, then ‘from beginning’or hit F5 to admire your handiwork.

**TIPS:**

1. Keep it Simple. PowerPoint uses slides with a horizontal or “Landscape” orientation. ...
2. Limit bullet points & text. ...
3. Limit transitions & builds (animation) ...
4. Use high-quality graphics. ...
5. Have a visual theme, but avoid using PowerPoint templates. ...
6. Use appropriate charts. ...
7. Use color well. ...
8. Choose your fonts well.

**LINKS:** [http://www.wikihow.com/Create-a-PowerPoint- Presentationhttp://www.garrreynolds.com/preso-tips/design/](http://www.wikihow.com/Create-a-PowerPoint-%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20Presentationhttp://www.garrreynolds.com/preso-tips/design/)

<https://www.youtube.com/watch?v=iqI149PD4v4>

<https://www.tutorialspoint.com/powerpoint/powerpoint_using_slide_master.htm>

**CLASS TASK: [Expected time=2 hrs]**

**Make a power point presentation on Android phone v.s. iphone. (minimum 10 slides)**

**Slide1:** Choose master slide and theme. Choose layout of first slide and add title and content. (topic, your name and id)

**Slide2:** Add some picture. Do formatting and apply animation.

**Slide3:** Give an outline of your presentation.

**Slide4:** Give salient features of android

**Slide5:** Give salient features of iOS

**Slide6,7:** Make slides with Headings covering security aspects, programming language, etc.

**Slide8:**Prepare a slide with graph showing the market share of both technologies

**Slide9:**Make a comparison using the given layout telling which one is better

**Slide10:**Give references

**Lab 3**

**Introduction to Microsoft Excel**

**Lab 3: Introduction to Microsoft Excel**

**Objective:**

* To introduce students to Microsoft Excel Tool
* To give tasks that enable students to develop an understanding of working on a spreadsheet

**Topics to be covered:**

* Introduction
* Familiarization with different tabs
* Practice tasks including excel formulas, charts, if else, etc.

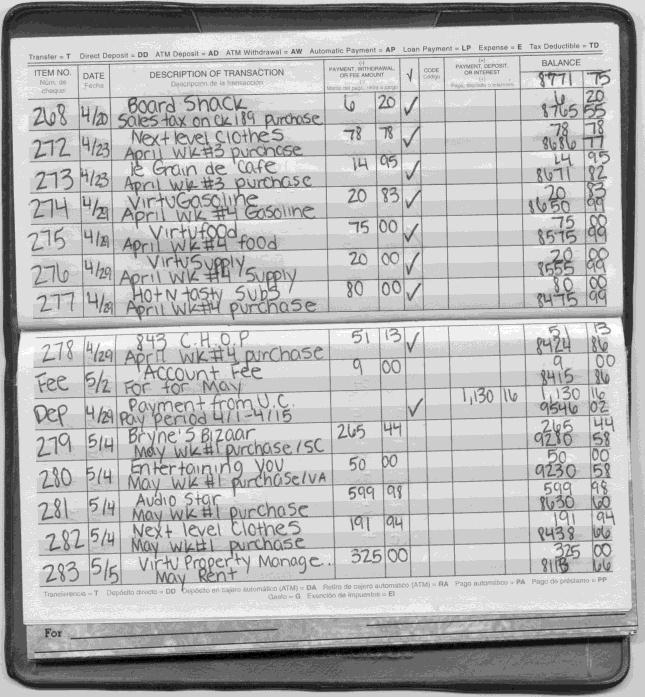
1. **Introduction [Expected time = 10 min]**

In this lab you will learn the basic working of Microsoft Excel. Like MS Word and MS PowerPoint, MS Excel is a part of Microsoft Office Suite and there are several features that you may find similar to MS Word and MS PowerPoint. We will however, concentrate only on those features that are specific to MS Excel.

MS Excel is an incredibly powerful tool for getting meaning out of vast amounts of data. But it also works really well for simple calculations and tracking almost any kind of information. Microsoft Excel can be used to create and manage business transactions that deal with accounting. The task you can complete with Excel ranges from preparing a simple family budget, preparing a purchase order, creating an elaborate 3-D chart, or managing a complex accounting ledger for a medium size business.

**What is a Spreadsheet?**

Suppose you would like to maintain personal finance register also known as check register that will keep track of the expenditures that you have been making during the current semester. Another scenario can be the budget management of a small house. You may be interested in question such as how much money has been spent on electricity bills and telephone bills. How much saving has been made? You may imagine a book that keeps record of all transaction/ expenditures and income. Such a book can be called a manual or paper spreadsheet. A sample is shown in the figure 1. In this case you may have to save data that can be numerical or alphanumeric (involving letters or numbers).



Spreadsheet is a tool that is used to organize data, such as a check register. Spreadsheets have been used for many, many years in business to keep track of expenses and other calculations.

**Electronic Spreadsheet:** Microsoft Excel is an example of spreadsheet application program that can be used for storing, organizing and manipulating data. The key benefit to using a spreadsheet program is that you can make changes easily, including correcting spelling or values, adding, deleting and formatting. It consists of a grid made from columns and rows similar to what you have seen in your Mathematics notebooks in school days. This grid environment makes number manipulation very easy.

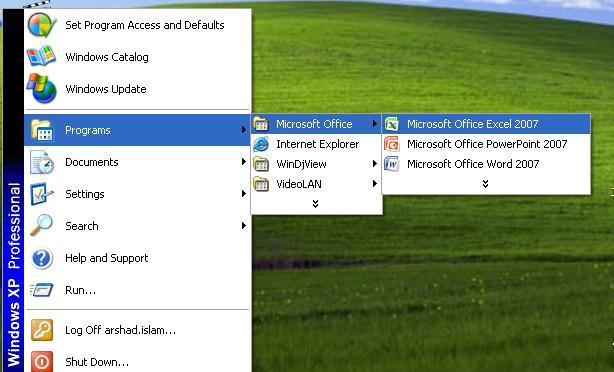
*Figure 1: Example Spread Sheet*

As always with computer programs, there is more than one way to go about these things. The instructions here are intended to be an easy introduction to the use of Excel. This lab manual includes a subset of Microsoft Excel features.

**Procedure and Tools**

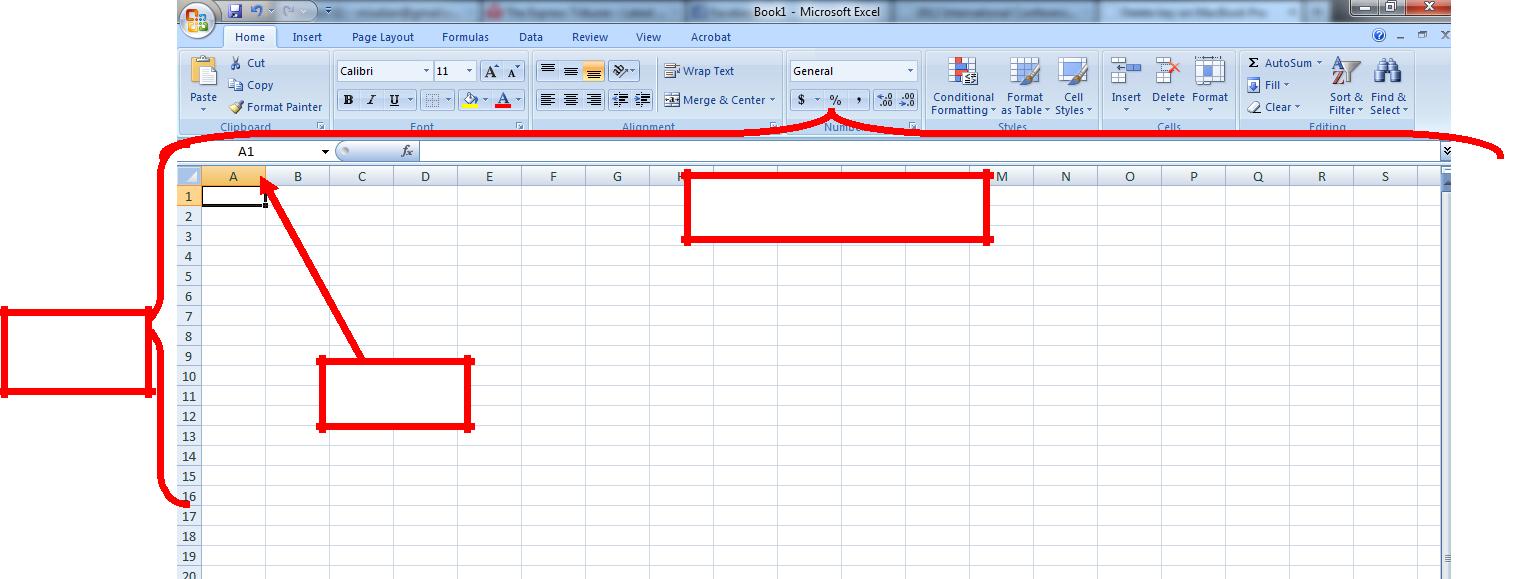
* + Desktop Computer
  + Microsoft Windows XP operating system
  + Microsoft Excel 2007

Microsoft Excel is an electronic spreadsheet. You can use it to organize your data into rows and columns. You can also use it to perform mathematical calculations quickly. Before you start working in Microsoft Excel, you need to open it. You can open Microsoft Excel by clicking on the Start button as shown in figure 2.



*Figure 2: Start Menu*

A screen similar to figure 3 will appear.



Columns

Rows

A1

*Figure 3: Excel Layout*

**1.1 Excel Layout**

Please refer to Figure 4 to observe the layout of MS Excel. On the top you will find a ribbon similar to MS Word and PowerPoint that we will be discussing shortly. The different thing you will notice is the rectangular boxes on the screen divided with the help of rows and columns

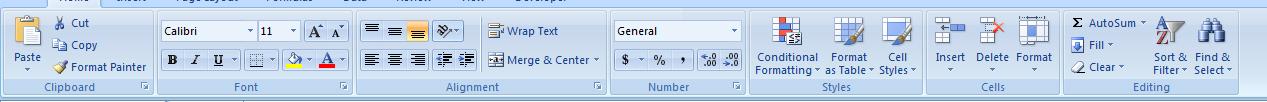
Rows are identified by Numbers as shown in the leftmost columns whereas Columns are identified by Alphabets as shown in the topmost row.

If you want to select a particular cell, you will have to identify its address. The selected cell in the figure 4.For.e.g. is A1, A being the first column and 1 being the first row.

**1.2 Walk Through Tasks**

|  |  |  |
| --- | --- | --- |
| 1.2.1 Home | **[Expected time = 20 min]** |  |
|  |  |

In this section we will go through step by step through each section of the Insert tab just like we did on the Home Tab. Below is a Snapshot of the insert tab in word ribbon (Figure 4). Later each section is explained individually in detail and practice exercises.



*Figure 4: Home Tab Ribbon*

Clipboard

This section of Home Tab is covered in MS WORD Lab

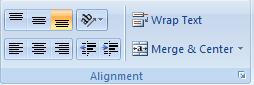
Font

This section of Home Tab is covered in MS WORD Lab

Alignment

This section of Home Tab is covered in MS WORD Lab-2.

In this section we would cover Merge and Center tab.



*Figure 5: Alignment*

For performing merge and center, you need to first select the cells that you want to merge. The selection of cells is shown in figure 5.



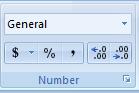
*Figure 6: Selection of Cells*

Once selection of cells is done, click on Merge and center command. When you would click the command the cells would be merged as shown in Figure 6.

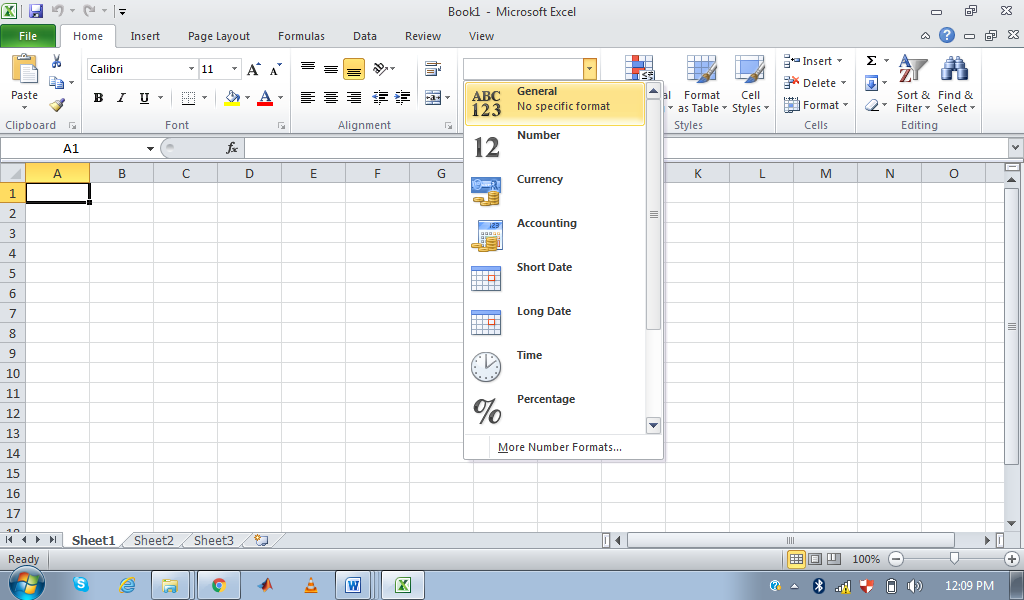
Number

In this section you can add various things with the number for e.g. you can add the currency, percentage sign, comma, decimal values after the number etc. Figure 7 shows the number section with different options displayed.

.



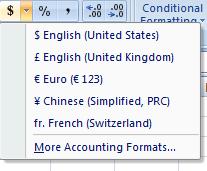
*Figure 7: Number Section*



*Figure 8: General command with options*

If you want to add currency to the number, click the drop down list of dollar sign. When you click the dollar sign various currency options are displayed e.g dollar sign, euro sign, pounds sign etc.

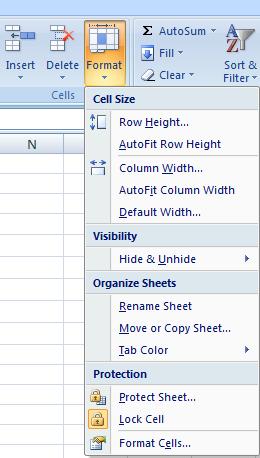
All these signs are shown in figure 9



*Figure 9: Adding Currency*

Cells

You can use a few clicks to adjust the row height and column width according to your desire. Click on “Format” in the cell section of the ribbon and a menu similar to the figure 10 below will appear.



*Figure 10: Cell Format Menu*

You can adjust the row height by clicking “Auto fit row height”. This will adjust the height of all the cells in a row according to the font size of the data you have inserted.

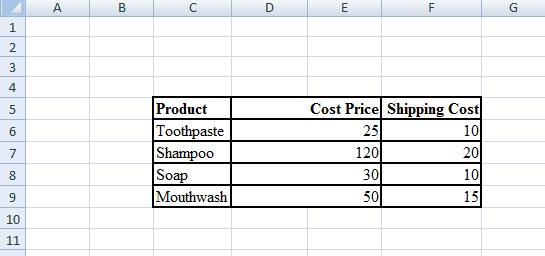
We will be doing tasks on the following example data:

1. Copy and paste this data into a newly created Excel File

Table 2: Sample data for practice tasks

|  |  |  |  |
| --- | --- | --- | --- |
| **Product** | **Cost Price** | **Sale Price** | **Shipping Cost** |
| Toothpaste | 20 | 25 | 10 |
| Shampoo | 110 | 120 | 20 |
| Soap | 20 | 30 | 10 |
| Mouthwash | 49 | 50 | 15 |

1. Now Left align the Product column while apply right alignment for the rest of the columns
2. Merge Cost Price and Sale Price in the new column “Price” and use Sale price value for it.
3. Click on Auto row height and column width to adjust the height and width of the columns of your data.
4. As the Column Price is a number and can be counted as currency, Modify setting of new column “Price” and add American dollar USD$ ad its currency.
5. In the end you will have something similar to the figure 11.



*Figure 11: Output of the First 6 Steps*

1. Note that Cost Price is now merged in two columns E and F instead of one column.
2. Now select the column for Cost Price and select Unmerge in the menu appearing after clicking the arrow with the merge button.

1. After Unmerging you will notice that cost price has been reduced to one column. Now right click on the empty column and delete to remove the empty column. As result you will have something similar to the figure 12.



*Figure 12: Output of 9 Steps*

Note that Cost price is now placed in the only Column D instead of both D and E as in the previous figure.

1.2.2 Insert Tab **[Expected time = 20 min]**

Insert Tab can be used if you want to add to graphics or link your document with other document. Insert tab in ribbon provides different options for inserting useful information to your excel sheet. The ribbon is shown in figure 13.



*Figure 13: Page Insert Tab in Ribbon*

Tables

This section of Home Tab is covered in MS WORD Lab

Illustrations

This section of Home Tab is covered in MS WORD Lab

Charts

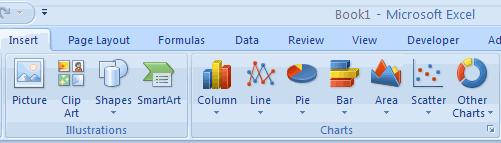
Charts allow you to present information contained in the worksheet in a graphic format. Excel offers many types of charts including: Column, Line, and Pie, Bar, Area, Scatter and more. To view the charts available click the Insert Tab on the Ribbon.

Creating a Chart:

– Select the cells that contain the data you want to use in the chart. In our example we can select the shipping cost column to be used as chart data

– Click the Insert tab on the Ribbon

– Click the type of Chart you want to create. There are different chart types asshown in figure 14.



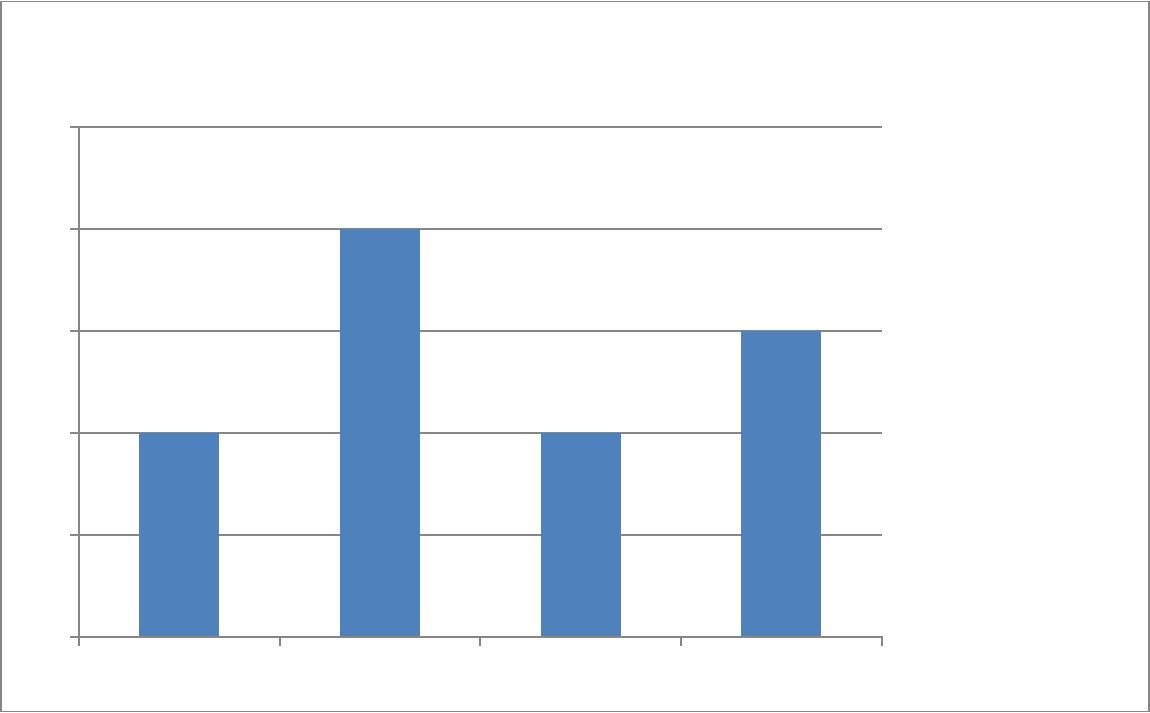
*Figure 14: Chart Tool*

– Click the Column chart button and then select the 2d chart

– A bar chart will appear showing the shipping cost as chart as shown in Figure 15.

– You can observe the costs at the Y-axis and as there are four items so you can see 4 bars at X-axis

– This chart has four components and you can right click on them separately to see the advance options available.



