Lab Manual

CSC101-Introduction to ICT

CUI

Department of Computer Science
Islamabad Campus

Lab Contents:

Introduction to MS Word; Introduction to MS PowerPoint; Introduction to MS Excel; Introduction to MS Access; Intoduction to HTML; Problem Solving: Sequential Structure, Decision Structure, Repitition Structure, and Functions; Python Elementary Programming; Conditional Statements; Loops & Nested Loops; Strings & Lists; and Functions.

Student Outcomes (SO)

Sr.#	Description
2	Identify, formulate, research literature, and solve complex computing problems reaching substantiated conclusions using fundamental principles of mathematics, computing sciences, and relevant domain disciplines.
4	Create, select, adapt and apply appropriate techniques, resources, and modern computing tools to complex computing activities, with an understanding of the limitations.

Intended Learning Outcomes

Sr.#	Description	Blooms Taxonomy Learning Level	SO
CLO -6	Demonstrate the usage of productivity software, problem solving tool, and web technology by performing appropriate tasks.	*Applying*	4
CLO -7	Implement an algorithm in a programming language to solve a simple problem.	*Applying*	2,4

Lab Assessment Policy

The lab work done by the student is evaluated using Psycho-motor rubrics defined by the course instructor, viva-voce, project work/performance. Marks distribution is as follows:

Assignments	Lab Mid Term Exam	Lab Terminal Exam	Total
25	25	50	100

Note: Midterm and Final term exams must be computer based.

`

List of Labs

Lab #	Main Topic	Page #
Lab 01	Introduction to MS Word	**3**
Lab 02	Introduction to MS PowerPoint	**43**
Lab 03	Introduction to MS Excel	**51**
Lab 04	Introduction to MS Access	**68**
Lab 05	Intoduction to HTML	**77**
Lab 06	Problem solving with Sequential Structure	**84**
Lab 07	Problem solving with Decision Structure	**96**
Lab 08	Problem solving with Repitition Structure	
Lab 09	**Mid Term Exam**	**102**
Lab 10	Problem solving using Functions	**109**
Lab 11	Elementary Programming in Python.	**113**
Lab 12	Conditional Statements.	**117**
Lab 13	Loops, and Nested Loops.	**121**
Lab 14	Strings.	**125**
Lab 15	Functions.	**129**
Final Term Exam		

`

Lab 01

Introduction to MS Word

Objective:

This lab will provide a hands-on experience of MS Word. Word is used for documentation. MS word is helpful in ceartion of official docuemts, letters, and other similar materials.

Activity Outcomes:

The lab will teach students to prepare different text documents.

The students will be able to:

- Create a formal text document with different formatting
- Use different merging tools
- Inserting tables and images in text documents.

Instructor Note:

As a lab activity, read “MicroSoft ”official site for guilines.

`

1) Useful concepts:

Microsoft Word is very helpful tool to create a wide variety of professional documents quickly and easily. This combination of ease of use and robust features makes it the go-to word processor in both homes and offices today. It's now also available for the Mac operating system as well as a web-based version through an Office 365 subscription.
You can also find templates to help you create the following:

- letter
- report or paper
- proposal
- newsletter
- brochure
- catalog
- poster
- flyer
- postcard
- sign
- banner
- resume
- business card
- invoice
- receipt
- product packaging
- mailing label

2) Solved Lab Activites

Sr.No	***Allocated Time***	***Level of Complexity***	***CLO Mapping***
1	***15***	***Low***	***CLO-6***
2	***15***	***Low***	***CLO-6***
3	***15***	***Low***	***CLO-6***
4	***15***	***Low***	***CLO-6***

Activity 1:

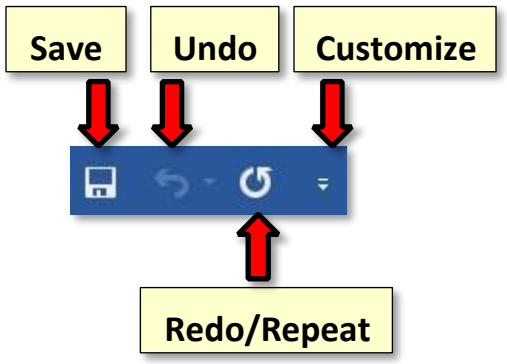
Exploring MS Word document

Solution:

Title Bar

1. **Note** the title bar section which has **window controls** at the right end, as in other Windows programs.
2. **Note** that a blank document opens with a default file name of **Document 1**.

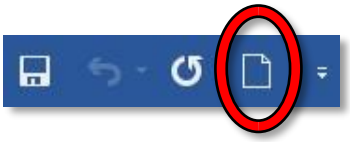
`



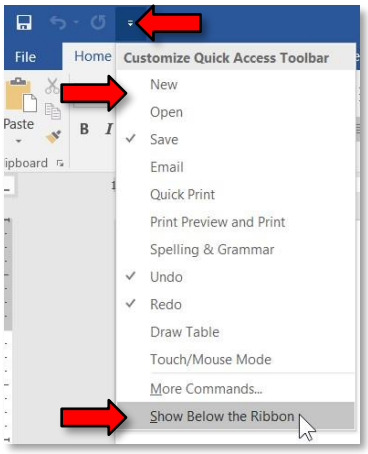
Quick Access Toolbar

The Quick Access Toolbar is located all the way to the left on the title bar. It contains frequently used commands and can be customized using the drop-down menu.

1. **Point** to each small icon to view its ScreenTip.
2. **Be aware** that the **Undo** button is not located anywhere else in the application except for the Quick Access Toolbar.
3. **Click** the **Customize Quick Access Toolbar** button, **click New** on the menu, and **see** the command get added to the Quick Access Toolbar.



4. **Click** the **Customize Quick Access Toolbar** button again, and **click** ***Show Below the Ribbon. Click Show Above the Ribbon*** *to move the Quick Access Toolbar back again.*



Ribbon

The ribbon contains all of the tools that you use to interact with your Microsoft Word file. It is located towards the top of the window underneath the title bar. All of the programs in the Microsoft Office suite have one.

The ribbon has a number of **tabs**, each of which contains **buttons**, which are organized into **groups**. Depending on the object you have selected in the document, several **contextual tabs** may appear, which provide additional formatting options for the selected object.

Try **clicking** on other **tabs** to view their buttons (do not click the File tab yet), and then **return** to Home tab.

Active Tab

By default, Word will open with the **Home tab** active on the ribbon. **Note** how the Active tab has a white background and blue letters, and the Inactive tabs have the opposite.

Contextual Tab

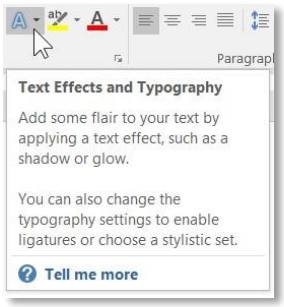
Contextual tabs are displayed when certain objects, such as an images and text boxes, are selected. They contain additional options for modifying the object. Contextual tabs stand out because they are darker in color and are located to the right of all the other tabs. As soon as we start being productive in the program, we will see contextual tabs appear.

`

Groups and Buttons

On each **Tab**, the **Buttons** (a.k.a. commands or tools) are organized into **Groups**. The groups have names, but the names are not clickable.

Hover over some of the buttons on the Home tab to **observe** the **ScreenTips**. The ScreenTip displays the name of the button, along with a short description of what the button does.



Buttons with Arrows

Note that some buttons have images on them and some have images *and an arrow*. The arrow indicates that more information is needed to carry out the function of the button. Some arrowed buttons have two parts: the button proper and the list arrow.

- A **one-part arrowed button**, called a **menu button**, will darken completely when you point to it:

  1. In the **Font group**, **point** to the **Text Effects and Typography** button.

  2. **Note** there is no difference in shading between the left and right of the button when you point to each section.

- On a **two-part arrowed button**, called a **split button**, only one section at a time will darken when you point to it.

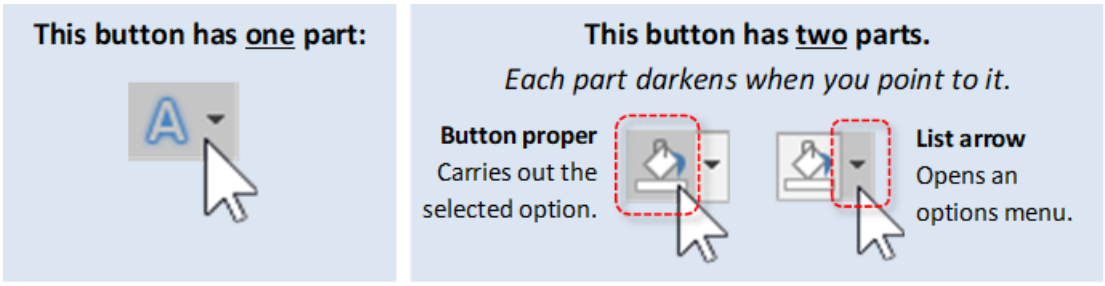
  1. In the **Paragraph group**, **point** to the left part of the **Shading** button. This is the "**button proper**" section of the button. **Note** how it is darkened separately from the arrow portion of the button.

  2. **Point** to the **right portion**, the section with the arrow. This is the "**list arrow**" section of the button. **Note** how it is darkened separately from the left portion.

  3. The **button proper** is the section of a two-part button that will carry out the default option or the last used option.

  4. The **list arrow** section will open an options menu.

`



Dialogue Box Launcher

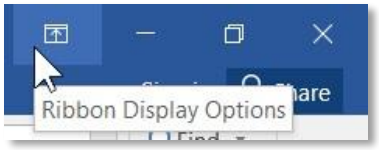
On some groups there is a **launcher** icon which will open a **dialogue box** or a **side panel** with related but less common commands.

Click any Dialogue Box Launcher icon, and then **close** the dialogue box or side panel.

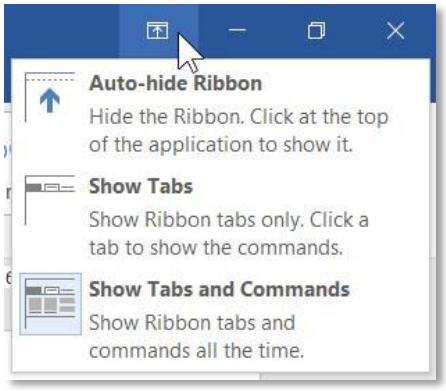
Ribbon Display Options button

This button provides options that will hide the Ribbon from view. The main benefit to this is that it allows your document to take up more of the screen.

1. **Locate** the **Ribbon Display Options** button (to the left of the window control buttons).



2. **Click** on it. Three options appear.



3. **Click Auto-hide Ribbon**. This option essentially makes Word go into "full screen" mode. It hides not only the ribbon, but also the Quick Access Toolbar, title bar, and Window Controls.

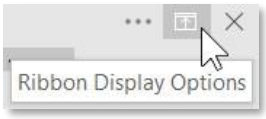
4. To get the ribbon to **show** after Auto-hiding it:

    a. **Point** to the **top-center** of the screen and **click**. (Clicking the three dots does the same thing.) The full ribbon can be seen and used. However, as as soon as the body of the document is clicked it will hide again.

`

b. **Click** in the middle of the document. **Notice** how the ribbon **hides** again.

5. To get a partial display of the ribbon to stay in view:
   a. **Click** the "mini" **Ribbon Display Options** button on the top right.



b. **Click Show Tabs**. **Note** this option has brought back our Quick Access Toolbar, title bar, Window Controls, and *part* of the ribbon; only the **Tabs** are visible. The buttons are not.

c. **Click** the **Home** tab. **Notice** how the buttons come into view.

d. **Click** in the middle of the document. **Notice** how the buttons disappear again.

> **Note:** A shortcut for changing to the "Show Tabs" view is to **double-click** the Active Tab. If the buttons in the ribbon suddenly disappear, then you may have done this by accident.

6. To get the entire ribbon to stay in view:
   a. **Click Ribbon Display Options**

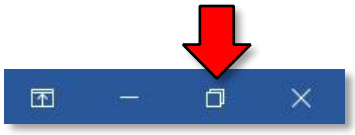
   b. **Click Show Tabs and Commands**. This option keeps **entire** ribbon visible at all times. It is the **default** option. We will keep this option selected for the remainder of class.

Dynamic Resizing

If you use Word on other computers, be aware that the button placement on the ribbon might look **slightly different**. For instance, a button might be a different size or be positioned in a slightly different place. The reason for this is that the ribbon auto-adjusts itself based on the size of the Word window.

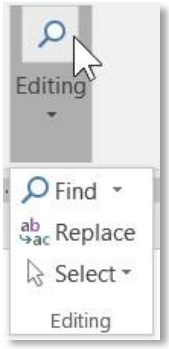
1. On the **Home** tab **notice** what the buttons in the **Editing** group currently look like.

2. **Click Restore Down** to shrink the size of the Word window.



`

3. **Notice** how the group looks different now. The entire group was collapsed into a **single button**. Click on the button to reveal the contents of the group.



4. **Click Maximize** to bring the window back to full screen.

File Tab

The **File** tab provides a **Backstage** view of your document. The Backstage view exposes information and metadata about the currently active document, lists recently opened documents, and provides a variety of user options, such as opening, saving, and printing. Instead of just a menu, it is a full-page view, which makes it easier to work with.

1. **Click** on the **File** tab.

2. **Notice** that the ribbon and the document are no longer in view. **Note** the commands, listed on the left side of the screen, are ones you would use to perform actions **TO** a document rather than **IN** a document.

3. Other things you can do in the **Backstage** view:

    a. **Click** the **Info** tab. The **Info** section of the **Backstage** view offers an easy to use interface for inspecting documents for hidden properties or personal information.

    b. **Click** the **New** tab. In this section you can create a new Blank document, or choose from a large selection of Templates.

    c. **Click** the **Open** tab. The **Open** section is used to open existing files on your computer.

        i. It immediately presents you with a list of documents that you have **recently** opened, so you can quickly find and open them again. (This is disabled in the computer lab.)

        ii. Clicking **Browse** opens a **File Explorer** dialogue, which allows you to find the file on your computer. We will be using this option in class.

    d. **Click** the **Save As** tab. This section allows you to save your file.

4. To return to the document from the Backstage view, **click** the large, left pointing arrow in the top-left corner of the screen.

`

Workspace
Underneath the ribbon is the workspace.

1. **Note** the **rulers** and **margin** settings.
2. **Note** the **scroll bar** on the right side of the screen.
    a. If the scroll bar is not visible, **move** the mouse and it will come into view.
3. **Note** the **blinking cursor/insertion point,** which is where **new input will display** when entered.
    a. If the insertion point is not blinking, **move** the mouse and it will start blinking.
4. **Point** somewhere on the blank page and **note** the mouse cursor with the **I-beam shape**, appropriate for a text environment.

Status Bar
The Status bar is located below the document window area.

Current Information
The **left end** displays a variety of information about the document, such as the page number, how many total words are in the document, and whether there are any spelling errors.

Views
At the **right end** are shortcuts to the different **views** that are available. Each view displays the document in a different way, allowing you to carry out various tasks more efficiently.

Read Mode	Displays the document full-screen, making it easier to read. You cannot edit the document in this view.
Print Layout	Shows what the document looks like when it's printed. This is overall the best view for editing documents. It is selected by default.
Web Layout	Shows what the document would look like if it were saved as a webpage.

Zoom Slider
Also at the **right end** of the Status bar is the **Zoom Slider**. This allows you to adjust how large the document is displayed on the screen. It does not adjust the actual size of the document—just how big or small it is displayed on the screen (like moving a newspaper away from or closer to your eyes).

Customization

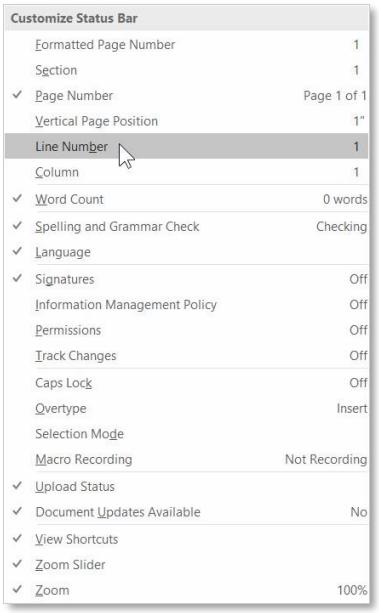
The **Status** bar can be customized.

1. **Right-click** on the **Status** bar to bring up the customize menu. Options that are enabled have a checkmark next to them.
2. **Click** on "Line Number" to enable this option.

`

3. **Notice** how the menu didn't disappear. **Click** in a clear space to dismiss the menu.

4. **Notice** how "Line: 1" appears in the **Status** bar.



`

Creating a document

1. When Word opens, it will display a blank document ready for you to type in. The words that you type and the formatting that you use become your document.

2. **Type** "My first document".

3. Each document you create is temporary unless you save it as a **file** with a unique name and location.

Preparing a Save to Location – a USB Device

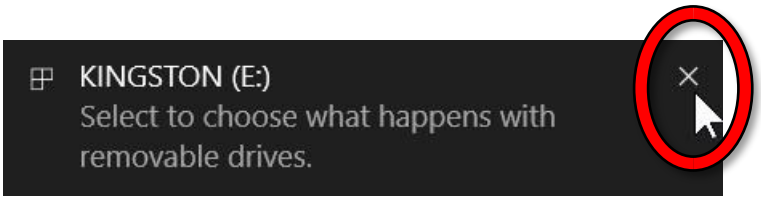
> **Note:** Home students can skip this section.

When we save a Word document, all the data in that document is collected and saved as a **file**. Normally files are saved on a computer's hard drive, but due to security restrictions on computer lab machines, files must be saved on removable storage devices.

2. **Fit** the connector into the port and **push** it in gently.

3. At this point, you ***may*** get a notice that the computer is installing a device driver – **wait** until the message disappears.

4. A notification may appear in the bottom-right corner of the screen, asking what you want to do with the flash drive. **Close** it by **pointing** to it and clicking its **Close** button.



You are now ready to begin saving your file.

Saving the File

1. **Click** the **File** tab.

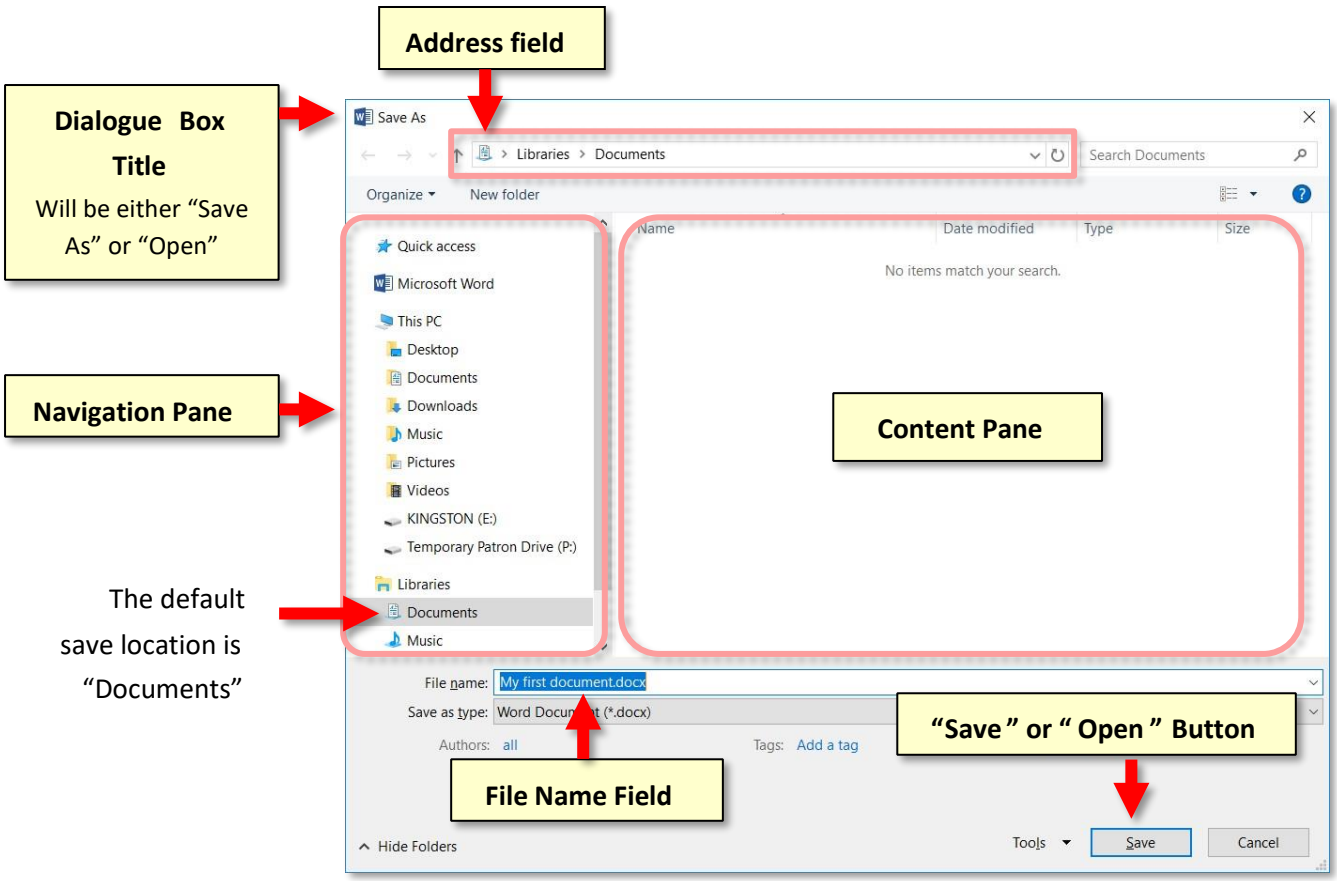
2. **Click Save As**. We use "Save As" instead of "Save" the **first** time we save a file because we need to tell the computer *where* to *put* the file (the file doesn't have a "home" yet).

`

"Save" assumes you've saved it before.

3. **Click Browse**.

4. **Notice** that a smaller window appears in front of our work. This small window is called a **dialogue box**. Because the computer needs to know more than just "OK, save," the dialogue box is where we tell it *how* we want to save our work.



5. When it comes to saving, there are two important things to *identify* for the computer:

    1. The **location** where the file is going to be saved to.
    2. What **name** you want to give the file.

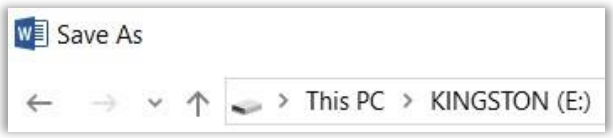
6. The **location** where it will be saved is displayed for us in the **Address field**. In this case, **note** that the **Documents directory** is the **default** save location, but we want to save our file to the **flash drive**.

7. **Notice** other available folders and devices can be seen in the left pane, called the **Navigation pane**. If we wanted to save to one of these alternate locations, we would have to click on it.

8. **Find** the location labeled **KINGSTON (E:)** and **click** it. Kingston is the name of the company that created our flash drive.

    Note: If you are taking this class from home and do not have a flash drive, use "Documents" as the location to save your files.

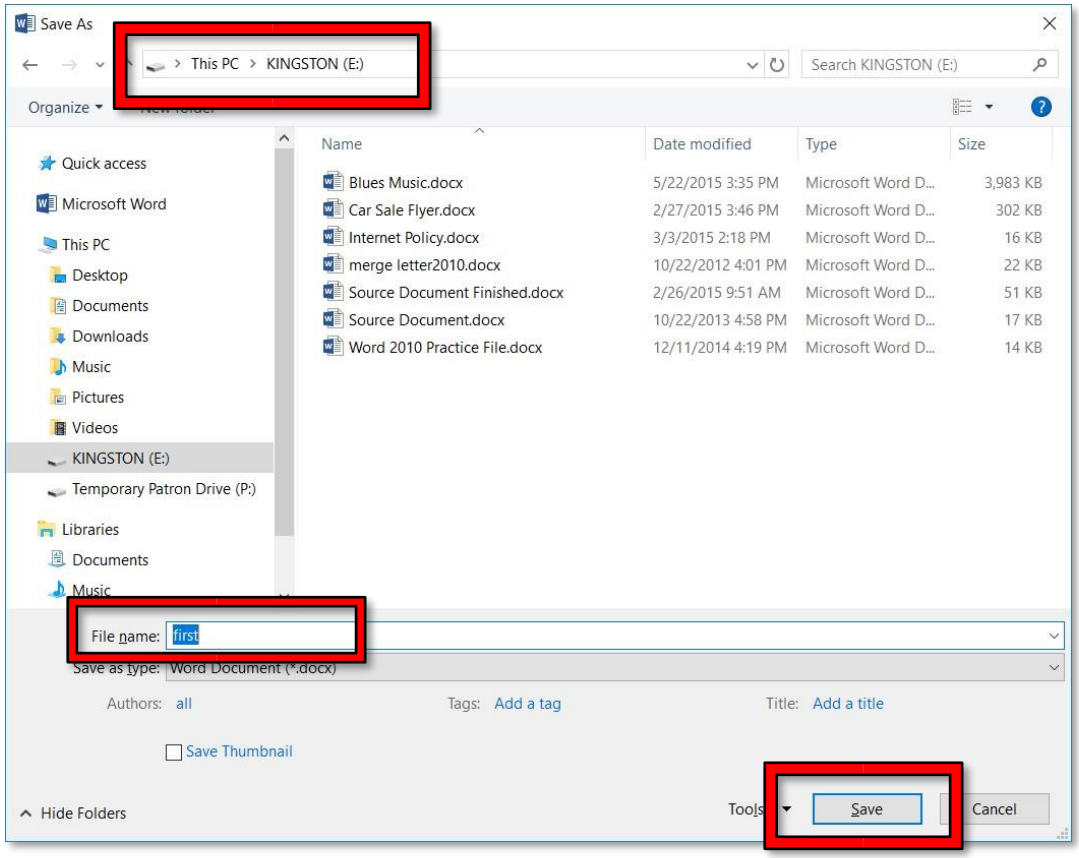
9. Your address field should now read **This PC > KINGSTON (E:)**.



10. Now we need to name our file. **Notice** that the file name field is towards the bottom of the dialogue box. By default, Word names the file after the first few words that were typed into the document.

11. **Click** into this box and the words will be highlighted. Then **type** the word **first** to name your file
'first'.

12. Once we have given the computer **a file name** and a **save location**, we are ready to save. At this point, your Save As dialogue box should look like the image below. To save, you will **click Save**.



13. Your Word window will still be open but **notice** the **title** bar will now show the file name **first.docx**.

`

Activity 2:

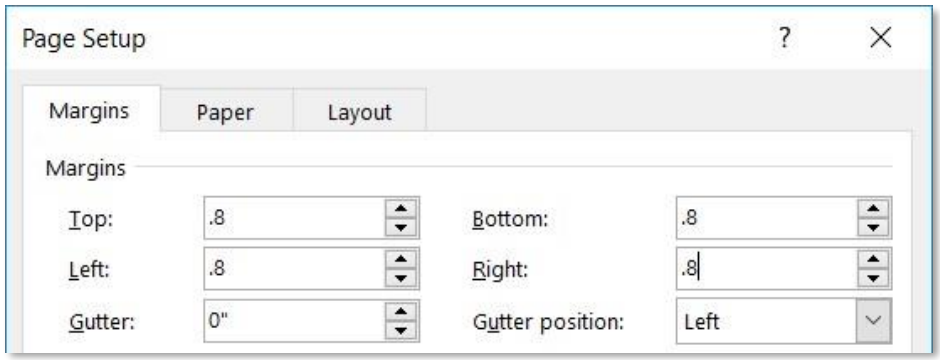
Controlling the Appearance of your Document

Solution:

Changing Page Margins

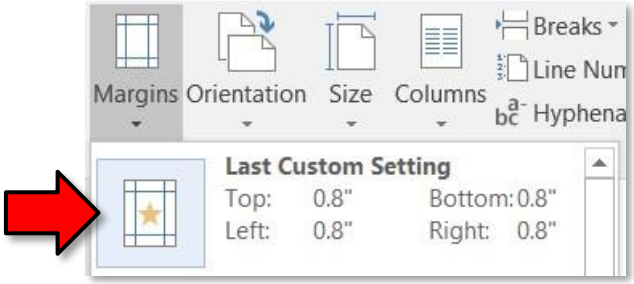
1. **Click** the **Layout** tab to access tools to change the appearance of your document.
2. In the **Page Setup** group**, click Margins**. A list will appear that will have your current settings highlighted. **Click Wide** to *see* how it will affect your document.
3. **Click Margins** again and **click Custom Margins** at the bottom of the list**.**

    a. When the **Page Setup dialogue box** opens, on the **Margins** tab, in the **Margins** section **click** the arrows to **change** the top, left, bottom and right margins to **0.8"**.



    b. The **Gutter** setting is an extra margin that is only used if you want to **bind** your printed pages together in some way (such as with a three-hole-punch). **Leave** this at **0"**. c. **Click OK**.

4. In the **Page Setup** group, **click Margins** again and **notice** how the margin list has now populated with your customization.



Page Breaks

1. **Place** your insertion point at the end of the blue paragraph.
2. On the **Insert** tab, **find** the **Pages** group and **click Page Break**.
3. **Notice** how the lines below are now on the next page.
4. **Notice** how Word has inserted some blank space at the top of the next page.
5. **Tap** the Delete **key** to remove this extra space.

`

Good stopping point for Session 1.
Close "Internet Policy.docx" and **save the changes**.

Describe how the Exercise is organized: The parts in parentheses are **hints** concerning how to complete each step.

Headers and Footers

A **header** is text that appears at the **top** of every page in your document. Similarly, a **footer** is text that appears at the **bottom** of every page.

1. First, let's insert a **header**.

    a. **Open** Internet Policy.docx.

    b. **Tap** Ctrl + Home to get to the top of the document.

    c. **Click** the **Insert** tab. In the **Header & Footer** group, **click Header** to open a list of different header options.

    d. **Scroll** down the menu to view all the options and **click Blank.**

    e. **Notice** that a new **contextual ribbon** has opened called **Header & Footer Tools.** It has one tab - **Design**.

    f. **Notice** how Word is calling out the header section with a **dotted-line**.

    g. **Notice** the words **"Type here"** enclosed in brackets on the left. This is a **placeholder** for an area of the header into which we can enter content. It is **colored gray**, which means it is already selected and ready for us to populate it with content.

    h. **Type** "Internet Policy".

2. **Close** header. This can be done in two ways:

    a. On the **Header & Footer Tools** contextual ribbon, on the **Design** tab, **click Close Header and Footer**.

    b. By **double-clicking** anywhere within the body of the document. (We'll try this in a minute)

3. **Scroll down** and **notice** that "Internet Policy" appears at the top of every page.

4. Also **notice** that the font color of the text in the header is light gray. This is not the **actual** font color. Microsoft Word makes the header text display in light grey to show that the header is **not currently active**.

5. **Double-click** on the header to make it active. **Notice** how the font color has changed to its real color (black) and the document body text is now dimmed. Again, this is to show that the header/footer is **active**, and the document body is **not active**.