**Shipping Cost**

25

20

15

 Shipping Cost

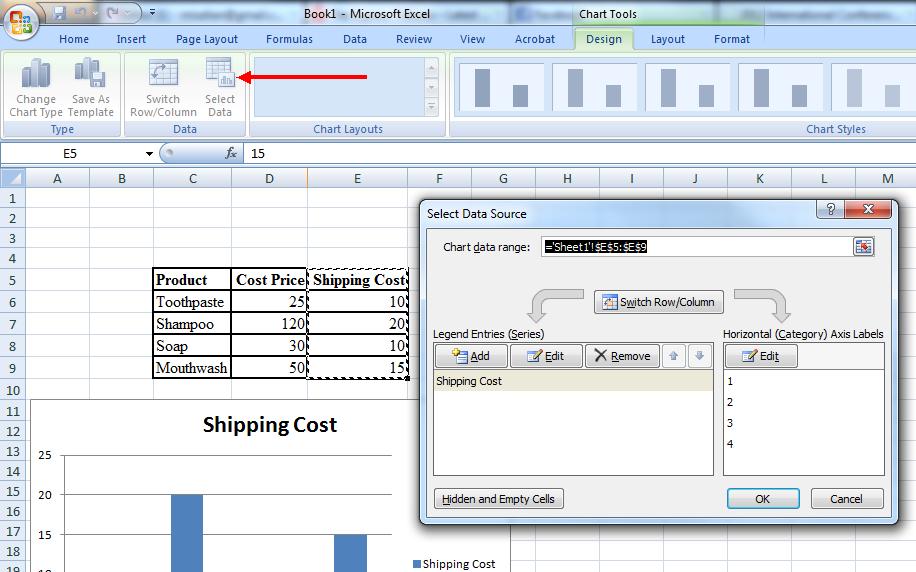
10

5

0

1 2 3 4

*Figure 15: Resulting Chart*



*Figure 16: Change Chart Data*

Modifying a Chart:

– Once you have created a chart you can do several things to modify the chart.

– Move the chart

To change the data included in the chart: o Click the Chart

O Click the Select Data button on the Design tab as shown in figure 16 o You can see the data selected in the “Chart data Range” box.

1. ='Sheet1'!$E$5:$E$9 tells us that we are using the data from sheet1 and our data starts from E5 and it goes till E9. The $ signs in the middle tell us that it is address of the cell and not the data inserted by user.

To reverse which data are displayed in the rows and columns figure 17 shows how to achieve this task.

– Click the Chart

– Click the Switch Row/Column button on the Design tab



*Figure 17: Reverse Rows and Columns Data*

Chart Tools

The Chart Tools appear on the Ribbon when you click on the chart. The tools are located on three tabs:

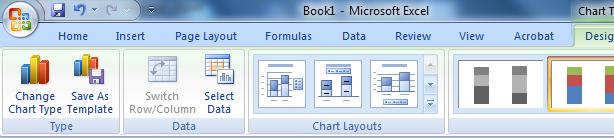
– Design

– Layout

– Format

**Design Tab**

Within the Design tab you can control the chart type, layout, styles, and location. This tab is shown in Figure 18.

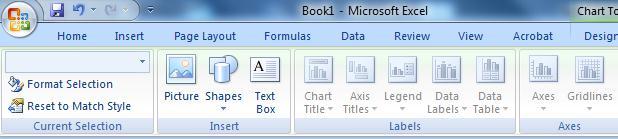


*Figure 18: Design Tab*

1. Select the chart that you have drawn and now apply different chart layout to see the differences.
2. Apply the different colors available in the ribbon such as green or orange.
3. Try moving the chart with the help of the Move chart button in the end and move it to another sheet of this Excel File.

**Layout Tab**

Within the Layout tab you can control inserting pictures, shapes and text boxes, labels, axes, background, and analysis. Layout Tab is shown in Figure 19.



*Figure 19: Layout Tab*

To modify the labels and titles:

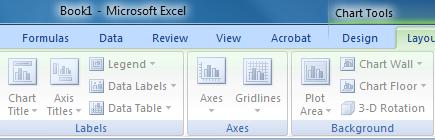
– Click the Chart

– On the Layout tab near the end of the ribbon,

– Click the Chart Title or the Data Labels button

– Change the Title and click Enter

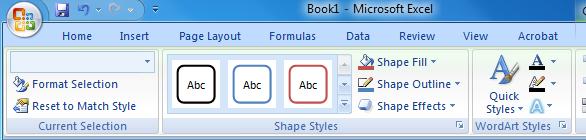
This step is represented in figure 20.



*Figure 20: Change X and Y Axis Label*

**Format Tab**

Within the Format tab you can modify shape styles, word styles and size of the chart. Format tab is shown in figure 21.



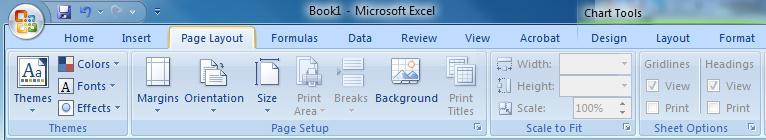
*Figure 21: Format Tab*

# 1.2.3 Page Layout

**[Expected time = 5 min]**

Page layout ribbon is shown in figure 22. This ribbon contains different options related to page

Setup. All these options have been covered in lab of Microsoft Word.



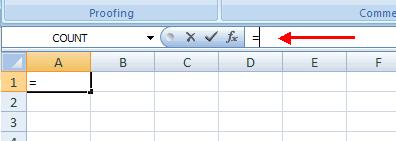
*Figure 22: Page Layout Tab*

1. Select the chart and apply different format type to see what effect it makes on your chart style and color.

# 1.2.4 Formulas

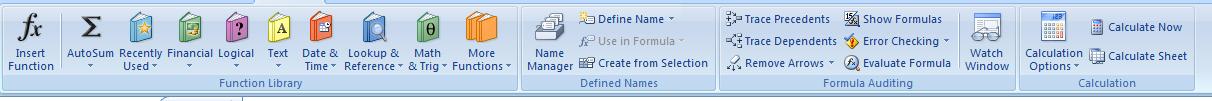
**[Expected time = 20-25 min]**

A formula is a set of mathematical instructions that can be used in Excel to perform calculations. Formals are started in the formula box starting with an = sign. A sample formula is shown in Figure 23.



*Figure 23: Formula Example*

The formula ribbon is shown in figure 24. This ribbon gives you different options for applying formulas. The first section represents Function library. Function library contains different formulas from various categories. Figure 24 gives you an overview of the Formula ribbon.



*Figure 24: Formula Tab*



*Figure 25: Data for Sum Formula*

Creating Basic Formula

To create a basic formula in Excel we need the data on which to apply the formula. Data is represented in Figure 25.

To create a basic formula you need to do the following:

– Select the cell for the formula

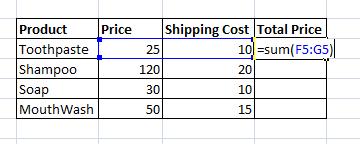
– Type “=” (the equal sign) and the formula

– The formula is of sum which would sum the values.

– “:” is used to specify the range of the cell on which you want to apply theformula

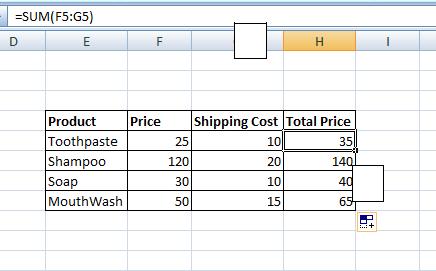
– Click Enter

Creating a basic formula is shown in figure 26.



*Figure 26: Sum Formula*

When you click enter you would get the result as shown in figure 27 the label 1 represents the formula applied to get the desired result and label 2 represents the result after applying the formula.



1

2

*Figure 27: Result of Sum Formula*

**Calculate with Functions**

A function is a built in formula in Excel. A function has a name and arguments (the mathematical function) in parentheses.

– Sum: Adds all cells in the argument

– Average: Calculates the average of the cells in the argument

– Min: Finds the minimum value

– Max: Finds the maximum value

– Count: Finds the number of cells that contain a numerical value within a range of

the argument **To calculate a function:**

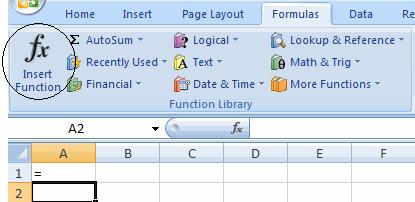
– Click the cell where you want the function applied

– Click the Insert Function button

– Choose the function

– Click OK

The process of calculating through a function is shown in figure 28.



*Figure 28: Function Computation*

**Function Library**

The function library is a large group of functions on the Formula Tab of the Ribbon.

– AutoSum: Easily calculates the sum of a range

– Recently Used: All recently used functions

– Financial: interest, cash flow return rates and additional financial functions

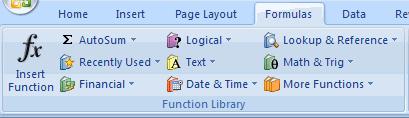
– Logical: And, If, True, False, etc.

– Text: Text based functions

– Date& Time: Functions calculated on date and time

– Math& Trig: Mathematical Functions

The categories of Function library is shown in figure 29



*Figure 29: Function Library*

**If-else Formula**

If-else is an important formula in Microsoft Excel. There are always situations in which you need to decide what you want to do when a particular stage is reached. Suppose you want to apply 5% of tax on all those products whose price is greater than 5000 rupees. To do such a task in Microsoft Excel, you need IF-ELSE formula. Before you apply IF-ELSE formula you need the data again on which you apply the IF-ELSE formula.

**Example 1:**

To apply the IF formula the data is given in figure 30.



*Figure 30: Sample Data*

Now we want to know whether the products are expensive or cheap. To do so we would apply the IF formula.The condition for expensive is greater than or equal to 50. If the product’s price is greater than or equal to 50 then it is expensive otherwise it is cheap.

To apply IF formula, you need to do the following:

– Select the cell where you want to apply the formula

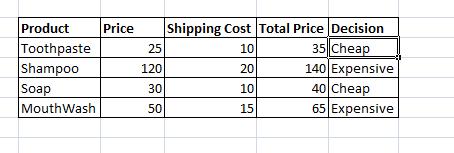
– Put = (equal sign) in that cell

– Then write the if formula as shown in figure 31



*Figure 31: IF formula*

When you apply the formula shown in Figure 6.26, you will get the result shown in figure 32.



*Figure 32: IF formula Result*

**Example 2:**

In the second example an Employee Salary table of a company is shown. We have to calculate the bonus for these employees. As you can see in the image the company has a strange criteria for giving bonus to their employees.

[](http://i1.wp.com/www.exceltrick.com/wp-content/uploads/2013/02/Excel-If-Statement-Example-2.png)

*Figure 33 EXAMPLE 2*

The criteria is, if the employee salary is greater than or equal to $ 8000 then bonus will be 15% of the salary otherwise the bonus will be 10% of the salary.

In this scenario we can use the Excel if Statement as: =IF(C2>=8000,C2\*15%,C2\*10%)

In this formula, first of all we check if the salary of first employee (in C2 cell) is greater than or equal to 8000. If this is true then the formula evaluates an expression (C2 \* 15%) otherwise the result should be calculated by the expression (C2 \* 10%).

As the salary of the first employee satisfies the condition i.e. (9,735 >= 8000). So, the result of this formula is (9,735 x 15%) which comes out to be $ 1460.25.

Similarly for the third employee the formula can be: =IF(C4>=8000,C4\*15%,C4\*10%)

Use of Logical Operators along with IF Statement:

Excel If Statement can also be used along with the logical operators (like AND, OR) for analyzing complex logics. Here I will help you to understand how can these operators be used with IF function.

|  |  |
| --- | --- |
| **Operator** | **Explanation** |
| AND | AND function only results into ‘True’ when all the conditions inside it are met. |
| OR | OR Function results into ‘True’ when any one of the conditions is met. |

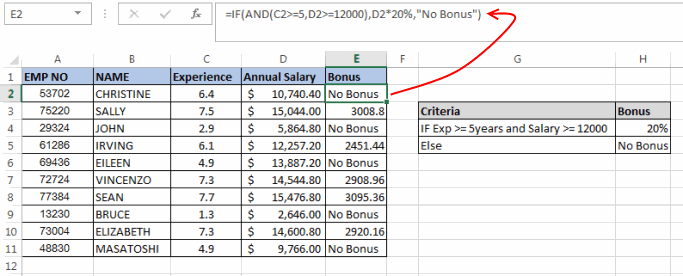
*Figure 34 LOGICAL OPERATORS*

The syntax of AND Function in Excel is: =AND(Logic1, Logic2, logic\_n)

The syntax of OR Function in Excel is: =OR(Logic1, Logic2, logic\_n)

Now let’s move to an example to understand how these functions can be used along with the IF function.

**Example 3:**

[](http://i1.wp.com/www.exceltrick.com/wp-content/uploads/2013/02/Excel-If-Statement-Example-3.png)

*Figure 35 EXAMPLE 3*

In this example, another company wants to give bonus to its senior employees. The company comes up with a criteria that any employee who has at-least 5 years of experience (5 years or more) and whose salary is greater than 12000 will be considered a senior employee. And such an employee will be eligible for a bonus equal to 20% of salary.

In such a scenario you cannot create an IF statement without using a logical operator. So, in this scenario we can create the if statement as:

=IF(AND(C2>=5,D2>=12000),D2\*20%,"No Bonus")

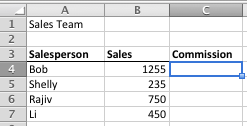
Here, in the AND function we have used two conditions, i.e. if C2 (Experience of 1st Employee) is greater than or equal to 5 and if D2 (Salary of 1st Employee) is greater than or equal to 12000. If both these condition are ‘True’ then only the output of AND will be ‘TRUE’ if any one of the value is ‘FALSE’ then AND Function will result into ‘FALSE’

In this example for the first employee the experience is 6.4 years but the salary is less than 12000. So, the first employee won’t be eligible for any bonus.

Nested IF STATEMENT:

**Calculating commission for a sales team based on monthly sales**

Imagine you have a sales team of five people, and you need to calculate their commission for the month based on their sales figures.

* You put the data into the following table:  
  
* Your commission plan works as follows:
  + If someone sells less than $400 in a month, they get 7% commission.
  + For sales between $400 and $750, they get 10% commission.
  + For sales between $750 and $1000, they get 12.5%
  + For sales over $1000, they get 16%
* Rather than calculate each of these commission figures individually, you decide to use a nested IF formula instead. The logical tests you would use in this case are these:
  + Is commission less than $400? If TRUE, then calculate commission.
  + If FALSE, then is commission less than $750? If TRUE then calculate commission.
  + If FALSE, then is commission less than $1000? If TRUE then calculate commission.
  + If FALSE, then calculate commission (because it must be more than $1000 - we don't need to do another logical test for this).
* The formula to represent this to calculate commission for Bob looks like this (I've highlighted the logical tests in bold to help you understand now the formula is put together:  
   =IF(**B4<400**,B4\*7%,IF(**B4<750**,B4\*10%,IF(**B4<1000**,B4\*12.5%,B4\*16%)))
* This formula can be confusing the first time you look at it. Let me run through it again.
  + The logical test in the first IF statement checks if the sales figure in B4 is less than $400. If it is, it calculates commission at 7% and stops calculating. Otherwise, it must be greater than or equal to $400, so we move on to the next IF statement.
  + The logical test in the second IF statement checks if the sales figure in B4 is less than $750. We already know it must be more than $400 or we wouldn't have got to this point. If it is less than $750, it calculates commission at 10%. Otherwise it must be greater than or equal to $750, so we move on to the next IF statement.
  + The logical test in the third and final IF statement checks if the sales figure in B4 is less than $1000. If it is, it calculates commission at 12.5%. Otherwise, it must be greater than or equal to $1000, so it calculates commission at 16%. At this point there are no more IF statements, no more logical tests we need to do, and we have our answer.
* Here's how our spreadsheet example looks once the formula has been entered into the Commission column. I've put the formula for each sales person in the cell next to the commission calculation so you can see how it looks:  
  
* Check Rajiv's commission figure in the table above. Rajiv sold $750 in the month, which is right on the threshold between 10% and 12.5%. What commission does he get? He actually gets 12.5% because the formula checks whether his sales were **less than** $750, not less than or equal to $750. That means he receives commission at 12.5%. So it's important to pay attention to how you construct your logical tests.

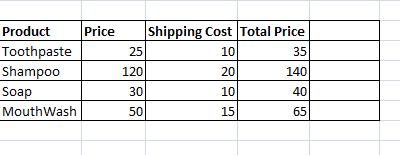
**Some things to remember when using nested IF functions**

Nested IF functions are very powerful, but there are some things to consider before you dive in and start using them.

* As you can see from the commission formula above, using nested IF statements gets complicated quite quickly. Trying to decipher this takes a moment or two, especially if you haven't looked at the spreadsheet in a while.
* Making changes is also challenging - imagine if you decided to introduce another commission threshold of $850. You'd need to add another IF function into the formula.
* You can have up to 64 IF functions nested in a formula in Excel 2007, 2010 and Excel 2013. Excel 2003 only supported 7 IF functions in one formula. That said, if I find myself needing more than 10 I start considering other approaches to my problem - managing that many IF functions in one formula quickly gets complicated, never mind if you have to manage 20, 30, 40 or more.
* The VLOOKUP function can sometimes be a better solution in a scenario like this. You can see a really good example of where you should [learn how the VLOOKUP function](http://fiveminutelessons.com/node/72) works in [one of the comments below](http://fiveminutelessons.com/comment/712#comment-712).
* That's not to say that VLOOKUP is automatically a better solution:
  + A commission spreadsheet like the one shown above is a good example where using VLOOKUP would be a better approach.
  + However, if you needed to use different calculations depending on the outcome of each logical test, then nested IF statements may well be a a better way to go.
  + An example might be where sales people who exceed $1000 in sales also get a bonus of $100. The formula would then look like the following, which is not so easy (although not impossible) to do with a VLOOKUP function:  
     =IF(**B4<400**,B4\*7%,IF(**B4<750**,B4\*10%,IF(**B4<1000**,B4\*12.5%,B4\*16%+100)))

**Percentage Formula**

There are some situations where you need to find some percent of a particular number, marks, price or the percentage of salary to incremented annually for employees. For all such tasks you need a percentage formula. We would apply a percentage formula for finding out how much sales tax to be applied on the products price. To do so we need the data i.e. prices of products. Figure 33 shows sample data for products



*Figure 33: Data for Percentage Formula*

We need to find 2% sales tax on product’s prices. To do so you need to do the following:

– Select the cell where you want to apply the formula

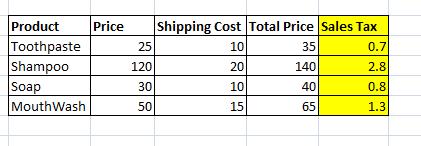
– Put = (equal sign) in that cell

– Apply the percentage formula as shown in figure 34



*Figure 34: Percentage Formula*

When you would apply the formula you would get the required sales tax as shown in figure 35. Sales tax is shown in Highlighted column.



*Figure 35: Percentage Formula*

# 1.2.5 Data

**[Expected time = 5 min]**

In Excel you need to play around data to achieve the desired goals. Therefore the Data tab has very much importance. In this tab you are given with various options that you can apply on the data in the Excel Sheet. We will not be covering everything from this ribbon shown in figure 36. We look at Sorting and filtering and how to get the external data from other sources.



*Figure 36: Data Tab*

Sort & Filter

Sorting and Filtering allow you to manipulate data in a worksheet based on given set of criteria. You can sort the data in ascending and descending order. There are two types of sorts in Microsoft Excel i.e. Basic Sorts and Custom Sorts.

Basic Sort

To execute a basic descending or ascending sort based on one column perform the following steps:

* Highlight the cells that will be sorted
* Click the Sort & Filter button on the Home tab
* Click the Sort Ascending (A-Z) button or Sort Descending (Z-A) button

These steps are shown in figure 37.



•

*Figure 37: Sort and Filter*

**Practice Tasks**

**TASK1** **[Expected time = 30 min]**

You are given the sheet below in Figure. You are supposed to create an Excel spreadsheet and fill in the data and then complete the tasks given below. The data is given in the table. Copy this data into a newly created Excel sheet.

Table 1: Sample data for Practice Tasks

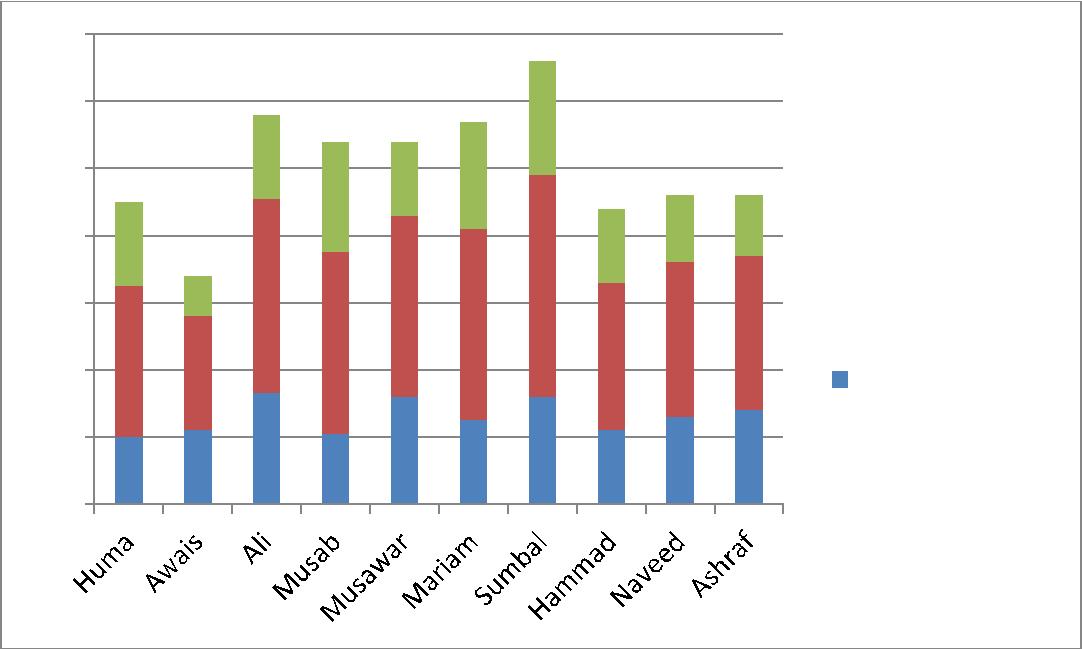
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **S. No** | **S. Name** | **1st Term (50)** | **2nd Term (50)** | **Obt. Marks** | **Total Marks** | **%age** | **Grade** |
| 1 | Huma | 20 | 25 |  | 100 |  |  |
| 2 | Awais | 22 | 12 |  | 100 |  |  |
| 3 | Ali | 33 | 25 |  | 100 |  |  |
| 4 | Musab | 21 | 33 |  | 100 |  |  |
| 5 | Musawar | 32 | 22 |  | 100 |  |  |
| 6 | Mariam | 25 | 32 |  | 100 |  |  |
| 7 | Sumbal | 32 | 34 |  | 100 |  |  |
| 8 | Hammad | 22 | 22 |  | 100 |  |  |
| 9 | Naveed | 26 | 20 |  | 100 |  |  |
| 10 | Ashraf | 28 | 18 |  | 100 |  |  |

* Calculate the obtained marks for the two terms
* After calculating the marks find the percentage
* Then apply the if formula and calculate the grades according to the MAJU scheme as given in the following table

Table 2: Grading Scheme for Practice Tasks

|  |  |
| --- | --- |
| **Grade** | **Marks** |
| A | >= 90 |
| A- | >85 &&<90 |
| B+ | >80 &&<=85 |
| B | >75 &&<=80 |
| B- | >71 &&<=75 |
| C+ | >65 &&<=71 |
| C | >61 &&<=65 |
| C- | >56 &&<= 60 |
| D+ | >52 &&<=56 |
| D | >50 &&<=52 |
| F | <50 |

* Sort the grades in ascending order
* Plot the data into a bar chart that shows the relationship between 1st term, 2nd term and total obtained marks. A sample is shown below in the figure.



140

120

100

80

60

 Obtained Marks

 2nd Term

1st te

**TASK 2** **[Expected time = 30 min]**

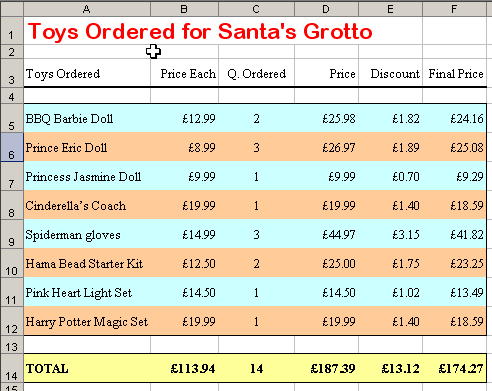
Open the workbook called:    **Plain Toys**from the folder shown above.

Make the following formatting changes:

* Insert some extra rows at the top and type in the title
* Format the font, size etc of the title
* Change the row heights to space them out more
* Change the vertical cell alignment of these rows to centered
* Insert more rows between the titles and first row of data and before the totals
* Format all the numbers except the Q Ordered column to show £ and 2 decimal places
* Change the column widths
* Line up the column titles with the numbers on the right
* Centre everything in the Q Ordered column
* Add borders, gridlines and shading as desired!

*Don't forget to change the £ sign right next to the number****s.***

Your formatted example will look something like this:



LAB NO. 4

Pivot Tables in MS Excel

**Objective:**

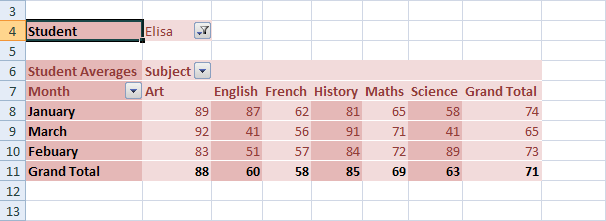
* To introduce students to Microsoft Excel Tool
* To familiarize students with the working of pivot tables

**Topics to be covered:**

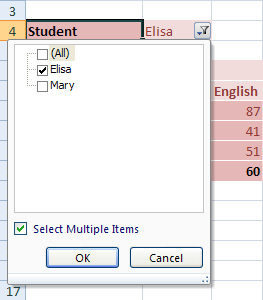
* Introduction to Pivot tables
* Practice tasks

## Introduction: [expected time: 1 hour]

A Pivot Table is way to present information in a report format. The idea is that you can click drop down lists and change the data that is being displayed. For example, choose just one student from a drop down list and view only his or her scores. Pivot tables are a lot easier to grasp when you see them in action. Here's the one we're going to create in this section:



Look at Row 4. This shows that the student is Elisa. If we click Elisa's drop down arrow, we'll see this:



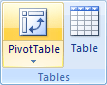
Now we have another student to select (we'll only use two students, for this tutorial). We could untick Lisa, and tick Mary instead. Then her scores would display.

The Subject and Month cells also have drop down lists. So we could view only January's scores, and just for Art and English, for example.  
So this is a **Pivot Table** - **a report that we can manipulate by selecting items from drop down lists**. Let's make a start.

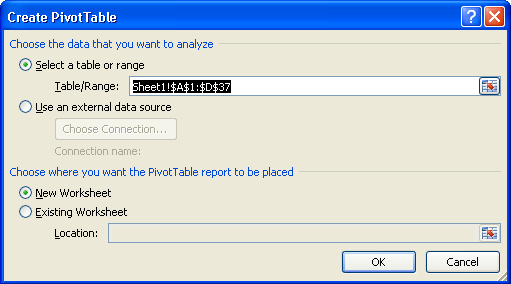
The first thing you need for a Pivot Table is some data to go in it.

[The Pivot Table Data in an Excel 2007 Spreadsheet](http://www.homeandlearn.co.uk/excel2007/excel2007s7p7.html) (New window)

Highlight the data that will be going in to your Pivot Table. On the Excel 2007 menu bar, click **Insert**. From the Insert menu, locate the **Tables Panel**:



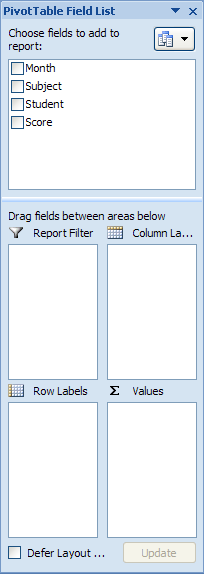
On the **Tables** panel click **Pivot Tables**. The **Create Pivot Tables** dialogue box appears:



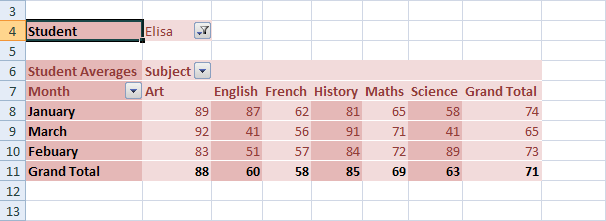
In the dialogue box above, the data that we highlighted is in the **Table/Range** textbox. You can select different cells by clicking the icon to the right of the Table/Range textbox. You can also specify an external data source, such as a text file, for the data in your Pivot Table.

We've selected a **New Worksheet** as the place where the Pivot Table will be placed. Click OK.

When you click OK, Excel 2007 presents you with a rather complex layout. The area on the right should look something like this one below:



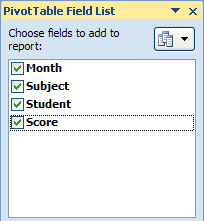
It helps to have a look again at what we're trying to create. Here's the completed Pivot Table again:



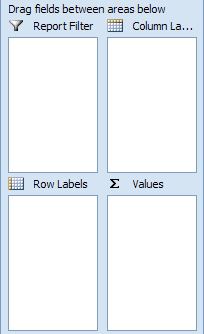
Now take a look at the **Pivot Table Field List** image again, the one above the completed pivot table. It has tick boxes for Month, Subject, Student, and Score. These are column headings from the original spreadsheet data. We've put the Month in cell A7 on our Pivot Table, Subject is in cell B6, Student is in cell B4, and Score is the Average scores in cells B8 to G10. You'll see how it works, though.

The idea is that you tick a box in the Pivot Table Field List, and then drag it to the four areas below. Excel 2007 will take care of the rest.

So, tick all four boxes in the field list:

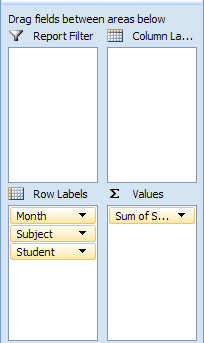


Excel will create a basic (and messy) Pivot Table for you. But we're going to put our 4 fields into the 4 areas below. Here's the 4 areas we can drag to:



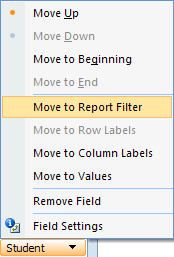
For the **Report Filter**, we want the name of a Student. For the **Column Labels**, we want the Subject, and for the **Row Labels**, we'll just have the Month. The **Values** will be the Average scores.

If you look at the Field areas after you have ticked all four boxes, however, you may see something like this:

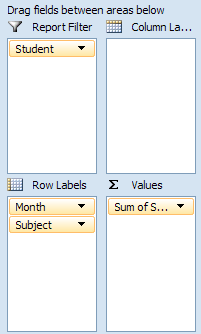


Month, Subject and Student have all been grouped under **Row Labels**. You can drag and drop these, though.

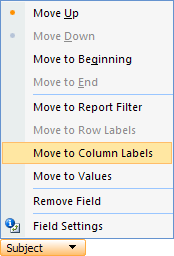
So click on **Student** in the **Row Labels** box. Hold down your left mouse button, and then drag it in to the **Report Filter** box. If you don't fancy dragging and dropping, simply click the Student item with your left button. From the menu that appears, select **Move to Report Filter**:



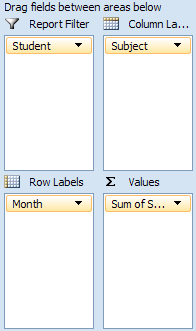
Your Field areas will then look like this:



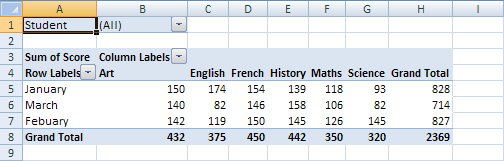
Move Subject from **Row Labels** to the **Column Labels** area:



Your Field areas will then look like this:



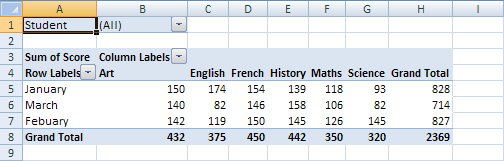
The Pivot Table on your spreadsheet will look a lot different, too. It should be looking like this:



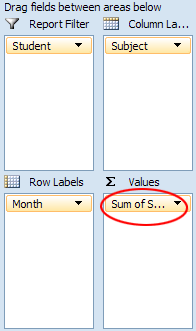
Our Pivot Table is coming along, but the scores are all wrong, and it needs tidying up a bit. We'll continue this tutorial in the next part.

# The reason why the scores from our Pivot Table are so strange is because Excel 2007 is using the wrong formula. It's using a Sum total when we want it to use an Average.

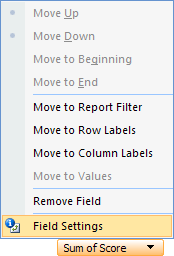
Here's the Pivot Table so far:



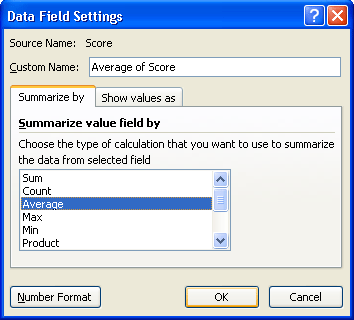
The numbers have all been added up. But we want averages, instead. To change the formula, click on **Sum of Score** under the **Values** field area:



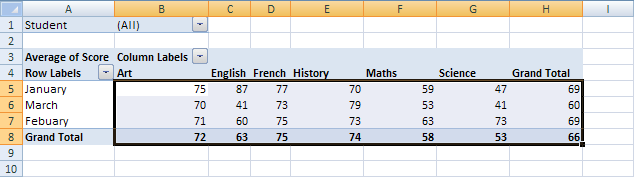
You'll see the following menu:



Select, Field Settings to see the following dialogue box:



Change the Formula from Sum to **Average**, and then click OK. Your Average formula won't be formatted to any decimal places. So highlight you data. On the Home menu in Excel 2007, locate the Number panel. Format your Averages so that it has no decimal places. Your Pivot Table will then look like this:



Almost there!

Look at cells A3, B3 and A4 above. These all have the not very descriptive names of Average of Score, Column Labels, and Row Labels. You can click inside of these cells and type your own headings, in exactly the same way as you would like to enter text in a normal cell.

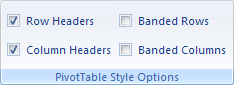
In the new version of the Pivot Table below, we have renamed these cells. We've also centered the data.



Only one thing left to do - spruce up the table by adding a bit of color.

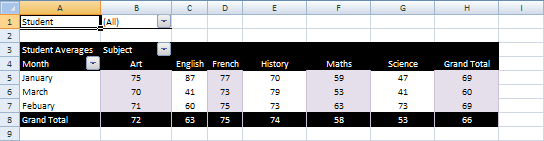
Click anywhere on your Pivot Table to highlight it. Now look at the menu bar at the top of Excel 2007. You'll notice a Design menu. Click on this to see the various design options.

The **Pivot Table Style Options** panel is interesting.

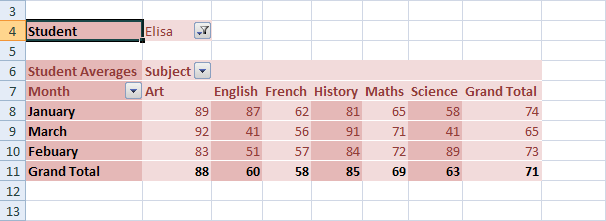


Select **Banded Rows** and see what happens. Now click **Banded Columns**.

Next to this panel, there are lots of Pivot Table Styles to choose from. Select one that catches your eye. Here's our finished Pivot Table again, only with a different Style:



And here's the original:

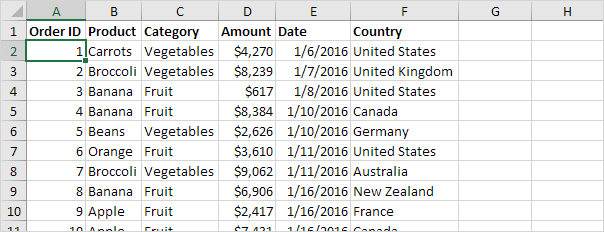


There's a lot more you can do with Pivot Tables, but we hope that this introduction has whetted your appetite!

## Practice Tasks: [expected time: 2 hours]

## TASK 1:

Pivot tables are one of Excel's most powerful features. A pivot table allows you to extract the significance from a large, detailed data set.

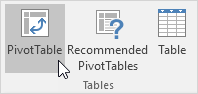
Our data set consists of 213 records and 6 fields. Order ID, Product, Category, Amount, Date and Count

### Insert a Pivot Table

To insert a **pivot table**, execute the following steps.

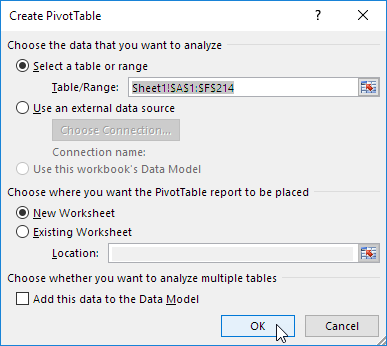
1. Click any single cell inside the data set.

2. On the Insert tab, in the Tables group, click PivotTable.



The following dialog box appears. Excel automatically selects the data for you. The default location for a new pivot table is New Worksheet.

3. Click OK.



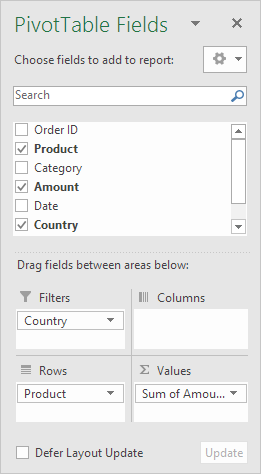
### Drag fields

The **PivotTable Fields pane** appears. To get the total amount exported of each product, drag the following fields to the different areas.

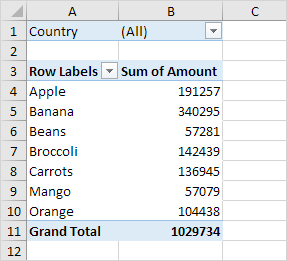
1. Product field to the Rows area.

2. Amount field to the Values area.

3. Country field to the Filters area.



Below you can find the pivot table. Bananas are our main export product. That's how easy pivot tables can be!



### Sort

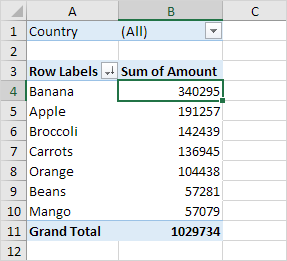
To get Banana at the top of the list, sort the pivot table.

1. Click any cell inside the Sum of Amount column.

2. Right click and click on Sort, Sort Largest to Smallest.



Result.

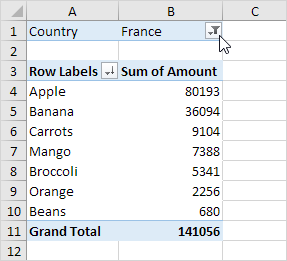


### Filter

Because we added the Country field to the Filters area, we can filter this pivot table by Country. For example, which products do we export the most to France?

1. Click the filter drop-down and select France.

Result. Apples are our main export product to France.



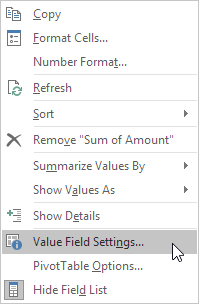
Note: you can use the standard filter (triangle next to Row Labels) to only show the amounts of specific products.

### Change Summary Calculation

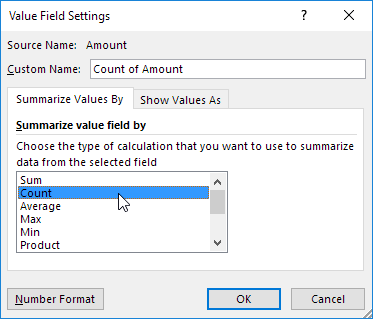
By default, Excel summarizes your data by either summing or counting the items. To change the type of calculation that you want to use, execute the following steps.

1. Click any cell inside the Sum of Amount column.

2. Right click and click on Value Field Settings.

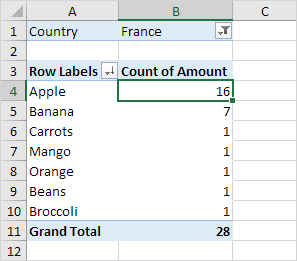


3. Choose the type of calculation you want to use. For example, click Count.



4. Click OK.

Result. 16 out of the 28 orders to France were 'Apple' orders.



### Two-dimensional Pivot Table

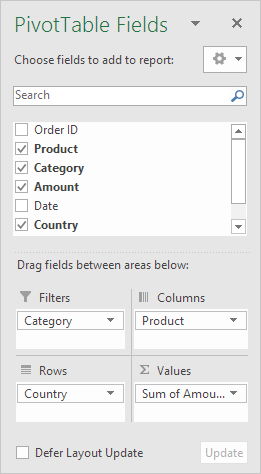
If you drag a field to the Rows area and Columns area, you can create a two-dimensional pivot table. First, [insert a pivot table](http://www.excel-easy.com/data-analysis/pivot-tables.html" \l "insert-pivot-table). Next, to get the total amount exported to each country, of each product, drag the following fields to the different areas.

1. Country field to the Rows area.

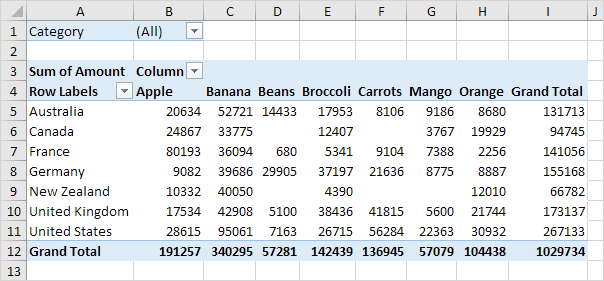
2. Product field to the Columns area.

3. Amount field to the Values area.

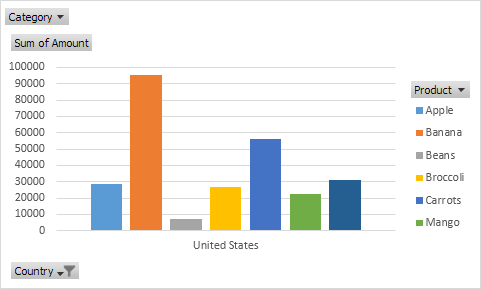
4. Category field to the Filters area.



Below you can find the two-dimensional pivot table.



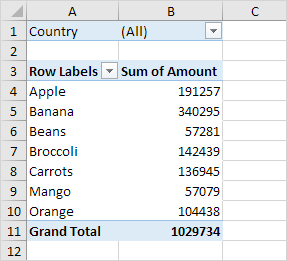
To easily compare these numbers, create a [pivot chart](http://www.excel-easy.com/examples/pivot-chart.html) and apply a filter. Maybe this is one step too far for you at this stage, but it shows you one of the many other powerful pivot table features Excel has to offer.