6. Next, let's insert a **footer**.

`

a. **Note** that, when the header is active, the footer is active as well. **Scroll down** to the bottom of the current page and notice that there is a **Footer** section called-out with a dotted-line.

b. On the **Header & Footer Tools** contextual ribbon, on the **Design** tab, locate the **Header & Footer** group. **Click Footer** to open a list of different footer options.

c. Again, **scroll down** the list to view all the options and then **click Blank (Three Columns).**

d. What we're going to do is, put our **name** in the left placeholder, the **current date** in the center placeholder, and the **page number** in the right placeholder. There are tools on the Header & Footer Tools contextual ribbon to facilitate this.

e. **Click** on the **left** placeholder to select it and **type** your name. *Do not tap Enter.*

f. Let's make our name bold. How would we do this? Because there is no Bold button visible, we have to switch to another ribbon. **Click** the **Home** tab, **locate** the **Font** group and **click**
Bold. (no need to highlight the name)

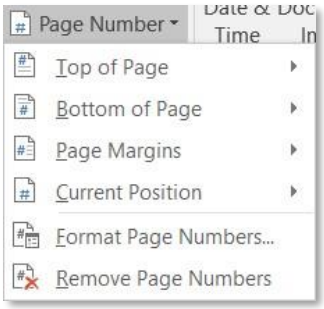
g. **Note** how our Header & Footer Tools contextual ribbon is no longer active since we switched to the Home tab. To bring the Header & Footer Tools contextual ribbon back, **click** on its **Design** tab.

h. **Click** on the middle placeholder in the footer to select it. On the **Header & Footer Tools** ribbon, **locate** the **Insert** group and **click Date & Time.** When the dialogue box opens, **click** any date format you wish under the **Available Formats** in the left pane.

i. **Note** the empty checkbox that says "Update automatically". This would need to be checked if you want the inserted date to change to the current date every time you open this document.

ii. **Click OK**.

i. **Click** on the **right** placeholder. On the **Header & Footer Tools** ribbon**, locate** the **Header & Footer** group and **click Page Number**. A list of options will be shown about where you want to insert the page numbers (see table below).



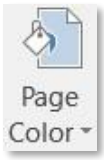
Top of Page	Puts the page number in the header. **Warning**: This will replace your entire header with a new header!
	Warning: This will replace your entire footer with a new footer!
Page Margins	Puts the page number in the left or right margins.
Bottom of Page	Puts the page number in the footer.
Current Position	Puts the page number wherever your insertion point is.

j. **Move** your pointer to **Current Position** and a list of options will open. **Scroll down** the list to the "**Page X of Y**" section and **click Bold Numbers**.

k. **Double-click** in the body of the document to close the Header and Footer Tools.

Adding Visual Interest

1. Changing the **Page Background**:



   a. On the **Design** tab, in the **Page Background** group, **click Page Color** to display a palette of colors.

   b. **Mouse-over** the colors and **observe Live Preview** changes to your document.

   c. **Click** a color that is **fairly dark** (fourth row of Theme Colors).

Tip: The document's text color automatically changes to white when a dark background color is select.

2. Adding a **Watermark:**

   a. On the **Design** tab, in the **Page Background** group, **click Watermark** to see a list of semi-transparent messages that can be added to your document. **Click** on one of the messages and **note** it's insertion into the document. These messages will be printed should you print the document.

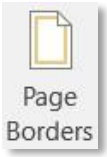
   b. You can also **customize** the watermark text. **Click Watermark** again and **click** "Custom Watermark". In the Printed Watermark dialogue box, find the Text field, **click** into it, **delete** the existing text, and **type** some different text. **Click** OK.

`

What's the difference between the "Apply" and "OK" buttons?

- **Apply** will commit your changes and **keep** the dialogue box **open**.
- **OK** will commit your changes and **close** the dialogue box.

3. Adding a **Page Border**:

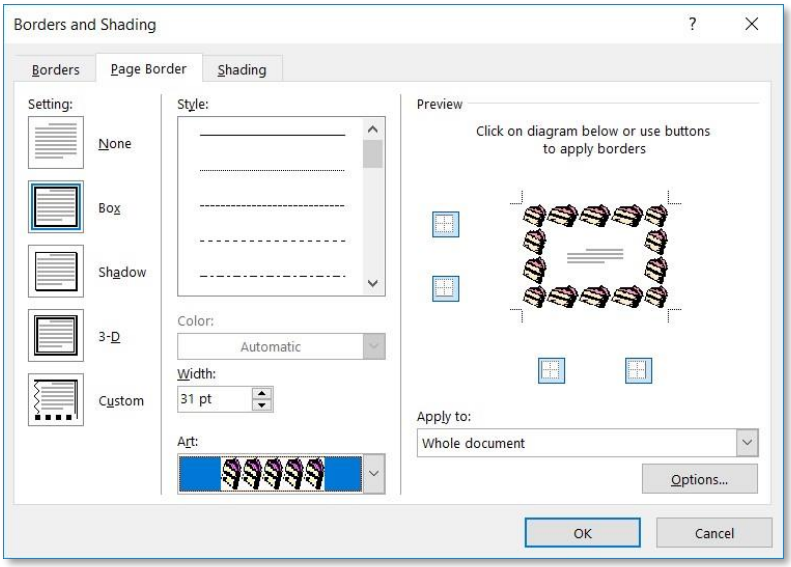


   a. To place a border around your document, on the **Design** tab, in the **Page Background** group, **click Page Borders.** A **Borders and Shading Dialogue Box** will open.

   b. In the **Borders and Shading Dialogue Box,** on the **Page Border tab,** there are options for customizing a border. As you **click** on different settings, styles, colors, etc. in the left and center panes, **note** a preview in the right pane.

   c. In the **Borders and Shading Dialogue Box,** on the **Page Border tab,** in the *left* pane, click on the **Box** setting.

   d. In the **Borders and Shading Dialogue Box,** on the **Page Border tab,** in the *center* pane, in the **Art** drop-down list box**, click** the drop-down arrow. **Scroll down** and **click** a border style that you like. **Click** OK to add the border.



   e. **Note** that, depending on how large the border is, it may **cover up** your **header and/or footer**. To fix this, you can adjust the distance between the edge of the page and the header/footer.

      i. **Double-click** on the **header** to make it active.

ii. On the **Header & Footer Tools** contextual ribbon, on the **Design** tab, in the **Position** group, there are two text boxes:

1. The **top** one controls the distance between the **header** and the edge of the page.

2. The **bottom** one controls the distance between the **footer** and the edge of the page.

iii. **Increase** the values in both of these text boxes until you can see your header and footer. Note – you will need to click into the footer before you adjust the bottom control.

Save the document and then **close** Word.

Activity 3:

Inserting Online Pictures, Shapes, Text Boxes, and Other Pictures:

Solution:

The insertion of specialized objects into a document can add visual interest. We will explore a few different types of objects in this section and also learn how to format the objects.

Online Pictures

The former name for this type of object was Clip Art. **Clip art** are small pictures and symbols made available for computer users to add to their documents. They can be used to enhance a narrative.

Inserting Online Pictures

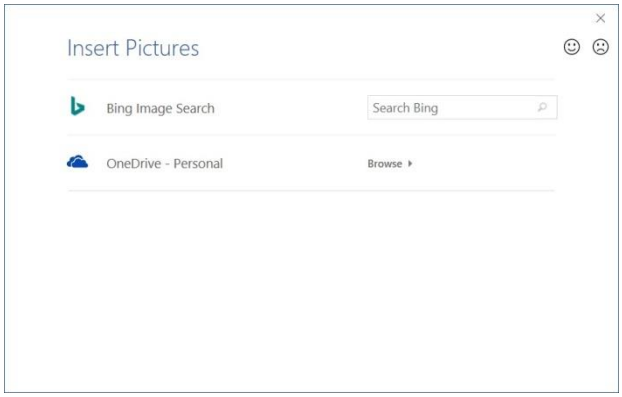
1. **Click** in a clear area underneath the table to move your insertion point off of the table.

Teacher's note:

If the student's table is too far down the page, the cursor will get stuck above the table. If this happens, the student can **double-click** below the table to place their insertion point there.

2. On the **Insert** tab, in the **Illustrations** group, **click Online Pictures**.

3. A dialogue box opens that prompts us to enter a search term.



`

4. **Note** the words **Bing Image Search**. Microsoft Word will search the Internet for images using its search engine, Bing (it is a competitor to Google).

5. **Type helmet** in the search field and tap . Enter

6. **Note** the gray checkbox labeled "Creative Commons only". Word has filtered the search results to only show images that are licensed under **Creative Commons** licenses.

    a. **What is Creative Commons?** It is a type of copyright license that is frequently used on the internet. In general, it is a liberal license that gives you permission to use the image for free.

    b. **HOWEVER**, Creative Commons allows content creators to add "gotchas" such as "cannot use for commercial purposes" or "must provide attribution to the original author". **MICROSOFT WORD DOES NOT SHOW YOU THESE.** So, at the end of the day, it is up to **you** to verify that you are abiding by the author's **SPECIFIC** license terms—and finding the specific license terms is not always an easy task.

    c. For this reason, if you are planning to use clip art for any professional purpose, we recommend using a clip art website that contains only **public use** images. One such website is **pixabay.com**. Public use images give you **complete control** over how you may use them. *Later in the lesson we will learn how to insert a picture from pixabay.com into a Word document.*

    d. Another alternative is to **purchase** clipart (also called "stock images"). This is how businesses typically obtain clip art.

7. **Find** a picture you like.

8. **Point** to the image.

9. **Click** on the **three dots** in the bottom-right corner of the image. A screen tip appears above the image containing additional information about the image, including its pixel dimensions and Internet URL.

`

10. **Click** in a clear space to dismi ss the screen tip.

11. Now, we will insert the image into our document.

a. **Click** on the picture to **select** it.

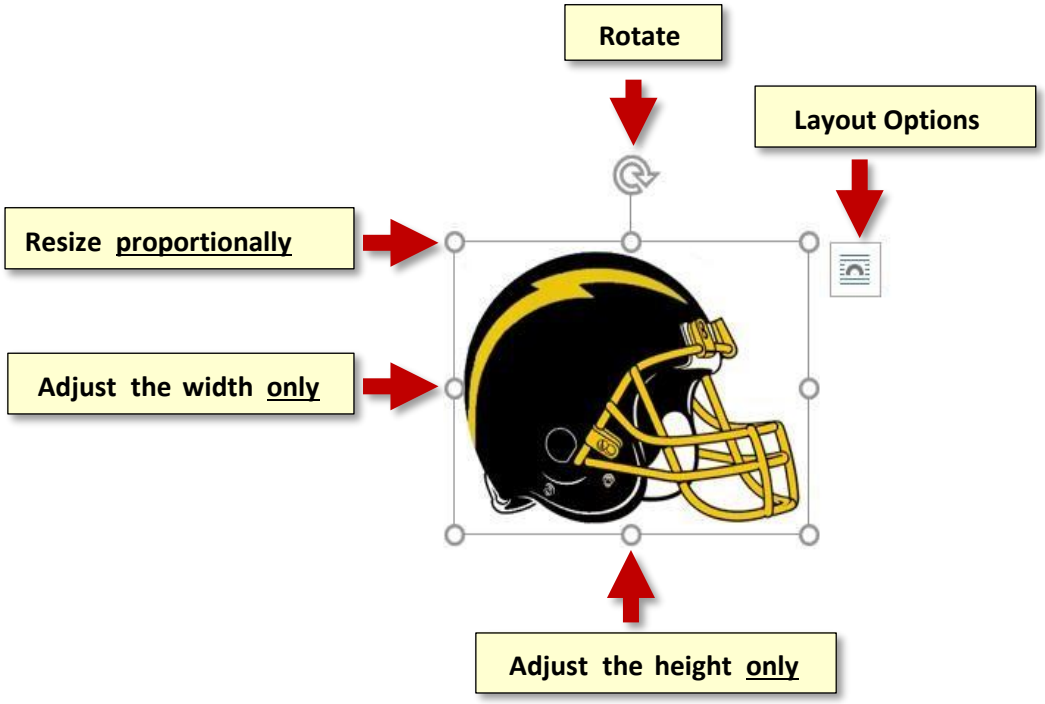
b. **Notice** the checkmark that appears in the **top-right** corner of the picture.

c. **Notice** how the Insert button indicates that **one picture** is selected. d. **Click Insert (1)**.

Resizing Inserted Objects

Objects can be resized by using "**handles**" that appear around a selected object.

`



1. If you cannot already see a border and small circles around your picture, **click** on the picture to select it.

2. **Point** your mouse to one of the circles. These circles are called **resizing handles**. **Notice** the pointer shape changes to a **white arrow with two ends**. As we learned before, this is a **resizing cursor**.

3. Using one of the **corner circles**, **click** and **drag** towards the center of the picture and **note** it resizes the picture **proportionately**. This does not always work with other types of objects. Other objects require you to Shift hold down while resizing to retain the original proportions.

> **Warning**: Increasing the size of an online picture can result in a **distorted look** when printed. To avoid this, try not to make an online picture any larger than it was when you originally inserted it into the document.

4. **Click** and **drag** from one of the **side circles** and **note** the change in shape of the picture. These circles do **not** resize the picture proportionally. **Click Undo**.

5. **Click** and **drag** the **circular arrow icon** above the picture **note** how the picture **rotates**. **Click Undo** to get it back to its original rotation.

6. **Using** the **rulers** on the top and left edges of document as a reference, **resize** the object proportionally to approximately 1 ½" square.

7. **Deselect** the object by clicking in a **blank area**.

Applying Wrapping Styles

"Objects can be placed in your document in two ways: either inline or floating. Inline objects are those that reside on the same layer as your text and are positioned within the stream of text that surrounds the object. Floating objects are those that are placed on a layer over the text". –Allen Wyatt

This positioning of objects is called the **text wrapping style.** The wrapping style can affect how difficult it is to move an object on the page. If it seems difficult to move an object where you want to, then you may need to change the wrapping style:

1. **Select** the picture object and **move** your pointer around on it until you see a **Move** cursor shape.

`

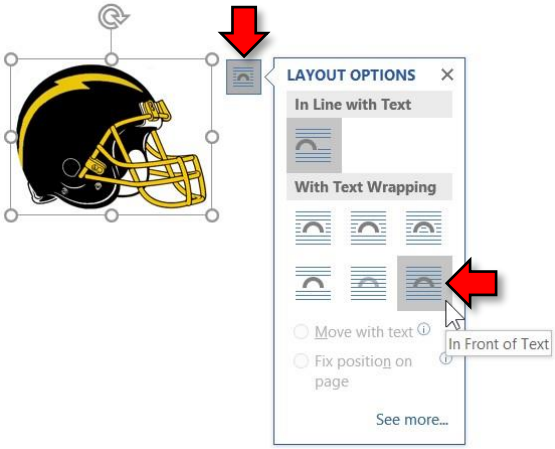
2. When you see this cursor, **click** and **drag**. You should **notice** that it is hard to move the object. We need **to change the wrapping style** so it is easier to move. The default wrapping style for pictures is **In Line with Text**, which means that only text can move it around. This can be very restricting.

3. **Click Layout Options** to the **right** of the selected object.

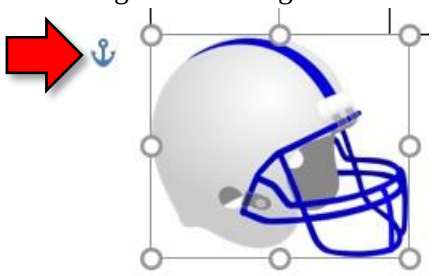
    a. It presents you with a set of icons, each of which represent a **wrapping style**.

    b. You can get an idea of what each wrapping style does by looking at the icons. The blue, horizontal lines represent your document's text, and the gray arch represents the image.

    c. **Point** to the icons to see a **ScreenTip** containing their names. **Click In Front of Text**.



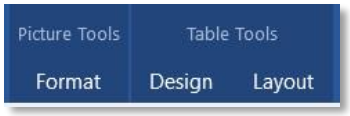
4. **Notice** the small "anchor" symbol that has appeared after we changed the wrapping style to a "floating" style. This is called an **object anchor** and it indicates where a floating object is located in relation to the text in your document. If we were to insert multiple lines of text somewhere in our document above the anchor, it would cause our object to get "bumped" down, even though it is floating.



5. **Using** the **move cursor**, **move** your object slightly to the left. You should **see** a **green line** appear. This green line helps you to align your object against the left margin of the document.

6. **Move** the object so it is on top of the last column in your table.

7. **Notice** how there are now **two** contextual ribbons: **Table Tools** and **Picture Tools**. This is because our picture object is selected *and* it is on top of the table.



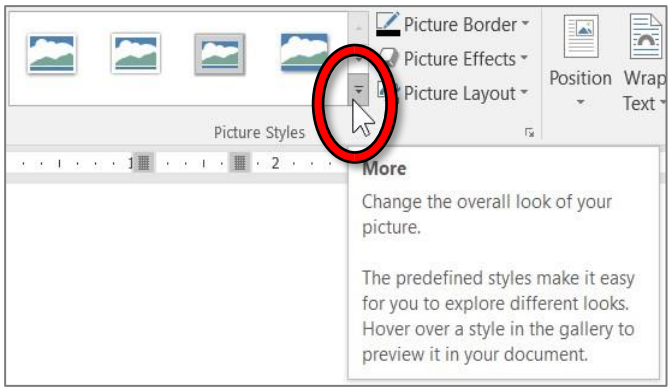
    a. **Picture Tools** has one tab: **Format**

    b. **Table Tools** has two tabs: **Design** and **Layout**

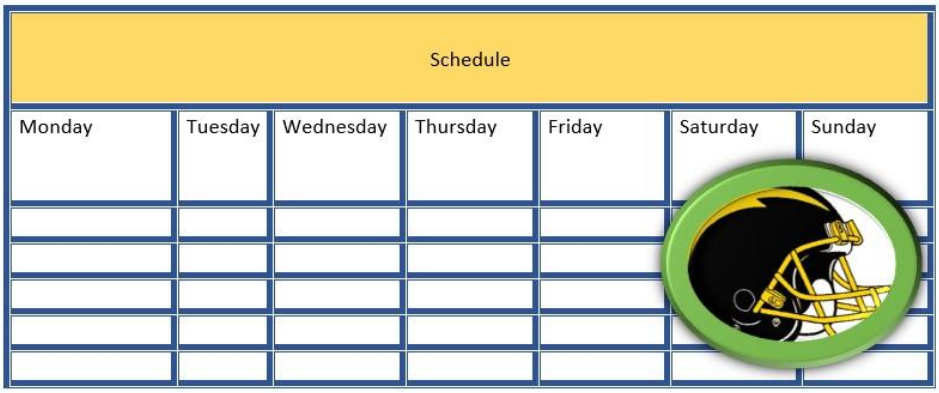
`

Applying Formatting to Pictures

1. **Click** the picture to select it if it is not already selected.
2. On the **Picture Tools** contextual ribbon, **click** the **Format** tab and, in the **Picture Styles** group, **move** your pointer over the predefined **Picture Styles** thumbnails to see a **Live Preview** of their effects.
3. **Click** the **More** button to see more predefined styles. As you point to the different styles **note** the ScreenTips that appear which contain the name of the style. **Click** on the **Metal Oval** style.



4. In the **Picture Styles** group, **click** on **Picture Border** and **click** on any color that you like.
5. **Deselect** the picture.



Shapes

A shape is another type of object that can be inserted into a Word document.

*Let's insert an **arrow shape** into our document.*

1. On the **Insert** tab, in the **Illustrations** group, **click Shapes**. A menu of shapes will open. The shapes are organized by type of shape.
2. In the **Line** section, **mouse** over the line shapes until you see a ScreenTip that says **Double Arrow** . **Click** the shape.
3. **Find** your mouse cursor in the document and **note** that it is shaped like a crosshair +.
4. We are going to "**draw"** an arrow from our **helmet picture** to the word "**Monday**" in our table.

   a. **Point** to the helmet picture.

`

b. **Click** and **drag** to the word Monday.

c. **Let go** of the mouse button.

5. **Note** the arrow is **selected**. You can tell by the **resize handles** at the ends.

6. With the arrow still selected, **note** the **Drawing Tools** contextual ribbon.  It has one tab: **Format**. **Click** the Format tab.

7. In the **Shape Styles** group, **click Shape Outline**. **Point** to **Weight** and, on the sub-menu, **click 6 pt**.

8. **Click Shape Outline** again. **Mouse over** the colors to see a live preview on your arrow. **Click** a color to select it.

9. **Point** your mouse at the **body** of the arrow until you see a **Move cursor**. **Click** and **drag** to move the arrow to another place.

10. **Deselect** your arrow.

11. **Insert** your cursor beneath the table.

Tip: To change the **default line styling** that is used when you create a new line, **right-click** on the line whose style you want to make the default and **click** "**Set as Default Line**".

*Next, we will insert a **star shape**.*

1. On the **Insert** tab, in the **Illustrations** group, **click Shapes**.

2. In the **Stars and Banners** section, **mouse over** the shapes until you see **5-point Star** ☆ and **click** it.

3. **Find** your cursor in the document and **note** that it is shaped like a crosshair +.

4. **Click** next to the insertion point and **drag** diagonally down and to the right. Don't let go of the mouse until the star until it is about 3" square (**use** the document rulers as a guide). **Let go** of the mouse button when you finish dragging.

5. **Remember** that as long as you don't let go of the mouse button, you have **full control** over the size and shape of the drawing.

Tip: Using the corner resize handles on a shape will **NOT** resize the shape proportionally, like with clipart. To resize a shape proportionally, you must hold down Shift while resizing.

5. **Find** the **move cursor** ☆ on the star object and **move** the star to the .**Use** right side of the document the green line to **align** it to the right margin of the document.
6. On the **Drawing Tools** contextual ribbon, on the **Format** tab, in the **Shape Styles** group, **click More** to see a gallery shape styles that could be used. **Mouse over** these Quick Styles to **see** a **live preview of the different styles**. **Scroll** down the gallery and **click** a quick style in the last row.

`

Clicking between objects may make tabs inactive

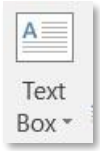
1. **Click** on the helmet object again. **Note** that your star shape object has become deselected and the Drawing Tools contextual ribbon associated with it has disappeared.
2. **Note** that **two contextual ribbons** now appear; one associated with the online picture object (Picture Tools) and one with the table (Table Tools). However, none of the tabs on either contextual ribbon are active.

3. **Click** on the **Design** tab and **note** the background color of the tab is **white**. This is how you can tell it is active.

4. **Click** on the star object again. **Note** that while the Drawing Tools contextual ribbon may appear, its **Format** tab might not be active. If you don't see the tools you expected to see, you may have to **click** on the tab to activate the tool selection.

Text Boxes

A text box is a freestanding object that can contain words. *Let's insert a text box.*



1. **Deselect** the star object.
2. **Click** on the **Insert** tab. In the **Text** group, **click Text Box**. A menu of Built-in text box styles will appear. However, to have more control over format, placement and size, we will draw our own text box. **Click Draw Text Box**.
3. **Click** into your document where you want the text box to start and **drag** diagonally and to the right to where you want it to end.
4. The insertion point within the text box indicates that what you type will be inserted there. **Type your name**.
5. **Resize** the text box to just fit around your name.
6. To move your text box, **make sure** it is selected, then **move** your pointer to the edge of the box until a **move cursor** appears . Then **click** and **drag** your text box to the **center** of the **star shape**.
7. **Note** that the text box has a **black border** and it is also **filled** with white color.
8. You can change formatting of the text box by using tools on the **Drawing Tools** contextual ribbon.
    a. With the text box selected, on the **Drawing Tools c**ontextual ribbon**, click** the **Format** tab. In the **Shape Styles** group, **click Shape Outline**, and **click No Outline.**

`

b. Next, In the **Shape Styles** group, **click Shape Fill** and **click No Fill**. **Deselect** the text box.

Other Pictures

In addition to inserting pictures via Online Pictures, you can also insert images of your own into a document. These images can be ones that you've made yourself (like photos taken with a camera) or ones that you've downloaded from the Internet. We have placed a couple of pictures on the flash drive for you to use in this section.

Inserting a Picture

1. **Open** a new blank Word document.
2. **Click** the **Insert** tab. In the **Illustrations** group, **click Pictures**.
3. In the Insert Picture dialogue box, **navigate** to the flash drive and **click Orchid.jpg**.

4. In the dialogue box, **click Insert.**

Color Effects and Artistic Effects

1. To prep for this section, let's move our picture to the right side of the page. See if you can remember how to do this. This is an important thing to know!
    a. **Change** the Text wrapping style to "In front of text".
    b. **Click** and **drag** your picture to the right side of your document.
2. On the **Picture Tools** contextual ribbon, on the **Format** tab, **find** the **Adjust** group, and **click** on **Color. Mouse over** the coloring effects that could be applied to your picture.
3. **Click** on the **title bar** to dismiss the gallery.
4. In the **Adjust** group, **click** on **Artistic Effects** and **mouse over** the artistic effects that could be applied to your picture.
5. **Click** on the **title bar** to dismiss the gallery.

Remove Background Effect

1. To prep for this section, we are going add a **dark background color** to the document.

    a. **Click** on the **Design** tab, and in the **Page Background** group, **click** on **Page Color**. b. **Click** a dark color.

    Note: By default, Word will **not print** a page's background color because of the amount of ink required.

2. On the **Picture Tools** contextual ribbon, **click** the **Format** tab and, in the **Adjust** group, **click Remove Background**. Your picture will look like the picture below. Also, a new contextual tab named **Background Removal** will open.

`

3. On the **Background Removal** tab, in the **Refine** group, **click Mark Areas to Keep**.
4. The pointer will change to the **shape of a pencil** when you point to the picture. **Use** the **tip** of the pencil to "**click away**" the bright pink sections.
5. In the **Close** group, **click Keep changes**.
6. **Deselect** your picture.

Crop Picture Effect

1. **Open** a new blank Word document.
2. **Insert** another picture from your flash drive. It is named **red-roses-photo.jpg**.

3. On the **Picture Tools** contextual ribbon, on the **Format** tab, **find** the **Size** group and **click** on the **list arrow** part of the **Crop** split button.
4. **Point** to **Crop to Shape**. In the **Basic Shapes** section of the Shapes menu, **click Heart** .

Format Painter

The Format Painter is used to **copy** the **formatting** of a piece of text or picture and **apply** it to something else. For instance, in the case of text, the formatting would be the font face, size, and color. We will use the Format Painter to apply the formatting of one object to another.

1. **Deselect** the **red roses** object which we just cropped to a heart shape (**click** the right margin).
2. **Insert** the **Orchid.jpg** picture from your flash drive again.
3. **Resize** each object **proportionally** (use the corner handles) until they appear side by side. (make them about **3" wide**).
4. **Select** the **red roses** object.
5. **Click** the **Home** tab and in the **Clipboard** group, **click Format Painter**.

`

6. **Move** the mouse pointer around the screen. **Notice** how the mouse cursor has changed to an arrow with a paintbrush next to it.

7. **Click** on the orchid object that you just inserted. **Notice** how it now has the same cropped shape as the red roses object.

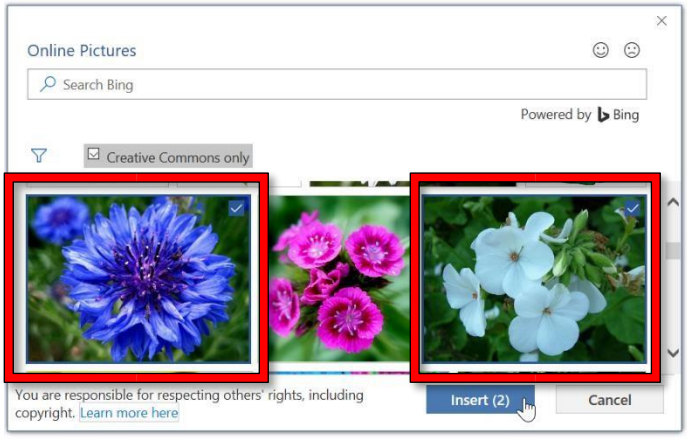
8. **Deselect** the orchid object and **note** your cursor shape has returned to an **I-beam shape**.

Double-clicking Format Painter makes it possible to apply a format to more than one object. *Let's try it.*

1. **Insert** your cursor to the right of the orchid object and tap . Enter

2. **Search** for **Online Pictures** pictures using the search term **flowers**.

3. In the search results, **locate** two pictures of **flowers** that do not have white backgrounds. Since the search dialogue allows the insertion of multiple objects at one time, **click** each of the two pictures and then **click Insert.**



4. **Resize** each image so they are each about **3" in width**.

5. **Click** one of the objects that is formatted with a heart shape.

6. **Double-click Format Painter**.

7. **Click** on one of the flower objects and **notice** it adopts the heart-shaped format.

8. **Deselect** the flower object and **notice** that the cursor does not change back to an I-beam.

9. **Click** on the **second flower object** and **notice** it adopts the heart-shaped format.

10. In order to "**turn off**" the Format Painter, **single-click** on its button in the Home ribbon.

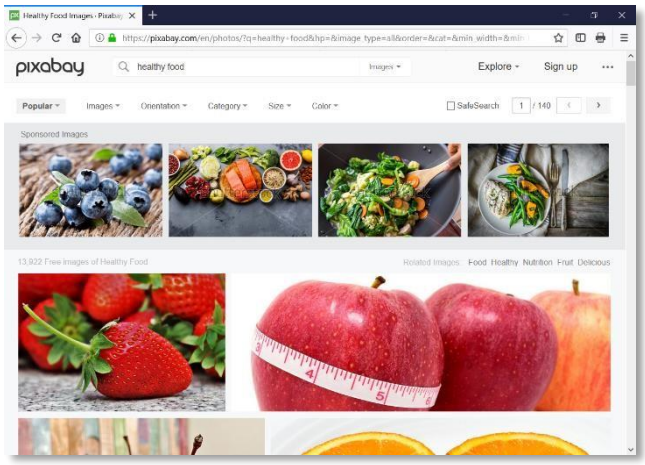
`

Inserting a Picture from a Webpage

You can also insert images from a webpage into your document. However, be aware that many images on the internet are **protected by copyright**. When you find an image you like, you should read the website's fine print to determine if you can use the image or not. This is especially important if you are making a presentation for commercial purposes (for example, as part of your job).

1. **Open** a new, blank document.
2. Using the Start Menu, **open Firefox**.
3. **Click** into the **address bar** at the top of the screen and **type www.pixabay.com**. Pixabay is a website that contains images that you can download and use for free, without restriction.

   Enter
4. **Type** "**healthy food**" into the search box and **tap** .
5. **Click** on an image you like (except for the ones in the first row—those cost money).



6. On the next screen, **notice** the **copyright notice** on the right. It says "**CC0 Creative Commons**". This means you can use the image however you want. You don't even have to give the original author credit.
7. **Right-click** in the **middle** of the image and **select Copy Image**.

Tip: If you're planning on printing the document, it's best to use a **high-resolution** version of the image. To do this, click the green "Free Download" button on the right. This will download an image file to your hard drive. Then, follow the instructions found in the "Inserting a Picture" section to insert the image file into the presentation.

8. Using the **taskbar, switch back** to **Word**.
9. In the **Home** tab, in the **Clipboard** group, **click** the **Paste button proper**.
10. **Close** Firefox.

`

Activity 4:

Using of Mail Merger:

Solution:

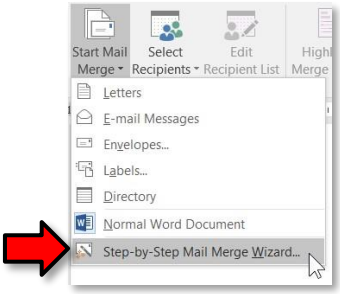
Mail merge is a feature of Word processing programs that enables you to generate form letters. Form letters are sent out en masse to people where much of the letter is the same for each recipient. What changes in the letter are the recipient's name, address, and maybe certain other items specific to each recipient.

To use a mail-merge system, you would first create a **data file** with a set of information, like a list of names and addresses. In a Word document, you would create a **sample letter**, substituting special symbols in place of names and addresses (or whatever other information will come from the first file). Through a series of small steps, you can create form letters that are personalized for each recipient.

The data file can be created with various programs such as Word, Excel, or Access. The beauty of the mail merge feature is that, while you can create your own data source in Word, if you already have a spreadsheet of data created in Excel or some other program, it makes sense to use that.