## TASK 2: GROUP PIVOT TABLE ITEMS

This example teaches you how to **group pivot table items**. Learn how to group products and how to group dates by months.

Below you can find a pivot table. Go back to [Pivot Tables](http://www.excel-easy.com/data-analysis/pivot-tables.html) to learn how to create this pivot table.



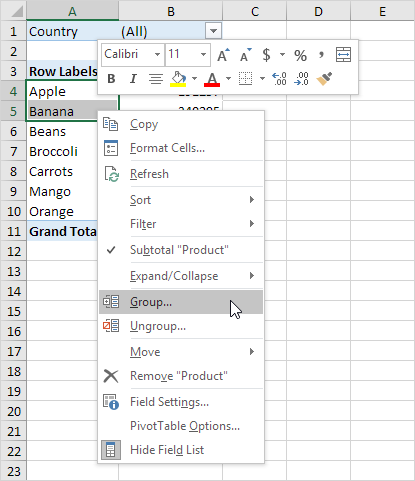
### Group Products

The Product field contains 7 items. Apple, Banana, Beans, Broccoli, Carrots, Mango and Orange.

To create two groups, execute the following steps.

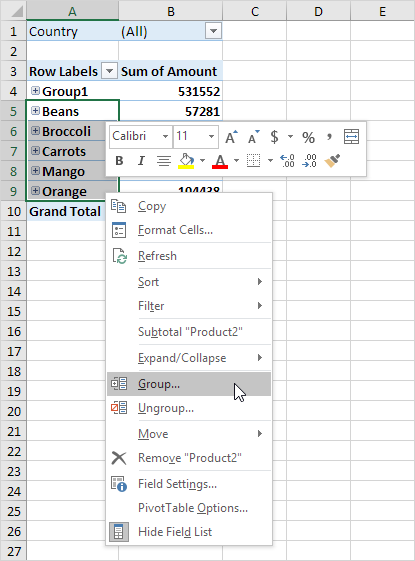
1. In the pivot table, select Apple and Banana.

2. Right click and click on Group.

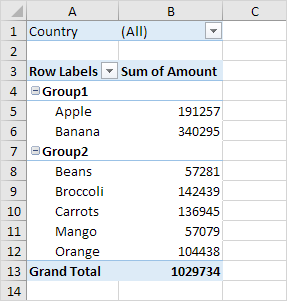


3. In the pivot table, select Beans, Broccoli, Carrots, Mango and Orange.

4. Right click and click on Group.

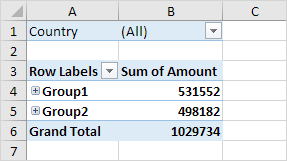


Result:



Note: to change the name of a group (Group1 or Group2), select the name, and edit the name in the formula bar. To ungroup, select the group, right click and click on Ungroup.

5. To collapse the groups, click the minus signs.



Conclusion: Apple and Banana (Group1) have a higher total than all the other products (Group2) together.

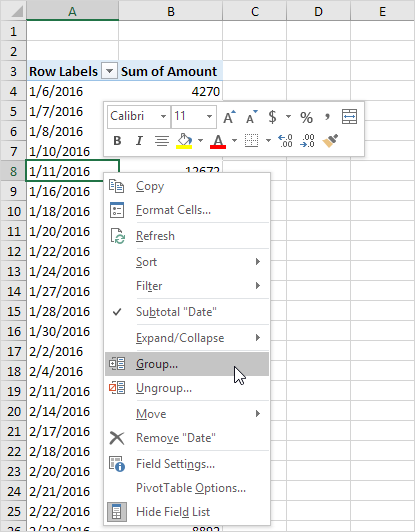
### Group Dates

To create the pivot table below, instead of the Product field, add the Date field to the Rows area. The Date field contains many items. 1/6/2016, 1/7/2016, 1/8/2016, 1/10/2016, 1/11/2016, etc.

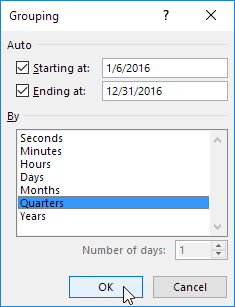
To group these dates by quarters, execute the following steps.

1. Click any cell inside the column with dates.

2. Right click and click on Group.

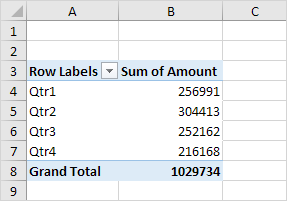


3. Select Quarters and click OK.



Note: also see the options to group by seconds, minutes, hours, etc.

Result:

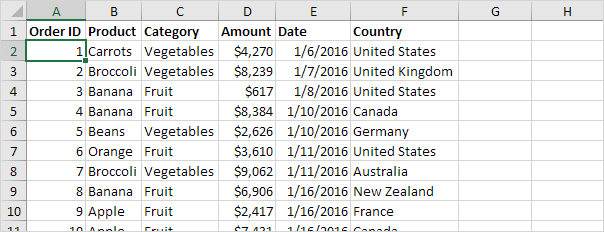


Conclusion: Quarter 2 is the best quarter.

## TASK 3: HISTOGRAM AND FREQUENCY DISTRIBUTUION

Did you know that you can use pivot tables to easily create a **frequency distribution** in **Excel**? You can also use the Analysis Toolpak to create a [histogram](http://www.excel-easy.com/examples/histogram.html).

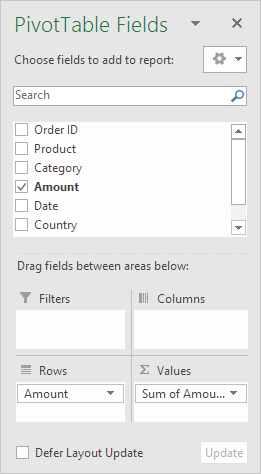
Remember, our data set consists of 213 records and 6 fields. Order ID, Product, Category, Amount, Date and Country.



First, [insert a pivot table](http://www.excel-easy.com/data-analysis/pivot-tables.html#insert-pivot-table). Next, drag the following fields to the different areas.

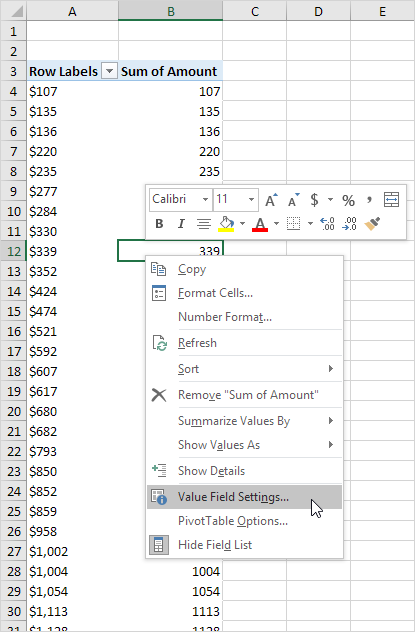
1. Amount field to the Rows area.

2. Amount field (or any other field) to the Values area.

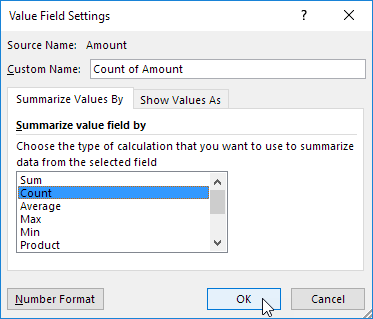


3. Click any cell inside the Sum of Amount column.

4. Right click and click on Value Field Settings.

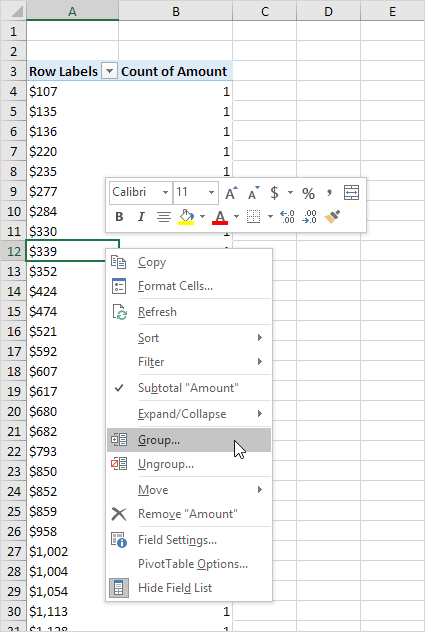


5. Choose Count and click OK.



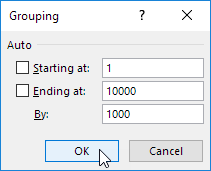
6. Next, click any cell inside the column with Row Labels.

7. Right click and click on Group.

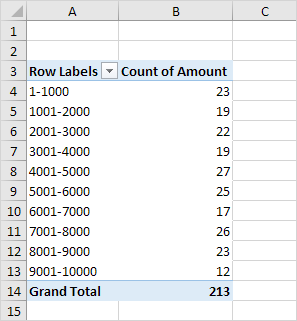


8. Enter 1 for Starting at, 10000 for Ending at, and 1000 for By.

9. Click OK.



Result:



To easily compare these numbers, create a pivot chart.

10. Click any cell inside the pivot table.

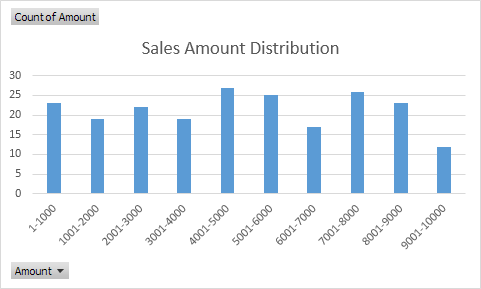
11. On the Analyze tab, in the Tools group, click PivotChart.



The Insert Chart dialog box appears.

12. Click OK.

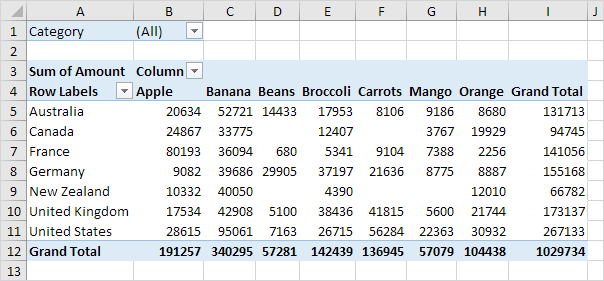
Result:



## TASK 4: PIVOT CHART

A **pivot chart** is the visual representation of a pivot table in **Excel**. Pivot charts and pivot tables are connected with each other.

Below you can find a two-dimensional pivot table. Go back to [Pivot Tables](http://www.excel-easy.com/data-analysis/pivot-tables.html#two-dimensional-pivot-table) to learn how to create this pivot table.



### Insert Pivot Chart

To insert a pivot chart, execute the following steps.

1. Click any cell inside the pivot table.

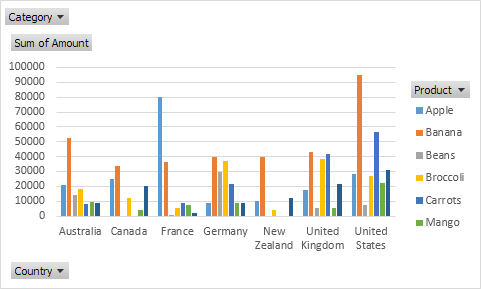
2. On the Analyze tab, in the Tools group, click PivotChart.



The Insert Chart dialog box appears.

3. Click OK.

Below you can find the pivot chart. This pivot chart will amaze and impress your boss.

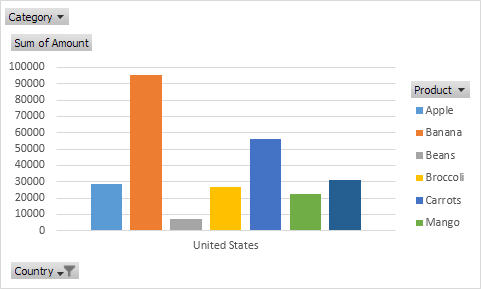


Note: any changes you make to the pivot chart are immediately reflected in the pivot table and vice versa.

### Filter Pivot Chart

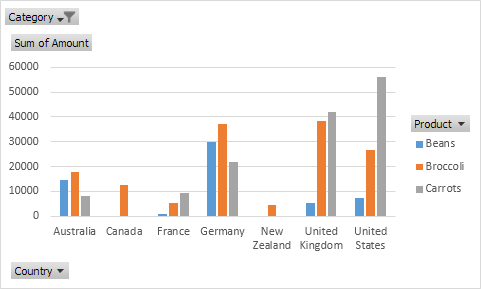
To filter this pivot chart, execute the following steps.

1. Use the standard filters (triangles next to Product and Country). For example, use the Country filter to only show the total amount of each product exported to the United States.



2. Remove the Country filter.

3. Because we added the Category field to the Filters area, we can filter this pivot chart (and pivot table) by Category. For example, use the Category filter to only show the vegetables exported to each country.

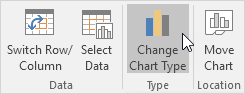


### Change Pivot Chart Type

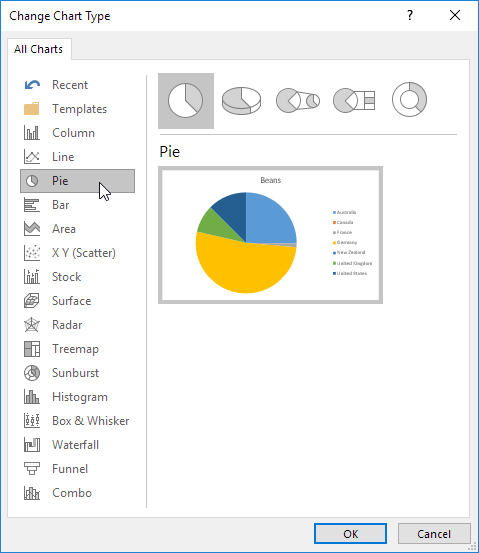
You can change to a different type of pivot chart at any time.

1. Select the chart.

2. On the Design tab, in the Type group, click Change Chart Type.

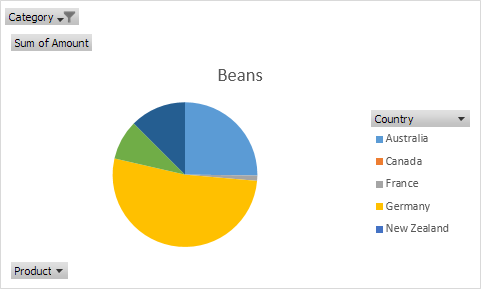


3. Choose Pie.



4. Click OK.

Result:

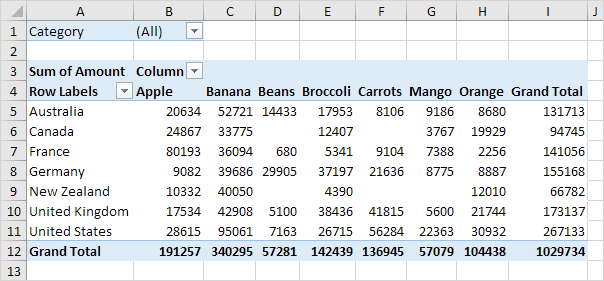


Note: pie charts always use one data series (in this case, Beans). To get a pivot chart of a country, swap the data over the axis. First, select the chart. Next, on the Design tab, in the Data group, click Switch Row/Column.

## TASK 5: SLICERS

You can insert **slicers** in **Excel** to quickly and easily filter pivot tables. However, using the report filter gives the exact same result.

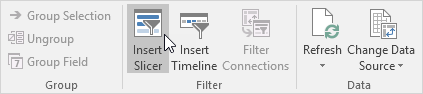
Below you can find a two-dimensional pivot table. Go back to [Pivot Tables](http://www.excel-easy.com/data-analysis/pivot-tables.html#two-dimensional-pivot-table) to learn how to create this pivot table.



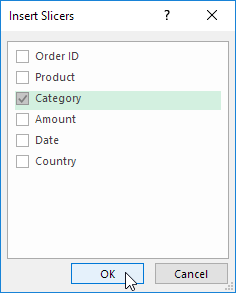
To insert a slicer, execute the following steps.

1. Click any cell inside the pivot table.

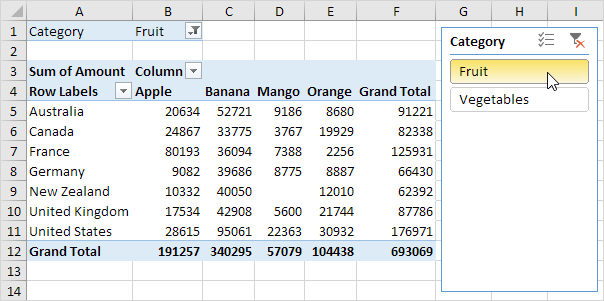
2. On the Analyze tab, in the Filter group, click Insert Slicer.



3. Check Category and click OK.



4. For example, click Fruit to only show the fruit exported to each country.



Note: the report filter (cell B1) changes to Fruit. Hold down CTRL to include fruit and vegetables.

## TASK 6: REFRESH/UPDATE PIVOT TABLE

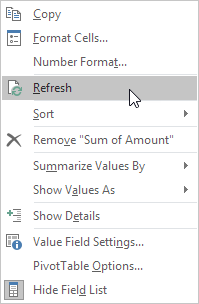
Any changes you make to the data set are not automatically picked up by the pivot table. **Refresh** the **pivot table** or **change the data source** to **update** the pivot table with the applied changes.

### Refresh

If you change any of the text or numbers in your data set, you need to refresh the pivot table.

1. Click any cell inside the pivot table.

2. Right click and click on Refresh.

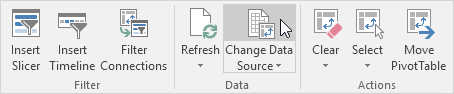


### Change Data Source

If you change the size of your data set by adding or deleting rows/columns, you need to update the source data for the pivot table.

1. Click any cell inside the pivot table.

2. On the Analyze tab, in the Data group, click Change Data Source.

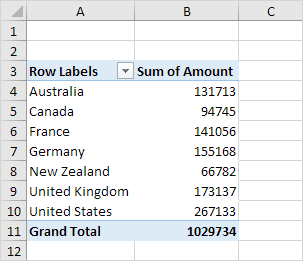


Tip: change your data set to a [table](http://www.excel-easy.com/data-analysis/tables.html) before you insert a pivot table. This way your data source will be updated automatically when you add or delete rows/columns. This can save time. You still have to refresh though.

## TASK 7: CALCULATED FIELD/ITEM

This example teaches you how to insert a **calculated field** or **calculated item** in a **pivot table**.

Below you can find a pivot table. Go back to [Pivot Tables](http://www.excel-easy.com/data-analysis/pivot-tables.html) to learn how to create this pivot table.

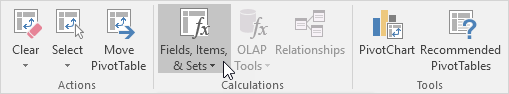


### Calculated Field

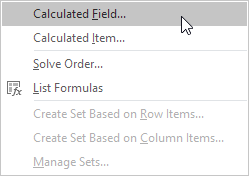
A calculated field uses the values from another field. To insert a calculated field, execute the following steps.

1. Click any cell inside the pivot table.

2. On the Analyze tab, in the Calculations group, click Fields, Items & Sets.



3. Click Calculated Field.

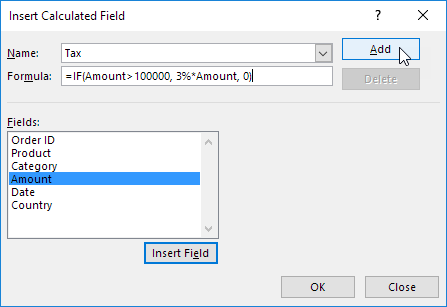


The Insert Calculated Field dialog box appears.

4. Enter Tax for Name.

5. Type the formula =IF(Amount>100000, 3%\*Amount, 0)

6. Click Add.

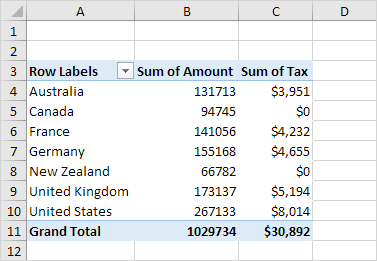


Note: use the Insert Field button to quickly insert fields when you type a formula. To delete a calculated field, select the field and click Delete (under Add).

7. Click OK.

Excel automatically adds the Tax field to the Values area.

Result:

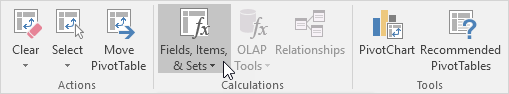


### Calculated Item

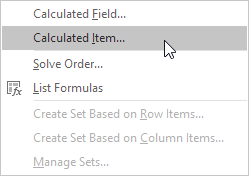
A calculated item uses the values from other items. To insert a calculated item, execute the following steps.

1. Click any Country in the pivot table.

2. On the Analyze tab, in the Calculations group, click Fields, Items & Sets.



3. Click Calculated Item.

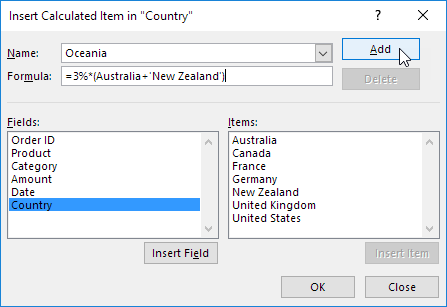


The Insert Calculated Item dialog box appears.

4. Enter Oceania for Name.

5. Type the formula =3%\*(Australia+'New Zealand')

6. Click Add.

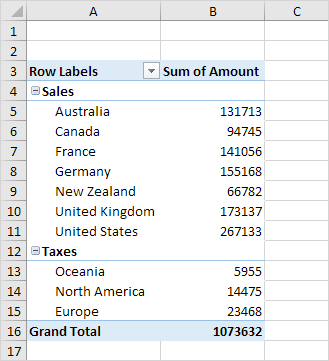


Note: use the Insert Item button to quickly insert items when you type a formula. To delete a calculated item, select the item and click Delete (under Add).

7. Repeat steps 4 to 6 for North America (Canada and United States) and Europe (France, Germany and United Kingdom) with a 4% and 5% tax rate respectively.

8. Click OK.

Result:



Note: we created two [groups](http://www.excel-easy.com/examples/group-pivot-table-items.html) (Sales and Taxes).

**Lab 5**

**Introduction to Flow Charts**

**Introduction to Flow Charts**

**Objective:**

* Introduction to flow charts and pseudo-code.
* Familiarization with flow chart symbols.
* Enable students to understand and develop logic for basic programs by practicing tasks.

**Topics covered:**

1. Flow charts, their symbols

2. Practice tasks

3. Examples

**Introduction** [expected time= 1 hr]

Flowchart is a diagrammatic representation of an algorithm. Flow charts are very helpful in writing program and explaining program to others. Each step in the process is represented by a different symbol and contains a short description of the process step. The flow chart symbols are linked together with arrows showing the process flow direction. A flowchart typically shows the flow of data in a process, detailing the operations/steps in a pictorial format which is easier to understand than reading it in a textual format.

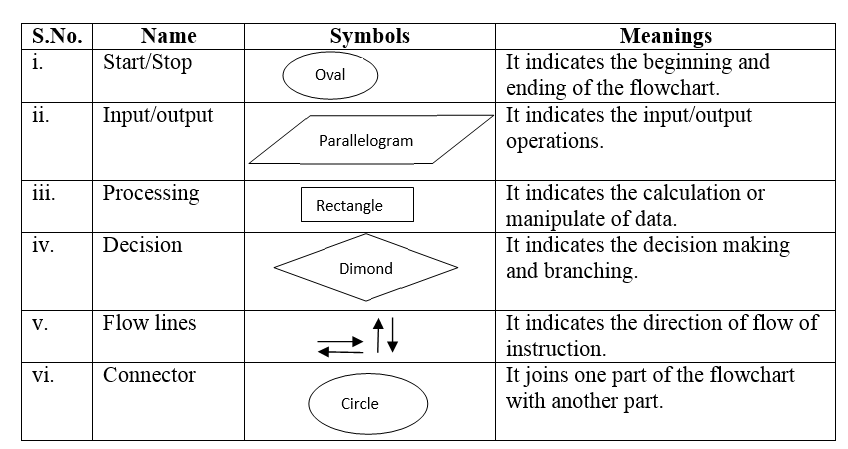
A flowchart describes what operations (and in what sequence) are required to solve a given problem. A flowchart can be likened to the blueprint of a building. As we know a designer draws a blueprint before starting construction on a building. Similarly, a programmer prefers to draw a flowchart prior to writing a computer program. Flowcharts are a pictorial or graphical representation of a process. The purpose of all flow charts is to communicate how a process works or should work without any technical or group specific jargon.

Flowcharts are used in analyzing, designing, documenting or managing a process or program in various fields. Flowcharts are generally drawn in the early stages of formulating computer solutions. Flowcharts often facilitate communication between programmers and business people. These flowcharts play a vital role in the programming of a problem and are quite helpful in understanding the logic of complicated and lengthy problems. Once the flowchart is drawn, it becomes easy to write the program in any high level language. Often we see how flowcharts are helpful in explaining the program to others. Hence, it is correct to say that a flowchart is a must for the better documentation of a complex program.

**Guidelines:**

The common guidelines (properties) of flowchart are:  
1.The flowchart should have only one start and ending points.  
2.Flow lines shouldn't intersect each other.  
3.Flowchart should not contain the programming language.  
4.The flowchart should be neat and clear for the user.

**Flowchart Symbols:**

****

**EXAMPLE 1**: For example, consider that we need to find the sum, average and product of 3 numbers given by the user.

Algorithm for the given problem is as follows:

Read X, Y, Z

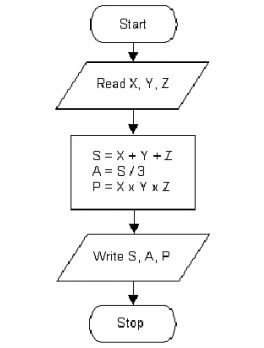
Compute Sum (S) as X + Y + Z

Compute Average (A) as S / 3

Compute Product (P) as X x Y x Z

Write (Display) the Sum, Average and Product

Flowchart for the above problem will look like:



**EXAMPLE 2**: Write algorithm and flow chart to find area and circumference of a circle.

Step 1 : Start

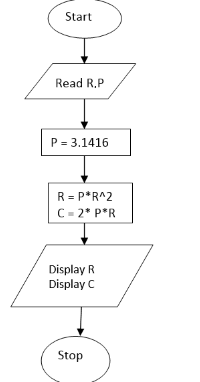
Step2 : Read radius and store to R

Step 3 : Assign value of pie to P

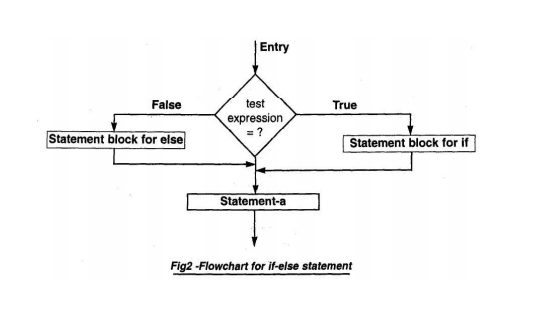
Step 4 : Find area and circumference

Step 5 : Display area and circumference

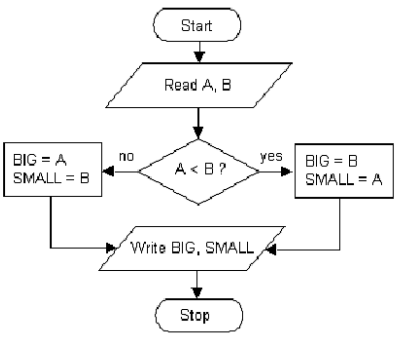
Step 6 : Stop



**Flow chart for decision making:**



**EXAMPLE 3:** Draw flow chart to find greater number among two numbers.



**EXAMPLE 4:**Draw flowchart for finding the sum of n numbers. (flow chart for loops)

Pseudocode Program:

Start

Sum = 0

**Display “Input value n”**

Input n

For(I = 1, n, 5)

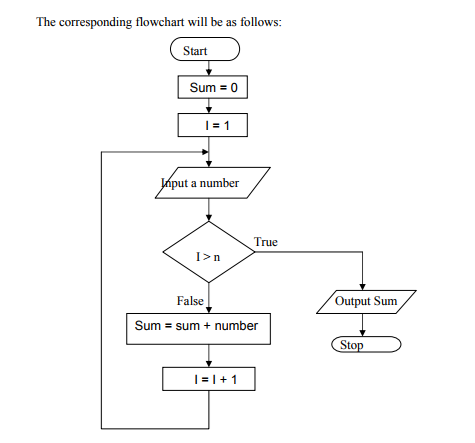
Input a value

Sum = sum + value

END FOR

Output sum

Stop

****

**PRACTICE TASKS:** [expected time= 1 hr]

1. Draw flowchart to find the largest among three different numbers entered by user.
2. Obtain a temperature in degrees Fahrenheit from the user. If the temperature is 80 degrees or more, display a message that says "Go play golf" otherwise display a message stating "It's too cold to be outside." Make a variable list, flowchart, and perform a desk check using the following values: 95, 70
3. Write an algorithm and draw a flow chart to determine whether the seller has made profit or incurred loss. Also determine how much profit he made or loss he incurred. Cost price and selling price of an item is input by the user.
4. Write an algorithm and draw a flowchart that will calculate the roots of a quadratic equation

**ax2 +bx +c =0**

Hint: d = sqrt (b2 - 4ac) and the roots are: x1 = (–b + d)/2a and x2 = (–b – d)/2a

1. Write an algorithm and draw a flow chart that prints ‘hello world’ 10 times.

**Lab 6**

**Flow charts, Algorithms**

**And Source codes**

**Lab 6: Flow Charts, algorithms and source codes (logic building)**

**Objective:**

* To review flow charts and algorithm writing.
* To develop logic building among students.
* Enable students to understand and develop algorithms/source codes by practicing tasks.

**Topics covered:**

1. Flow charts
2. Algorithm writing
3. Developing source codes
4. Practice tasks
5. Examples

**Review and Examples** [expected time= 45 min]

Flowchart is a diagrammatic representation of an algorithm. Flow charts are very helpful in writing program and explaining program to others. Each step in the process is represented by a different symbol and contains a short description of the process step. The flow chart symbols are linked together with arrows showing the process flow direction. A flowchart typically shows the flow of data in a process, detailing the operations/steps in a pictorial format which is easier to understand than reading it in a textual format.

**Algorithm:**

An algorithm gives a solution to a particular problem as a well-defined set of steps. A recipe in a cookbook is a good example of an algorithm. When a computer is used for solving a particular problem, the steps to the solution should be communicated to the computer. This makes the study of algorithms a very important part in computer science. An algorithm is executed in a computer by combining lot of elementary operations such as additions and subtractions to perform more complex mathematical operations.

In simple terms, an *algorithm* is executable code logic defined by a sequence of steps to resolve a problem or complete a task while pseudo-code is a narrative describing algorithm logic.

**Pseudo-code:**

Pseudo-code is one of the methods that could be used to represent an algorithm. It is a detailed yet readable description of what a computer program or algorithm must do, expressed in a formally-styled natural language rather than in a programming language.

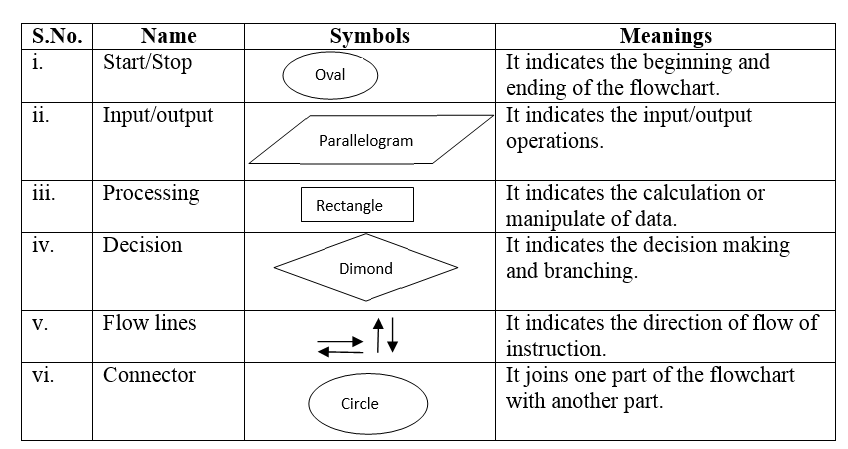
What is the difference between Algorithm and Pseudo-code?

An algorithm is a well-defined sequence of steps that provides a solution for a given problem, while a pseudo-code is one of the methods that can be used to represent an algorithm. While algorithms can be written in natural language, pseudo-code is written in a format that is closely related to high level programming language structures. But pseudo-code does not use specific programming language syntax and therefore could be understood by programmers who are familiar with different programming languages. Additionally, transforming an algorithm presented in pseudocode to programming code could be much easier than converting an algorithm written in natural language.

**Guidelines:**

The common guidelines (properties) of flowchart are:  
1.The flowchart should have only one start and ending points.  
2.Flow lines shouldn't intersect each other.  
3.Flowchart should not contain the programming language.  
4.The flowchart should be neat and clear for the user.

**Flow chart Symbols:**

****

**EXAMPLE 1**: Draw a flow chart for a program that sums all the even numbers between 1 and 20 inclusive and then displays the sum. It uses a repeat loop and contains a null else within the repeat loop.

The equivalent pseudo-code is:

sum = 0

count = 1

REPEAT

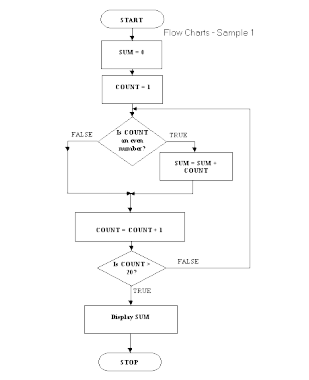
IF count is even THEN sum = sum + count

count = count + 1

UNTIL count > 20

DISPLAY sum

Flowchart for the above problem will look like:



**EXAMPLE 2**: Write algorithm, source code and draw a flow chart to find sum of digits of given integer number.

**Pseudo code:**

* Input a Number
* Initialize Sum to zero
* While Number is not zero
* Get Remainder by Number Mod 10
* Add Remainder to Sum
* Divide Number by 10
* Print sum

**Detailed Algorithm:**

Step 1:  Input N

Step 2:  Sum = 0

Step 3:  While (N != 0)

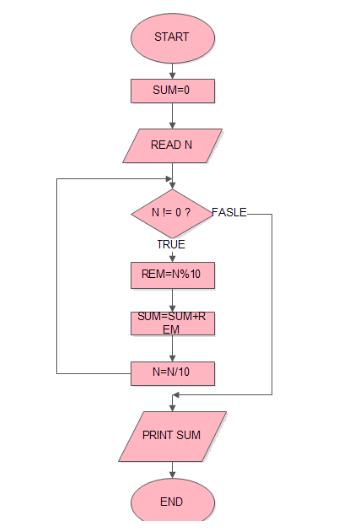
                        Rem = N % 10;

                        Sum = Sum + Rem;

                        N = N / 10;

Step 4:  Print Sum

**Flowchart:-**

[](http://cssimplified.com/wp-content/uploads/2014/05/FCSumOfDigits.jpg)

SOURCE CODE:

#include<iostream>

using namespacestd;

int main()

{

intn,t,r,sum=0;

cout<<"Enter any number : ";

cin>>n;

t=n;

while(t>0)

{

r=t%10;

sum+=r;

t=t/10;

}

cout<<"Sum of digits of number "<<n<<" is "<<sum;

return 0;

}

**EXAMPLE 3:** Draw flow chart and write source code to calculate the total expenses:

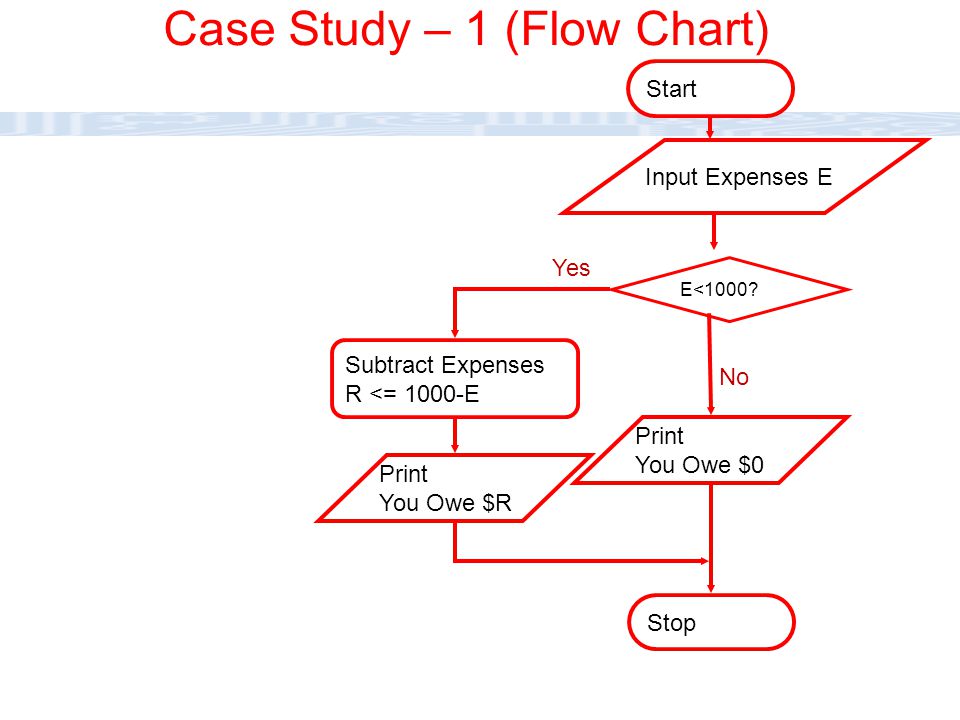
A certain Company gives each of its sales people $1,000 at the beginning of each month to cover travel, lodging, and food expenses. At the end of the month, a salesperson must total his/her expense receipts. If the amount is less than $1,000 then the difference must be returned to the company.

Here is an algorithm for figuring how much money, if any, must be returned:

1-Total the expense receipts for the month.

2. Subtract the amount of the expense receipts from 1,000.

3. If the remainder is more than 0, return that amount to the company.

****

**SOURCE CODE:**

#include<iostream>

using namespacestd;

int main()

{

int E,R;

cout<<"Enter expenses : ";

cin>>E;

if(E>1000)

{ cout<<"You owe 0$ "<<endl;}

else

{

R=1000-E;

cout<<"You owe R$ "<<endl;

}

return 0;

}

**PRACTICE TASKS:** [expected time= 2hrs and 15 min]

1. Write an algorithm and source code to find factorial of a number entered by the user.
2. Write an algorithm and source code to reverse any given integer number.
3. Write an algorithm and source code to calculate 24 (two raised to the power four) using a loop.
4. In a company an employee is paid as under:

If his basic salary is less than Rs. 1500, then House Rent Allowance (HRA) = 10% of basic salary and Transport Allowance (TA) = 90% of basic salary.

If his salary is either equal to or above Rs. 1500, then HRA = Rs. 500 and TA = 98% of basic salary.

If the employee's salary is input by the user, write an algorithm and source code to find his gross salary.

**LAB 7**

**Basic Input/ output**

The example programs of the previous sections provided little interaction with the user, if any at all. They simply printed simple values on screen, but the standard library provides many additional ways to interact with the user via its input/output features. This section will present a short introduction to some of the most useful.  
  
C++ uses a convenient abstraction called *streams* to perform input and output operations in sequential media such as the screen, the keyboard or a file. A stream is an entity where a program can either insert or extract characters to/from. There is no need to know details about the media associated to the stream or any of its internal specifications. All we need to know is that streams are a source/destination of characters, and that these characters are provided/accepted sequentially (i.e., one after another).  
  
The standard library defines a handful of stream objects that can be used to access what are considered the standard sources and destinations of characters by the environment where the program runs:

|  |  |
| --- | --- |
| **stream** | **Description** |
| cin | standard input stream |
| cout | standard output stream |
| cerr | standard error (output) stream |
| clog | standard logging (output) stream |

We are going to see in more detail only cout and cin (the standard output and input streams); cerr and clog are also output streams, so they essentially work like cout, with the only difference being that they identify streams for specific purposes: error messages and logging; which, in many cases, in most environment setups, they actually do the exact same thing: they print on screen, although they can also be individually redirected.

### Standard output (cout)

On most program environments, the standard output by default is the screen, and the C++ stream object defined to access it is cout.  
  
For formatted output operations, cout is used together with the *insertion operator*, which is written as << (i.e., two "less than" signs).

|  |  |  |
| --- | --- | --- |
| 1 2 3 | cout << "Output sentence"; // prints Output sentence on screen  cout << 120; // prints number 120 on screen  cout << x; // prints the value of x on screen |  |

The << operator inserts the data that follows it into the stream that precedes it. In the examples above, it inserted the literal string Output sentence, the number 120, and the value of variable x into the standard output stream cout. Notice that the sentence in the first statement is enclosed in double quotes (") because it is a string literal, while in the last one, x is not. The double quoting is what makes the difference; when the text is enclosed between them, the text is printed literally; when they are not, the text is interpreted as the identifier of a variable, and its value is printed instead. For example, these two sentences have very different results:

|  |  |  |
| --- | --- | --- |
| 1 2 | cout << "Hello"; // prints Hello  cout << Hello; // prints the content of variable Hello |  |

Multiple insertion operations (<<) may be chained in a single statement:

|  |  |  |
| --- | --- | --- |
|  | cout << "This " << " is a " << "single C++ statement"; |  |

This last statement would print the text This is a single C++ statement. Chaining insertions is especially useful to mix literals and variables in a single statement:

|  |  |  |
| --- | --- | --- |
|  | cout << "I am " << age << " years old and my zipcode is " << zipcode; |  |

Assuming the age variable contains the value 24 and the zipcode variable contains 90064, the output of the previous statement would be:  
  
I am 24 years old and my zipcode is 90064  
What cout does not do automatically is add line breaks at the end, unless instructed to do so. For example, take the following two statements inserting into cout:  
cout << "This is a sentence.";  
cout << "This is another sentence.";  
  
The output would be in a single line, without any line breaks in between. Something like:  
  
This is a sentence.This is another sentence.  
To insert a line break, a new-line character shall be inserted at the exact position the line should be broken. In C++, a new-line character can be specified as \n (i.e., a backslash character followed by a lowercase n). For example:

|  |  |  |
| --- | --- | --- |
| 1 2 | cout << "First sentence.\n";  cout << "Second sentence.\nThird sentence."; |  |

This produces the following output:  
  
First sentence.  
Second sentence.  
Third sentence.  
  