Steps to Create a Mail Merge Document

1. To save time, we have already created a data file in Excel and a sample letter in Word and placed them on your flash drive.
    a. **Insert** your **flash drive.**
    b. **Use File Explorer** to **open Donations List.xlsx.**
    c. **Use File Explorer** to **open merge letter.docx.**
2. Before we start the mail merge process, let's look at the donations list that was created in Excel.
    a. **Notice** that the **list** is on the worksheet named Sheet 1.
    b. Notice that the **first row** of the worksheet contains headings.
    c. **Close Donations List.xlsx**
3. In **merge letter.docx, notice** that the **address** of the establishment, the **body** of the letter and the **closing** are all in place. We will use mail merge to personalize each letter with an **address block**, a **salutation**, and a **donation** amount for each person in our list.
4. **Click** the **Mailings** tab.
5. **Find** the **Start Mail Merge** group. **Click Start Mail Merge** and then **click Step-by-Step Mail Merge Wizard**.



6. A **Mail Merge pane** will open on the right that will take you through the mail merge process in 6 steps.

`

7. **Step 1** - **Notice Select document type** prompt. By default, the radio button for letters is checked, which is what we want. **Click** on **Next: Starting Document** at the bottom of the Mail Merge pane.

8. **Step 2** - **Notice** the **Select starting document** prompt. Since we will be using this letter, do not change the default selection of use the current document. **Click** on **Next: Select Recipients** at the bottom of the Mail Merge pane.

9. **Step 3** - **Notice** the **Select recipients** prompt

    a. The default selection, Use an existing list is the one we need. To find the list, **click** on the Browse button.

    b. A **Select Data Source** dialogue box will open. **Navigate** to your flash drive and **open Donations List.xlsx**

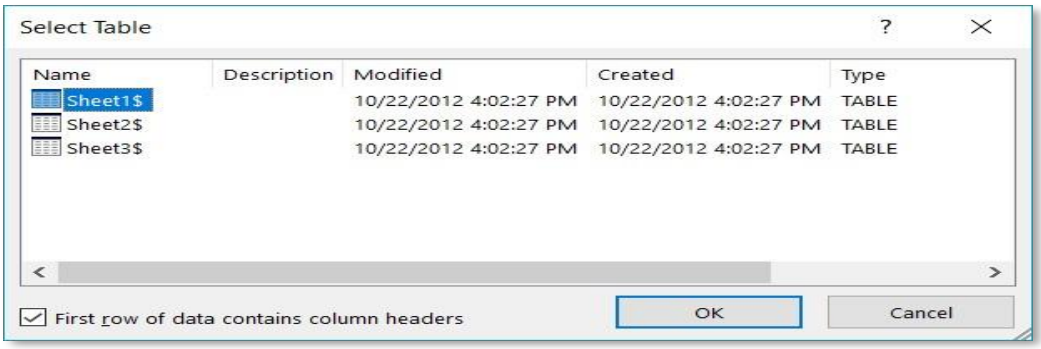
    c. A **Select Table** dialogue box will open. We need to provide some information about our document.

        i. **Notice** that by default, Sheet 1 is selected.

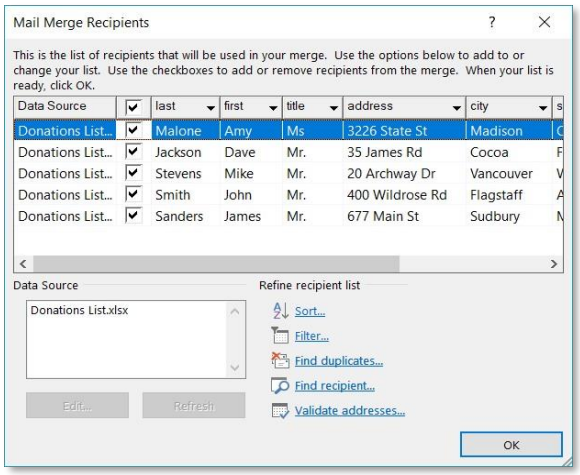
        ii. **Remember** that when we examined our Excel file our data was on the Worksheet entitled Sheet 1.

        iii. Also **be sure** that the **check box** before First row of data contains column headers is **checked**.

        iv. **Click OK**.



    d. A **Mail Merge Recipients** dialogue box will open which will allow you to review, delete or change your list. Since we are not making any changes, **click OK**.

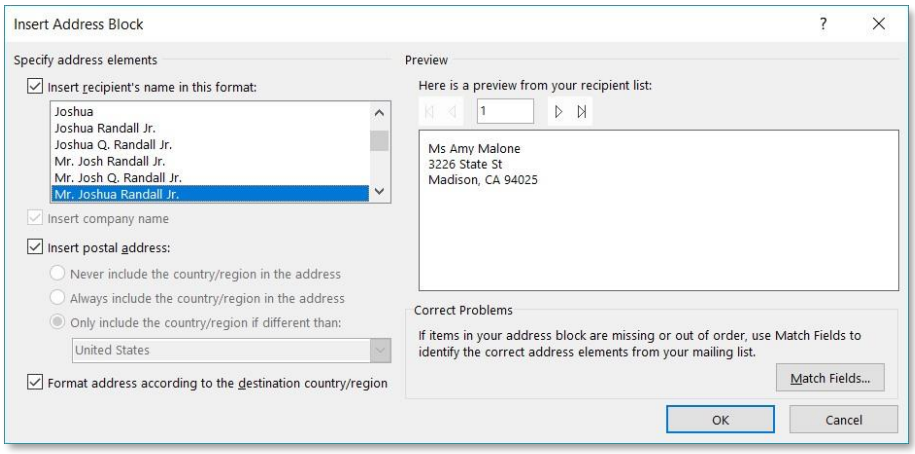


e. **Click Next: Write your letter** at the bottom of the Mail Merge pane.

10. **Step 4** - **Notice** the **Write your letter** prompt. Since we are using an existing letter, we merely have to click into a location within the letter to insert different items.

a. **Place** your insertion point below the return address for the Animal Shelter, then **click** Address block in the Mail Merge pane.

b. An **Insert Address Block** Dialogue box will appear with a **preview** of how your address block will look in the letter. If you select a different format, your preview will change. Make no changes. **Click** OK.



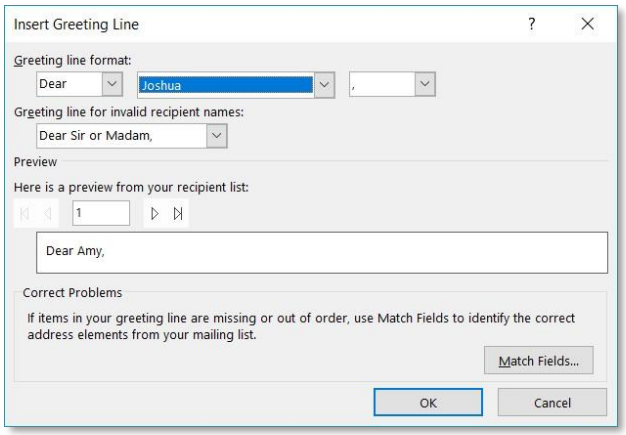
c. **Notice** how the text **«AddressBlock»** has appeared. This will be replaced with an actual address when we complete the mail merge.

d. **Place** your insertion point below the address block and **click** Greeting Line in the Mail Merge Pane.

e. An **Insert Greeting Line** dialogue box will appear to allow you to control the way you want your greeting to appear.

i. **Click** on the **list arrow** next to the box that reads "Mr . Randall" and **click** "Joshua". This will use the person's first name as the greeting.

ii. **Click OK**.



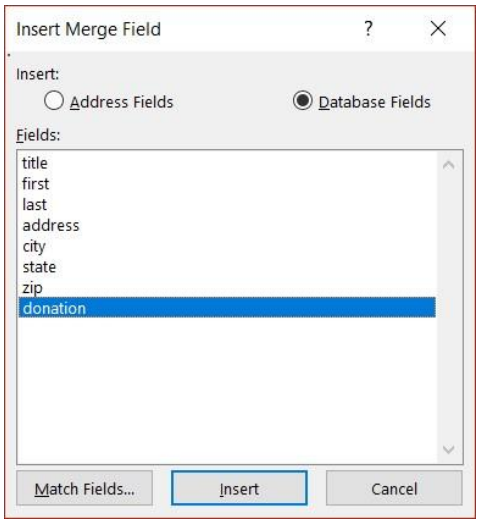
f. Within the body of the letter, **place** your insertion point immediately after the words **contribution of** in the first sentence. **Click** on **More items** in the Mail Merge pane.

g. An **Insert Merge Field** dialogue box will appear.

i. **Click donation**

ii. **Click Insert**.

iii. **Click Close**.



h. **Be sure** that there is a space before and after **«donation»**.

i. **Click Next: Preview your letters** at the bottom of the Mail Merge pane.

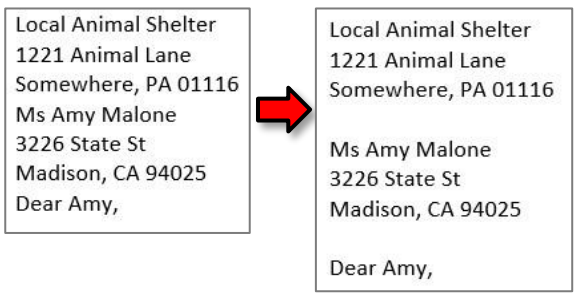
11. **Step 5** – **Notice** the **Preview your letters** prompt.

a. The letter to the first person on your list will appear.

b. **Use** the **arrows** to **scroll** through each succeeding letter.

c. **Notice** how the addresses and greeting are all clumped together. Let's insert some blank lines to separate them. Even though only one person's letter is showing, these changes will affect each letter.

i. **Click** at the end of the **first zip code** of the first address and press Enter. ii. **Click** after the **second zip code** and press Enter.



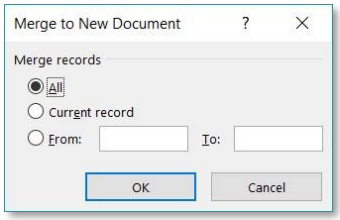
d. **Notice** that there is no $ before the donation amount in your letter. **Place** your insertion point directly before the donation amount, and **type** a **$**. This change will affect each letter.

e. **Click Next: Complete the merge** at the bottom of the Mail Merge pane.

12. **Step 6** – **Notice** the choices on the **Complete the merge** prompt.

a. **Click** on the **Edit individuals letters** choice.

b. This will open a **Merge to New Document** dialogue which allow us to merge all our records into a specific new document, separate and apart from the merge letter.docx, rather than just printing the letters out, so that in future we can have a record of to whom we sent letters. **Click OK**.



c. A **new document** called "Letters1" opens with each letter on a separate page.

d. **Scroll down** the document to see the other pages.

e. **Save** your file as **Fall Thank You Letters**. **Close** the file.

f. **Click Print** in the Mail Merge pane. The **Merge to Printer** dialogue box will open allowing you to select which records you wish to print letter for. Be warned: This option sends the records directly to the printer without showing you a print preview.

g. We will not be printing. **Click Cancel**.

h. **Close** Word. Do not save changes to merge letter.docx.

Steps to Use Mail Merge for Address Labels

1. To save time, we will use **Donations List.xlsx** again.

2. **Open** a new Word document

3. **Click** the **Mailings** tab.

4. **Find** the **Start Mail Merge** group. **Click Start Mail Merge** to **open** the menu and then **click Step-byStep Mail Merge Wizard**.

5. A Mail Merge pane will open on the right that will take you through the mail merge process in 6 steps.

6. **Step 1** - **Notice Select document type** prompt. By default, the radio button for letters is checked, **check** the radio button next to **Labels**. **Click** on **Next: Starting Document** at the bottom of the Mail Merge pane.

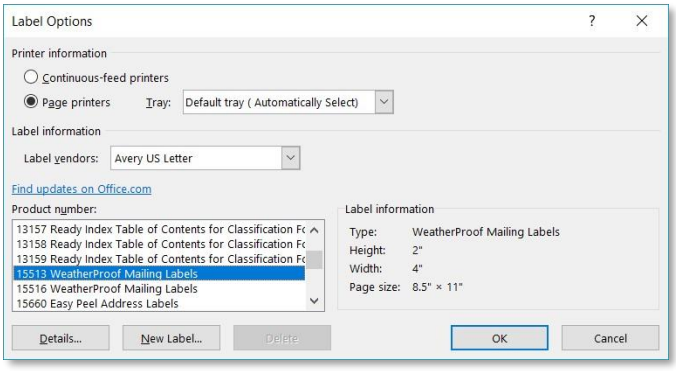
7. **Step 2** - **Notice** the **Select starting document** prompt. Accept the default selection of Change document layout. **Click Label Options**.

a. A dialogue box will open allowing you to select the type of printer (dot matrix or laser), the type of label product (such as Avery), and the product number.

`

b. **Click Avery US Letter** and product number **15513** for this lesson.

c. **Click OK**. (If you are using a custom label, click **Details**, and then type the size of the label.)



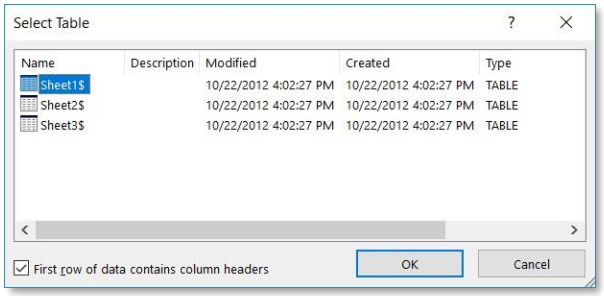
d. **Click** on **Next: Select Recipients** at the bottom of the Mail Merge pane.

8. **Step 3** – **Select recipients** prompt

a. The default selection, Use an existing list is the one we need. To find the list, **click Browse**.

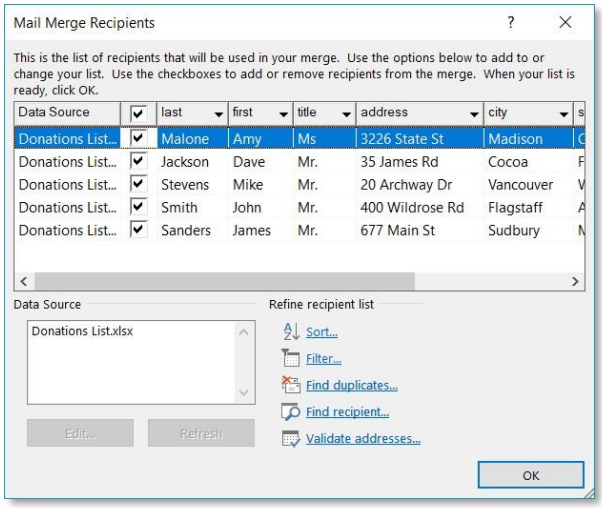
b. A **Select Data Source** dialogue box will open. **Navigate** to your flash drive and **open** DonationsList.xlsx

c. A **Select Table** dialogue box will open. We need to provide some information about our document. **Notice** that by default, Sheet 1 is selected. **Remember** that when we examined our Excel file our data was on the Worksheet entitled Sheet 1. Also **be sure** that the **check box** before First row of data contains column headers is selected. Click OK.



i. A **Mail Merge Recipients** dialogue box will open which will allow you to review, delete or change your list. Since we are not making any changes, **click OK**.

`



d. You will now see that the first label is blank and every other label has **«Next Record»**. This is because the first label begins on the first record. The other labels must be instructed to move on to the next record.

e. **Click Next: Arrange your labels** at the bottom of the Mail Merge pane

9. **Step 4** – **Arrange your labels** prompt

   a. **Scroll** to the **left** and **notice** the insertion point is in the first label.

   b. Since these are address labels, **click Address block**.

   c. An insert address block dialogue box will appear allowing you to make the same kinds of choices as is the merge letter above. **Click OK**.

   d. **Notice** how **«Address»** gets inserted into only the **first label**.

   e. Under the **Replicate labels** prompt, **click Update all labels**. This causes the address block to propagate into the other labels.

   f. **Click Next: Preview your labels** at the bottom of the Mail Merge pane

10. **Step 5** – **Notice** the **Preview your labels** prompt

    a. **Check** your labels

    b. **Click Next: Complete the merge** at the bottom of the Mail Merge pane.

11. **Step 6** – **Notice** the choices on the **Complete the merge** prompt.

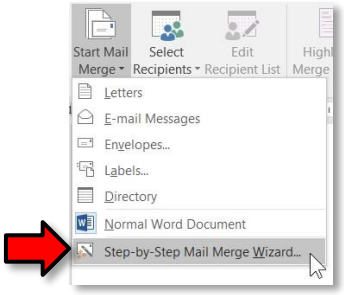
    a. The same options apply as for Merge letters above.

Steps to Use Mail Merge for Envelopes

1. To save time, we will use **Donations List.xlsx** again.
2. **Open** a new Word document
3. On the **ribbon**, **click** on the **Mailings** tab.

`

4. **Find** the **Start Mail Merge** group. **Click Start Mail Merge** to **open** the menu and then **click Step-byStep Mail Merge Wizard**.

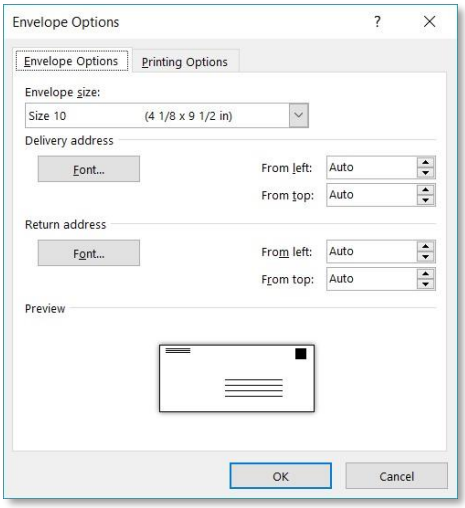


5. A Mail Merge pane will open on the right that will take you through the mail merge process in 6 steps.

6. **Step 1** - **Notice Select document type** prompt. By default, the radio button for letters is checked, **check** the radio button next to Envelopes. **Click** on **Next: Starting Document** at the bottom of the Mail Merge pane.

7. **Step 2** - **Select starting document** prompt. **Accept** the default selection of Change document layout. **Click** on **Envelope Options**.

    a. A dialogue box will open allowing you to select the envelope size, the type of font and position. **Select Size 10** (the default setting) for this lesson. **Click OK**.



    b. **Click** on **Next: Select Recipients** at the bottom of the Mail Merge pane.

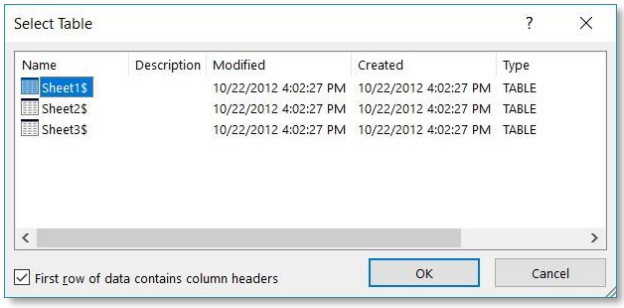
8. **Step 3** – **Notice** the **Select recipients** prompt

    a. The default selection, **Use an existing list** is the one we need. To find the list, **click Browse**.

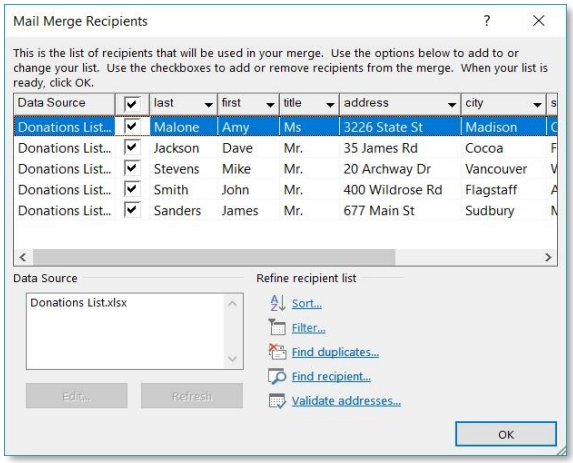
    b. A **Select Data Source** dialogue box will open. **Navigate** to your flash drive and **open Donations List.xlsx**

    c. A **Select Table** dialogue box will open. We need to provide some information about our document. **Notice** that by default, Sheet 1 is selected. **Remember** that when we examined our Excel file our data was on the Worksheet entitled Sheet 1. Also **be sure** that the **check box** before First row of data contains column headers is *selected*. **Click OK**.

`



j. A **Mail Merge Recipients** dialogue box will open which will allow you to review, delete or change your list. Since we are not making any changes, **click OK**.



d. **Click** on **Next: Arrange Your Envelope** at the bottom of the Mail Merge pane.

9. **Step 4** – **Notice** the **Arrange your Envelope** prompt

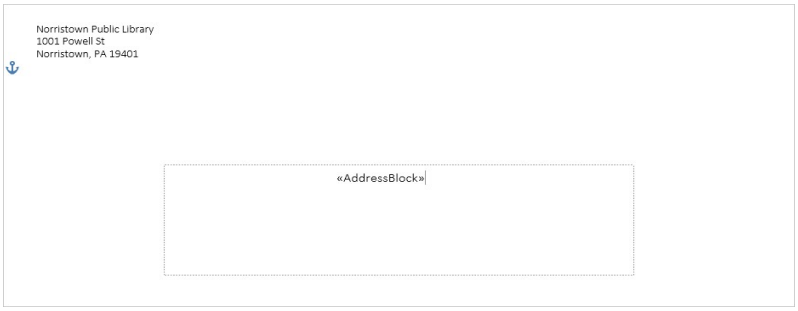
a. The insertion point will be located where a return address should be placed. Most businesses will have preprinted envelopes. If yours do not, **type** your **return address**.

b. **Click around** in the center of the envelope towards the bottom until a **text box** appears.

c. **Click Address block** in the Mail Merge pane.

d. An insert address block dialogue box will appear allowing you to make the same kinds of choices as in the merge letter above. **Click OK**.

e. **Click Next: Preview your envelopes** at the bottom of the Mail Merge pane.



10. **Step 5** – **Notice** the **Preview your envelopes** prompt

a. **Check** your envelopes.

b. **Click Next: Complete the merge** at the bottom of the Mail Merge pane.

11. **Step 6** – **Notice** the choices on the **Complete the merge** prompt.

`

3) Graded Lab Tasks:

Note: The instructor can design graded lab activities according to the level of difficulty and complexity of the solved lab activities. The lab tasks assigned by the instructor should be evaluated in the same lab.

Lab Task 1

Create a short report on topic of your choice which includes the following:

- *Title page*
- *Table of Content*
- *Bibliograpghy and referencing.*
- *Tables/Figures with captions.*

Lab Task 2

A flyer is a pamphlet, which is a form of paper advertisement intended for wide distribution in a public place, handed out to individuals or sent through the mail. Consider you work at a company. Your boss has asked you to prepare a flyer that advertises company's major products and/or services.

`

Lab 02

Introduction to Power Point

Objective:

In this lab the students will be able to combine text, graphics, and predesigned backgrounds to create professional presentations.

Activity Outcomes:

The students will be able to:

- Prepare a professional Presentation
- Add animations and sounds to the slides.

Instructor Note:

As a pre-lab activity, read "MicroSoft" official site for detail guidance.

1) Useful Concepts

MS PowerPoint is a program that is included in the Microsoft Office suite. It is used to make presentations for personal and professional purposes.

In this manual, we shall discuss in detail the functions and features of a PowerPoint presentation.

The following elements can be added to a Powerpoint slide:

- Clip Art
- Graphs
- Tables
- Photographs
- Charts
- Media Clips
- Videos

2) Solved Lab Activites

Sr.No	***Allocated Time***	***Level of Complexity***	***CLO Mapping***
1	***30***	***Medium***	***CLO-6***
2	***30***	***Medium***	***CLO-6***

Activity 1:

Making Presentation Slides

Solution:

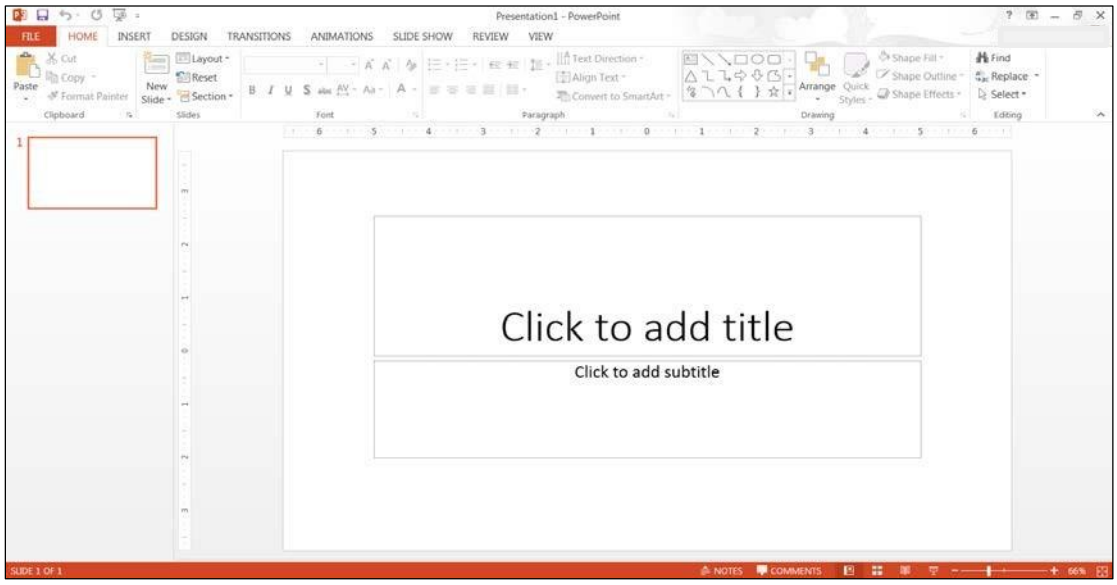
Slides in a presentation are similar to pages in a word processing document. All slides and graphics are saved in one file (example: **keys.xppt**). Use the PowerPoint file to present the information in the following ways:

`

- **On-screen slide show:** The keys.xppt file displays the slide show on a monitor or computerprojected large screen.
- **Web pages:** The keys.xppt file can be saved as Web page and then published on the Web.
- **Overhead transparencies:** The keys.xppt file can be printed as transparencies (**Important:** Make sure the appropriate transparencies are used for your printer model. The wrong type of transparencies can melt inside your printer.).
- **Handouts:** The keys.xppt file can print two to nine mini slides per page.

Create Slide Presentation:

This section will teach the basics of opening PowerPoint and beginning a presentation. When PowerPoint is launched the **Presentation** window will appear.



When creating a new presentation, you have choices about how to proceed. PowerPoint gives you a range of ways with which to start creating a presentation. You can start your presentation with:

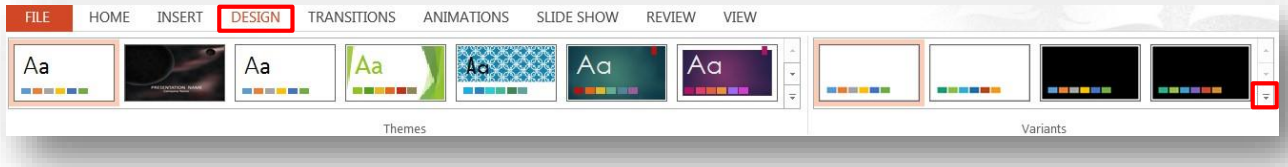
- **Blank:** Slides that are unformatted and have no color scheme.
- **Design:** Slide Themes that have design concepts, fonts, and color schemes.

- **Template on Microsoft.com:** Microsoft Office Templates and Theme Gallery which are arranged according to type (Click on the **File** tab, select the **New** option, and then click on **PowerPoint presentations and slides** from the **Available Templates and Themes**.).

This workshop section will focus on using a Design Theme.

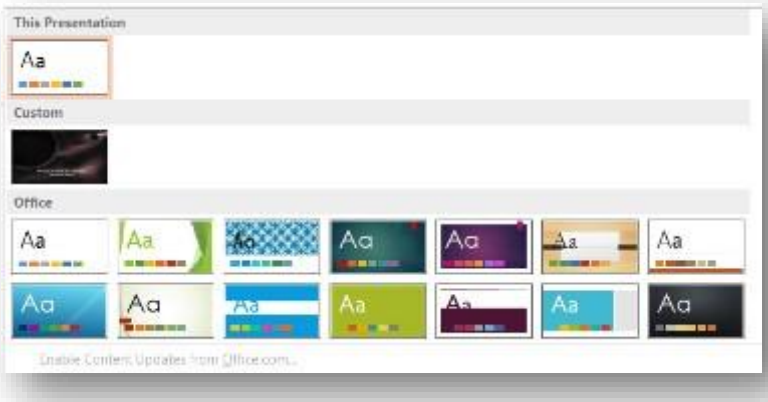
A. Design Theme

A Theme gives your slides a consistent appearance throughout your presentation. Themes contain color schemes with custom formatting, styled fonts, and layouts. When you apply a design template to your presentation, the slide master and color scheme of the template replaces the original blank slide.



1. Select the **Design** tab, then on the **Theme** group, click on the drop-down arrow next to the last Theme.

2. The **All Themes** window will appear with available presentation Themes.



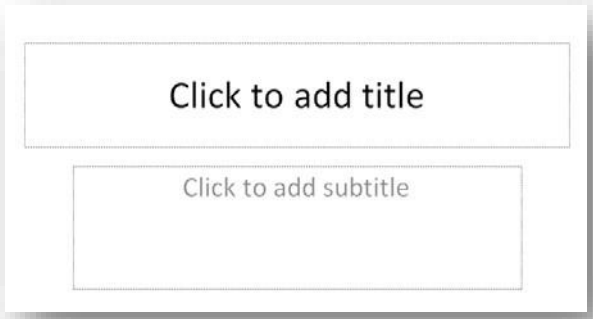
3. Hover the mouse pointer over a Theme to preview it.

4. Click on a **Theme** of your choice.

Note: You can change the Theme during or after the creation of your PowerPoint file.

B. Add Text

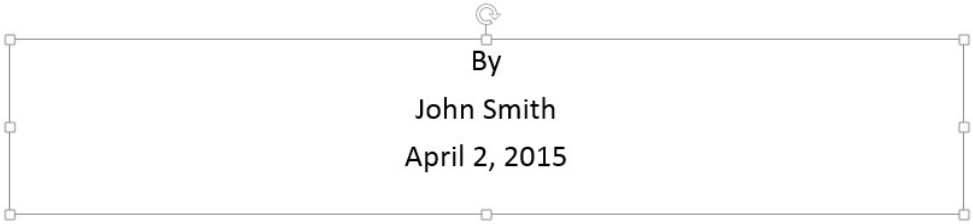
The template for the design Theme you select will determine the font type and text alignment. PowerPoint places all information (text and graphics) contained on a slide in separate **Placeholders**. Placeholders are designated by dotted lines; they appear on a slide as guides, but they will not appear on the finished presentation. In order to edit text, click once inside of the **Text Placeholder** and the insertion point will appear; then begin to type your text.



1. Click in the **Title Placeholder** and type the text title below.

2. Click in the **Subtitle Placeholder**.

3. Type the text below (You will need to press the **Enter** key after each line of text.).

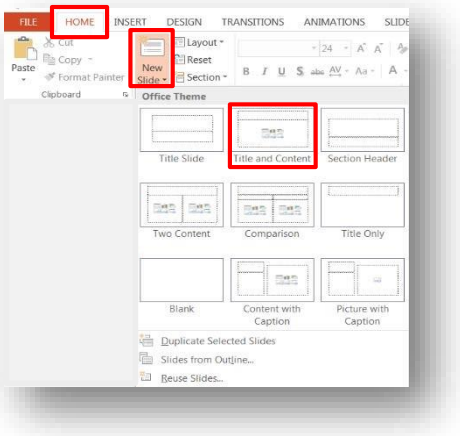


4. **Save** the presentation. Click on the **File** tab and then click on **Save As**. The **Save As** window will open. In the **File name** box, type **Keys to Success** for the presentation name.
   The instructor will indicate where to save the file. Click on the **Save** button.