Alternatively, the endl manipulator can also be used to break lines. For example:

|  |  |  |
| --- | --- | --- |
| 1 2 | cout << "First sentence." << endl;  cout << "Second sentence." << endl; |  |

This would print:  
  
First sentence.  
Second sentence.  
  
The endl manipulator produces a newline character, exactly as the insertion of '\n' does; but it also has an additional behavior: the stream's buffer (if any) is flushed, which means that the output is requested to be physically written to the device, if it wasn't already. This affects mainly *fully buffered* streams, and cout is (generally) not a *fully buffered* stream. Still, it is generally a good idea to use endl only when flushing the stream would be a feature and '\n' when it would not. Bear in mind that a flushing operation incurs a certain overhead, and on some devices it may produce a delay.

### Standard input (cin)

In most program environments, the standard input by default is the keyboard, and the C++ stream object defined to access it is cin.  
  
For formatted input operations, cin is used together with the extraction operator, which is written as >> (i.e., two "greater than" signs). This operator is then followed by the variable where the extracted data is stored. For example:

|  |  |  |
| --- | --- | --- |
| 1 2 | int age;  cin >> age; |  |

The first statement declares a variable of type int called age, and the second extracts from cin a value to be stored in it. This operation makes the program wait for input from cin; generally, this means that the program will wait for the user to enter some sequence with the keyboard. In this case, note that the characters introduced using the keyboard are only transmitted to the program when the ENTER (or RETURN) key is pressed. Once the statement with the extraction operation on cin is reached, the program will wait for as long as needed until some input is introduced.  
  
The extraction operation on cin uses the type of the variable after the >> operator to determine how it interprets the characters read from the input; if it is an integer, the format expected is a series of digits, if a string a sequence of characters, etc.

|  |  |  |  |
| --- | --- | --- | --- |
| 1 2 3 4 5 6 7 8 9 10 11 12 13 14 | // i/o example  #include <iostream>  using namespace std;  int main ()  {  int i;  cout << "Please enter an integer value: ";  cin >> i;  cout << "The value you entered is " << i;  cout << " and its double is " << i\*2 << ".\n";  return 0;  } | Please enter an integer value: 702  The value you entered is 702 and its double is 1404. |  |

As you can see, extracting from cin seems to make the task of getting input from the standard input pretty simple and straightforward. But this method also has a big drawback. What happens in the example above if the user enters something else that cannot be interpreted as an integer? Well, in this case, the extraction operation fails. And this, by default, lets the program continue without setting a value for variable i, producing undetermined results if the value of i is used later.  
  
This is very poor program behavior. Most programs are expected to behave in an expected manner no matter what the user types, handling invalid values appropriately. Only very simple programs should rely on values extracted directly from cin without further checking. A little later we will see how *stringstreams* can be used to have better control over user input.  
Extractions on cin can also be chained to request more than one datum in a single statement:

|  |  |  |
| --- | --- | --- |
|  | cin >> a >> b; |  |

This is equivalent to:

|  |  |  |
| --- | --- | --- |
| 1 2 | cin >> a;  cin >> b; |  |

In both cases, the user is expected to introduce two values, one for variable a, and another for variable b. Any kind of space is used to separate two consecutive input operations; this may either be a space, a tab, or a new-line character.

### cin and strings

The extraction operator can be used on cin to get strings of characters in the same way as with fundamental data types:

|  |  |  |
| --- | --- | --- |
| 1 2 | string mystring;  cin >> mystring; |  |

|  |  |  |
| --- | --- | --- |
| 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 | // cin with strings  #include <iostream>  #include <string>  using namespace std;  int main ()  {  string mystr;  cout << "What's your name? ";  getline (cin, mystr);  cout << "Hello " << mystr << ".\n";  cout << "What is your favorite team? ";  getline (cin, mystr);  cout << "I like " << mystr << " too!\n";  return 0;  } | What's your name? Homer Simpson  Hello Homer Simpson.  What is your favorite team? The Isotopes  I like The Isotopes too! |

**Tasks:**

**1.** Write a program in C++ to print a welcome text in a separate line.  **2.** Write a program in C++ to print the sum of two numbers.    
Sample Output:  
Print the sum of two numbers :  
-----------------------------------  
The sum of 29 and 30 is : 59

**3.** Write a program in C++ to find Size of fundamental data types.  Sample Output:  
Find Size of fundamental data types :  
------------------------------------------  
The sizeof(char) is : 1 bytes  
The sizeof(short) is : 2 bytes  
The sizeof(int) is : 4 bytes  
The sizeof(long) is : 8 bytes  
The sizeof(long long) is : 8 bytes  
The sizeof(float) is : 4 bytes  
The sizeof(double) is : 8 bytes  
The sizeof(long double) is : 16 bytes  
The sizeof(bool) is : 1 bytes

**4.**Write a program in C++ to print the sum of two numbers using variables.  Print the sum of two numbers :  
-----------------------------------  
The sum of 29 and 30 is : 59

**5.** Write a program in C++ to check the upper and lower limits of integer.  Expected Output:  
Check the upper and lower limits of integer :  
--------------------------------------------------  
The maximum limit of int data type : 2147483647  
The minimum limit of int data type : -2147483648  
The maximum limit of unsigned int data type : 4294967295  
The maximum limit of long long data type : 9223372036854775807  
The minimum limit of long long data type : -9223372036854775808  
The maximum limit of unsigned long long data type : 18446744073709551615  
The Bits contain in char data type : 8  
The maximum limit of char data type : 127  
The minimum limit of char data type : -128  
The maximum limit of signed char data type : 127  
The minimum limit of signed char data type : -128  
The maximum limit of unsigned char data type : 255  
The minimum limit of short data type : -32768  
The maximum limit of short data type : 32767  
The maximum limit of unsigned short data type : 65535

**6.** Write a program in C++ to check whether the primitive values crossing the limits or not.    
Check whether the primitive values crossing the limits or not :  
--------------------------------------------------------------------  
The Gender is : F  
Is she married? : 1  
Number of sons she has : 2  
Year of her appointment : 2009  
Salary for a year : 1500000  
Height is : 79.48  
GPA is 4.69  
Salary drawn upto : 12047235  
Balance till : 995324987

**7.** Write a program in C++ to display various type or arithmetic operation using mixed data type   
Sample output:  
Display arithmetic operations with mixed data type :  
---------------------------------------------------------  
5 + 7 = 12  
3.7 + 8.0 = 11.7  
5 + 8.0 = 13.0  
5 - 7 = -2  
3.7 - 8.0 = -4.3  
5 - 8.0 = -3.0  
5 \* 7 = 35  
3.7 \* 8.0 = 29.6  
5 \* 8.0 = 40.0  
5 / 7 = 0  
3.7 / 8.0 = 0.5  
5 / 8.0 = 0.6

**Lab 8**

**Conditions: IF ELSE statements**

In computer programming, we use the if statement to run a block code only when a certain condition is met.

For example, assigning grades (A, B, C) based on marks obtained by a student.

* if the percentage is above **90**, assign grade **A**
* if the percentage is above **75**, assign grade **B**
* if the percentage is above **65**, assign grade **C**

There are three forms of if...else statements in C++.

1. if statement
2. if...else statement
3. if...else if...else statement

## C++ if Statement

The syntax of the if statement is:

if (condition) {

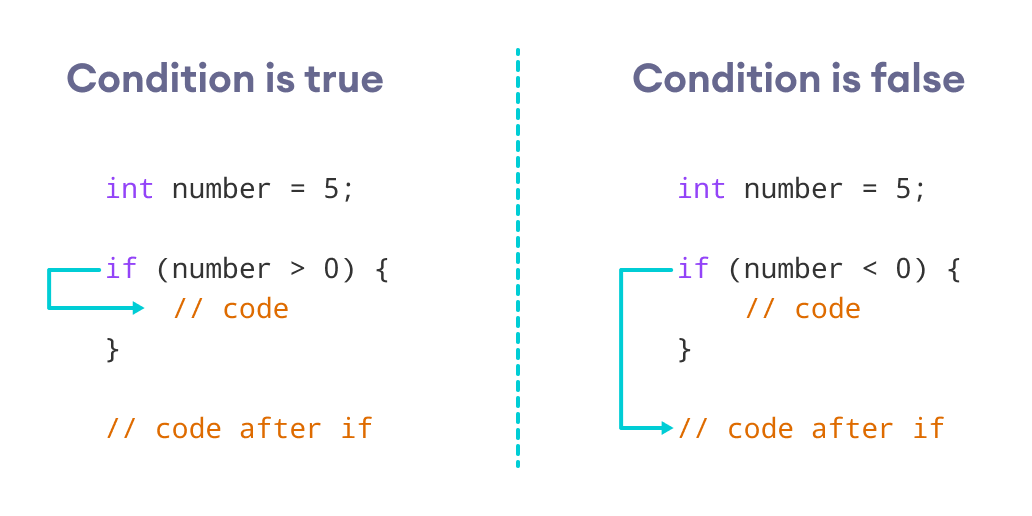
// body of if statement

}

The if statement evaluates the condition inside the parentheses ( ).

* If the condition evaluates to true, the code inside the body of if is executed.
* If the condition evaluates to false, the code inside the body of if is skipped.

**Note:** The code inside { } is the body of the if statement.

Working of C++ if Statement

### Example 1: C++ if Statement

// Program to print positive number entered by the user

// If the user enters a negative number, it is skipped

#include <iostream>

using namespace std;

int main() {

int number;

cout << "Enter an integer: ";

cin >> number;

// checks if the number is positive

if (number > 0) {

cout << "You entered a positive integer: " << number << endl;

}

cout << "This statement is always executed.";

return 0;

}

**Output 1**

Enter an integer: 5

You entered a positive number: 5

This statement is always executed.

When the user enters 5, the condition number > 0 is evaluated to true and the statement inside the body of if is executed.

**Output 2**

Enter a number: -5

This statement is always executed.

When the user enters -5, the condition number > 0 is evaluated to false and the statement inside the body of if is not executed.

## C++ if...else

The if statement can have an optional else clause. Its syntax is:

if (condition) {

// block of code if condition is true

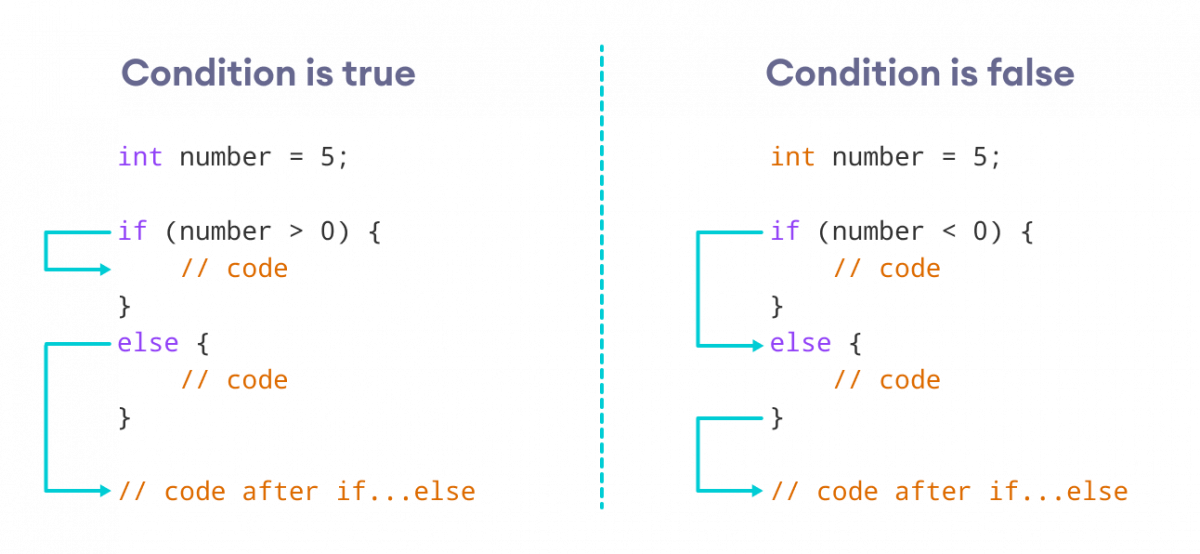
}

else {

// block of code if condition is false

}

The if..else statement evaluates the condition inside the parenthesis.

Working of C++ if...else

If the condition evaluates true,

* the code inside the body of if is executed
* the code inside the body of else is skipped from execution

If the condition evaluates false,

* the code inside the body of else is executed
* the code inside the body of if is skipped from execution

### Example 2: C++ if...else Statement

// Program to check whether an integer is positive or negative

// This program considers 0 as a positive number

#include <iostream>

using namespace std;

int main() {

int number;

cout << "Enter an integer: ";

cin >> number;

if (number >= 0) {

cout << "You entered a positive integer: " << number << endl;

}

else {

cout << "You entered a negative integer: " << number << endl;

}

cout << "This line is always printed.";

return 0;

}

**Output 1**

Enter an integer: 4

You entered a positive integer: 4.

This line is always printed.

In the above program, we have the condition number >= 0. If we enter the number greater or equal to 0, then the condition evaluates true.

Here, we enter 4. So, the condition is true. Hence, the statement inside the body of if is executed.

**Output 2**

Enter an integer: -4

You entered a negative integer: -4.

This line is always printed.

Here, we enter -4. So, the condition is false. Hence, the statement inside the body of else is executed.

## C++ if...else...else if statement

The if...else statement is used to execute a block of code among two alternatives. However, if we need to make a choice between more than two alternatives, we use the if...else if...else statement.

The syntax of the if...else if...else statement is:

if (condition1) {

// code block 1

}

else if (condition2){

// code block 2

}

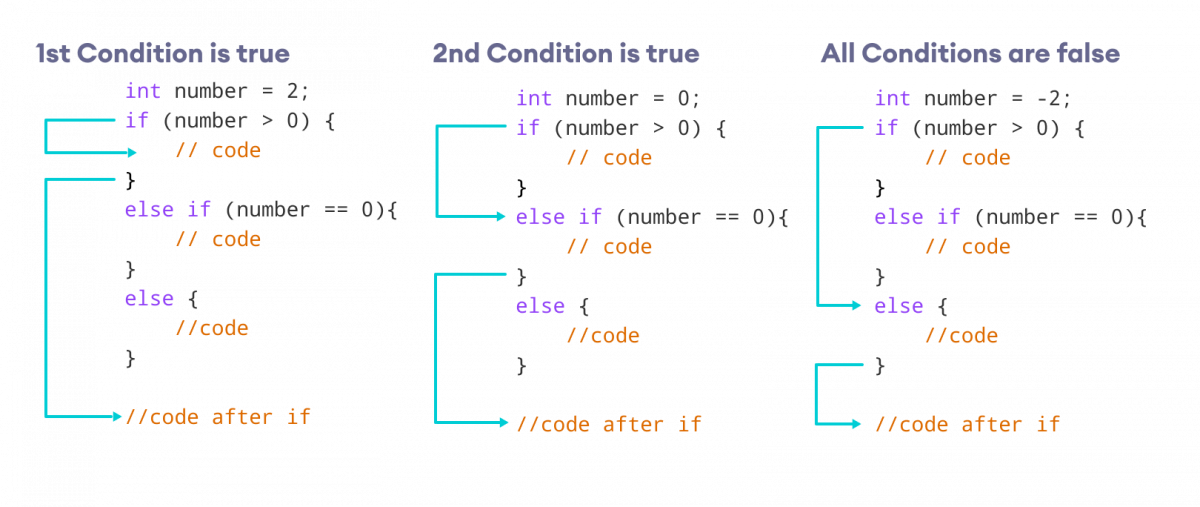
else {

// code block 3

}

Here,

* If condition1 evaluates to true, the code block 1 is executed.
* If condition1 evaluates to false, then condition2 is evaluated.
* If condition2 is true, the code block 2 is executed.
* If condition2 is false, the code block 3 is executed.

How if...else if...else Statement Works

**Note:** There can be more than one else if statement but only one if and else statements.

### Example 3: C++ if...else...else if

// Program to check whether an integer is positive, negative or zero

#include <iostream>

using namespace std;

int main() {

int number;

cout << "Enter an integer: ";

cin >> number;

if (number > 0) {

cout << "You entered a positive integer: " << number << endl;

}

else if (number < 0) {

cout << "You entered a negative integer: " << number << endl;

}

else {

cout << "You entered 0." << endl;

}

cout << "This line is always printed.";

return 0;

}

**Output 1**

Enter an integer: 1

You entered a positive integer: 1.

This line is always printed.

**Output 2**

Enter an integer: -2

You entered a negative integer: -2.

This line is always printed.

**Output 3**

Enter an integer: 0

You entered 0.

This line is always printed.

In this program, we take a number from the user. We then use the if...else if...else ladder to check whether the number is positive, negative, or zero.

If the number is greater than 0, the code inside the if block is executed. If the number is less than 0, the code inside the else if block is executed. Otherwise, the code inside the else block is executed.

**C++ Nested if...else**

Sometimes, we need to use an if statement inside another if statement. This is known as nested if statement.

Think of it as multiple layers of if statements. There is a first, outer if statement, and inside it is another, inner if statement. Its syntax is:

// outer if statement

if (condition1) {

// statements

// inner if statement

if (condition2) {

// statements

}

}

**Notes:**

* We can add else and else if statements to the inner if statement as required.
* The inner if statement can also be inserted inside the outer else or else if statements (if they exist).
* We can nest multiple layers of if statements.

### Example 4: C++ Nested if

// C++ program to find if an integer is even or odd or neither (0)

// using nested if statements

#include <iostream>

using namespace std;

int main() {

int num;

cout << "Enter an integer: ";

cin >> num;

// outer if condition

if (num != 0) {

// inner if condition

if ((num % 2) == 0) {

cout << "The number is even." << endl;

}

// inner else condition

else {

cout << "The number is odd." << endl;

}

}

// outer else condition

else {

cout << "The number is 0 and it is neither even nor odd." << endl;

}

cout << "This line is always printed." << endl;

}

**Output 1**

Enter an integer: 34

The number is even.

This line is always printed.

**Output 2**

Enter an integer: 35

The number is odd.

This line is always printed.

**Output 3**

Enter an integer: 0

The number is 0 and it is neither even nor odd.

This line is always printed.

In the above example,

* We take an integer as an input from the user and store it in the variable num.
* We then use an if...else statement to check whether num is not equal to 0.
  + If true, then the **inner** if...else statement is executed.
  + If false, the code inside the **outer** else condition is executed, which prints "The number is 0 and neither even nor odd."
* The **inner** if...else statement checks whether the input number is divisible by 2.
  + If true, then we print a statement saying that the number is even.
  + If false, we print that the number is odd.

Notice that 0 is also divisible by 2, but it is actually not an even number. This is why we first make sure that the input number is not 0 in the outer if condition.

**Note:** As you can see, nested if...else makes your logic complicated. If possible, you should always try to avoid nested if...else.

**Body of if...else With Only One Statement**

If the body of if...else has only one statement, you can omit { } in the program. For example, you can replace

int number = 5;

if (number > 0) {

cout << "The number is positive." << endl;

}

else {

cout << "The number is negative." << endl;

}

with

int number = 5;

if (number > 0)

cout << "The number is positive." << endl;

else

cout << "The number is negative." << endl;

The output of both programs will be the same.

**Note:** Although it's not necessary to use { } if the body of if...else has only one statement, using { } makes your code more readable.

**Tasks:**

**1.**Write a C program to accept two integers and check whether they are equal or not.    
Test Data : 15 15**Expected Output :**  
Number1 and Number2 are equal

**2.**Write a C program to check whether a given number is even or odd.    
Test Data : 15  
**Expected Output :**  
15 is an odd integer

**3.** Write a C program to check whether a given number is positive or negative.    
Test Data : 15  
**Expected Output :**  
15 is a positive number

**4.**Write a C program to find whether a given year is a leap year or not.    
Test Data : 2016  
**Expected Output :**  
2016 is a leap year.

**5.**Write a C program to read the age of a candidate and determine whether it is eligible for casting his/her own vote.    
Test Data : 21  
**Expected Output :**  
Congratulation! You are eligible for casting your vote.

**6.**Write a C program to read the value of an integer m and display the value of n is 1 when m is larger than 0, 0 when m is 0 and -1 when m is less than 0.    
Test Data : -5  
**Expected Output :**  
The value of n = -1

**7.** Write a C program to accept the height of a person in centimeter and categorize the person according to their height.   
Test Data : 135  
**Expected Output :**  
The person is Dwarf.

**8.**Write a C program to find the largest of three numbers.    
Test Data : 12 25 52  
**Expected Output :**  
1st Number = 12,        2nd Number = 25,        3rd Number = 52  
The 3rd Number is the greatest among three

**9.**Write a C program to accept a coordinate point in a XY coordinate system and determine in which quadrant the coordinate point lies.    
Test Data : 7 9  
**Expected Output :**  
The coordinate point (7,9) lies in the First quadrant.

**10.** Write a C program to find the eligibility of admission for a professional course based on the following criteria.