`

Add New Slide

A slide layout defines the placement of text, pictures, tables, and graphs. If you change the layout of a slide, the text and graphics remain intact. You can resize text and graphic boxes to conform to the new layout



1. On the **Home** ribbon, located in the **Slides** group, click on the **New Slide** drop-down arrow. The Office Theme panel will appear with multiple slide layouts. Select your desired slide layout.

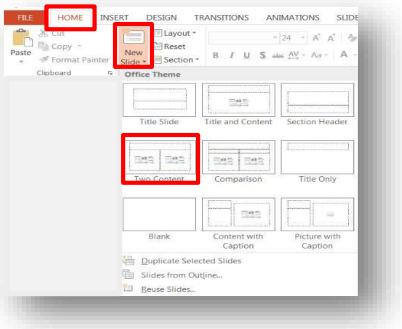
2. For this exercise, click on the second Layout (**Title and Content**) in the first row.

3. In the **Title Placeholder**, type the text **Agenda** as seen above.

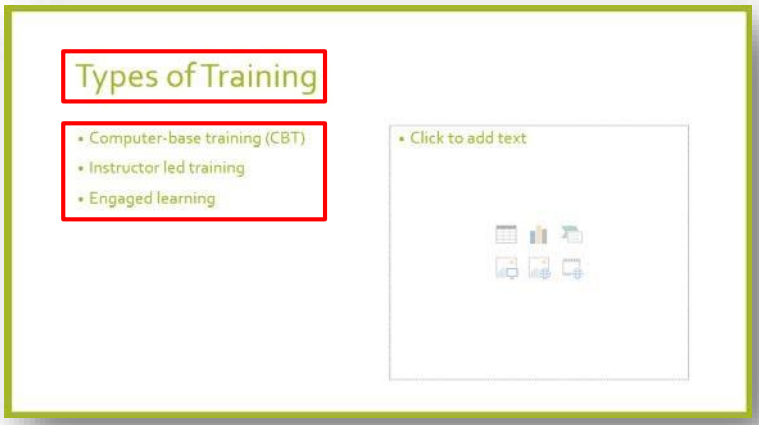
4. In the **Text Placeholder**, type the bulleted text as seen above (You will need to press **Enter** after each line of text.).

5. Add another new slide.

`



6. On the **Home** ribbon, click on the **New Slide** drop-down arrow and then select the **Two Content** slide layout (This slide contains a title, text, and clip art placeholders.).



7. Click inside the **Title and Text Placeholders** and type the text shown above.

8. On the **Quick Access Toolbar**, click on the **Save** button to save your presentation changes.

Activity 2:

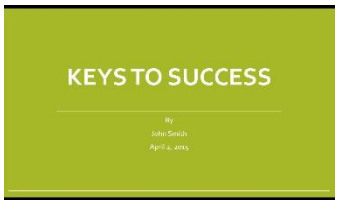
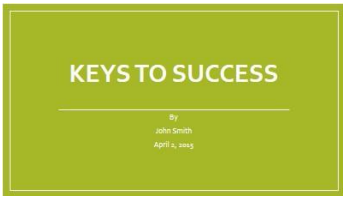
Editing Techniques

Solution:

This section will teach you basic techniques for editing slides.

View Modes for Editing

The **Normal**, **Slide Sorter**, **Reading**, and **Slide Show** Views allow you to type, edit, and view your presentation. To switch between views, click the **View** Options at the lower right-hand side of the PowerPoint window.

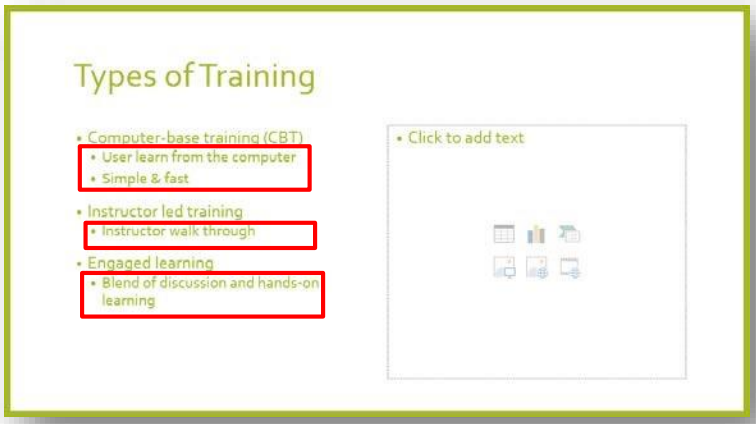
Normal View 	Normal View is the main editing View, which you use to write and design your presentation. The View has three working areas: on the left, tabs that alternate between an outline of your slide text (**Outline** tab), and your slides displayed as thumbnails (**Slides** tab); on the right, the slide pane, which displays a large view of the current slide; and on the bottom, the notes pane.
Slide Sorter View 	Slide Sorter View is an exclusive view of your slides in thumbnail form. When you are finished creating and editing your presentation, Slide Sorter gives you an overall picture of it — making it easy to reorder, add, or delete slides, and preview your transition and animation effects.
Reading View 	Reading View is new in PowerPoint 2013. It is similar to Slide Show View. The difference between the two Views is that while Slide Show View takes over the whole screen, the slide in Reading View is shown in full screen, but you will see the PowerPoint title band at the top of the screen. The PowerPoint status bar and the Windows task bar are also displayed at the bottom of the screen.
Slide Show View 	Slide Show View takes up the full computer screen, like an actual slide show presentation. In this full-screen View, you see your presentation the way your audience will. You can see how your graphics, timings, movies, transition effects, and animation elements will look in the actual show.

B. Edit Bulleted List

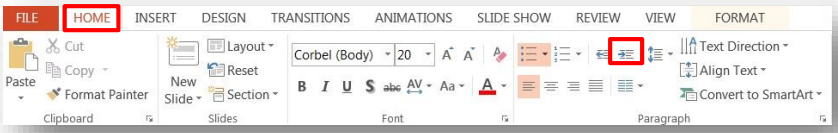
Typing text in PowerPoint is similar to typing in other applications. However, since PowerPoint deals with bulleted lists, a few keystrokes will be identified to help in typing multiple lines.

1. Confirm you are on **slide three**.

`



2. In **Normal** View, edit the bulleted list to include the circled text above. Place the Insertion bar after each line of the bulleted text and then press the **Enter** key.



3. To add a sub-bullet, click on the **Increase Indent** button, located on the **Home** ribbon. The **Tab** key can also be used to indent text.

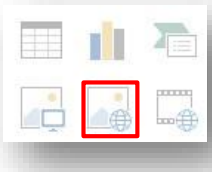
4. Type the text and if additional sub-bullets lines are needed press the **Enter** key, after your line of text.

Note: Pressing **Enter** after any text returns the cursor to the same indent (paragraph) level for the next line. The Tab key is also used before typing to indent text to the next indent level (sub-bullet point) and pressing **Shift+Tab** before typing to return to the previous indent level.

C. Add Pictures

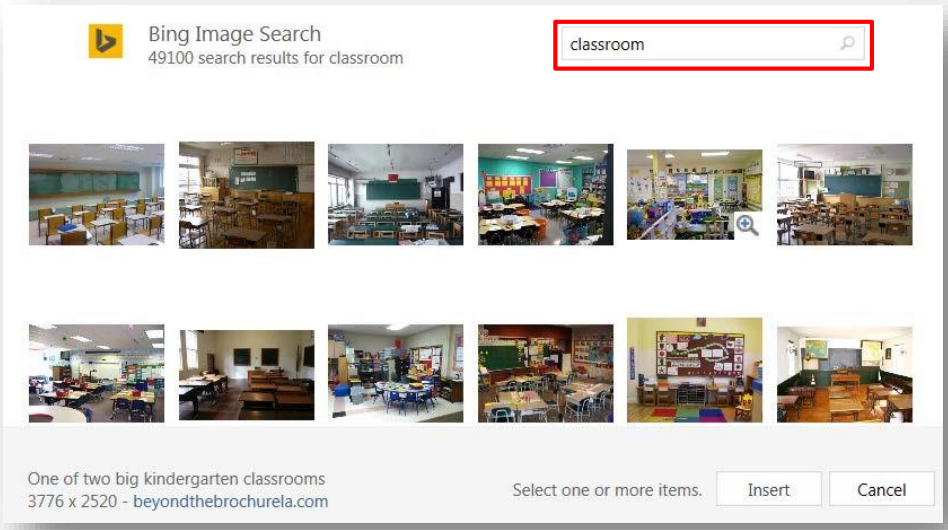
Online Pictures are any type of computerized images such as artwork and photos. You can make your presentation more eye-catching and entertaining by adding Pictures.

1. Confirm you are on **slide three**.



2. Click on the **Online Picture** button in the **Illustration** box, located in the **Text Placeholder**.

3. The **Bing Image Search** window will appear.



4. In the **Search** box, type the word **Classroom**, and then press the **Enter** key. A variety of online images associated with your search will appear.

5. Scroll through the **Pictures** window to find your desired image.

6. To insert the image, place the mouse pointer on the image and then click on the left mouse button twice.

3) Graded Lab Tasks

Note: The instructor can design graded lab activities according to the level of difficulty and complexity of the solved lab activities. The lab tasks assigned by the instructor should be evaluated in the same lab.

Lab Task 1

Make a presentation(15 slides) on different Input/Output devices (Chapters 2 and 3 of textbook). Use different effects, animations, transitions and a custom Master Slide.

***Note:** this should be a formal presentation. Take care of font size, font color, font names, and other styles and animations accordingly.*

`

Lab 03

Introduction to MS Excel

Objective:

It will enable students to understand and use MS Excel. How to create MS Excel document, use rows and columns.

Activity Outcomes:

The students will be able to:

- Create Spread Sheets .
- Making reports.
- To filtration the data

Instructor Note:

As a pre-lab activity, read “MicroSoft” official site for detail guidelines.

1) Useful Concepts

MS Excel is a commonly used Microsoft Office application. It is a spreadsheet program which is used to save and analyse numerical data. Excel is typically used to organize data and perform financial analysis. It is used across all business functions and at companies from small to large.

The main uses of Excel include:

- Data entry
- Data management
- Accounting
- Financial analysis
- Charting and graphing
- Programming
- Time management
- Task management
- Financial modeling
- Customer relationship management (CRM)
- Almost anything that needs to be organized!

2) Solved Lab Activites

Sr.No	***Allocated Time***	***Level of Complexity***	***CLO Mapping***
1	***10***	***Low***	***CLO-6***
2	***25***	***Medium***	***CLO-6***
3	***25***	***Medium***	***CLO-6***

`

Activity 1:

Conditional Formatting

Solution:

Conditional formatting allows you to change the appearance of a cell, based on criteria that you define, using predetermined rules in Excel.

Highlight Cells Rules

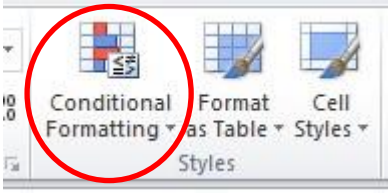
Using the highlight cells rules, you can highlight cells in your data that are greater or less than a value, between or equal to a value or contain a specified or duplicate value.

Greater Than

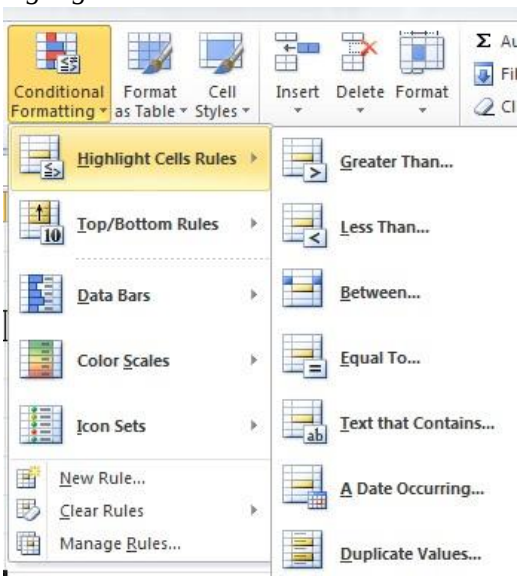
To highlight cells which contain data greater than a specific value:

1. Highlight the data range.

2. Select the *Conditional Formatting* tool



3. Hover over *Highlight Cells Rules* to reveal the menu of different rules.



4. Select *Greater Than* from the menu to open the Greater Than dialogue box:

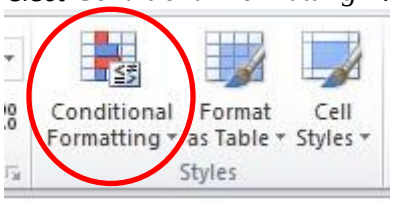
5. Enter the value that you want to set as your lower limit for the Greater Than condition.

6. Select the type of formatting from the dropdown menu.
7. Select Ok.

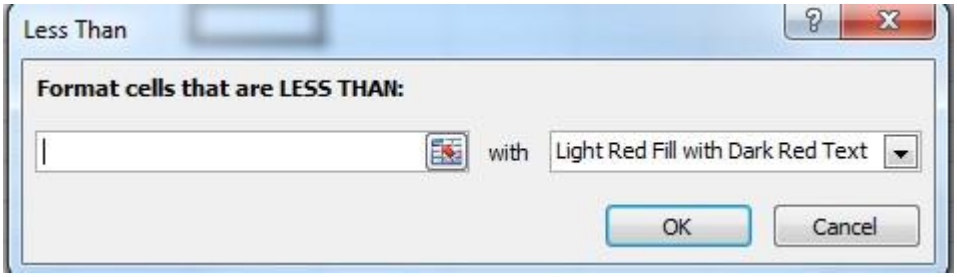
The cells which contain a value greater than the value you specified will now appear with the cell formatting which you selected.

Less Than

To highlight cells that contain data less than a specific value:
1. Highlight the data range.
2. Select *Conditional Formatting* .



3. Hover over *Highlight Cell Rules*.
4. Select *Less Than* to open the Less Than dialogue box.

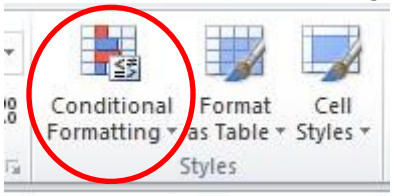


5. Enter the value that you want to set as your upper limit for the Less Than condition
6. Select *Ok.*

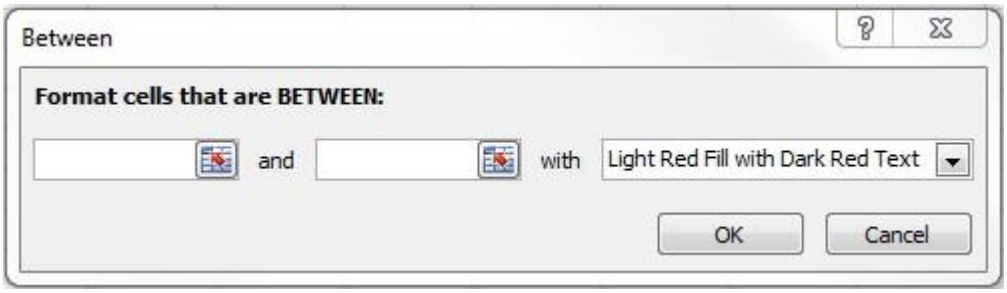
The cells which contain a value less than the value you specified will now appear with the cell formatting which you selected.

Between

To highlight cells between two specific values:
1. Highlight the data range.
2. Select *Conditional Formatting* .



3. Hover over *Highlight Cells Rules* to reveal the menu of different rules.
4. Select *Between* to open the Between dialogue box.



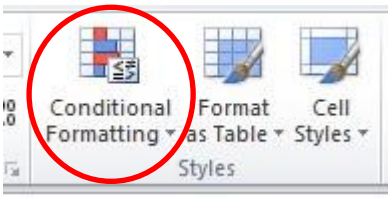
5. Enter the lower limit in the first box and the upper limit in the second box.
6. Select the cell formatting.
7. Select *Ok*.

The cells which contain a value between the two specified values will now appear with the cell formatting which you selected.

Equal To

To highlight cells equal to a specific value:

1. Highlight the data range.
2. Select *Conditional Formatting*.



3. Hover over *Highlight Cells Rules*.
4. Select *Equal To* to open the Equal To dialogue box.

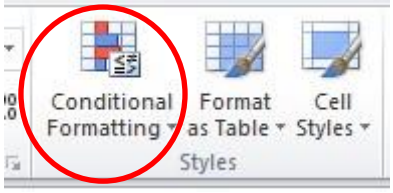
5. Enter the value that you're looking for.
6. Select the type of cell formatting you wish to use.
7. Select *Ok*.

The cells which contain the specified value will now appear with the cell formatting which you selected.

Text That Contains

To highlight cells that contain a certain character(s):

1. Highlight the data range.
2. Select *Conditional Formatting* .



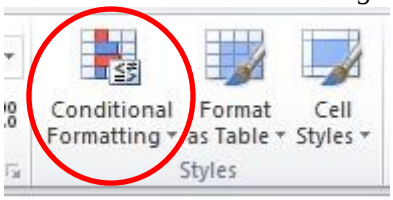
3. Hover over the *Highlight Cells Rules*.
4. Select *Text That Contains* to open the Text That Contains dialogue box.

5. Enter the character(s) you're looking for.
6. Select the type of cell formatting you wish to use.
7. Select *Ok*.

The cells which contain the specified character(s) will now appear with the cell formatting which you selected.

A Date Occurring

To highlight cells that contain a certain date or date range:
1. Highlight the data range.
2. Select *Conditional Formatting* .



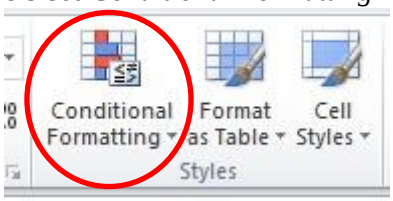
3. Hover over the *Highlight Cells Rules*.
4. Select *A Date Occurring* to open the Date Occurring dialogue box.

5. Select the date or date range that you're looking for.
6. Select the type of cell formatting.
7. Select *Ok*.

The cells which contain the specified date or date range will now appear with the cell formatting which you selected.

Duplicate Values

To highlight cells that contain either duplicate or unique values:
1. Highlight the data range.
2. Select *Conditional Formatting* .



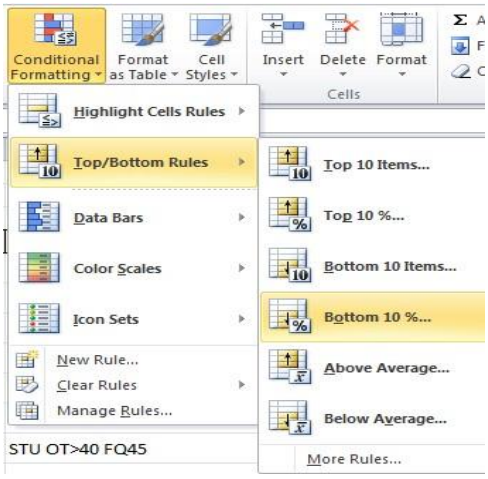
3. Hover over *Highlight Cells Rules*.
4. Select *Duplicate Values* to open the Duplicate Values dialogue box.

5. Select either *Duplicate* or *Unique* from the drop down menu.
6. Select the type of cell formatting you wish to use.
7. Select *Ok.*

The cells which contain either duplicate or unique values will now appear with the cell formatting which you selected.

Top/Bottom Rules

Top and bottom rules can be used to highlight cells that are the top or bottom ten items or the top or bottom ten percent. They can also be used to identify items above or below the average.



Activity 2:

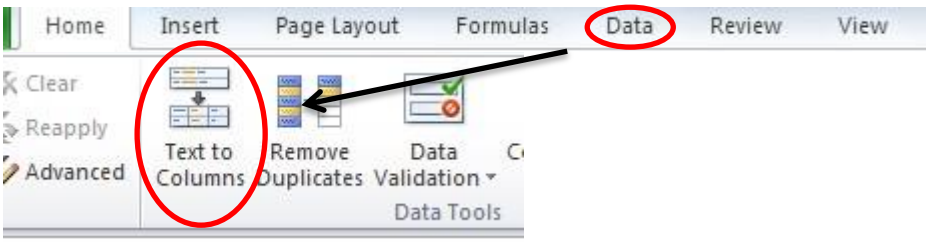
Separating Text within a Cell

Solution:

When data is combined within a cell, such as a first and last name, Excel is able to separate this data into two cells.

To separate data within a cell:

1. Insert a blank column to the right of the column containing the merged data.
2. Highlight the column of full names.
3. Select the *Data* tab.
4. Select *Text to Columns* .

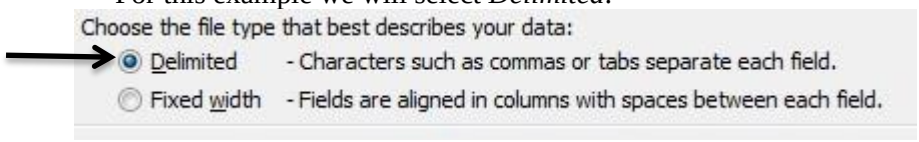


The Convert Text to Columns Wizard dialogue box will.

5. Choose the appropriate data type.

To separate a column based on punctuation characters, select *Delimited*.
To separate a column based on spaces between each field, select *Fixed Width*.

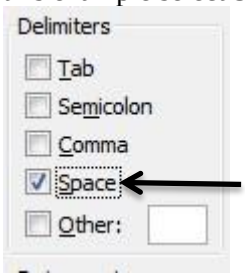
For this example we will select *Delimited*.



6. Select *Next*.

7. Choose your delimiters for the text separation.

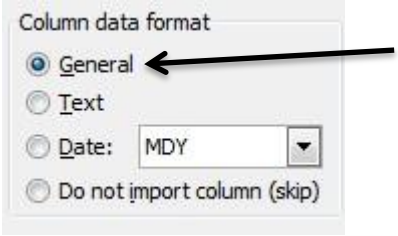
   For this example select *Space*.



8. Select *Next*.

`

9. Select the data format for each column.
   For this example select *General*.



10. Select *Finish*.

Data will be displayed as separate columns

Sallie Shaffer		Sallie	Shaffer
Joseph Garcia		Joseph	Garcia
Sallie Shaffer		Sallie	Shaffer
Surhid Gautam		Surhid	Gautam
Charles Dennis		Charles	Dennis
Yasmine Johnson		Yasmine	Johnson
Kathleen Chilton	→	Kathleen	Chilton
Kathleen Chilton		Kathleen	Chilton
Helen Martin		Helen	Martin
Helen Sue Martin		Helen	Martin
Joseph Garcia		Joseph	Garcia
Sallie Shaffer		Sallie	Shaffer
Mark Abraham		Mark	Abraham
Mark Abraham		Mark	Abraham
Surhid Gautam		Surhid	Gautam
Carmen Roberts		Carmen	Roberts

Activity 3:

Use of Functions and Formulas

Solution:

Excel has many different functions and formulas which can be used to manipulate data in a variety of ways, such as sums, subtotals, averages, number counts, maximums, and minimums.

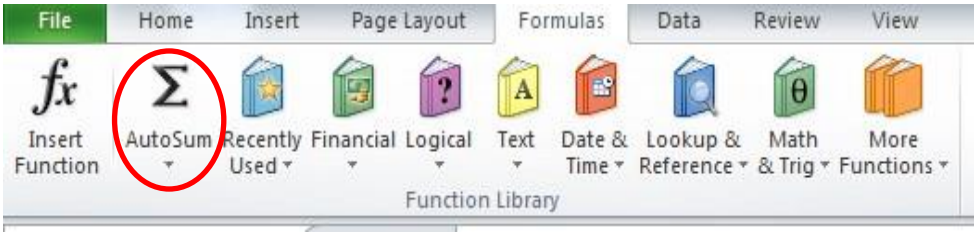
Sums: One of the most commonly used functions of Excel is summation. If you have a data table for a single student with amounts and dates of payment, to find the sum of all payments, you would use the summation function.

Last Name	First Name	Banner ID	Amount Paid	Date Paid
Smith	John	745082	3,000	9/8/2013
			2,500	10/12/2013
			1,500	12/10/2013
			2,000	3/15/2014
			1,000	6/10/2014

`

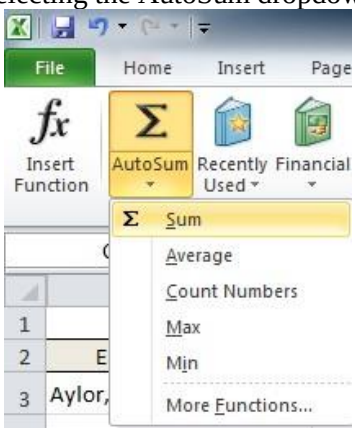
To add numbers in a column:

1. Select the cell directly beneaththe last entry.
2. Select *Auto Sum* located on the*Formulas* tab.



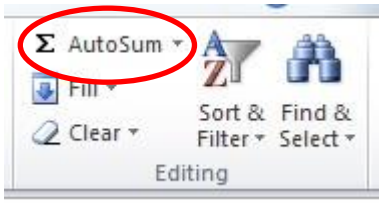
3. Select the *AutoSum* button
4. This will select all items within the column
5. Click the *Enter* key on your keyboard to calculate the sum of all fields.

Other functions are available by selecting the AutoSum dropdown



Other functions include: averaging the numbers in a column, counting the numbers in a column and finding the minimum and/or maximum numbers in the column.

Additionally, there is an *AutoSum* button and dropdown menu also located on the Home toolbar.



Subtotaling

The Subtotal tool is used sum data by group. Subtotaling data eliminates the need to manually insert a row and perform a summation.

Below is a sample data sheet for which we need to calculate the total amount paid for each semester.

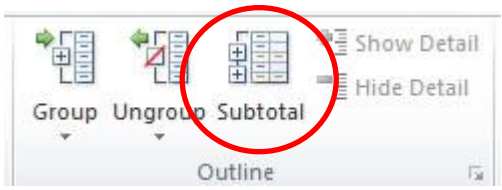
Last Name	First Name	Banner ID	Amount Paid	Semester
Smith	John	745082	$ 3,000.00	Fall
			$ 2,500.00	Fall
			$ 1,500.00	Fall
			$ 2,000.00	Spring
			$ 1,000.00	Spring

`

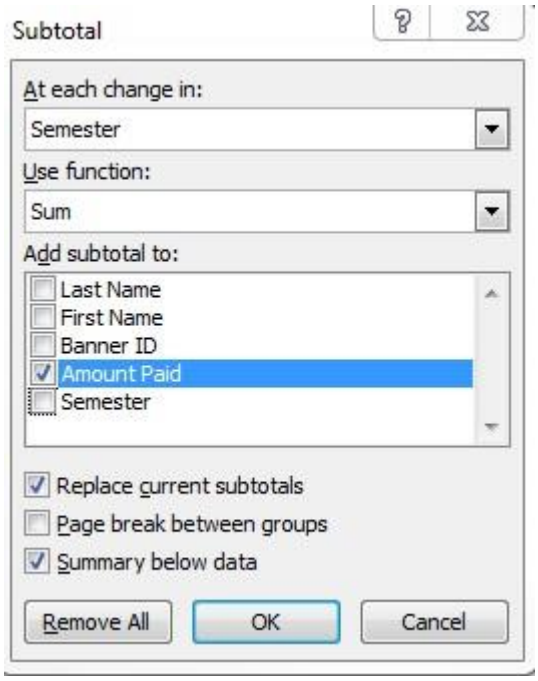
One Level Subtotals

To Subtotal a data sheet:

1. Select the *Subtotal* button located on the Data toolbar.



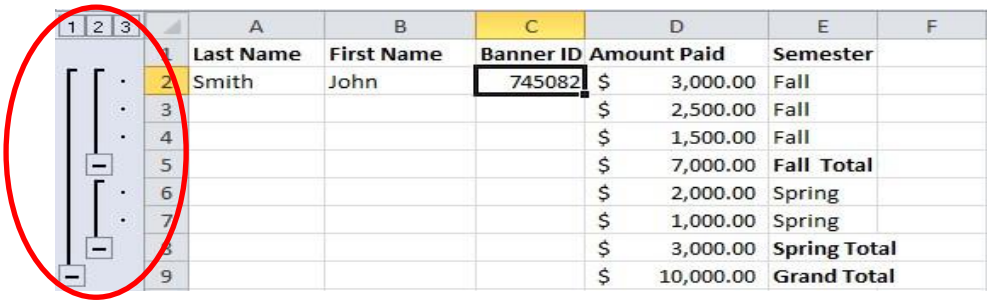
The Subtotal dialogue box will open



To subtotal this data sheet by semester:

1. Choose *Semester* for the *At Each Change* In dropdown.
2. Select *Sum* for the *Use Function* dropdown.
3. Choose *Amount Paid* for the *Add Subtotal To* field.
4. Click *Ok*.

Subtotals will automatically be added to your data.



	A	B	C	D	E	F
1	**Last Name**	**First Name**	**Banner ID**	**Amount Paid**	**Semester**	
2	Smith	John	745082	$ 3,000.00	Fall	
3				$ 2,500.00	Fall	
4				$ 1,500.00	Fall	
5				$ 7,000.00	**Fall Total**	
6				$ 2,000.00	Spring	
7				$ 1,000.00	Spring	
8				$ 3,000.00	**Spring Total**	
9				$ 10,000.00	**Grand Total**	

The subtotal hierarchy located to the left of the spreadsheet can be used to hide some of the data within the spreadsheet.

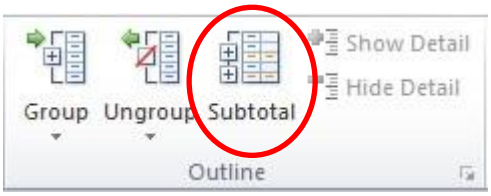
To view only the grand total, select column 1
To view the total for each subsection, select column 2 To view all data, select column 3.

Nested Level Subtotals

Nested Level Subtotals are used to subtotal more than one level of data.
For this example our list of data contains individual payers and semesters

Semester	Last Name	First Name	Banner ID	Amount Paid
Fall 2013	Smith	John	745082	$ 3,000.00
Fall 2013	Smith	John	745082	$ 2,500.00
Fall 2013	Smith	John	745082	$ 1,500.00
Fall 2013	Jones	Katherine	642986	$ 1,500.00
Fall 2013	Jones	Katherine	642986	$ 2,000.00
Fall 2013	Jones	Katherine	642986	$ 3,250.00
Spring 2014	Smith	John	745082	$ 2,000.00
Spring 2014	Smith	John	745082	$ 1,000.00
Spring 2014	Jones	Katherine	642986	$ 3,000.00
Spring 2014	Jones	Katherine	642986	$ 2,750.00
Fall 2014	Smith	John	745082	$ 2,750.00
Fall 2014	Smith	John	745082	$ 3,250.00
Fall 2014	Jones	Katherine	642986	$ 1,750.00
Fall 2014	Jones	Katherine	642986	$ 2,000.00
Spring 2015	Smith	John	745082	$ 1,750.00
Spring 2015	Smith	John	745082	$ 2,250.00
Spring 2015	Smith	John	745082	$ 2,000.00
Spring 2015	Jones	Katherine	642986	$ 2,250.00
Spring 2015	Jones	Katherine	642986	$ 2,500.00
Spring 2015	Jones	Katherine	642986	$ 3,000.00

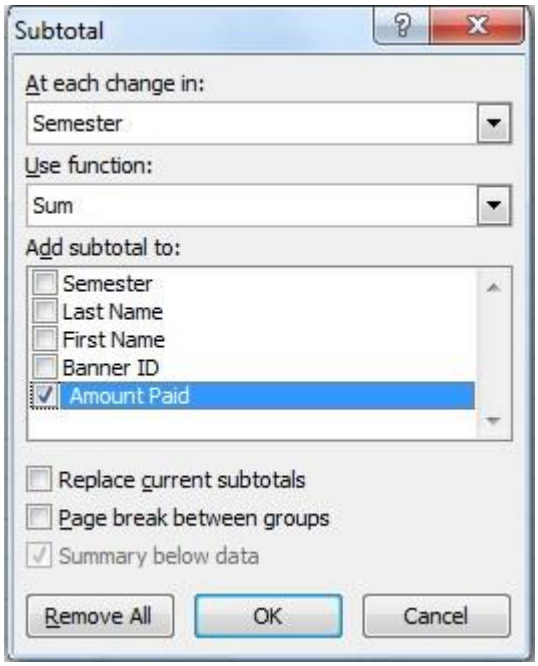
1. Select any cell within your range of data
2. Select *Subtotal* on the Data tab.



The Subtotal dialogue box will open.

3. For the *At Each Change* in dropdown menu, select *Semester*.
4. Choose to *Use Function*, *Sum*.
5. Choose to *Add Subtotal To*, *Amount Paid*.

`



6. Click *Ok*.

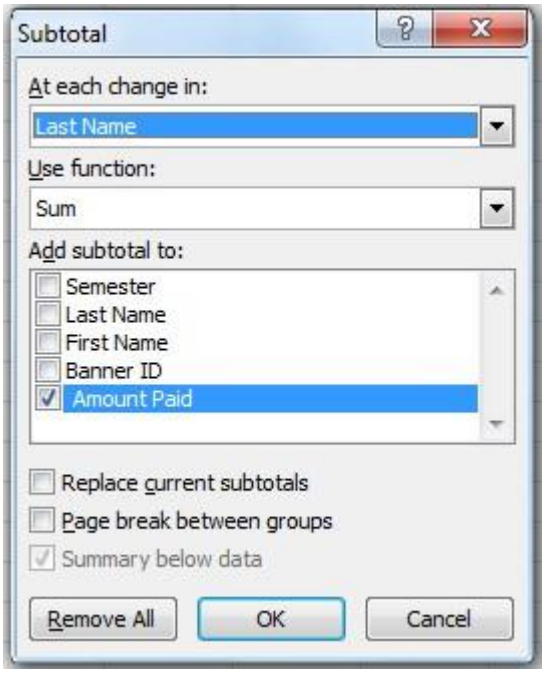
The first level of subtotal will be added to the data.

	A	B	C	D	E
1	**Semester**	**Last Name**	**First Name**	**Banner ID**	**Amount Paid**
2	Fall 2013	Smith	John	745082	$ 3,000.00
3	Fall 2013	Smith	John	745082	$ 2,500.00
4	Fall 2013	Smith	John	745082	$ 1,500.00
5	Fall 2013	Jones	Katherine	642986	$ 1,500.00
6	Fall 2013	Jones	Katherine	642986	$ 2,000.00
7	Fall 2013	Jones	Katherine	642986	$ 3,250.00
8	**Fall 2013 Total**				$ 13,750.00
9	Spring 2014	Smith	John	745082	$ 2,000.00
10	Spring 2014	Smith	John	745082	$ 1,000.00
11	Spring 2014	Jones	Katherine	642986	$ 3,000.00
12	Spring 2014	Jones	Katherine	642986	$ 2,750.00
13	**Spring 2014 Total**				$ 8,750.00
14	Fall 2014	Smith	John	745082	$ 2,750.00
15	Fall 2014	Smith	John	745082	$ 3,250.00
16	Fall 2014	Jones	Katherine	642986	$ 1,750.00
17	Fall 2014	Jones	Katherine	642986	$ 2,000.00
18	**Fall 2014 Total**				$ 9,750.00
19	Spring 2015	Smith	John	745082	$ 1,750.00
20	Spring 2015	Smith	John	745082	$ 2,250.00
21	Spring 2015	Smith	John	745082	$ 2,000.00
22	Spring 2015	Jones	Katherine	642986	$ 2,250.00
23	Spring 2015	Jones	Katherine	642986	$ 2,500.00
24	Spring 2015	Jones	Katherine	642986	$ 3,000.00
25	**Spring 2015 Total**				$ 13,750.00
26	**Grand Total**				$ 46,000.00

To add an additional level of subtotals:

1. Select *Subtotal*
2. For the *At Each Change* in dropdown menu, select *Last Name*.
3. Choose to *Use Function*, *Sum*.
4. Choose to *Add Subtotal To*, *Amount Paid*.
5. Ensure the checkbox *Replace Current Subtotals* is unchecked.

`



6. Click *Ok*.

The second level of subtotals will be added to the data range:

	A	B	C	D	E
1	**Semester**	**Last Name**	**First Name**	**Banner**	**Amount Pai**
2	Fall 2013	Smith	John	745082	$ 3,000.00
3	Fall 2013	Smith	John	745082	$ 2,500.00
4	Fall 2013	Smith	John	745082	$ 1,500.00
5		**Smith Total**			$ 7,000.00
6	Fall 2013	Jones	Katherine	642986	$ 1,500.00
7	Fall 2013	Jones	Katherine	642986	$ 2,000.00
8	Fall 2013	Jones	Katherine	642986	$ 3,250.00
9		**Jones Total**			$ 6,750.00
10	**Fall 2013 Total**				$ 13,750.00
11	Spring 2014	Smith	John	745082	$ 2,000.00
12	Spring 2014	Smith	John	745082	$ 1,000.00
13		**Smith Total**			$ 3,000.00
14	Spring 2014	Jones	Katherine	642986	$ 3,000.00
15	Spring 2014	Jones	Katherine	642986	$ 2,750.00
16		**Jones Total**			$ 5,750.00
17	**Spring 2014 Total**				$ 8,750.00
18	Fall 2014	Smith	John	745082	$ 2,750.00
19	Fall 2014	Smith	John	745082	$ 3,250.00
20		**Smith Total**			$ 6,000.00
21	Fall 2014	Jones	Katherine	642986	$ 1,750.00
22	Fall 2014	Jones	Katherine	642986	$ 2,000.00
23		**Jones Total**			$ 3,750.00
24	**Fall 2014 Total**				$ 9,750.00
25	Spring 2015	Smith	John	745082	$ 1,750.00
26	Spring 2015	Smith	John	745082	$ 2,250.00
27	Spring 2015	Smith	John	745082	$ 2,000.00
28		**Smith Total**			$ 6,000.00
29	Spring 2015	Jones	Katherine	642986	$ 2,250.00
30	Spring 2015	Jones	Katherine	642986	$ 2,500.00
31	Spring 2015	Jones	Katherine	642986	$ 3,000.00
32		**Jones Total**			$ 7,750.00
33	**Spring 2015 Total**				$ 13,750.00
34	**Grand Total**				$ 46,000.00

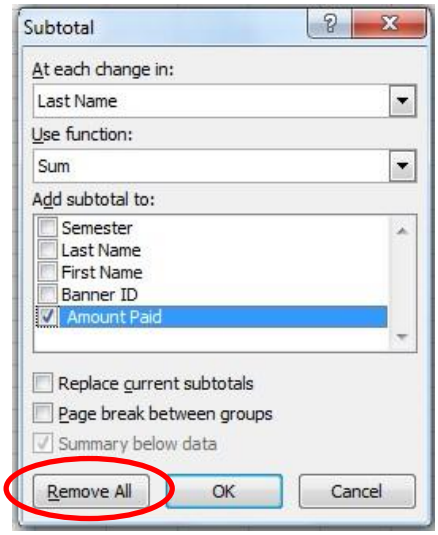
Removing Subtotals

To remove subtotals from a data sheet:

1. Select the *Subtotal t*ool

The Subtotal Dialogue box will appear.

2. Select *Remove All* to remove all subtotals.



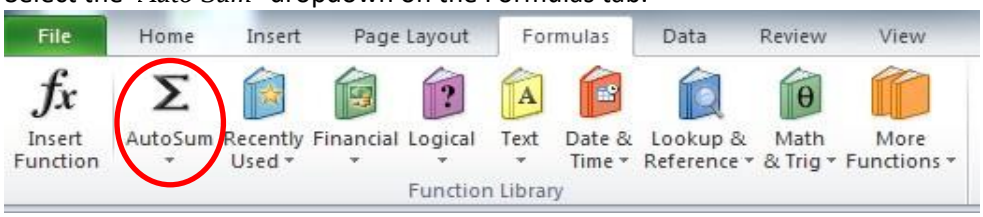
Average

To find the average of a select range of data:

1. Select the cell directly beneath the range of data

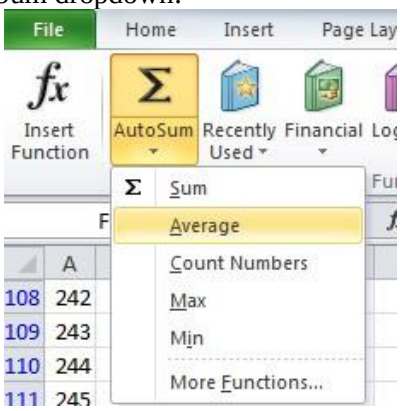
Last Name	First Name	Banner ID	Amount Paid	Date Paid
Smith	John	745082	3000	9/8/2013
			2500	10/12/2013
			1500	12/10/2013
			2000	3/15/2015
			1000	6/10/2015

2. Select the *Auto Sum* dropdown on the Formulas tab.



3.

Choose *Average* from the Auto Sum dropdown:



4. Select the range of cells to calculate
5. Click *Enter* on your keyboard

Last Name	First Name	Banner ID	Amount Paid	Date Paid
Smith	John	745082	$3,000	9/8/2013
			$2,500	10/12/2013
			$1,500	12/10/2013
			$2,000	3/15/2015
			$1,000	6/10/2015
			$2,000	

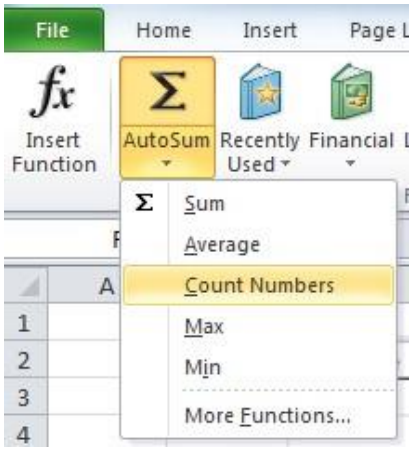
Count Numbers

To count the number of items in a range of data:

1. Select the cell directly beneath the range of data.

Last Name	First Name	Banner ID	Amount Paid	Date Paid
Smith	John	745082	3000	9/8/2013
			2500	10/12/2013
			1500	12/10/2013
			2000	3/15/2015
			1000	6/10/2015

2. Select the *Auto Sum* dropdown.
3. Select *Count Numbers*.



`

4. Select the range of cells to calculate.
5. Click *Enter* on your keyboard.

Last Name	First Name	Banner ID	Amount Paid	Date Paid
Smith	John	745082	3000	9/8/2013
			2500	10/12/2013
			1500	12/10/2013
			2000	3/15/2015
			1000	6/10/2015
			5	

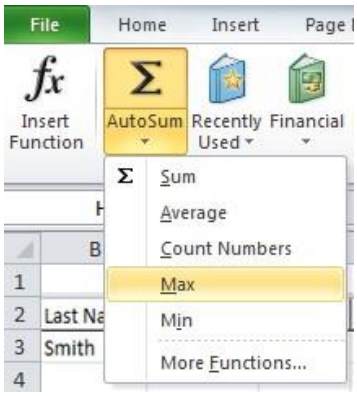
Maximum and Minimum

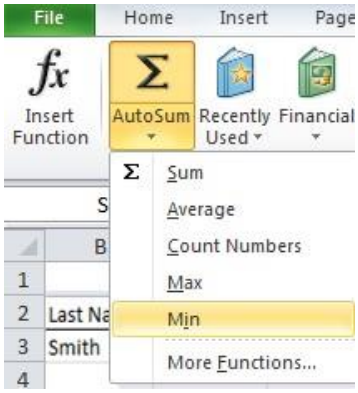
To calculate the Maximum or Minimum for a range of data:

1. Select the cell directly beneath the range of data.

Last Name	First Name	Banner ID	Amount Paid	Date Paid
Smith	John	745082	3000	9/8/2013
			2500	10/12/2013
			1500	12/10/2013
			2000	3/15/2015
			1000	6/10/2015

2. Select the *Auto Sum* dropdown.
3. Select *Max* or *Min* to calculate the maximum or minimum values





6. Select the range of cells to calculate.
7. Click *Enter* on your keyboard to calculate the value.

3) Graded Lab Tasks

Note: The instructor can design graded lab activities according to the level of difficulty and complexity of the solved lab activities. The lab tasks assigned by the instructor should be evaluated in the same lab.

Lab Task 1

The Sales Invoice

Use the template below and create your own invoice for purchased products. You can sell any product at any price, but you must include General sales tax, discount percentages and at least four items in your invoice. Do all formulas required. Format the invoice for clarity.

Item	Quantity	List Price	Discount	Your Price	Total
Lays	10	9	0.5	8.5	85
Subtotal					
G. Sales Tax(GST @17% of the price)					
Amount Due					

Lab Task 2

The CSC101 Grade Book

Recreate the table shown below. Determine the average for each student's final grade. Alphabetize the list. Using the IF statements, calculate the letter grade for each student whereby anything over 90 is an A, 80 is a B, 70 is a C, 60 is a D and anything lower is an F. Chart the final grades in a bar chart.

Name	Exam 1	Exam 2	Exam 3	Final	Final Grade	Letter Grade
Asad	80	88	87	94		
Amir	99	92	96	100		
Atique	85	99	82	95		
Ayesha	56	76	74	70		
Khurram	45	35	56	60		
Rizwan	100	90	95	100		
Nusrat	75	88	97	89		
Salman	90	90	85	89		

`

Lab Task 3

Class Make-up

Create a chart for each:

- *There are 28 students in a class. 10 are freshmen, 8 sophomores, 4 juniors, and 1 senior. 5 people did not answer the question;*
- *There are 11 men and 17 women – do a pie chart for this.*
- *If only 23 of the students did the homework assignment, what percent of the class is that? Another pie chart.*

LAB Task 4

Vacation Budget

Before you create the worksheet in Excel, explain where you want to go and what kind of activities you would like to do while you are there in a Word Document. What is your proposed budget? Set up your worksheet for transportation costs (airfare, car rental, trains, etc.), accommodations, food, sightseeing and shopping. Include any other activities you might enjoy. Include a 10% contingency plan for emergencies.

`

Lab 4

Introduction to MS Access

Objective:

This lab will introduce Microsoft Access database management system (DBMS) that combines the relational Microsoft Jet Database Engine with a graphical user interface and software-development tools.

Activity Outcomes:

The activities provide hands - on practice on

- How to create database
- How to create ghaphical view

Instructor Note:

As pre-lab activity, read “Microsoft” official site for detail guidelines.

1) Useful Concepts

Microsoft Access is a Database Management System (DBMS) from Microsoft that combines the relational Microsoft Jet Database Engine with a graphical user interface and software-development tools. It is a member of the Microsoft Office suite of applications, included in the professional and higher editions.

Microsoft Access is just one part of Microsoft’s overall data management product strategy.It stores data in its own format based on the Access Jet Database Engine.Like relational databases, Microsoft Access also allows you to link related information easily. For example, customer and order data. However, Access 2013 also complements other database products because it has several powerful connectivity features.

It can also import or link directly to data stored in other applications and databases. As its name implies, Access can work directly with data from other sources, including many popular PC database programs, with many SQL (Structured Query Language) databases on the desktop, on servers, on minicomputers, or on mainframes, and with data stored on Internet or intranet web servers.

Access can also understand and use a wide variety of other data formats, including many other database file structures.

You can export data to and import data from word processing files, spreadsheets, or database files directly.

Access can work with most popular databases that support the Open Database Connectivity (ODBC) standard, including SQL Server, Oracle, and DB2. Software developers can use Microsoft Access to develop application software.

`

2) Solved Lab Activites

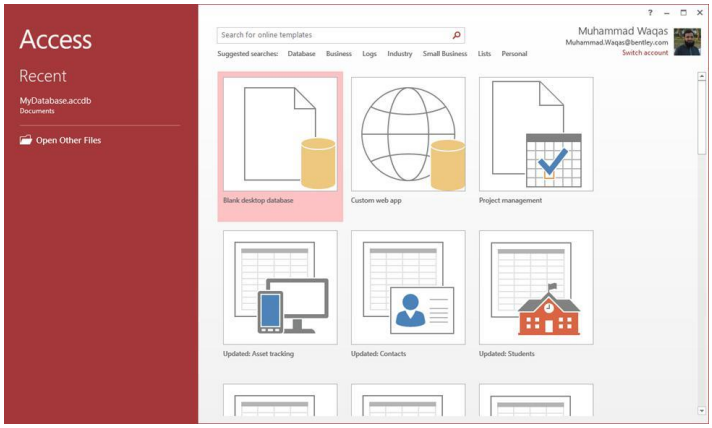
Sr.No	*Allocated Time*	*Level of Complexity*	*CLO Mapping*
1	*20*	*Low*	*CLO-6*
2	*40*	*Medium*	*CLO-6*

Activity 1:

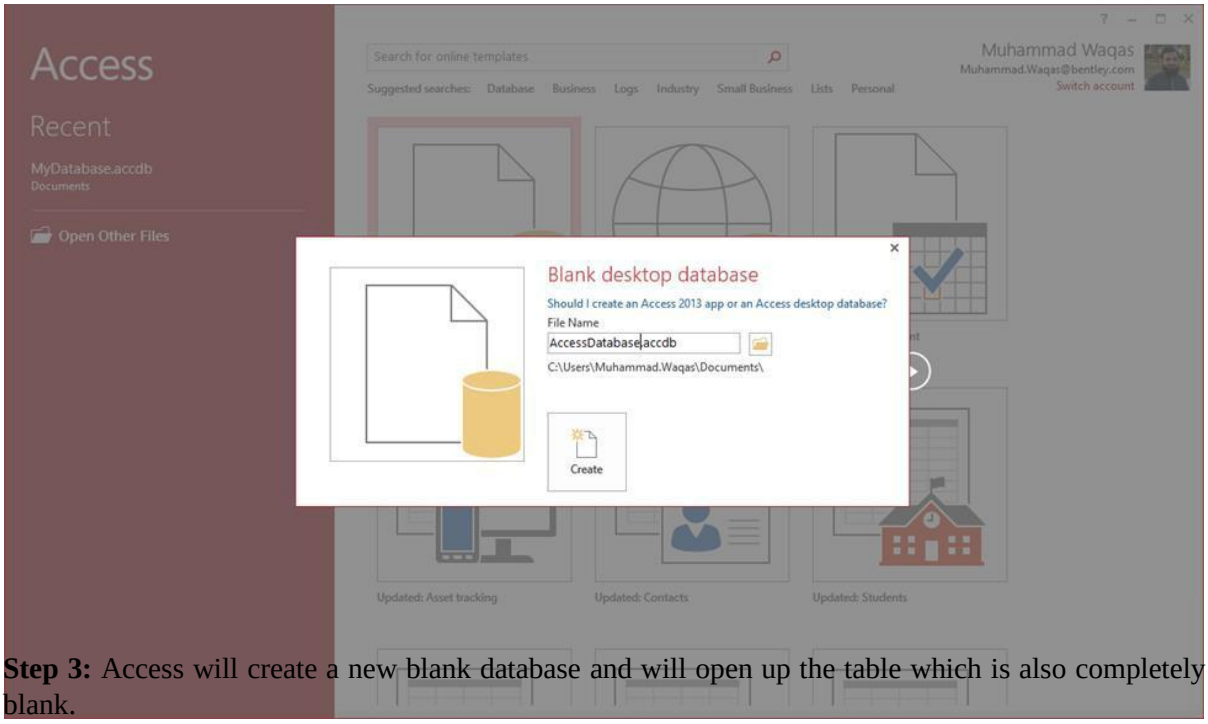
Create a Blank Database.

Solution:

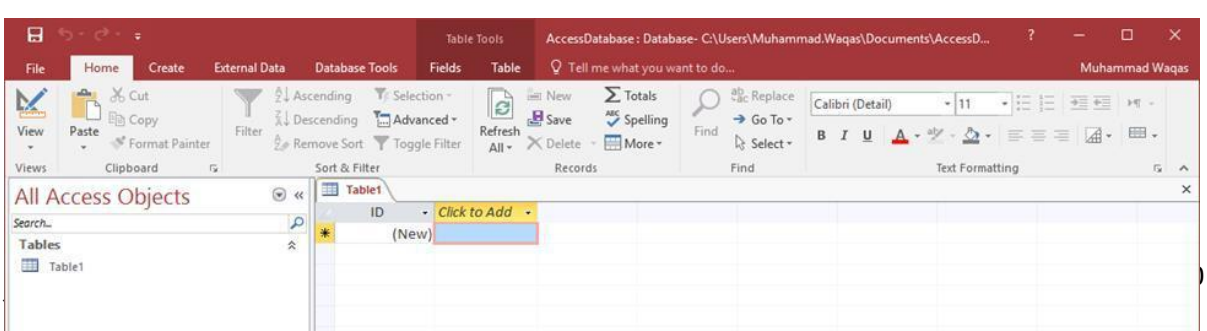
Step 1: Let us now start by opening MS Access.



Step 2: Select Blank desktop database. Enter the name and click the Create button.



Step 3: Access will create a new blank database and will open up the table which is also completely blank.



Activity 2:

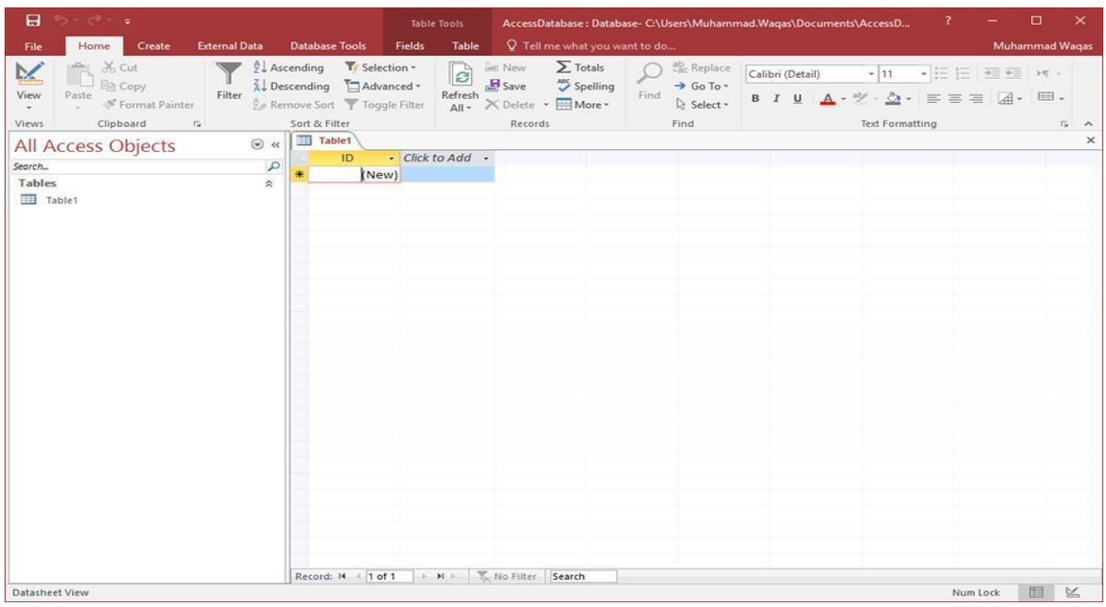
Creating Tables..

Solution:
When you create a database, you store your data in tables. Because other database objects depend so heavily on tables, you should always start your design of a database by creating all of its tables and then creating any other object. Before you create tables, carefully consider your requirements and determine all the tables that you need.

Let us try and create the first table that will store the basic contact information concerning the employees as shown in the following table:

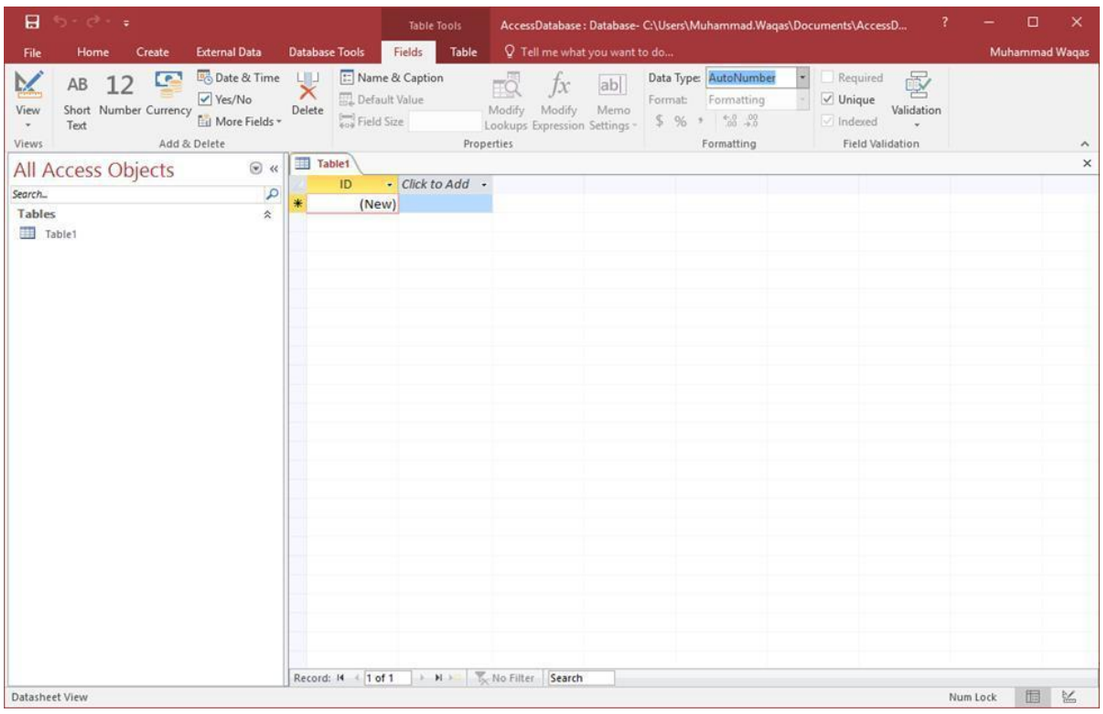
Field Name	Data Type
EmployeelD	AutoNumber
FirstName	Short Text
LastName	Short Text
Address1	Short Text
Address2	Short Text
City	Short Text
State	Short Text
Zip	Short Text
Phone	Short Text
PhoneType	Short Text

Let us now have short text as the data type for all these fields and open a blank database in Access.

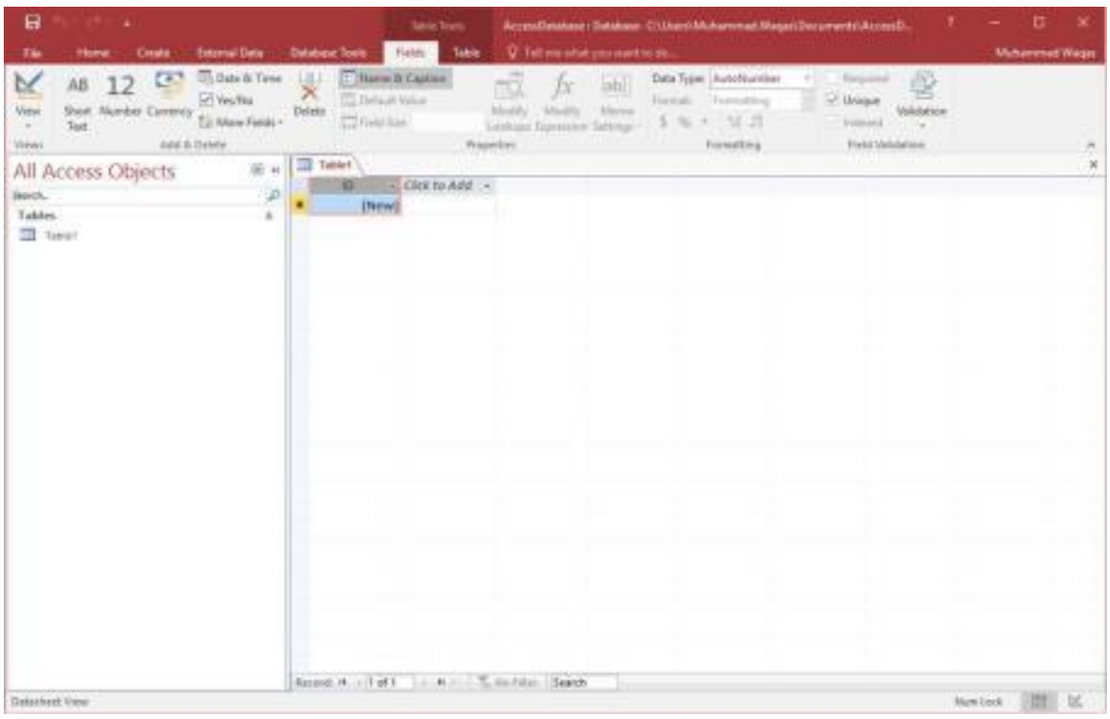


`

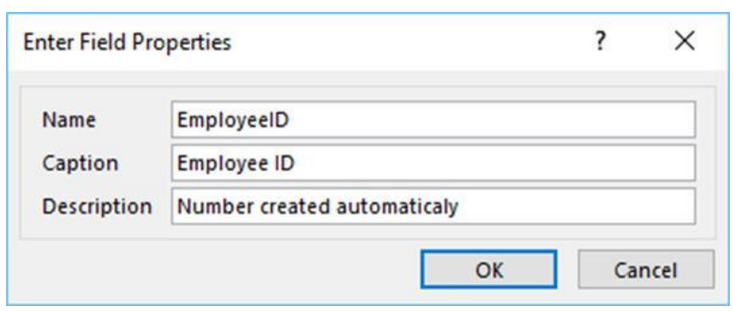
This is where we left things off. We created the database and then Access automatically opened up this table-one-datasheet view for a table.



Let us now go to the Field tab and you will see that it is also automatically created. The ID which is an AutoNumber field acts as our unique identifier and is the primary key for this table. The ID field has already been created and we now want to rename it to suit our conditions. This is an Employee table and this will be the unique identifier for our employees.

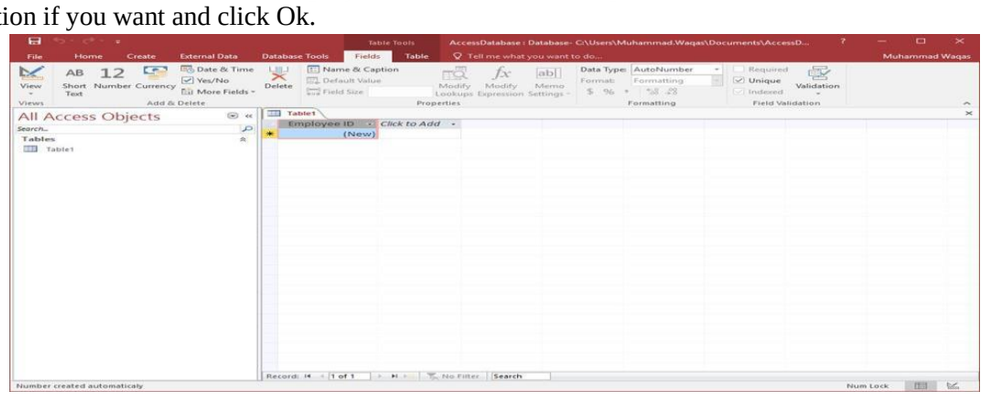


Click on the **Name & Caption** option in the Ribbon and you will see the following dialog box.