**Eligibility Criteria :**

Marks in Maths >=65 and Marks in Phy >=55 and Marks in Chem>=50 and Total in all three subject >=190

Or

Total in Maths and Physics >=140 ------------------------------------- Input the marks obtained in Physics :65 Input the marks obtained in Chemistry :51 Input the marks obtained in Mathematics :72 Total marks of Maths, Physics and Chemistry : 188 Total marks of Maths and Physics : 137 The candidate is not eligible.  
**Expected Output :**  
The candidate is not eligible for admission.

**Lab 9**

**Nested IF-ELSE**

In C programming, we can have if-else statement within another if-else statement. When if-else statement comes within another if-else statement then this is known nesting of if-else statement.

Syntax for Nested if-else Statement

if (Condition1)

{

if(Condition2)

{

Statement1;

}

else

{

Statement2;

}

}

else

{

if(Condition3)

{

Statement3;

}

else

{

Statement4;

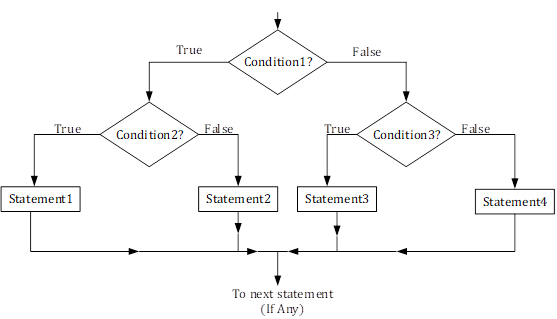
}

}

**Working of Nested if-else Statement**

In this example of nested if-else, it first tests Condition1, if this condition is TRUE then it tests Condition2, if Condition2 evaluates to TRUE then Statement1 is executed otherwise Statement2 will be executed. Similarly, if Condition1 is FALSE then it tests Condition3, if Condition3 evaluates to TRUE then Statement3 is executed otherwise Statement4 is executed. The working of nested if-else presented in this example can be further illustrated by following flowchart.

Flowchart for Nested if-else Statement



Working of Nested if-else Statement

**Examples for Nested if-else Statement**

***Example 1: C program to find largest from three numbers given by user to Explain Nested if-else***

#include<stdio.h>

int main()

{

int num1, num2, num3;

printf("Enter three numbers:\n");

scanf("%d%d%d",&num1, &num2, &num3);

if(num1>num2)

{

/\* This is nested if-else \*/

if(num1>num3)

{

printf("Largest = %d", num1);

}

else

{

printf("Largest = %d", num3);

}

}

else

{

/\* This is nested if-else \*/

if(num2>num3)

{

printf("Largest = %d", num2);

}

else

{

printf("Largest = %d", num3);

}

}

return(0);

}

Output of Above Program to Illustrate Nested if-else

Run 1:

---------------

Enter three numbers:

12↲

33↲

-17↲

Largest = 33

Run 2:

---------------

Enter three numbers:

-18↲

-31↲

-17↲

Largest = -17

Run 3:

---------------

Enter three numbers:

118↲

19↲

-87↲

Largest = 118

**Tasks**

1. [Write a C program to input marks of five subjects Physics, Chemistry, Biology, Mathematics and Computer. Calculate percentage and grade according to following:](https://codeforwin.org/2015/05/c-program-to-enter-student-marks-and-calculate-percentage-and-grade.html)  
   Percentage >= 90% : Grade A  
   Percentage >= 80% : Grade B  
   Percentage >= 70% : Grade C  
   Percentage >= 60% : Grade D  
   Percentage >= 40% : Grade E  
   Percentage < 40% : Grade F
2. [Write a C program to input basic salary of an employee and calculate its Gross salary according to following:](https://codeforwin.org/2015/05/c-program-to-calculate-gross-salary-of-employee.html)  
   Basic Salary <= 10000 : HRA = 20%, DA = 80%  
   Basic Salary <= 20000 : HRA = 25%, DA = 90%  
   Basic Salary > 20000 : HRA = 30%, DA = 95%
3. [Write a C program to input electricity unit charges and calculate total electricity bill according to the given condition:](https://codeforwin.org/2015/05/c-program-to-calculate-electricity-bill.html)  
   For first 50 units Rs. 0.50/unit  
   For next 100 units Rs. 0.75/unit  
   For next 100 units Rs. 1.20/unit  
   For unit above 250 Rs. 1.50/unit  
   An additional surcharge of 20% is added to the bill

**LAB 10**

**LOOPS**

# Introduction

In this lab, you will learn about **nested loops.** Moreover, we will discuss some preliminaries about single-dimensional **arrays**.

The section 2 presents a table that outlines some major activities and tasks you will do as the part of this lab. Table 1 also provides the estimated-time for each activity, which will help you to organize your tasks well. Section 3 presents some of the learning objectives for this lab. Section 4 (“Concept Map”) discusses and provides a comprehensive introduction of the topic. Section 5 lists the set of home-tasks you are required to complete before this lab. Section 6 presents a “walkthrough task” that you will do as the first practical activity during your lab. The walkthrough task has many small steps which you should follow as directed in-order to complete the task and to get the desired output. After that, you will be ready to work on some tasks on your own. The section 7 lists practice tasks for this purpose. As the part of section 8, your lab instructor will give you some tasks at runtime and will evaluate those according to the criteria mentioned in section 9. Section 10 lists some further reading links.

# Objective of the Experiment

* To get basic understanding of loops.
* To practice the use of nested loop structures.
* Learning basics of constants and arrays.

# Concept Map

## Nested Loops

In lab-6, you have already learned and practiced the loop structures which can be used to repeat/iterate set of statements based on some number of iterations or based on some logical condition. Basically C++ provides three types of loops: *for-loop*, *while-loop*, and *do-loop*. To solve different problems, you have exercised the use of loops in you program.

Consider a scenario, where you are required to repeat a complete loop. For example: you are asked to write a C++ program that prints 6 lines (where each line contains 25 even numbers starting from 2 and ending at 50) as shown below:

**2, 4, 6, 8, 10, 12, 14, 16, 18, 20, 22, 24, 26, 28, 30, 32, 34, 36, 38, 40, 42, 44, 46, 48, 50,**

**2, 4, 6, 8, 10, 12, 14, 16, 18, 20, 22, 24, 26, 28, 30, 32, 34, 36, 38, 40, 42, 44, 46, 48, 50,**

**2, 4, 6, 8, 10, 12, 14, 16, 18, 20, 22, 24, 26, 28, 30, 32, 34, 36, 38, 40, 42, 44, 46, 48, 50,**

**2, 4, 6, 8, 10, 12, 14, 16, 18, 20, 22, 24, 26, 28, 30, 32, 34, 36, 38, 40, 42, 44, 46, 48, 50,**

**2, 4, 6, 8, 10, 12, 14, 16, 18, 20, 22, 24, 26, 28, 30, 32, 34, 36, 38, 40, 42, 44, 46, 48, 50,**

**2, 4, 6, 8, 10, 12, 14, 16, 18, 20, 22, 24, 26, 28, 30, 32, 34, 36, 38, 40, 42, 44, 46, 48, 50,**

How will you code this program? Like below?

**#include<iostream>**

**using namespace std;**

**void main()**

**{**

**for(int i=2;i<=50;i+=2)**

**cout<<i<<’, ’;**

**cout<<endl;**

**for(int i=2;i<=50;i+=2)**

**cout<<i<<’, ’;**

**cout<<endl;**

**for(int i=2;i<=50;i+=2)**

**cout<<i<<’, ’;**

**cout<<endl;**

**for(int i=2;i<=50;i+=2)**

**cout<<i<<’, ’;**

**cout<<endl;**

**for(int i=2;i<=50;i+=2)**

**cout<<i<<’, ’;**

**cout<<endl;**

**for(int i=2;i<=50;i+=2)**

**cout<<i<<’, ’;**

**}**

As you can see that it is not an elegant looking code although the output was correctly produced. Assume that you are required to produce the similar output but instead of only 6 lines of output you were asked to output 1000 lines.If you code you program as shown above, you can imagine how untidy looking source code will be produced.

As you have noticed that in the above program we actually need to repeat the entire for loop (shown below) to **6** or **1000** number of times.

**for(int i=2;i<=50;i+=2)**

**cout<<i<<’, ’;**

The more elegant solution to the problem would be to place the above shown loop inside another loop (nesting of a loop) which iterates 6 or 1000 times. Following code shows the improved version of the program:

**#include<iostream>**

**using namespace std;**

**void main()**

**{**

**for(int j=0;j<6;j++)**

**{**

**for(int i=2;i<=50;i+=2)**

**cout<<i<<’, ’;**

**cout<<endl;**

**}**

**}**

Loop nesting (similar to nested if) is a code arrangement where we place a complete loop inside another loop. Above code shows that we have placed***i-loop*** (which prints even numbers on the screen) inside another loop (which repeats the inner loop 6 times). Loop nesting can be done at two or some higher levels as required. In C++, you are allowed to nest loops (loop inside another loop) at any level of nesting. For example a level 3 loop nesting means that there are three loops which are nested in a hierarchy such as:

**#include<iostream>**

**using namespace std;**

**void main()**

**{**

**for(int i=0;i<3;i++)**

**{**

**for(intj=0;j<3;j++)**

**{**

**for(int k=0;k<3;k++)**

**cout<<”\nHello World!”; }**

**}**

**}**

In the above shown code “Hello World!” will be printed on screen 27 times (3 \* 3 \* 3 = 27). In C++, you can nest any kind of loop (for, while, or do loop). For example: a while loop may contain for loop (as a nested loop). Or a do loop may contain a while loop which further contains a for loop, etc. Some of the nested loops example is given below:

|  |  |
| --- | --- |
| **do**  **{**  **cin>>a;**  **for(int i=0;i<a;i++)**  **cout<<(i\*i)<<” ”;**  **} while(a>10);** | **for(int i=0;i<10; i++)**  **{**  **cin>>n;**  **total = i \* n;**  **while(total>20)**  **{**  **cout<<”\nHello World”;**  **total--;**  **}**  **}** |

**Example 1: Program to print half pyramid using \***

\*

\* \*

\* \* \*

\* \* \* \*

\* \* \* \* \*

**Source Code**

#include<iostream>

usingnamespacestd;

int main()

{

int rows;

cout<<"Enter number of rows: ";

cin>> rows;

for(int i = 1; i <= rows; ++i)

{

for(int j = 1; j <= i; ++j)

{

cout<<"\* ";

}

cout<<"\n";

}

return0;

}

**Example 2: Program to print half pyramid a using numbers**

1

1 2

1 2 3

1 2 3 4

1 2 3 4 5

**Source Code**

#include<iostream>

usingnamespacestd;

int main()

{

int rows;

cout<<"Enter number of rows: ";

cin>> rows;

for(int i = 1; i <= rows; ++i)

{

for(int j = 1; j <= i; ++j)

{

cout<< j <<" ";

}

cout<<"\n";

}

return0;

}

**Example 3: Program to print half pyramid using alphabets**

A

B B

C CC

D DDD

E EEEE

**Source Code**

#include<iostream>

usingnamespacestd;

int main()

{

char input, alphabet = 'A';

cout<<"Enter the uppercase character you want to print in the last row: ";

cin>> input;

for(int i = 1; i <= (input-'A'+1); ++i)

{

for(int j = 1; j <= i; ++j)

{

cout<< alphabet <<" ";

}

++alphabet;

cout<<endl;

}

return0;

}

**Programs to print inverted half pyramid using \* and numbers**

**Example 4: Inverted half pyramid using \***

\* \* \* \* \*

\* \* \* \*

\* \* \*

\* \*

\*

**Source Code**

#include<iostream>

usingnamespacestd;

int main()

{

int rows;

cout<<"Enter number of rows: ";

cin>> rows;

for(int i = rows; i >= 1; --i)

{

for(int j = 1; j <= i; ++j)

{

cout<<"\* ";

}

cout<<endl;

}

return0;

}

**Example 5: Inverted half pyramid using numbers**

1 2 3 4 5

1 2 3 4

1 2 3

1 2

1

**Source Code**

#include<iostream>

usingnamespacestd;

int main()

{

int rows;

cout<<"Enter number of rows: ";

cin>> rows;

for(int i = rows; i >= 1; --i)

{

for(int j = 1; j <= i; ++j)

{

cout<< j <<" ";

}

cout<<endl;

}

return0;

}

**Programs to display pyramid and inverted pyramid using \* and digits**

**Example 6: Program to print full pyramid using \***

\*

\* \* \*

\* \* \* \* \*

\* \* \* \* \* \* \*

\* \* \* \* \* \* \* \* \*

**Source Code**

#include<iostream>

usingnamespacestd;

int main()

{

int space, rows;

cout<<"Enter number of rows: ";

cin>> rows;

for(int i = 1, k = 0; i <= rows; ++i, k = 0)

{

for(space = 1; space <= rows-i; ++space)

{

cout<<" ";

}

while(k != 2\*i-1)

{

cout<<"\* ";

++k;

}

cout<<endl;

}

return0;

}

**Example 7: Program to print pyramid using numbers**

1

2 3 2

3 4 5 4 3

4 5 6 7 6 5 4

5 6 7 8 9 8 7 6 5

**Source Code**

#include<iostream>

usingnamespacestd;

int main()

{

int rows, count = 0, count1 = 0, k = 0;

cout<<"Enter number of rows: ";

cin>> rows;

for(int i = 1; i <= rows; ++i)

{

for(int space = 1; space <= rows-i; ++space)

{

cout<<" ";

++count;

}

while(k != 2\*i-1)

{

if (count <= rows-1)

{

cout<<i+k<<" ";

++count;

}

else

{

++count1;

cout<< i+k-2\*count1 <<" ";

}

++k;

}

count1 = count = k = 0;

cout<<endl;

}

return0;

}

**Example 8: Inverted full pyramid using \***

\* \* \* \* \* \* \* \* \*

\* \* \* \* \* \* \*

\* \* \* \* \*

\* \* \*

\*

**Source Code**

#include<iostream>

usingnamespacestd;

int main()

{

int rows;

cout<<"Enter number of rows: ";

cin>> rows;

for(int i = rows; i >= 1; --i)

{

for(int space = 0; space < rows-i; ++space)

cout<<" ";

for(int j = i; j <= 2\*i-1; ++j)

cout<<"\* ";

for(int j = 0; j < i-1; ++j)

cout<<"\* ";

cout<<endl;

}

return0;

}

**Example 9: Print Pascal's triangle**

1

1 1

1 2 1

1 3 3 1

1 4 6 4 1

1 5 10 10 5 1

**Source Code**

#include<iostream>

usingnamespacestd;

int main()

{

int rows, coef = 1;

cout<<"Enter number of rows: ";

cin>> rows;

for(int i = 0; i < rows; i++)

{

for(int space = 1; space <= rows-i; space++)

cout<<" ";

for(int j = 0; j <= i; j++)

{

if (j == 0 || i == 0)

coef = 1;

else

coef = coef\*(i-j+1)/j;

cout<<coef<<" ";

}

cout<<endl;

}

return0;

}

Another logic:

1. */\**
2. *\* C++ Program to Print Pascal's Triangle*
3. *\*/*
5. #include<iostream>
6. using namespace std;
8. int main()
9. {
10. int rows;
11. cout<<"Enter the number of rows : ";
12. cin>> rows;
13. cout<<endl;
15. for(int i =0; i < rows; i++)
16. {
17. intval=1;
18. for(int j =1; j <(rows - i); j++)
19. {
20. cout<<" ";
21. }
22. for(int k =0; k <= i; k++)
23. {
24. cout<<" "<<val;
25. val=val\*(i - k)/(k +1);
26. }
27. cout<<endl<<endl;
28. }
29. cout<<endl;
30. return0;
31. }

**Example 10: Print Floyd's Triangle.**

1

2 3

4 5 6

7 8 9 10

**Source Code**

#include<iostream>

usingnamespacestd;

int main()

{

int rows, number = 1;

cout<<"Enter number of rows: ";

cin>> rows;

for(int i = 1; i <= rows; i++)

{

for(int j = 1; j <= i; ++j)

{

cout<< number <<" ";

++number;

}

cout<<endl;

}

return0;

}

**Lab 11**

*NESTED FOR LOOPS:*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **iv)** | \*       \*\*\*     \*\*\*\*\*   \*\*\*\*\*\*\* \*\*\*\*\*\*\*\*\* | **v)** | **1       222     33333   4444444 555555555** | **vi)** | **1       212     32123   4321234 543212345** |

*//Solution of (iv)*

#include<iostream>

using namespacestd;

int main()

{

inti,j,k;

for(i=1;i<=5;i++)

{

for(j=5;j>i;j--)

cout<<' ';

for(k=1;k<2\*i;k++)

cout<<'\*';

cout<<endl;

}

return 0;

}

*//Solution of (v)*

#include<iostream>

using namespacestd;

int main()

{

inti,j,k;

for(i=1;i<=5;i++)

{

for(j=5;j>i;j--)

cout<<' ';

for(k=1;k<2\*i;k++)

cout<<i;

cout<<endl;

}

return 0;

}

*//Solution of (vi)*

#include<iostream>

using namespacestd;

int main()

{

inti,j,k,l;

for(i=1;i<=5;i++)

{

for(j=5;j>i;j--)

cout<<' ';

for(k=i;k>=1;k--)

cout<<k;

for(l=2;l<=i;l++)

cout<<l;

cout<<endl;

}

return 0;

}

**Example 8: Inverted full pyramid using \***

\* \* \* \* \* \* \* \* \*

\* \* \* \* \* \* \*

\* \* \* \* \*

\* \* \*

\*

**Source Code**

#include<iostream>

usingnamespacestd;

int main()

{

int rows;

cout<<"Enter number of rows: ";

cin>> rows;

for(int i = rows; i >= 1; --i)

{

for(int space = 0; space < rows-i; ++space)

cout<<" ";

for(int j = i; j <= 2\*i-1; ++j)

cout<<"\* ";

for(int j = 0; j < i-1; ++j)

cout<<"\* ";

cout<<endl;

}

return0;

}

**Example 9: Print Pascal's triangle**

1

1 1

1 2 1

1 3 3 1

1 4 6 4 1

1 5 10 10 5 1

**Source Code**

#include<iostream>

usingnamespacestd;

int main()

{

int rows, coef = 1;

cout<<"Enter number of rows: ";

cin>> rows;

for(int i = 0; i < rows; i++)

{

for(int space = 1; space <= rows-i; space++)

cout<<" ";

for(int j = 0; j <= i; j++)

{

if (j == 0 || i == 0)

coef = 1;

else

coef = coef\*(i-j+1)/j;

cout<<coef<<" ";

}

cout<<endl;

}

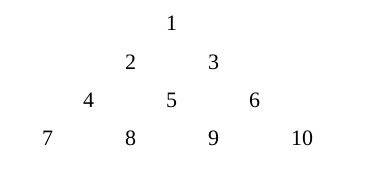
return0;

}

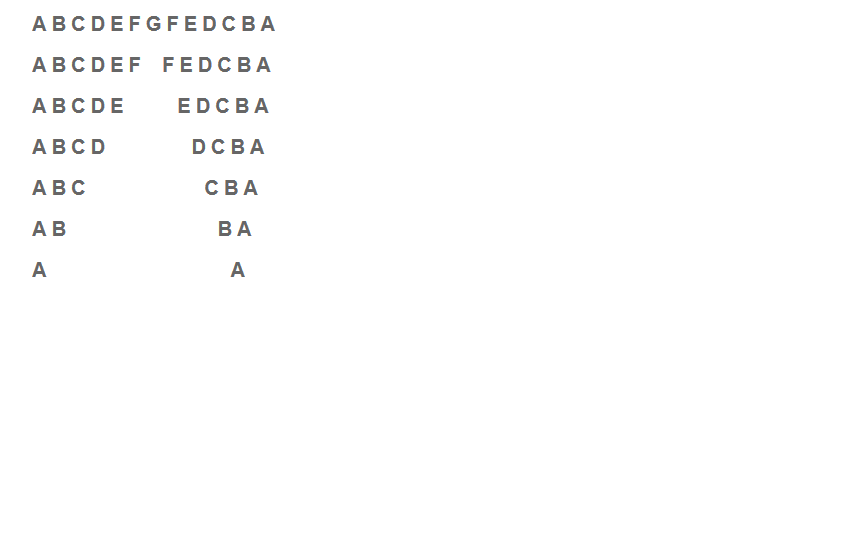
Another logic:

1. */\**
2. *\* C++ Program to Print Pascal's Triangle*
3. *\*/*
5. #include<iostream>
6. using namespace std;
8. int main()
9. {
10. int rows;
11. cout<<"Enter the number of rows : ";
12. cin>> rows;
13. cout<<endl;
15. for(int i =0; i < rows; i++)
16. {
17. intval=1;
18. for(int j =1; j <(rows - i); j++)
19. {
20. cout<<" ";
21. }
22. for(int k =0; k <= i; k++)
23. {
24. cout<<" "<<val;
25. val=val\*(i - k)/(k +1);
26. }
27. cout<<endl<<endl;
28. }
29. cout<<endl;
30. return0;
31. }

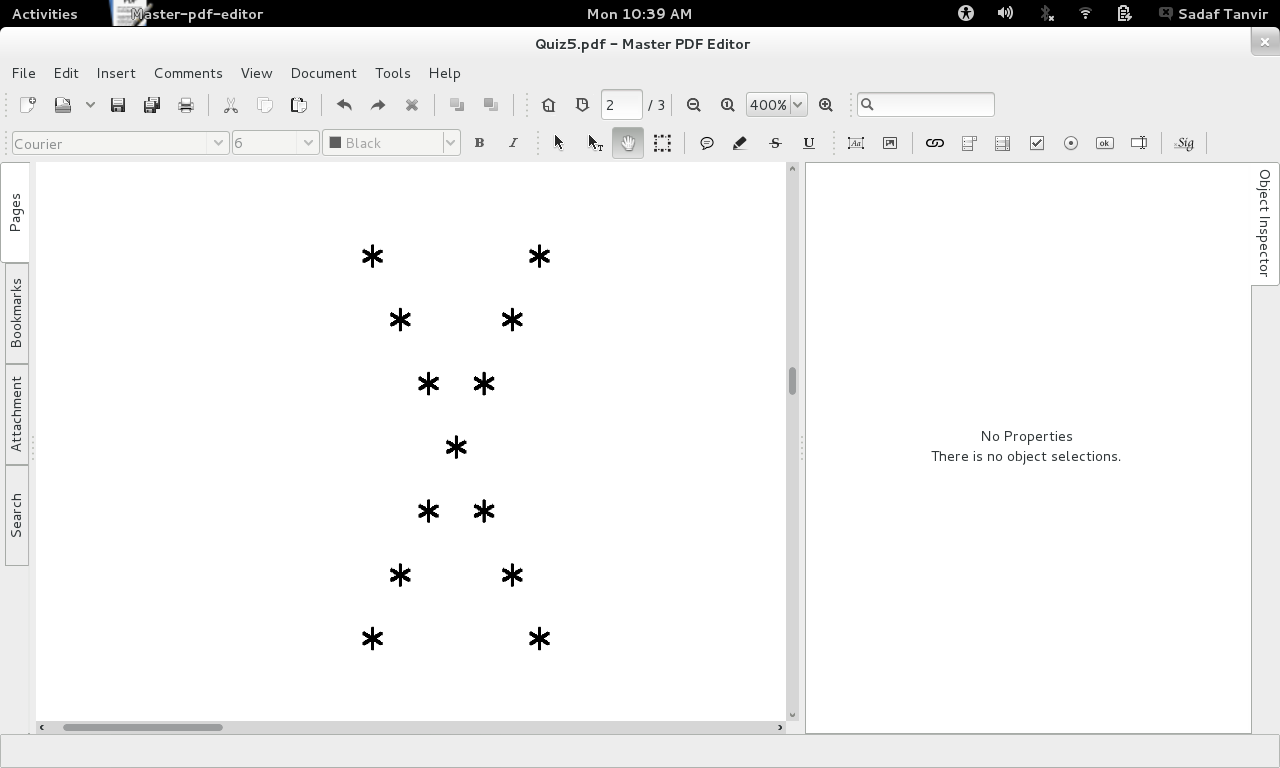
Write source code for following pattern:



Write source code for following pattern:



Print following pattern:



**Lab 12: Repetition Structure Loops**

Lab 12: Repetition Structures - Loops

# Introduction

In the previous labs, you have learned how to use a typical control structure i.e conditions. There is another control structure which is known as repetition structure or loops. The loops also control the flow of your program. This lab will introduce you to different loops and their usage. You will also learn the scenarios where a typical type of loop can be used.