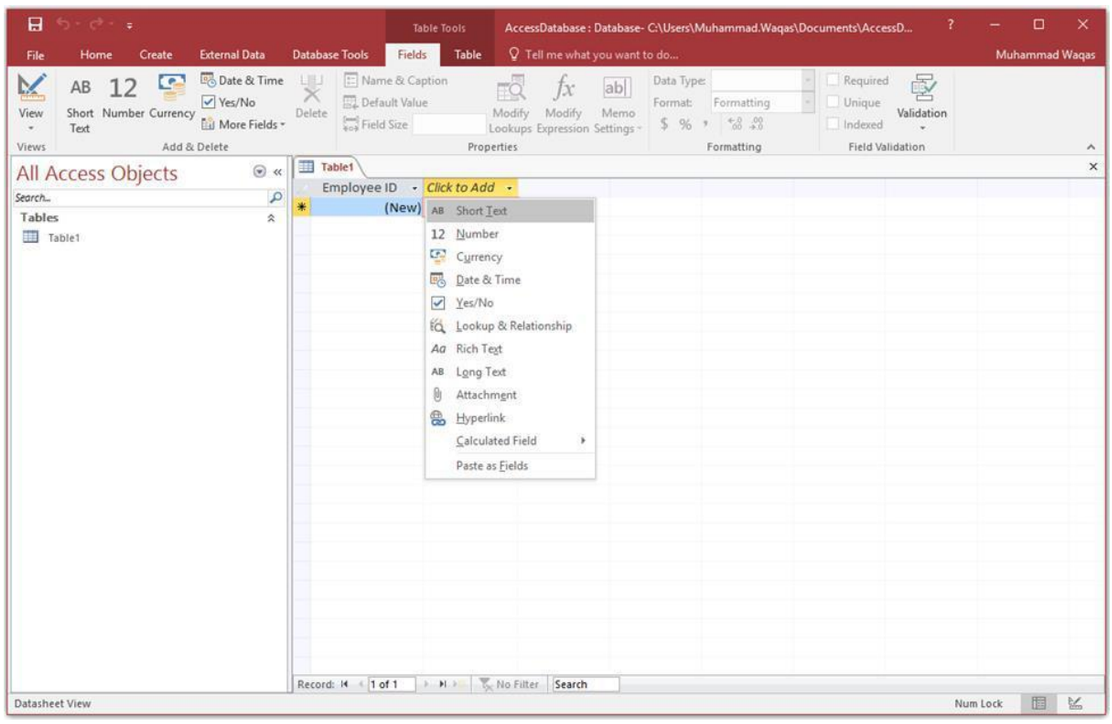
`

Change the name of this field to **EmployeeID** to make it more specific to this table. Enter the other optional information if you want and click Ok.

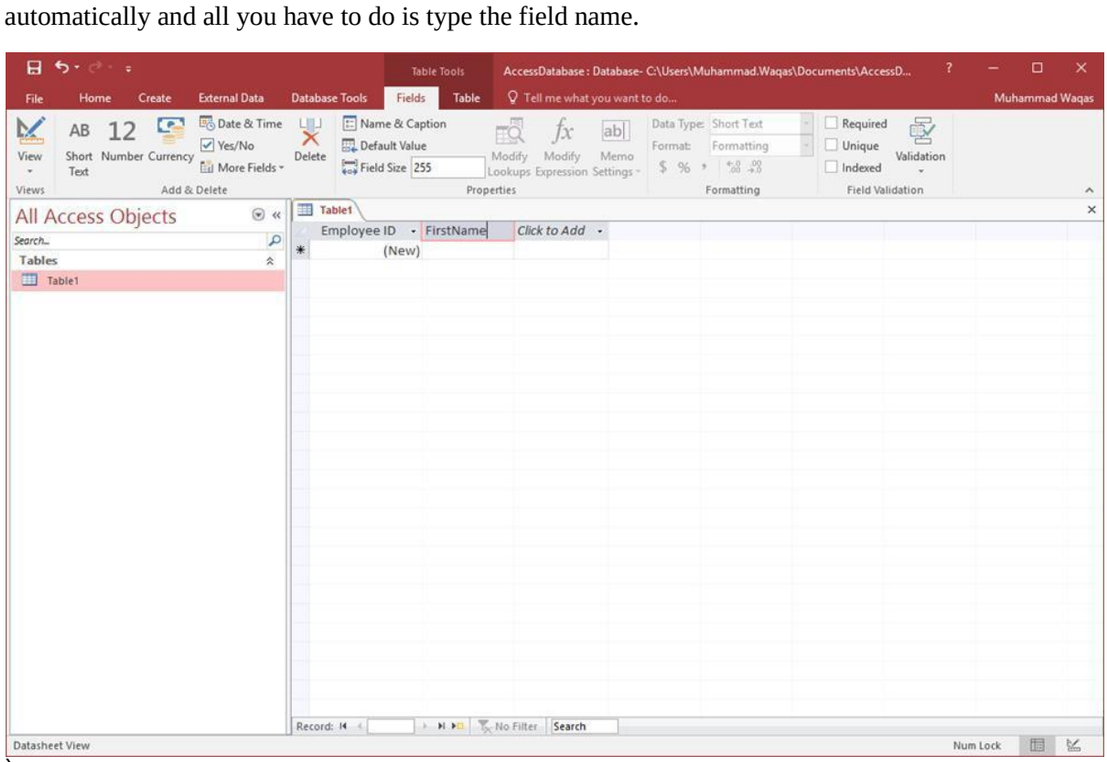


We now have our employee ID field with the caption Employee ID. This is automatically set to auto number so we don't really need to change the data type.

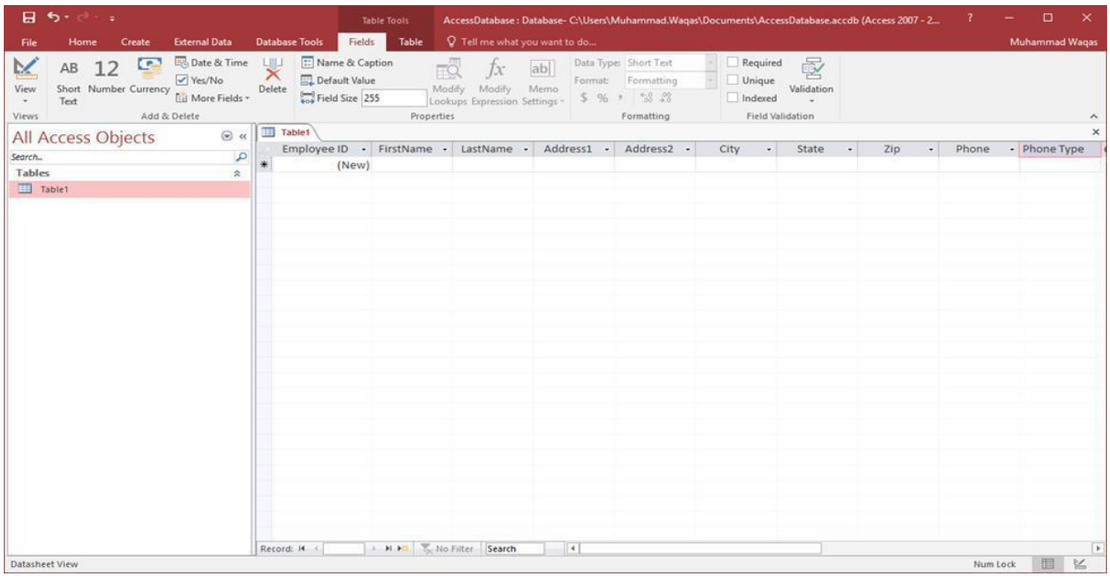
Let us now add some more fields by clicking on **click to add**.



Choose **Short Text** as the field. When you choose short text, Access will then highlight that field name automatically and all you have to do is type the field name.

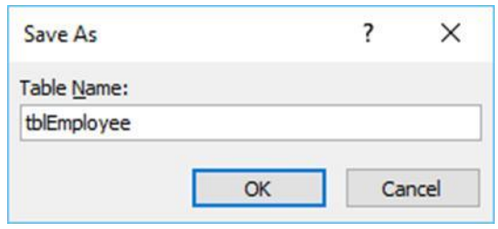


Type **FirstName** as the field name. Similarly, add all the required fields as shown in the following screenshot.

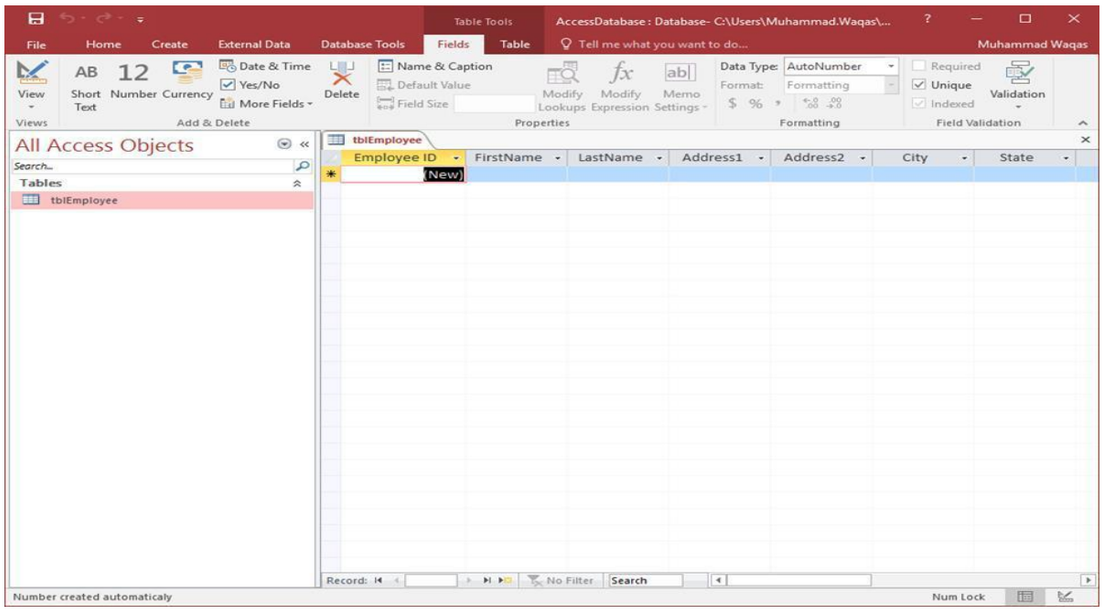


Once all the fields are added, click the Save icon.

You will now see the **Save As** dialog box, where you can enter a table name for the table.

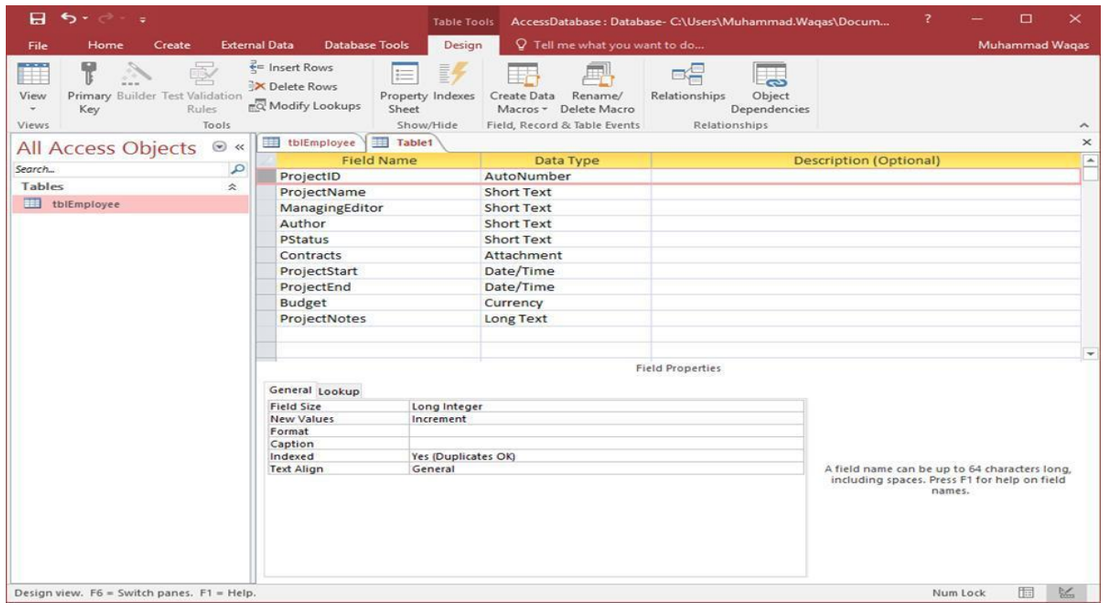


Enter the name of your table in the Table Name field. Here the **tbl** prefix stands for table.
Let us click Ok and you will see your table in the navigation pane.

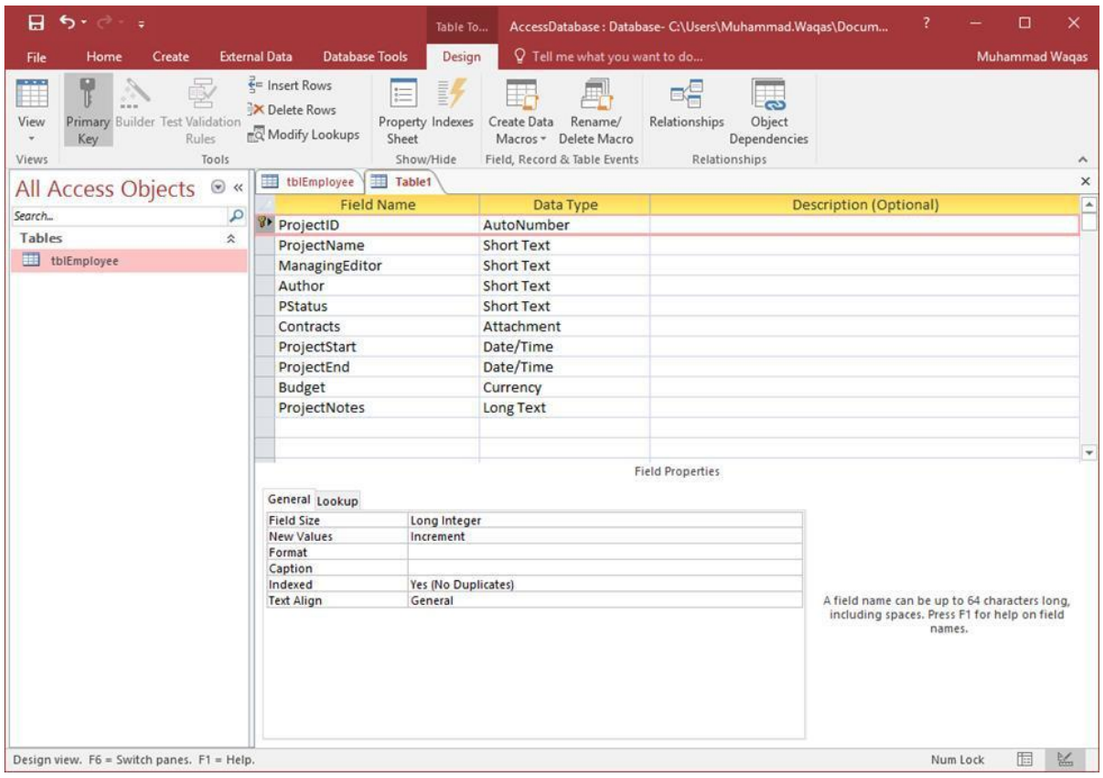


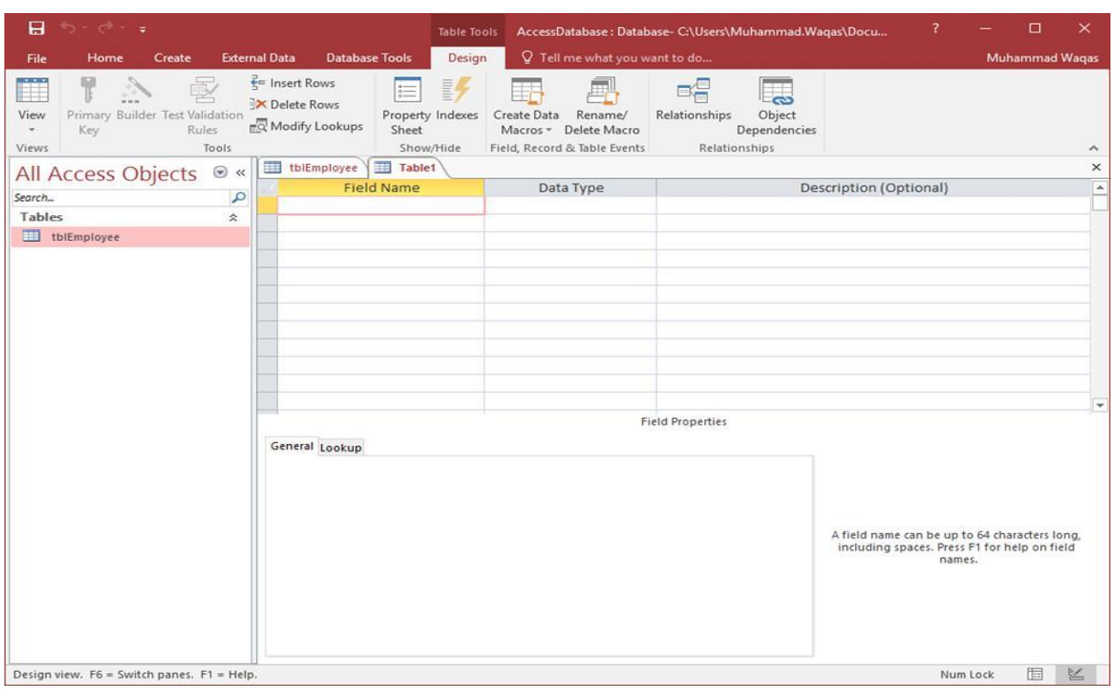
In the tables group, click on Table and you can see this looks completely different from the Datasheet View. In this view, you can see the **field name** and **data type** side by side.

`



We now need to make **ProjectID** a primary key for this table, so let us select **ProjectID** and click on **Primary Key** option in the ribbon.

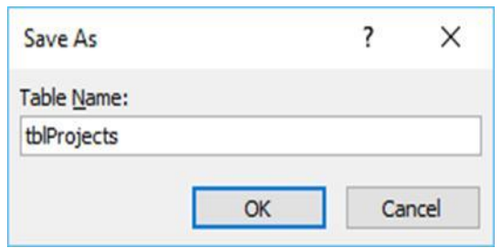




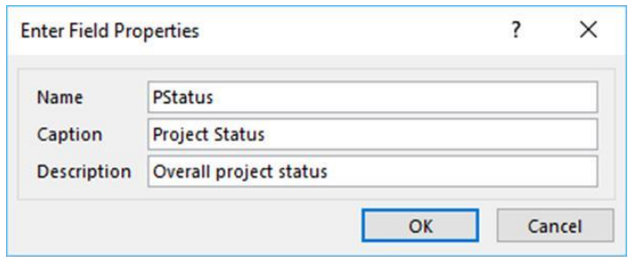
`

You can now see a little key icon that will show up next to that field. This shows that the field is part of the table's primary key.

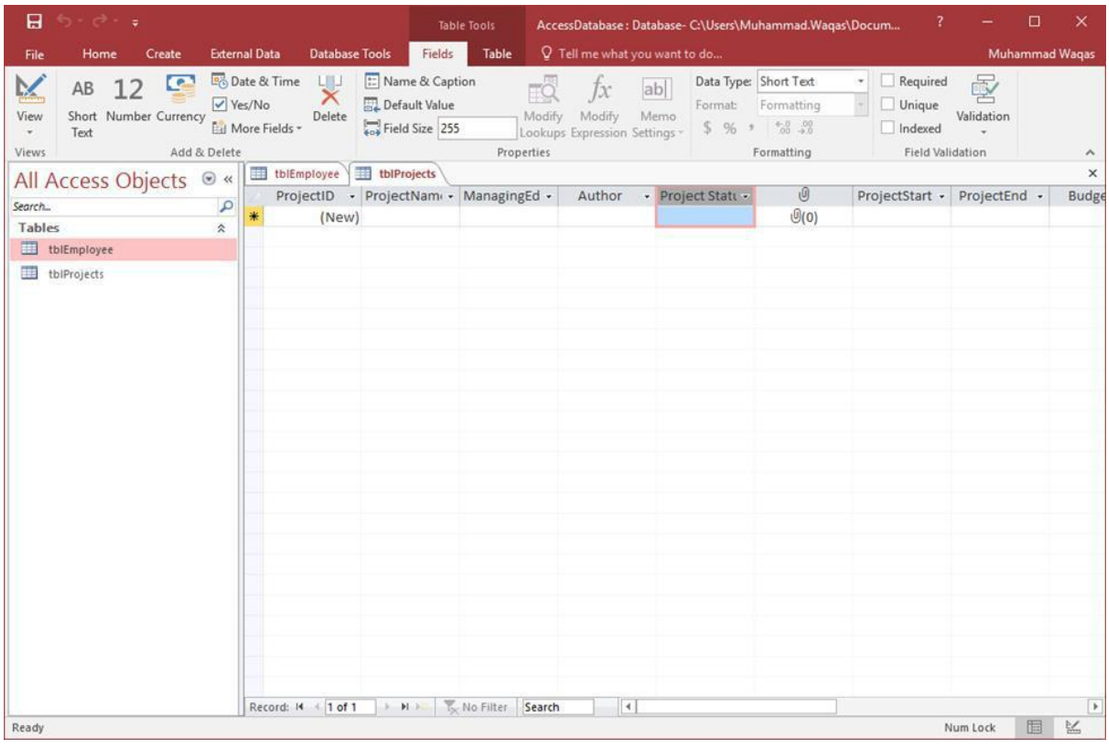
Let us save this table and give this table a name.



Click Ok and you can now see what this table looks like in the Datasheet View. If you ever want to make changes to this table or any specific field, you don't always have to go back to the Design View to change it. You can also change it from the Datasheet View. Let us update the PStatus field as shown in the following screenshot.



Click Ok and you will see the changes.



`

3) Graded Lab Task

Note: The instructor can design graded lab activities according to the level of difficulty and complexity of the solved lab activities. The lab tasks assigned by the instructor should be evaluated in the same lab.

Lab Task 1

Create database of your lab project. Create tables for each activity. Apply the following queries on the tables:

- *Insert data in the tables.*
- *Retriew data from the tables.*
- *Update the data.*
- *Create forms for all the activites.*

`

Lab 05

Intoduction to HTML

Objective:

The objective of this lab will be to learn about HTML and create web pages.

Activity Outcomes:

The activities provide hands - on practice with the following topics

- Basics of HTML
- Create own web pages.

Instructor Note:

As a pre-lab activity, read concepts of Html (Online).

1) Useful Concepts

HTML, or HyperText Markup Language, allows web users to create and structure sections, paragraphs, and links using elements, tags, and attributes. However, it's worth noting that HTML is not considered a programming language as it can't create dynamic functionality.

HTML has a lot of use cases, namely:

- **Web development**. Developers use HTML code to design how a browser displays web page elements, such as text, hyperlinks, and media files.
- **Internet navigation**. Users can easily navigate and insert links between related pages and websites as HTML is heavily used to embed hyperlinks.
- **Web documentation**. HTML makes it possible to organize and format documents, similarly to Microsoft Word.

2) Solved Lab Activities

Sr.No	***Allocated Time***	***Level of Complexity***	***CLO Mapping***
1	***1hr***	***Medium***	***CLO-6***

Activity 1:

Home page Development static pages (using Only HTML) of an online Book store. The website should consist the following pages.

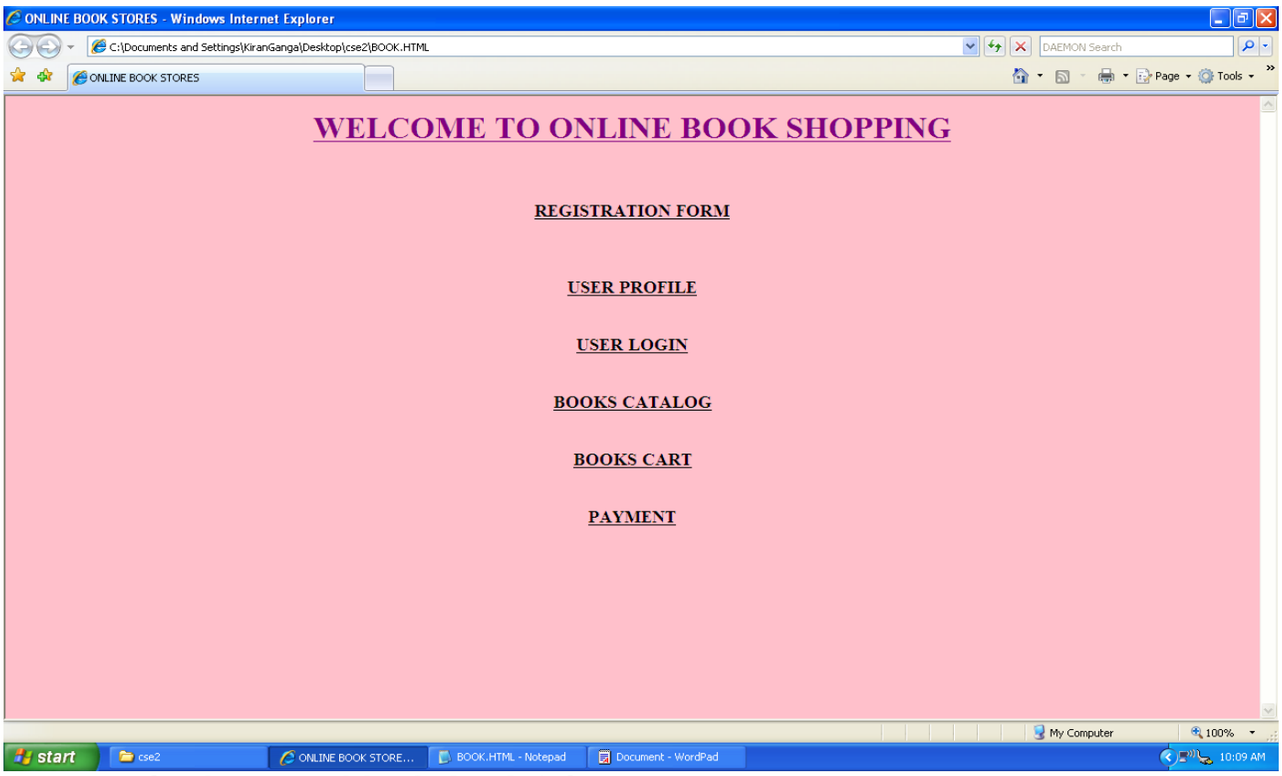
- ***Registration and user Login***
- ***User Profile Page***
- ***Books catalog***
- ***Shopping Cart***
- ***Payment By credit card***
- ***Order Conformation***

Solution:

Source Code for home page

```
<html>
<head>
```


`

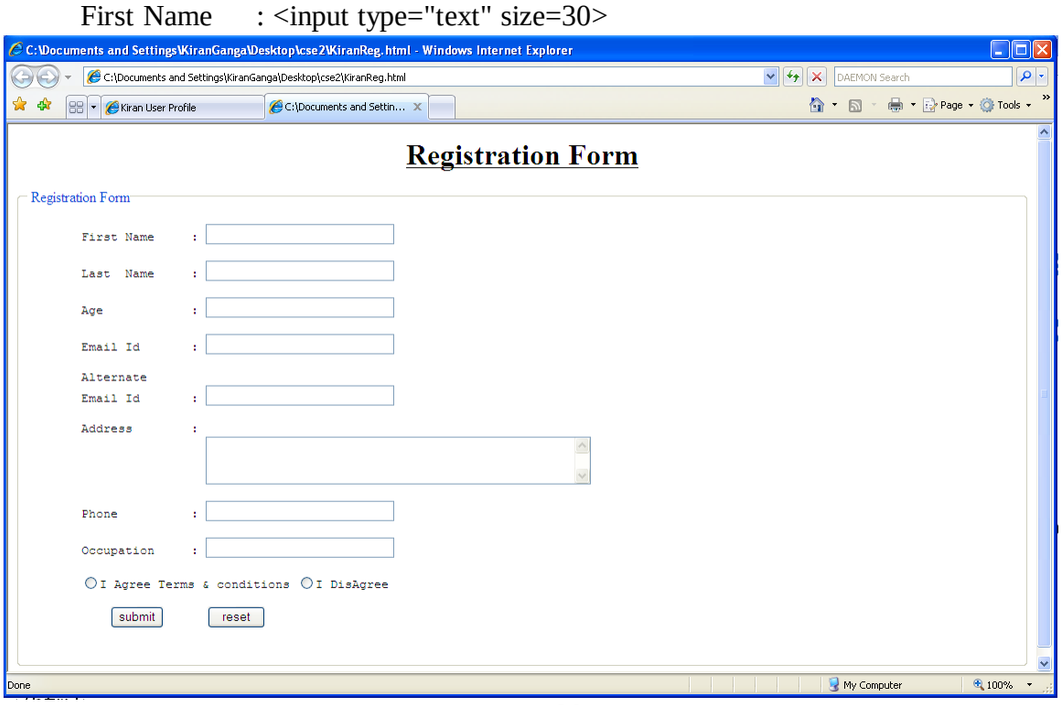



```
<title> ONLINE BOOK STORES</title>
</head>
<body bgcolor="pink">
<FRAMESET ROWS="20%,*">
<FRAME NAME="A2" SCROLLING="YES" SRC="head.html">

</FRAMESET><H1 ALIGN="CENTER"><U><FONT
COLOR="PURPLE">WELCOME TO ONLINE BOOK
SHOPPING<ITALIC></ITALIC>
</U></FONT></H1>
<H2> <FONT COLOR="WHITE"></FONT></H2>
<H3 ALIGN="CENTER"><A HREF="reg.html"><BR><BR>
<FONT COLOR="black"><ITALIC>REGISTRATION
FORM</FONT></ITALIC><BR><BR>
<BR><BR><A HREF="user profile .html"><FONT COLOR="black"><ITALIC>USER
PROFILE</FONT></ITALIC><BR>
<BR><BR><A HREF="user login.html"><FONT COLOR="black"><ITALIC>USER
LOGIN</FONT></ITALIC><BR>
<BR><BR><A HREF="book catalog.html"><FONT
COLOR="black"><ITALIC>BOOKS CATALOG</FONT></ITALIC><BR>
<BR><BR><A HREF="confim.HTML"><FONT COLOR="black"><ITALIC>BOOKS
CART</FONT></ITALIC><BR>
<BR><BR><A HREF="payment.HTML"><FONT
COLOR="black"><ITALIC>PAYMENT</H3></FONT></ITALIC><BR>
</BODY>
</HTML>
```


`

Source Code for Registration and user Login

```
<html>
<head>
 <center><u><h1>Registration
       Form</h1></u></center>
 </head>
 <body>
 <fieldset width=300><legend class=heading>Registration Form</legend>
 <form actions="http://localhost:8080//servlet-examples//KiranUserProfile">
 <pre class="reg">
       First Name     : <input type="text" size=30>
```



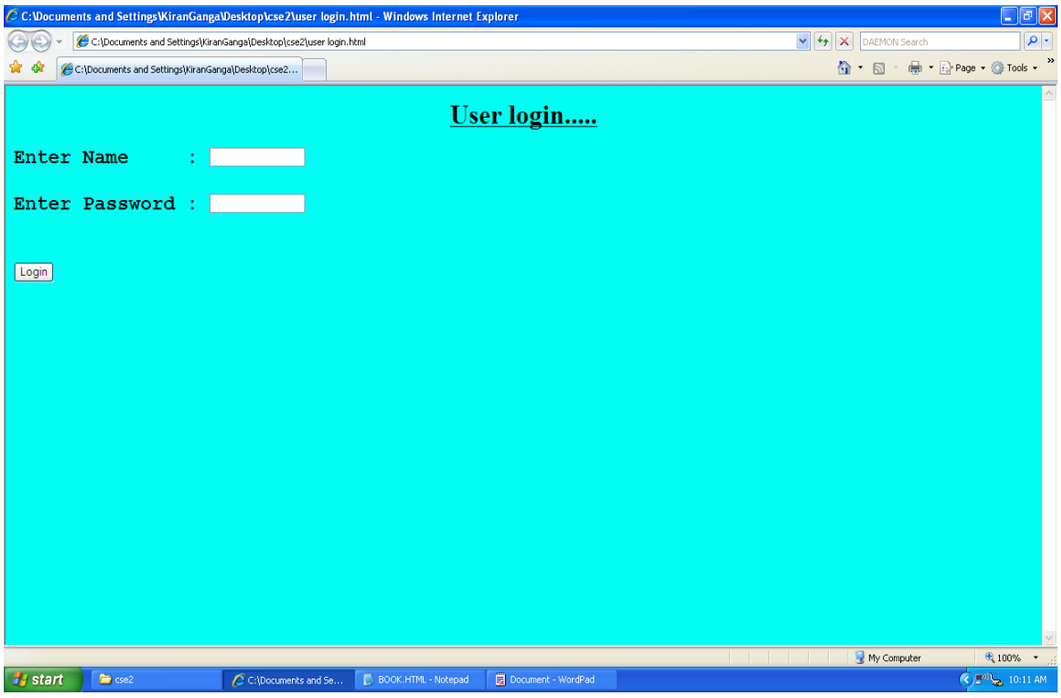

```
       Last Name     : <input type="text" size=30>
        Age          : <input type="text" size=30 onFocus="kirandisp()">
        Email Id     : <input type="email" size=30>
        Alternate
        Email Id     :  <input  type="email"  size=30>
        Address      :<textarea cols=50 rows=3></textarea>
        Phone              :  <input  type="text"  size=30>
        Occupation   : <input type="text" size=30>
 <input type="radio" name="agree" value="i agree">I Agree Terms & conditions  <input
type="radio" name="agree" value="i disagree">I DisAgree
 <input type="submit" value=submit>     <input type="Reset" value="  reset  ">

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

Source-Code for User-login

```
 <html>
 <head><center>
 <font size="6"  color="Black" ><h1><u>User login.....</u><h1></font></center>
 </head>
 <form method="POST" ACTION="http://localhost:8080//servlet-examples//KiranPass">
 <body  bgcolor="0x1ff0f2">
 <pre>
 Enter Name    : <input  type="text"  name="name"  size=15  >  <br>
 Enter Password : <input type="password" name="pwd" size=15><br>
 <input type= Submit name=sbmt value="Login">
```

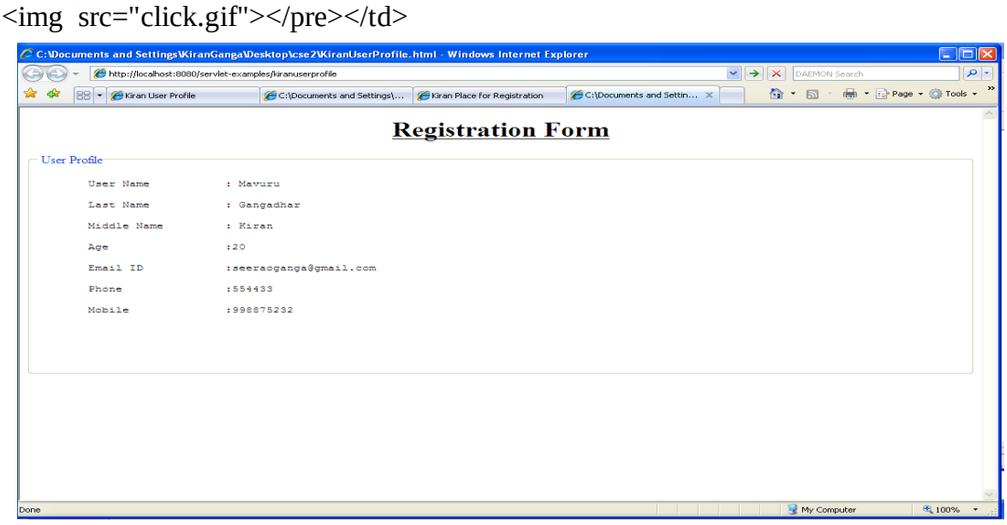

`

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



Source code for books catalog

```
<html>
<head>
</head>
<body>
<center><font color="red"  size=7 face="Vivaldi"><b><i>Book
Catlog</i></font></center></b>
<table>
<td><img src=java.jpg  width=100></td><td><b>Java Complete <br>references</b>
<ul><pre><li><font color=blue>Author          :Gangadhar</font>
<li><font color=blue>edition    :2007</font>
<li><font color=blue>price      :450</font>
<li><font color=blue>Publicher  :Ideal publishers</pre></font></pre>
</ul>
<td><pre><b>add    to    cart</b>
<img  src="click.gif"></pre></td>
</td>
<tr>
<td><img src="vbnet.gif" width=80></td><td><b>Visual Basics</b>
<ul><pre><li><font color=blue>Author          :Kiran</font>
<li><font color=blue>edition    :2007</font>
<li><font color=blue>price      :950</font>
<li><font color=blue>Publicher  :gangii publishers</pre></font></pre>
</ul>
<td><pre><b>add    to    cart</b>
<img  src="click.gif"></pre></td>
```



```
</tr>
<tr>
<td><img src="web tech.gif" width=80></td><td><b>Web Technology &
Designing</b>
<ul><pre><li><font color=blue>Author          :Kiran-Gangadhar</font>
<li><font color=blue>edition    :2007</font>
<li><font color=blue>price      :5950</font>
<li><font color=blue>Publicher  :jyothi publishers</pre></font></pre>
</ul>
<td><pre><b>add    to    cart</b>
<img  src="click.gif"></pre></td>
</tr>
<tr>
<td><img src="quick.gif" width=80></td><td><b>Web Technology & Designing</b>
<ul><pre><li><font color=blue>Author          :Kiran-Gangadhar</font>
<li><font color=blue>edition    :2007</font>
<li><font color=blue>price      :5950</font>
<li><font color=blue>Publicher  :jyothi publishers</pre></font></pre>
</ul>
<td><pre><b>add    to    cart</b>
<img  src="click.gif"></pre></td>
</tr>
<tr>
<td><img src="harry.jpg" width=80></td><td><b>Harry reference to java</b>
<ul><pre><li><font color=blue>Author          :Harry Potter</font>
<li><font color=blue>edition    :2007</font>
<li><font color=blue>price      :950</font>
<li><font color=blue>Publicher  :jyothi publishers</pre></font></pre>
</ul>
<td><pre><b>add    to    cart</b>
<img  src="click.gif"></pre></td>
</tr>
</table>
</body>
</html>
```

`

3) Graded Lab Tasks

Note: The instructor can design graded lab activities according to the level of difficulty and complexity of the solved lab activities. The lab tasks assigned by the instructor should be evaluated in the same lab.

Lab Task 1

Create your own website for Hosptal Management system.The site may includes the following pages.

- *Doctor Login*
- *User/patient Login*
- *Appointment Resevation*
- *Patient History*
- *Doctor profile.*

Lab 06

Problem Solving with Sequential Structure

Objective:

This lab will provide a hands-on experience of Raptor. RAPTOR is a visual programming development environment based on flowcharts. A flowchart is a collection of connected graphic symbols, where each symbol represents a specific type of instruction to be executed. The connections between symbols determine the order in which instructions are executed. In this lab we will be particularly learning the sequential flow of the instructions that is also the default flow of execution.

Activity Outcomes:

The lab will teach students to represent a specific problem in terms of a flowchart using Raptor.

The students will be able to:

- Understand problems and develop a flowchart in raptor for those problems
- Gain an insight of sequential program flow in raptor
- Insert data into a program
- Process data in raptor
- Show output in raptor

Instructor Note:

As a pre-lab activity, read "An Introduction to Programming and Algorithmic Reasoning using RAPTOR" by Hadfield, Weingart and Brown to gain an insight about flowchart representation in raptor.

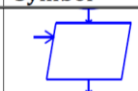
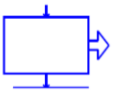
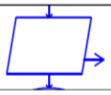
1) Useful concepts:

RAPTOR is a flowchart-based programming environment, designed specifically to help students visualize their algorithms and avoid syntactic baggage. RAPTOR programs are created visually and executed visually by tracing the execution through the flowchart. Required syntax is kept to a minimum. Students prefer using flowcharts to express their algorithms, and are more successful creating algorithms using RAPTOR than using a traditional language or writing flowcharts without RAPTOR.

- The RAPTOR development environment minimizes the amount of syntax you must learn to write correct program instructions.
- The RAPTOR development environment is visual. RAPTOR programs are diagrams (directed graphs) that can be executed one symbol at a time. This will help you follow the flow of instruction execution in RAPTOR programs.
- RAPTOR is designed for ease of use. Other programming development environments are extremely complex.
- RAPTOR error messages are designed to be more readily understandable by beginning programmers.
- The major goal is to teach students how to design and execute algorithms. Also, familarise them wih the sequential flow of execution.

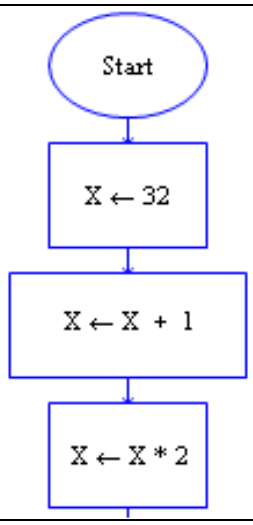
1. Raptor Symbols and Statements:
   RAPTOR has six (6) basic symbols, where each symbol represents a unique type of instruction. The top four statement types, Assignment, Call, Input, and Output, are explained in this reading, The bottom two types, Selection and Loops, will be explained in a future reading.
   - The typical computer program has three basic components:
     - INPUT – get the data values that are needed to accomplish the task.
     - PROCESSING – manipulate the data values to accomplish the task.
     - OUTPUT – display (or save) the values which provide a solution to the task.

   These three components have a direct correlation to RAPTOR instructions as shown in the following table.

Purpose	Symbol	Name	Description
INPUT		input statement	Allow the user to enter data. Each data value is stored in a ***variable***.
PROCESSING		assignment statement	Change the value of a ***variable*** using some type of mathematical calculation.
PROCESSING		procedure call	Execute a group of instructions defined in the named procedure. In some cases some of the procedure arguments (i.e., ***variables***) will be changed by the procedure's instructions.
OUTPUT		output statement	Display (or save to a file) the value of a ***variable***.

The common thread among these four instructions is that they all do something to variables! To understand how to develop algorithms into working computer programs, you must understand the concept of a variable. Please study the next section carefully!

2. Raptor Variables:
Variables are computer memory locations that hold a data value. At any given time a variable can only hold a single value. However, the value of a variable can vary (change) as a program executes. That's why we call them "variables"! As an example, study the following table that traces the value of a variable called X.

Description	Value of X	Flowchart
• When the program begins, no variables exist. In RAPTOR, variables are automatically created when they are first used in a statement. • The first assignment statement, X←32, assigns the data value 32 to the variable X. • The next assignment statement, X←X+1, retrieves the current value of X, 32, adds 1 to it, and puts the result, 33, in the variable X. • The next assignment statement, X←X*2, retrieves the current value of X, 33, multiplies it by 2, and puts the result, 66, in the variable X.	Undefinied 32 33 66	

During the execution of the previous example program, the variable X stored three distinct values. Please note that the order of statements in a program is very important. If you re-ordered these three assignment statements, the values stored into X would be different.

A variable can have its value set (or changed) in one of three ways:
- By the value entered from an input statement.
- By the value calculated from an equation in an assignment statement.
- By a return value from a procedure call (more on this later).

It is variables, and their changing data values, that enable a program to act differently every time it is executed.

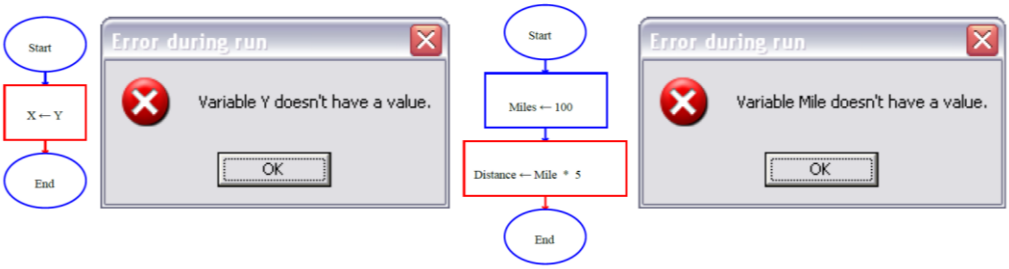
All variables should be given meaningful and descriptive names by the programmer. Variable names should relate to the purpose the variable serves in your program. A variable name must start with a letter and can contain only letters, numerical digits, and underscores (but no spaces or other special characters). If a variable name contains multiple "words," the name is more "readable" if each word is separated by an underscore character. The table below shows some examples of good, poor, and illegal variable names.

Good variable names	Poor variable names	Illegal variable names
tax_rate	a (not descriptive)	4sale (does not start with a letter)
sales_tax	milesperhour (add underscores)	sales tax (includes a space)
distance_in_miles	my4to (not descriptive)	sales$ (includes invalid character)
mpg		

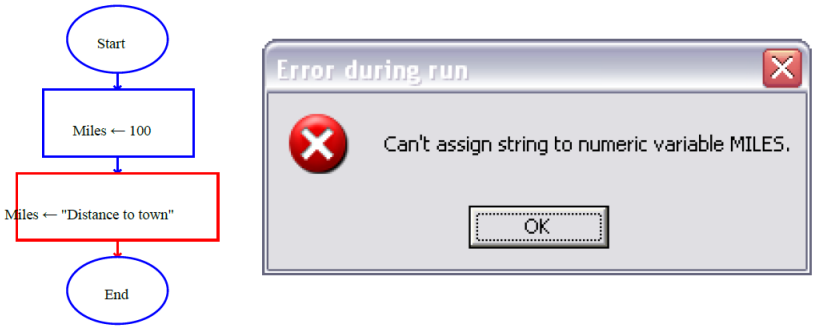
IMPORTANT: If you give each value in a program a meaningful, descriptive variable name, it will help you think more clearly about the problem you are solving and it will help you find errors in your program.
One way of understanding the purpose of variables is to think of them as a means to communicate information between one part of a program and another. By using the same variable name in different parts of your program you are using the value that is stored at that location in different parts of your program. Think of the variable as a place holder or storage area for values between each use in your program computations.
When a RAPTOR program begins execution, no variables exist. The first time RAPTOR encounters a new variable name, it automatically creates a new memory location and associates this variable name with the new memory. The variable will exist from that point in the program execution until the program terminates. When a new variable is created, its initial value determines whether the variable will store numerical data or textual data. This is called the variable's data type. A variable's data type cannot change during the execution of a program. In summary, variables are automatically created by RAPTOR and can hold either:
• Numbers e.g., 12, 567, -4, 3.1415, 0.000371, or
• Strings e.g., “Hello, how are you?”, “James Bond”, “The value of x is ”

Common errors when using variables:
Error 1: "Variable ____ does not have a value"
There are two common reasons for this error:

1. The variable has not been given a value.
2. The variable name was misspelled.



Error 2: "Can't assign string to numeric variable _____" or
"Can't assign numeric to string variable _____"
This error will occur if your statements attempt to change the data type of a variable.



2) Solved Lab Activites

Sr.No	*Allocated Time*	*Level of Complexity*	*CLO Mapping*
1	*10*	*Low*	*CLO-6*
2	*10*	*Low*	*CLO-6*
3	*10*	*Low*	*CLO-6*
4	*15*	*Medium*	*CLO-6*
5	*15*	*Medium*	*CLO-6*

Activity 1:

Understanding Program Structure

Solution:

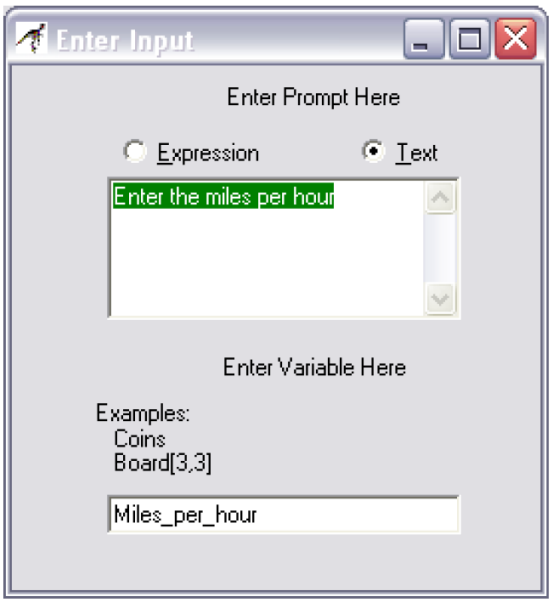
A RAPTOR program is a set of connected symbols that represent actions to be performed. The arrows that connect the symbols determine the order in which the actions are performed. When executing a RAPTOR program, you begin at the **Start** symbol and follow the arrows to execute the program. A RAPTOR program stops executing when the **End** symbol is reached. The smallest RAPTOR program (which does nothing) is depicted at the right. By placing additional RAPTOR statements between the **Start** and **End** symbols you can create meaningful RAPTOR programs

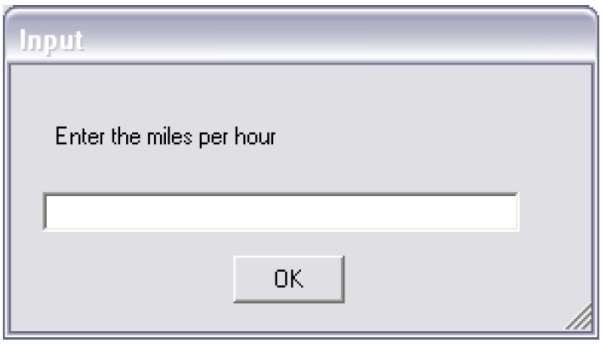
Activity 2:

Getting Input from user

Solution:

An input statement/symbol allows the user of a program to enter a data value into a program variable during program execution. It is important that a user know exactly what type of value is expected for input. Therefore, when you define an input statement you specify a string of text that will be the prompt that describes the required input. The prompt should be as explicit as possible. If the expected value needs to be in particular units (e.g., feet, meters, or miles) you should mention the units in the prompt.





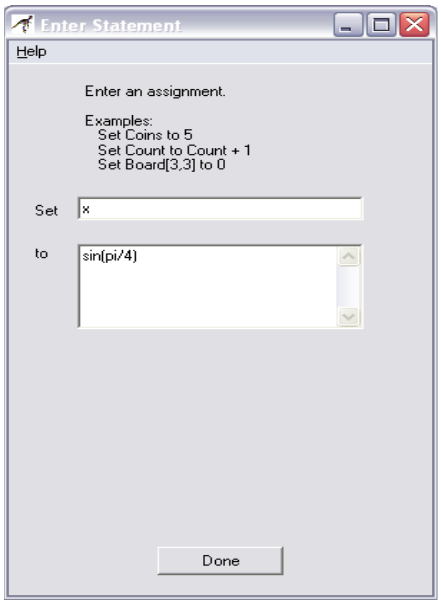
When you define an input statement, you must specify two things: a prompt and the variable that will be assigned the value enter by the user at run-time. As you can see by the "Enter Input" dialog box at the right there are two types of input prompts: **Text** and **Expression** prompts. An Expression prompt enables you to mix text and variables together like the following prompt:

```
"Enter a number between " + low + " and " + high + ":".
```

At run-time, an input statement will display an input dialog box, an example of which is shown to the right. After a user enters a value and hits the enter key (or clicks OK), the value entered by the user is assigned to the input statement's variable.
Make sure you distinguish between the "**definition** of a statement" and the "**execution** of a statement". The dialog box that is used to define a statement is totally different from the dialog box that is used at run-time when a program is executing.

Activity 3:

Assigning value to a Variable:

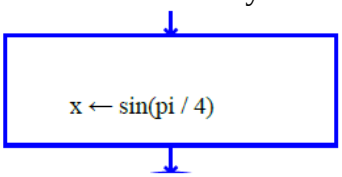


Solution:

The assignment symbol is used to perform a computation and then store the results in a variable. The definition of an assignment statement is performed using the dialog box shown on the right. The variable to be assigned a value is entering into the "Set" field, and the computation to perform is enter into the "to" field. The example on the right sets the value of the variable x to 0.707106781186547.
An assignment statement is displayed inside its RAPTOR symbol using the syntax:

```
Variable ← Expression
```

For example, the statement created by the dialog box to the right is displayed as:



One assignment statement can only change the value of a single variable, that is, the variable on the left hand side of the arrow. If this variable did not exist prior to the statement, a new variable is created. If this variable did exist prior to the statement, then its previous value is lost and its new value is based on the computation that is performed. No variables on the right hand side of the arrow (i.e., the expression) are ever changed by the assignment statement.

Activity 4:

Evaluating Expressions

Solution:

The expression (or computation) of an assignment statement can be any simple or complex equation that computes a single value. An expression is a combination of values (either constants or variables) and operators. Please carefully study the following rules for constructing valid expressions.

A computer can only perform one operation at a time. When an expression is computed, the operations of the equation are not executed from left to right in the order that you typed them in. Rather, the operations are performed based on a predefined "order of precedence." The order that operations are performed can make a radical difference in the value that is computed. For example, consider the following two examples:

x ← (3+9)/3 x ← 3+(9/3)

In the first case, the variable x is assigned a value of 4, whereas in the second case, the variable x is assigned the value of 6. As you can see from these examples, you can always explicitly control the order in which operations are performed by grouping values and operators in parenthesis. The exact "order of precedence" is

1. compute all functions, then
2. compute anything in parentheses, then
3. compute exponentiation (^,**) i.e., raise one number to a power, then
4. compute multiplications and divisions, left to right, and finally
5. compute additions and subtractions, left to right.

An operator or function directs the computer to perform some computation on data. Operators are placed between the data being operated on (e.g. X/3) whereas functions use parentheses to indicate the data they are operating on (e.g. sqrt(4.7)). When executed, operators and functions perform their computation and return their result. The following lists summarize the built-in operators and functions of RAPTOR.

basic math: `+`, `-`, `*`, `/`, `^`, `**`, `rem`, `mod`, `sqrt`, `log`, `abs`, `ceiling`, `floor`

trigonometry: `sin`, `cos`, `tan`, `cot`, `arcsin`, `arcos`, `arctan`, `arccot`

miscellaneous: `random`, `Length_of`

The following table briefly describes these built-in operators and functions. Full details concerning these operators and functions can be found in the RAPTOR help screens.

Operation	Description	Example
+	addition	3+4 is 7
-	subtraction	3-4 is -1
-	negation	-3 is a negative 3
*	multiplication	3*4 is 12
/	division	3/4 is 0.75
^ **	exponentiation, raise a number to a power	3^4 is 3*3*3*3=81 3**4 is 81
rem mod	remainder (what is left over) when the right operand divides the left operand	10 rem 3 is 1 10 mod 4 is 2
sqrt	square root	sqrt(4) is 2
log	natural logarithm (base e)	log(e) is 1
abs	absolute value	abs(-9) is 9
ceiling	rounds up to a whole number	ceiling(3.14159) is 4
floor	rounds down to a whole number	floor(9.82) is 9
sin	trig sin(angle_in_radians)	sin(pi/6) is 0.5
cos	trig cos(angle_in_radians)	cos(pi/3) is 0.5
tan	trig tan(angle_in_radians)	tan(pi/4) is 1.0
cot	trig cotangent(angle_in_radians)	cot(pi/4) is 1
arcsin	trig $\sin^{-1}$(expression), returns radians	arcsin(0.5) is pi/6
arcos	trig $\cos^{-1}$(expression), returns radians	arccos(0.5) is pi/3
arctan	trig $\tan^{-1}$(y,x), returns radians	arctan(10,3) is 1.2793
arccot	trig $\cot^{-1}$(x,y), returns radians	arccot(10,3) is 0.29145
random	generates a random value in the range [1.0, 0.0)	random * 100 is some value between 0 and 99.9999
Length_of	returns the number of characters in a string variable	Example ← "Sell now" Length_of(Example) is 8

The result of evaluating of an expression in an assignment statement must be either a single number or a single string of text. Most of your expressions will compute numbers, but you can also perform simple text manipulation by using a plus sign (+) to join two or more strings of text into a single string. You can also join numerical values with strings to create a single string. The following example assignment statements demonstrate string manipulation.