The introduction includes some “walkthrough tasks/examples” that you will do as the first practical activity during your lab. The walkthrough tasks have many small steps which you should follow as directed in-order to get the desired output. After that, you will be ready to work on some tasks on your own. The end of manual contains tasks that you will do as the part of this lab.

# Objective of the Experiment

* To get basic understanding of loops
* To learn about post-condition and pre-condition loops.
* To learn different possible expressions which can be used in loops
* To practice how the loops can be used in the program.

# Concept Map

## Why using loops

Loops are used in the program when certain types of statements are repeated in the program to achieve a specific task. For example, consider the following scenario:

Suppose you want to display **“Welcome to CP Course”** 10 times on the screen

|  |
| --- |
| **cout<<” Welcome to Cp Course\n”;**  **cout<<” Welcome to Cp Course\n”;**  **cout<<” Welcome to Cp Course\n”;**  **cout<<” Welcome to Cp Course\n”;**  **cout<<” Welcome to Cp Course\n”;**  **cout<<” Welcome to Cp Course\n”;**  **cout<<” Welcome to Cp Course\n”;**  **cout<<” Welcome to Cp Course\n”;**  **cout<<” Welcome to Cp Course\n”;**  **cout<<” Welcome to Cp Course\n”;** |

To achieve this, you know that you can write 10 statements in the following manner:

However, this is not elegant way to achieve this task, although the above statements will produce the required output.

You can notice that the statement **cout<<” Welcome to Cp Course\n”;** is repeating 10 times. This type of statements can be written using a loop. For example, consider the following code which will print the same output:

|  |
| --- |
| **for(int i=1;i<=10;i++)**  **cout<<”Welcome to Cp Course\n”;** |

Wow, this looks great. You have achieved the task of printing a message 10 times using just two lines. Please notice, if we want to print the same message 20 times, we will just replace 10 with 20 in the above code just like the followings:

|  |
| --- |
| **for(int i=1;i<=20;i++)**  **cout<<”Welcome to Cp Course\n”;** |

How this program works, we will learn in the following sections. We will start formally describing loops. The loops can be categorized into two major types such as: Pre-condition loops and post-condition loops

## Pre-condition Loops

The pre-condition loops first checks the condition, if the condition is true then the statement associated with loops are executed, otherwise not. There are two pre-conditions loops such as: for-loop and while-loop. First we will discuss for-loop followed by the other pre-condition loop : while-loop.

### For-Loop

Consider the code written in the previous section

|  |
| --- |
| **for(int i=1;i<=20;i++)**  **cout<<”Welcome to Cp Course\n”;** |

This loop is called for-loop. This loop has initialization such as int i=1 from where the loop is going to be started. Then after semi-colon, there is a condition i<=20, and then the increment i++. The working of this loop will be as follows:

1. First the initialization will be done.
2. Then condition will be checked. Then the statements associated with this loop will be executed. At the moment, there is only one statement that is associated with this loop.
3. After the statement execution, the increment will be done i.e i will become 2.
4. Then the step (b) and (c) will be repeated until i becomes 21. At that stage the condition will be false and the loop will terminate. Then the next statements will be executed in a flow.

You will learn about many different categories of controlling the conditions of the loop in the next section where we will discuss the while loop.

### while-Loop

The while-loop is also a pre-conditioned loop. The condition is evaluated first before executing the statements of the loop. The previous code can be re-written with while loop in the following way.

|  |
| --- |
| **int i=1;**  **while(;i<=20;)**  **{**  **cout<<” Welcome to Cp Course\n”;**  **cout<<”Hello class”;**  **i++;**  **}** |

This loop is executed in the similar way as discussed previously about for-loop. This type of loop is called counter controlled loop.

**Example 1:** Print Numbers from 1 to 10 using a while loop

#include <iostream.h>

int main()

{

int counter = 1;

while (counter <= 10) // condition

{

cout<<“Counter now reads \n”<<counter;

counter++; // Same as counter=counter + 1 (increment)

}

return 0;

}

Our 2nd example is based on a while-loop that keeps on running until a certain condition is reached (a certain value is entered by the user).

**Example 2:** Printing the numbers you entered using a while loop

#include <iostream.h>

int main()

{

int flag; //flag is just an integer variable

cout<<“Enter any number: ( -1 to quit) \n”;

cin>>flag;

cout<<“Entering the while loop now...\n”;

while(flag != -1) {

cout<<“Enter any number: ( -1 to quit)\n ”;

cin>>flag;

cout<<“You entered \n”<<flag;

}

Cout<<“Out of loop now \n”;

return 0;

}

However, there are some other types of controlling a loop.

**Sentinel-Controlled while loop**

Some times, we do not know how many times we will execute a loop. For example, we want to sum the marks of all registered courses of a student. However, we do not know how many courses a typical student has registered. Foe example, for a student ‘a’, we want to execute the loop 4 times, while for a student ‘b’, we want to execute the loop 5 times etc. in this scenario, we define a sentinel or read sentinel from the user, and when this sentinel is entered the loop is terminated. For example, consider the following scenario:

|  |
| --- |
| **int marks;**  **int sum=0;**  **constint sentinel=-1; /// the marks of a student can not be negative.**  **cout<<”Enter the marks of the student”;**  **cin>>marks;**  **while(marks!=sentinel)**  **{**  **sum=sum+marks;**  **cin>>marks;**  **}**  **cout<<”The total marks of this students are”<<sum;** |

Now the user is free to enter the marks of a student for any un-specified quantity. However, once the user enters the -1, the loop is terminated.

## 3.2.3 do-while-Loop

The loops discussed in the above section are known as pre-condition loops. This means first the condition is checked and then the associated statements are executed. However, there is another type of loops, which are known as post-conditioned loops. This means the loop will execute at least once and then will check the condition, if the conditions true the loop will execute again, otherwise will be terminated. Such a loop can be implemented in C++ using do-while structure. The example discussed in the above section has be re-written using do-while loop below:

|  |
| --- |
| **int i=1;**  **do {**  **cout<<” Welcome to Cp Course\n”;**  **cout<<”Hello class”;**  **i++;**  **} while(i<=20);** |

### To understand the usage of Do-While loop

Starting from the very basic program

#include <stdio.h>

#include <iostream.h>

#include <conio.h>

int main( )

{

int x = 0; // Create a local variable 'x'

do

{

x=x+1; // Increment the variable 'x' by 1

} while (x < 3);

return 0;

}

**A more advanced example**

Following loop reads marks until a valid mark is entered:

#include <stdio.h>

#include <iostream.h>

#include <conio.h>

int main( )

{

do

{

cout<< "Please enter a mark: ";

cin>> mark;

if (mark < 0 || mark > 100)

cout<< "Invalid mark. Try again. " <<endl<<endl;

}while (mark < 0 || mark > 100);

Return 0;

}

The following C++ code shows a count-controlled Do-while statement:

sum = 0;

counter = 0;

do

{

cout<< "Please enter a mark: ";

cin>> mark;

sum = sum + mark;

counter++;

} while (counter < 10);

cout<< "The average of the marks entered is " << sum/counter <<endl;

# Procedure & Tools

### Visual Studio 2013.

# Practice Tasks [Expected time = 2.5 hours]

This section will provide more practice exercises which you need to finish during the lab. You need to finish the tasks in the required time.

## Practice Task 1

Write a program to enter the numbers till the user wants and at the end it should display the count of positive, negative and zeros entered. (use do while)

## Practice Task 2

Write a C++ program that asks user for two numbers x and y. Write a program to find the value of one number raised to the power of another.

Put program in a **do-while loop.** Once the program gives the output, the user will see another message “Do you want to continue again [Y/N]”, if user enters ‘Y’ or ‘y’, the program will execute again, however, on pressing any key other than ‘Y’ or ‘y’, the program will terminate.

## Practice Task 3

Write a program to enter the numbers till the user wants and at the end it should display the maximum and minimum number entered.

## 

## Practice Task 4

Write a program to print Fibonacci series of n terms where n is input by user :

0 1 1 2 3 5 8 13 24 .....

## Practice Task 5

Write a program that computes the natural logarithm of 2, by adding up to n terms in the series

1 - 1/2 + 1/3 - 1/4 + 1/5 -... 1/n

where n is a positive integer and input by user.

## Out comes

The outcomes of this lab were:

1. You have learnt loops control structures in C++
2. You have learned different scenarios to use while, and do-while loops
3. You have practiced different tasks how to use loops.

LAB No. 13

Functions

# Objectives of this lab:

### To understand the Concept of functions

Let’s do an example which calls a function which prints ten asterisks (\*) in line. (\*\*\*\*\*\*\*\*\*\*)

#include<iostream.h>

voidasteriks(); // prototype declaration

void main(){

asteriks(); // Function calling

}

voidasteriks(){ // Function definition

int i=0;

for(;i<10;i++)

cout<<"\*";

}

Lets go one step ahead, function asterisks (int a) with a single argument.

#include<iostream.h>

voidasteriks(int n); // prototype declaration

void main(){

asteriks(7); // Function calling

}

voidasteriks(intnum){ // Function definition

int i=0;

for(;i<num;i++)

cout<<"\*";

}

Now an example of a function with two parameters

Here is an example of a function which calculates the sum of two numbers passed as arguments to the function and returns the sum to the calling function.

int add(int no1,int no2 ){

int sum = no1+no2;

return sum;

}

Let’s do another example of making a program using functions which will tell us whether the input number is even or odd.

#include <iostream.h>

intis\_even(int n); // (Prototype declaration)

void main()

{

int number, test;

cout<<"Enter a number to test even or odd";

cin>>number;

test = is\_even(number); // (Function calling)

if(test==0)

cout<<"\nThe number is odd";

else

cout<<"\nThe number is even";

}

intis\_even(int n){ //Function definition

int remainder;

remainder= n%2;

if(remainder==1)

return 0;

else

return 1;

}

One more way to write the same example is

#include <iostream.h>

intis\_even(int n){ //Function definition

int remainder;

remainder= n%2;

if(remainder==1)

return 0;

else

return 1;

}

void main()

{

int number, test;

cout<<"Enter a number to test even or odd\n";

cin>>number;

test = is\_even(number); //(Function calling)

if(test==0)

cout<<"\nThe number is odd";

else

cout<<"\nThe number is even";

}

Another way to write the same program

#include <iostream.h>

intis\_even(int n); // (Prototype declaration)

void main()

{

int number;

cout<<"Enter a number to test even or odd\n";

cin>>number;

if(is\_even(number)) //Function call within if condition

cout<<"\nThe number is even";

else

cout<<"\nThe number is odd";

}

intis\_even(int n){ //Function definition

int remainder;

remainder= n%2;

if(remainder==1)

return 0;

else

return 1;

}

**Compile and run this program**.

#include <iostream.h>

floatavg(float no1, float no2, float no3); //(Prototype declaration)

void main()

{

float n1,n2,n3,result;

cout<<"Enter three number to find avg");

cin>>n1>>n2>>n3;

result=avg(n1,n2,n3); // (Function calling)

cout<<"\nThe average is \n”<<result;

}

floatavg(float a, float b, float c){

cout<<"\nEntering the function\n";

float average= (a+b+c)/3;

return average;

}

**NOTE:**

When you pass parameters by valuethen the actual parameters are copied to formal parameters. The changes made within the function are not visible to the place from you called the function; however, if you pass parameters by reference, then the address of formal parameters is copied to the actual parameters. This means that both formal and actual parameters are pointing to the same memory location. The changes made within the function in the formal parameters are visible to the actual parameters. You know that a function can return only one value; however, if you pass parameters using pass-by-reference, then you can return more than one values from a function.

**PRACTICE TASKS:**

TASK 1: (pass by value)

Write a function to calculate the factorial value of any integer as an argument. Call this function from main( ) and print the results in main( ).

Task 2: (pass by value)

Write a program that lets the user perform arithmetic operations on two numbers. Your program must be menu driven, allowing the user to select the operation (+, -, \*, or /) and input the numbers. Furthermore, your program must consist of following functions:

1. Function showChoice: This function shows the options to the user and explains how to enter data.

2. Function add: This function accepts two number as arguments and returns sum.

3. Function subtract: This function accepts two number as arguments and returns their difference.

4. Function multiply: This function accepts two number as arguments and returns product.

5. Function divide: This function accepts two number as arguments and returns quotient.

TASK 3:

Write a function called zero\_small() that has two integer arguments being **passed by reference** and sets the smaller of the two numbers to 0. Write the main program to access the function.

TASK 4: (pass by value)

Write a function that receives two numbers as an argument and display all prime numbers between these two numbers. Call this function from main( ).

**Assignment: 10 marks**

Q1. Raising a number to a power p is the same as multiplying n by itself p times. Write a function called power that takes two arguments, a double value for n and an int value for p, and return the result as double value. Use default argument of 2 for p, so that if this argument is omitted the number will be squared. Write the main function that gets value from the user to test power function.

Q2. Write a function named "sum\_from\_to" that takes two integer arguments, call them "first" and "last", and returns as its value the sum of all the integers between firstand lastinclusive. Thus, for example,

cout<<sum\_from\_to(4,7) <<endl; //

will print 22 because 4+5+6+7 = 22

cout<<sum\_from\_to(-3,1) <<endl; //will print −5

cout<<sum\_from\_to(7,4) <<endl; //

will print 22 because 7+6+5+4 = 22

cout<<sum\_from\_to(9,9) <<endl; //

will print 9

Q3. Write a function that receives two numbers as an argument and displays odd numbers between these two numbers. Call this function from main.

LAB 14

Arrays

# Objectives of this lab:

### Declaration of Array

### Initialization of Array

### Accessing elements of Array

### Printing arrays

### Copying arrays

### Scanning array elements using cin

### Dealing with characters using arrays

Let's start by looking at the following code where a single variable is used to store a person's age.

Code

#include <stdio.h>

int main()

{

int age;

age=23;

cout<<endl<< age;

return 0;

}

**INTRODUCTION :**An array is basically a collection of data-items or values all based on same data-type. For example: A student’s marks in five courses in a semester: 78, 91, 83, 67, and 89; represent a collection of five **int**type numbers where each value represents his/her marks in a certain course. An array is a structured datatype (collection of values) of some fixed size (number of elements/values) where all values are of same type.

To define an array you have to mention its ***type, name*** and ***size*** (number of elements/values the array will contain). For example:

**int marks[5];**

Above statements defines an **int type** array named ***marks*,** having **size: 5** (capable of storing five values).In the similar way, you can create array of any basic type, Examples:

**float GPA[4]; //To store GPA of last 4 semesters**

**doublearea\_of\_circles[10]; //To store area of 10 circles**

Individual array elements can be accessed using **same name (array name)** along its **index**. An **index** (which must be an integer value**) indicates relative position of an element/value within an array.** In C++, array are 0-index based, it means that index of the starting/first element of an array is 0. For example: int marks[5]; defines an int-type array having five elements, where first value or element will be at index 0, second on index 1, thirds element at index 2, fourth at index 3, and the fifth element at index 4. Examples:

**cin>>marks[2]; //Gets marks from user and stores at 3rd position**

1. **Declaration of Array**

Here's is the code snippet to create an array and one way to initialize an array:

#include <iostream.h>

int main()

{

int age[4]; //declaration of Array

age[0]=23;//initialization of Array elements

age[1]=34;

age[2]=65;

age[3]=74;

return 0;

}

1. **Initialization of Array**

It is like a variable, an array can be initialized. To initialize an array, we provide initializing values which are enclosed within curly braces in the declaration and placed following an equals sign after the array name. Here is an example of initializing an integer array.

int age[4]={23,34,65,74};

1. **Printing arrays**

#include <iostream.h>

int main()

{

int age[4];

age[0]=23;

age[1]=34;

age[2]=65;

age[3]=74;

cout<< age<<endl;

return 0;

}

How about printing out each of the values separately? Try this:

#include <iostream.h>

int main()

{

int age[4];

age[0]=23;

age[1]=34;

age[2]=65;

age[3]=74;

cout<<age[0]<<endl;

cout<<age[1]<<endl;

cout<<age[2]<<endl;

cout<<age[3]<<endl;

return 0;

}

Lines (10) through line (13) produce the output we are expecting.

Thus there is no single statement in the language that says "print an entire array to the screen". Each element in the array must be printed to the screen individually.

1. **Copying arrays**

Suppose that after filling our 4 element array with values, we need to copy that array to another array of 4 int ? Try this:

#include <iostream.h>

int main()

{

int age[4];

intsame\_age[4];

age[0]=23;

age[1]=34;

age[2]=65;

age[3]=74;

for (int i=0;i<4;i++)

same\_age[i]=age[i];

for (i=0;i<4;i++)

cout<<same\_age[i]<<endl;

return 0;

}

1. **Scanning array elements using cin**

int a[5];

cin>>a[0]; // this will scan the value for the very first location of the array.

cout<<a[0]<<endl;

You can also scan the entire elements of the array using a loop.

1. **Dealing with characters using arrays**

You can also store characters and other type data (float etc.) in the arrays. Just declare it as we’ve done in the case with int. There is no difference in dealing with characters except you’ve to enclose the value in a single quote.

charar[3];

ar[0]=’a’ ; ar[1]=’b’ …..

As you have learned that array elements can be accessed using index values. Therefore, it is *common* and a very *convenient* way to access array elements using **loops** (mainly **for-loop**). For example, below shown code gets input from the user for all five elements of array *marks*.

**for(int i=0; i<5; i++)**

**cin>>marks[i];**

Arrays are very valuable data-structure that helps to process large amount of data of same-type. Arrays simplify coding of the programs (processing large amount of similar data) using fewer code lines as compared to the non-array based C++ program having same functionality.

**Task-1**

Write a C++ program to find the sum and average of one dimensional integer array.

**Task-2**

Write a C++ program to swap first and last element of an integer 1-d array.

**Task-3**

Write a C++ program to reverse the element of an integer 1-D array.

**Task-4**

Write a C++ program to find the largest and smallest element of an array.

**Task-5**

Write a C++ program creates an one dimensional array of 10 integers. Get input from the user for each array element. After that, calculate square (N \* N) of each array element and store at the corresponding array position. In the end, display the contents of the array on the screen.

**Sample Input:**

Enter values in the array:

6

4

3

8

2

9

2

1

7

5

**Sample Output:**

36

16

9

64

4

81

4

1

49

25

# Appendix A: Lab Evaluation Criteria

### Labs with projects

|  |  |  |
| --- | --- | --- |
| Experiments and their report |  | 50% |
| a. Experiment | 30% |  |
| b. Lab report | 20% |  |
| Quizzes (3-4) |  | 10% |
| Final evaluation |  | 40% |