```
Full_name ← "Joe " + "Alexander " + "Smith"
Answer ← "The average is " + (Total / Number)
```

RAPTOR defines several symbols that represent commonly used *constants*. You should use these constant symbols when you need their corresponding values in computations.

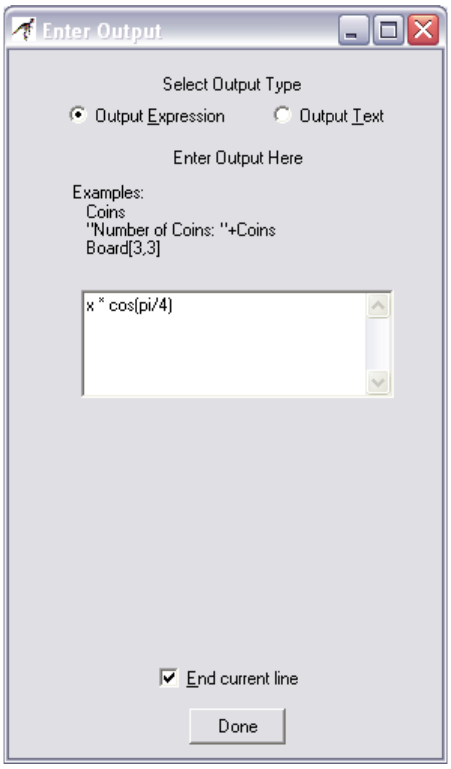
pi is defined to be 3.14159274101257.

e is defined to be 2.71828174591064.

Activity 5:

Showing Output

Solution:



Master Console window when it is executed. When you define an output statement, the "Enter Output" dialog box asks you to specify three things:

- Are you displaying text, or the results of an expression (computation)?
- What is the text or expression to display?
- Should the output be terminated by a new line character?

The example output statement on the right will display the text, "The sales tax is" on the output window and terminate the text with a new line. Since the "End current line" is checked, any future output will start on a new line below the displayed text.

When you select the "Output Text" option, the characters that you type into the edit box will be displayed exactly as you typed them, including any leading or trailing spaces. If you include quote marks (") in the text, the quote marks will be displayed exactly as you typed them.

When you select the "Output Expression" option, the text you type into the edit box is treated as an expression to be evaluated. When the output statement is executed at run-time, the expression is evaluated and the resulting single value that was computed is displayed. An example output statement that displays the results of an expression is shown on the right.

You can display multiple values with a single output statement by using the "Output Expression" option and building a string of text using the string plus (+) operator. When you build a single string from two or more values, you must distinguish the text from the values to be calculated by enclosing any text in quote marks ("). In such cases, the quote marks are not displayed in the output window. For example, the expression,

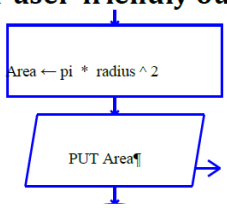
```
"Active Point = (" + x + "," + y + ")"
```

will display the following if x is 200 and y is 5:
Active Point = (200,5)

Notice that the quote marks are not displayed on the output device. The quote marks are used to surround any text that is not part of an expression to be evaluated.

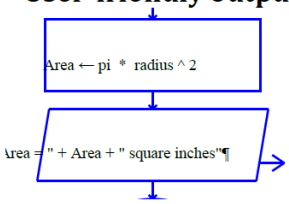
Your instructor (or a homework assignment) will often say “Display the results in a user-friendly manner”. This means you should display some explanatory text explaining any numbers that are output to the MasterConsole window. An example of "non-user-friendly output" and "user-friendly output" is shown below.

Non-user-friendly output



Example output: 2.5678

User-friendly output



Example output: Area = 2.5678 square inches

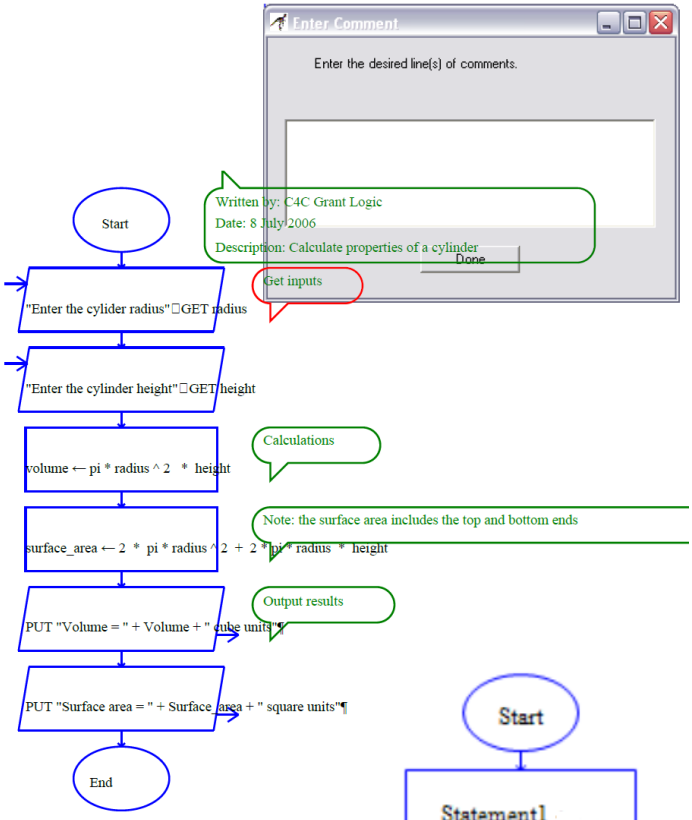
Activity 6:

Adding Comments in Raptor

Solution:

The RAPTOR development environment, like many other programming languages, allows *comments* to be added to your program. Comments are used to explain some aspect of a program to a human reader, especially in places where the program code is complex and hard to understand. Comments mean nothing to the computer and are not executed. However, if comments are done well, they can make a program much easier to understand for a human reader.

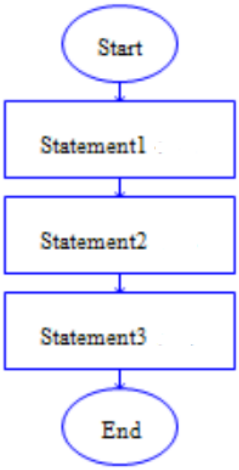
To add a comment to a statement, right-click your mouse over the statement symbol and select the "Comment" line before releasing the mouse button. Then enter the comment text into the "Enter Comment" dialog box, an example of which is shown to the right. The resulting comment can be moved in the RAPTOR window by dragging it, but you typically do not need to move the default location of a comment.



There are three general types of comments:

- Programmer header – documents who wrote the program, when it was written, and a general description of what the program does. (Add to the "Start" symbol)
- Section description – mark major sections of your program to make it easier for a programmer to understand the overall program structure.
- Logic description – explain non-standard logic.

Typically you should **not** comment every statement in a program. An example program that includes comments is shown below.



All the activities we have performed so far, we have done sequential execution of the instructions. You have controlled which statement is executed next by

ordering them one after the other. Essentially you placed each statement in the order that you wanted them to be executed. The first statement of a program that is executed is the first statement after the Start statement. Once that statement is finished executing (i.e. the semantics associated with that statement have been accomplished), then the statement immediately following that statement is executed. Once the second statement is executed, the third is executed, and then the fourth, and so on until the End is reached. As you can see by the inset diagram, the arrows linking the statements depict the execution flow. If you have 20 programming statements in the instruction section, then when your program runs successfully, it will execute those 20 statements in order and then quit.

Sequential Control	Used to execute statements, one after the other, in the order in which they appear in the program.

Notice that you, as a programmer, have total control over which statements are executed before others, merely by your placement of those instructions relative to each other in the instruction sequence. It is your job as a programmer to determine the statement that is needed and its placement. Writing the correct statement is one task. Determining where to place that statement is equally important. As an example, when you want to get and process data from the user you have to `GET` the data before you can use it. Switching the order of these statements means that your program is trying to use data that it hasn't gotten yet.

Sequential control is "free," in the sense that you accomplish sequential control just by placing statements in the order you want them to be executed. Now, perform the following activities using the sequential control structure.

3) Graded Lab Tasks:

Note: The instructor can design graded lab activities according to the level of difficult and complexity of the solved lab activities. The lab tasks assigned by the instructor should be evaluated in the same lab.

Lab Task 1

A painter wants to know the amount of paint needed to paint only the walls and the interior side of the door in a room. The chosen paint covers 100 square feet per gallon. There are two windows. Test the problem with the following data:
The room is 12 feet long, 10 feet wide, and 8 feet tall.
The two windows are 5 by 3 feet, and 6 by 2 feet, respectively.

Lab Task 2

One of the jobs that Joe Roberts has been given at work is to order special paper for a report for a board meeting. The paper comes in reams of 500 sheets. He always makes five more copies than the number of people that will be there. Joe wants to know how many reams of paper he needs for a meeting. He can order only whole, not partial, reams. Assume the required number of pages will not equal an exact number of reams. Test your solution with the following data:
The report is 140 pages long.
There will be 25 people at the meeting.

Lab 07

Problem Solving with Decision Structure

Objective:

It will enable students to understand and use selection structures for data processing.

Activity Outcomes:

The students will be able to:

- Solve problems related to selection control
- Solve problems related to nested selection control

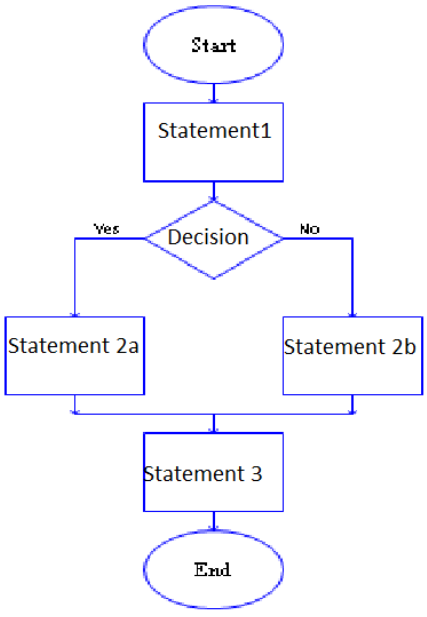
Instructor Note:

As a pre-lab activity, read "An Introduction to Programming and Algorithmic Reasoning using RAPTOR" by Hadfield, Weingart and Brown to gain an insight about flowchart representation in raptor.

1) Useful Concepts

One of the most important parts of programming is controlling which statement will execute next. In this reading, you will learn the statements, called control structures or control statements that enable you, as a programmer, to determine the order in which your program statements are executed. Using these control structures, you can determine the order that statements are executed and whether or not statements are executed at all.

1. if statement: The *if* is a decision statement that allows the code to be executed based on a given boolean condition.



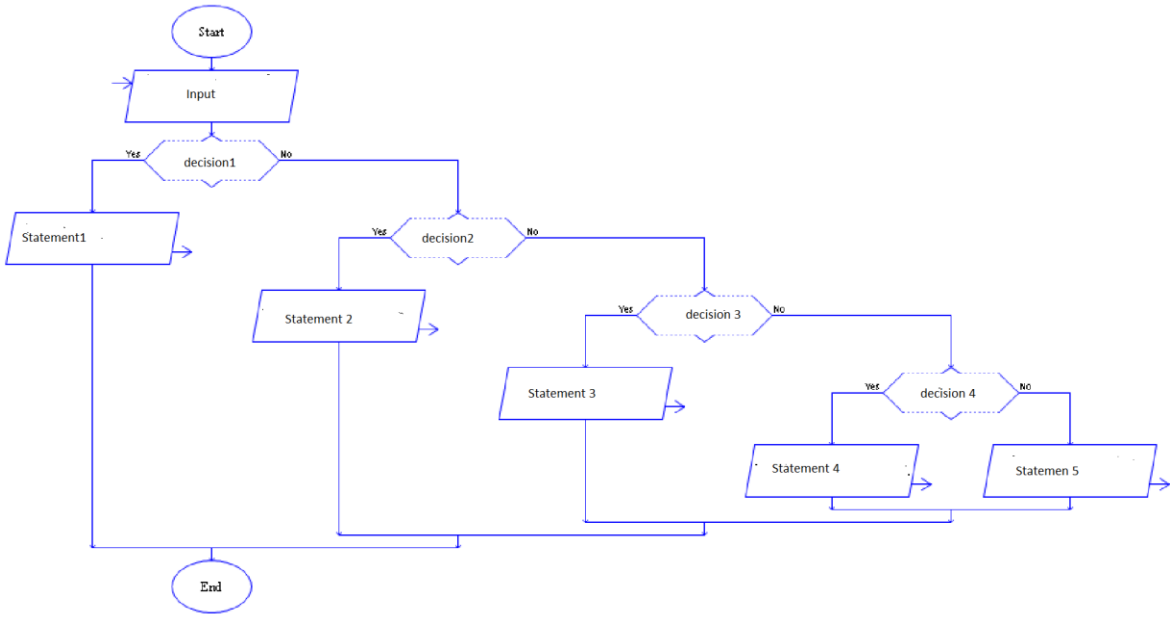
The algorithm you are developing may need to do some actions based upon a decision. For example, an algorithm that is evaluating a formula can check to see if a number is negative before taking the square root of that number and then proceed differently depending on whether the result will be a normal number or a complex number. Another example is processing grades. If the grade is above 90%, an A is assigned to the student, between 80% and 90%, a B is assigned to the student, and so on.

A selection-control statement controls whether or not a collection of code is executed or which collection of code is executed. In the inset diagram, the diamond shape represents a decision that when executed could result in an answer of Yes or No (True or False). Depending on what the answer is, the flow of control will follow the appropriate path. In the example, either statement 2a or statement 2b will be executed. One of them will be executed, but not both.

However, regardless of which statement 2 was executed, statement 3 will always be executed as it doesn't depends on any decision.

There need not be a statement 2a or a statement 2b. Either path could be empty or could contain several statements. It would be silly for both of them to be empty (or both have the exact same statements), as your decision, Yes or No, would have no effect (nothing different would happen based on the decision).

2. Cascading Selection statements: Sometime you are not making a selection between two alternatives, but making a decision amongst multiple alternatives. If this is the case, you need to have multiple selection statements.



2)Solved Lab Activites

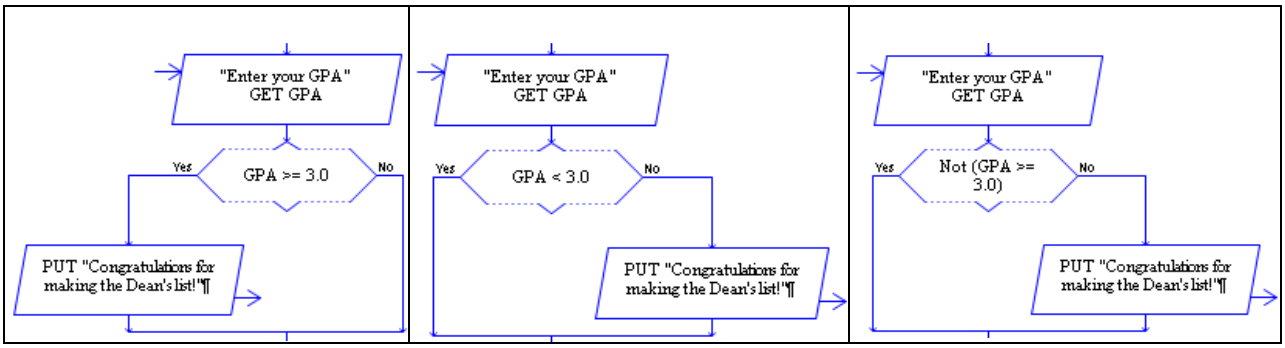
Sr.No	*Allocated Time*	*Level of Complexity*	*CLO Mapping*
1	*10*	*Medium*	*CLO-6*
2	*25*	*Medium*	*CLO-6*
3	*25*	*Medium*	*CLO-6*

Activity 1:

Raptor flowchart to illustrate decision structure

Solution:

The exact decision you write down affects whether the Yes or No paths are taken. It is very easy to change the decision to switch the yes and no paths. For example, on the left below is a simple decision on whether a student's GPA makes the grade for being on the Dean's list. On the right are two different rephrasings of the decision such that the No path is taken if the student is on the Dean's list. Some people prefer the version on the left and some the version in the center. Very few prefer the right because of the double negative implicit in the decision.

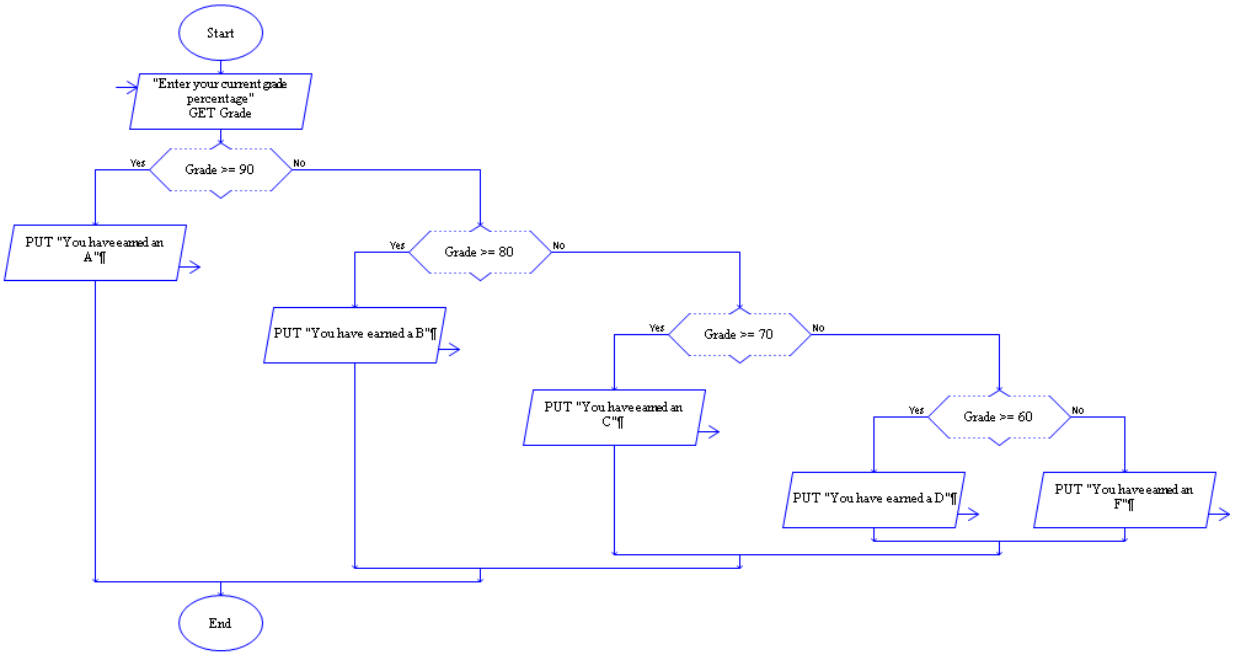


Activity 2:

Raptor flowchart to illustrate cascading decision structure

Solution:

Sometime you are not making a selection between two alternatives, but making a decision amongst multiple alternatives. For example, if you are assigning a grade (A, B, C, D, or F) you need to select

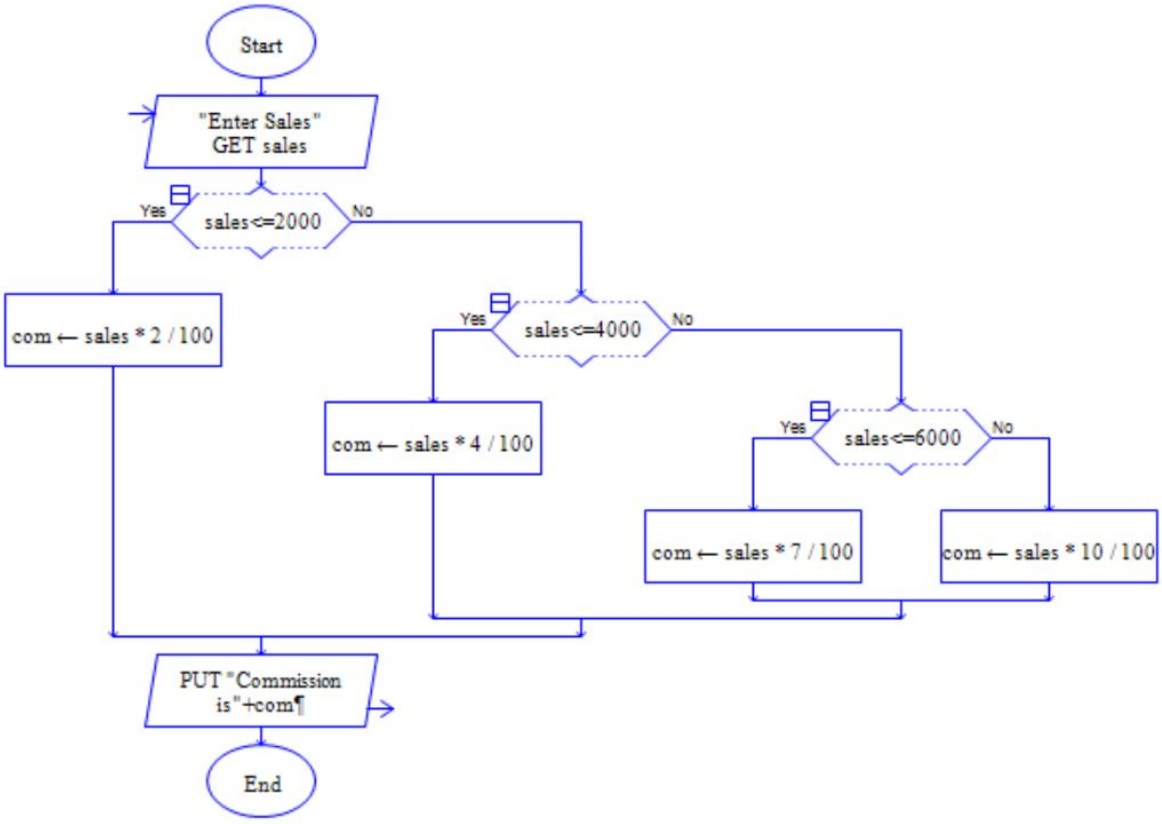


between the multiple choices. The following RAPTOR program includes multiple selection statements. If you follow the control logic, you can see how the selections cascade (as in water cascading over a series of falls).

Activity 3:

The problem illustrated in this flowchart is to calculate the commission rate for a salesperson, given the amount of sales. When the salesperson has sold less than or equal to $2,000 worth of goods, the commission is 2%. When the sales total is more than $2,000 and less than or equal to $4,000, the commission is 4%. When the sales total is more than $4,000 and less than or equal to $6,000, the commission is 7%. When the person has sold more than $6,000, the commission is 10%.

Solution:



3. Graded Lab Tasks

Note: The instructor can design graded lab activities according to the level of difficulty and complexity of the solved lab activities. The lab tasks assigned by the instructor should be evaluated in the same lab.

Lab Task 1

Draw a flowchart to calculate the water bill given the cubic feet of water used for a Water Company, which charges the homeowner one of the following:

1. *A flat rate of $15.00 for usage up to and including 1000 cubic feet.*
2. *$0.0175 per cubic foot for usage over 1000 cubic feet and up to and including 2000 cubic feet.*
3. *$0.02 per cubic foot for usage over 2000 cubic feet and up to and including 3000 cubic feet.*
4. *A flat rate of $70.00 for usage over 3000 cubic feet.*

Test your flowchart with actual data.

Lab Task 2

A company that issues check-cashing cards uses an algorithm to create card numbers. The algorithm adds the digits of a four-digit number, and then adds a fifth digit of 0 or 1 to make the sum of the digits even. The last digit in the number is called the check digit. Draw a flowchart to develop a solution that accepts a four-digit number into one variable, adds the check digit, and prints the original number and the new number. Test your flowchart with the following data:

Original number = 4737 New number = 47371

And Original Number= 4631 New Number = 46310

Lab 09

Problem Solving with Repetition Structure

Objective:

It will enable students to understand and use repitition structures with the help of examples and learning tasks.

Activity Outcomes:

The students will be able to:

- Solve problems related to repitition structure
- Solve problems related to nested repitition structure
- Solve problems using both decision and repitition structure

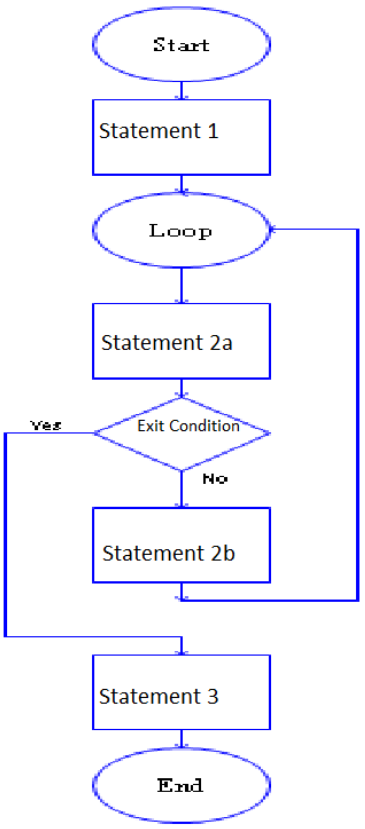
Instructor Note:

As a pre-lab activity, read "An Introduction to Programming and Algorithmic Reasoning using RAPTOR" by Hadfield, Weingart and Brown to gain an insight about flowchart representation in raptor.

1) Useful Concepts

Looping in programming languages is a feature which facilitates the execution of a set of instructions/functions repeatedly while some condition evaluates to true.

An iteration control statement controls how many times a block of code is executed. Often you need to execute some code until a condition occurs. As you may not know in advance how many times you will need to execute the code, you can't simply cut and paste the code a specific number of times. Even if you did know how many times the code would repeat, copying it that many times is not a good idea, as any error in your code would be replicated that many times as well.



In the diagram on the right, statement 1 and statement 2a are always executed. If the exit condition is true, then the loop exits and statement 3 is executed. If the exit condition is false the statement 2b and then statement 2a are executed and the exit condition is checked again. As long as the exit condition is false, statement 2b and then statement 2a will be executed again and again. 9=]

The exit condition may never be true. We'll have some examples of that later. You have to have some code, in statement 2a or statement 2b, that makes a change that turns the exit condition to true, without that, if it starts out being false, it will remain false forever.

As with the generic selection example, any of the statements in the example could be replaced by several statements. As with the selection statement, it is possible to nest loop statements. It is also possible to nest selection statements in loop

statements and vice versa.

2) Solved Lab Activites

Sr.No	*Allocated Time*	*Level of Complexity*	*CLO Mapping*
1	***10***	***Medium***	***CLO-6***
2	***15***	***Medium***	***CLO-6***
3	***15***	***Medium***	***CLO-6***
4	***20***	***High***	***CLO-6***

Activity 1:

Raptor flowchart to illustrate Loop Control examples

Solution:

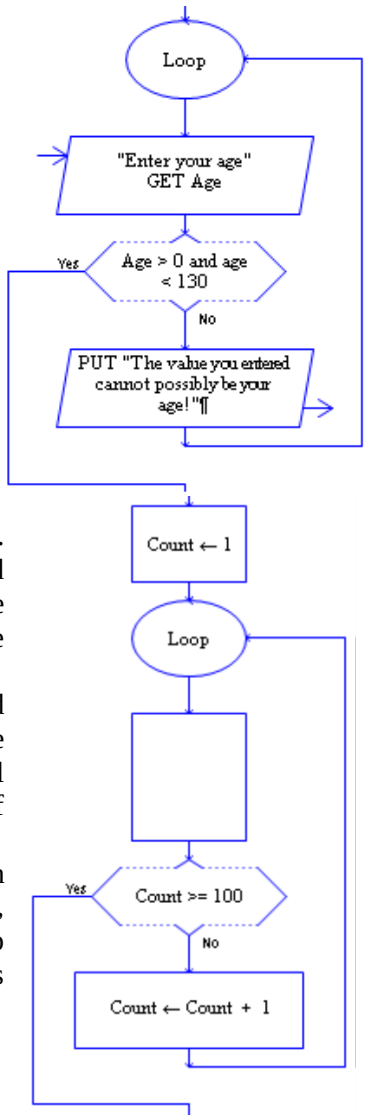
One of the most common uses of a loop is to **validate user input**. If you want the user to input data that meets certain constraints, such as entering a persons age (see example), or a number between 1 and 10, then validating the user input will ensure such constraints are met before processing continues.

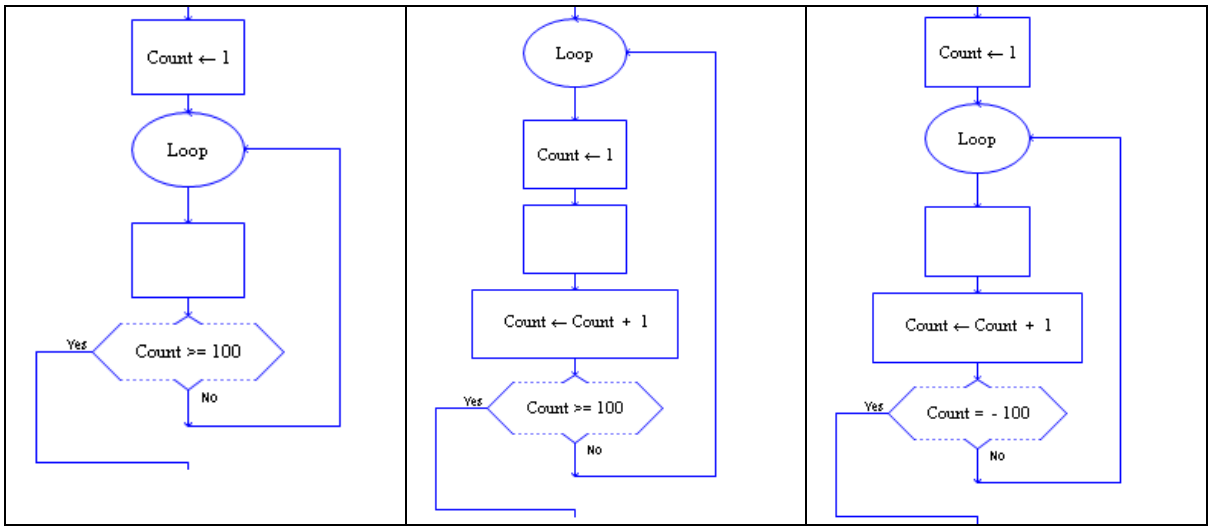
As we do not know if, or how many times, the user will attempt to enter data that does not meet the constraints, a loop must be used to ensure that correct data is entered. The exit condition establishes exactly what criteria must be met for the data to be validated.

Another common use of a loop is to execute code a specific number of times. To do this you need to count the number of loop executions and to exit. This type of loop is called a **counter-controlled loop**. In the folowing example, the long box signifies some sort of processing that must (in this example) be accomplished 100 times. The **loop control variable**, in this example is Count. There is nothing magical about the variable name Count, we could have used any legal, meaningful name. The important thing is that the loop control variable must be initialized before the loop, modified in the middle of the loop, and tested to see if the loop has executed enough times. If any of these three elements are missing, the loop will not execute the correct number of times.

Some loops are called **infinite loops** because the exit condition could never be true. The three examples below are slight variations of the counter-controlled loop example. Each of them has a problem which will cause the loop to be infinite. See if you can spot the error in each piece of code below.

Typically one or more variables are used to control whether the iteration construct exits or loops again. The acronym I.T.E.M (Initialize, Test, Execute, and Modify) can be used to check whether the loop and loop control variable(s) are being used correctly. See if you can spot what is wrong (I.T.E. or M.) with each of the code fragments below.





If you can't spot the error in each of the programs above, compare the programs with the counter-controlled loop example on the previous page.

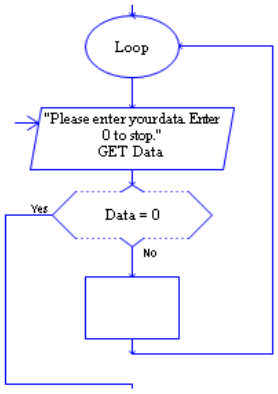
Activity 2:

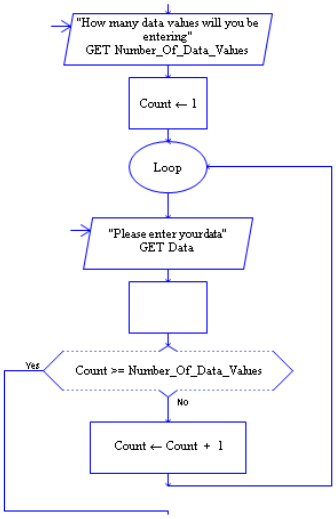
Raptor flowchart to illustrate sentinel controlled loop

Solution:

Sometimes you want users to enter a bunch of values that you can process. There are several ways that this can be accomplished in RAPTOR. The first method is to have the user enter a "special" value, called sentinel vaulue, that signifies they are finished entering data.

A second method is to ask the user in advance how many values that they will be entering and then to use that value to control the loop that asks for the data. These two methods are depicted below.





In both cases the empty boxes signify where the entered data should be processed. Don't worry about how the data is processed, just look at these examples to see how the user controls how much data is entered.

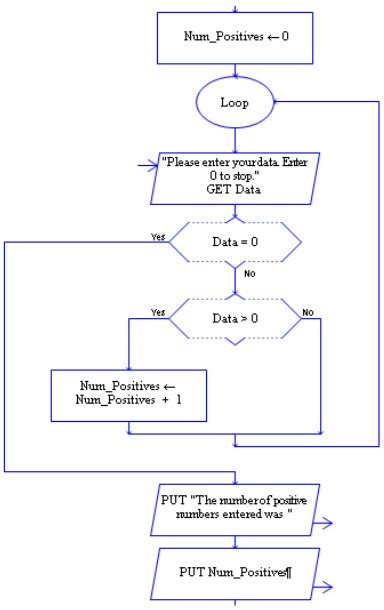
Activity 3:

Raptor flowchart for counting the number of times an event occurs

Solution:

The example to the right counts the number of positive numbers the user enters.

The first statement that initializes Num_Positives is before the loop. The second statement is "guarded" by a selection statement. The "guard" ensures you only increment the Num_Positives variable when Data > 0 is true. By replacing the guard test with some other test you can count some other event.



Activity 4:

Raptor flowchart combining decision and repitition controls

Solution:

The next example will be a complete program instead of snippets of code. We want to ask the user to type in a number and then return all of the factors of the number or, if it has none, display that the number is prime. Unfortunately, while the logic of the program that we write works well for small numbers, it would take millions of years to find the factors for large 500 digit numbers.

One of the goals of CSC101 is to teach you how solve problems and program your solutions. Showing you the code would not serve that purpose. Instead, we will describe the process by which the program was developed, and then show you the code. By reading about the process of solving a problem like that above, and then trying it out yourself in the lab, we hope to give you the skills to do it on your own.

A good way to start writing a program is to clearly understand what the inputs to the program are and what outputs the program should produce. It often helps to have several specific examples of this input/output behavior. Often a program can be thought of as a "black box" that changes inputs to outputs. Having several examples helps us understand that "black box" better.

We want our program to exhibit the following behavior. If, for example, the user types in a 77, the program should respond with "7 is a factor of 77.", and "11 is a factor of 77.". If the user types in a 41, though, the program should respond "41 is a prime number." as 41 has no factors.

We can understand the "black box" better by coming up with a process that turns the inputs into outputs and then use that process as the code for your program. Before you can develop a program for a problem you must be able to do the following:

(1) Solve the problem yourself for particular instances of the problem (like 41 and 77).
(2) Be able to clearly describe the process by which you arrived at the answer, and have that process be generic enough to be used for any number (not just the two examples).

It is critical that it be a general process. While you may be able to describe exactly how you determined the factors of 77 and that 41 was prime that is not sufficient. You must be able to describe a process that could find all of the factors of any number, and whether any number was prime. If you cannot do so then it will be VERY difficult to write a program to do so!

The following code snippet is a naïve attempt to solve the problem:

```
if Number = 77 then
   PUT "7 is a factor of 77."
   PUT "11 is a factor of 77."
else if Number = 41 then
   PUT "41 is a prime number."
else if etc.
 ...
```

Obviously there are too many different numbers that could be entered by the user for you to code all of them. In addition, if you were to attempt to code a solution to the problem using code like that above, you would be doing the grunt work of determining what the factors of a number were and whether it was prime or not, not the computer. The point of using a computer is for it to do all of the tedious work. You, as the programmer, do the work that requires brains. You need to come up with a process and then instruct the computer by writing code that it can use to follow the process and arrive at a solution. Think about how you would describe to anyone, let alone a computer, how to determine all of the factors of a number and whether a number was prime or not… then read on.

One way to determine the factors of a number, X, is to start at the number 2 and then check all of the numbers between 2 and X - 1 to see if any evenly divide X. If none of the numbers in that range evenly divide X then we know X is prime. If any numbers in that range do evenly divide X, then they are factors and we can display them to the user.

The above text describes a process that could be used to determine the factors of any number and if it had none declare the number to be prime. Sometimes just describing a process like that is good enough to start coding from. In this case, because it is the first really complex program you are seeing, we will take the narrative form of the process one step further and create an algorithm, using pseudo code, which makes the steps in the process more like the eventual code.

Prompt for and get a number, storing it as “Possible_Prime”
Assume initially that the number is prime (Set “Is_Prime” to True)
Loop with X ranging from 2 to “Possible_Prime”-1
If a particular X divides “Possible_Prime” evenly then
Display that X is a factor of “Possible_Prime”
Remember that the number is not prime (Set “Is_Prime” to False)
If “Is_Prime” is True then
Display “Possible_Prime” is prime

One of the keys to the above algorithm is the use of the variable “Is_Prime”. First we assume that “Possible_Prime” is a prime number. Then we check all of the numbers, 2 through “Possible_Prime” – 1, one at a time, to determine if any of them divide the “Possible_Prime” number evenly. If any of the numbers are factors then we display the number immediately. However, we also need to “remember” that we now know that “Possible_Prime” is not a prime number. We do that by setting “Is_Prime” to False. Once we have checked all of the numbers in the range we will have displayed all of the factors or there were no factors. In the latter case we can determine that “Possible_Prime” is a prime number because we never “corrected” our original assumption. After the loop is finished we can use the value of “Is_Prime” to determine whether or not to display that “Possible_Prime” is prime.

So far we have accomplished the first two steps of programming. We have understood the problem and better characterized the problem by describing its input/output behavior. We have also designed a solution to the problem that appears to work for all positive integers. The third step is to translate the solution (algorithm) to RAPTOR code. The following code is the translated solution. We have annotated it to make it easier for you to understand. Review the program and make sure you understand how the code relates to the algorithm and the process we came up with for solving the problem.

Annotations to help you understand the factoring program

Get the number to check from the user

Assume that the number is prime

Use 2 as the starting point for checking factors

Stop checking possible factors when you count up to "Possible_Prime"

Check if the Possible_Factor evenly Divides the Possible_Prime

If it does evenly divide then the "Possible_Factor" is a real factor

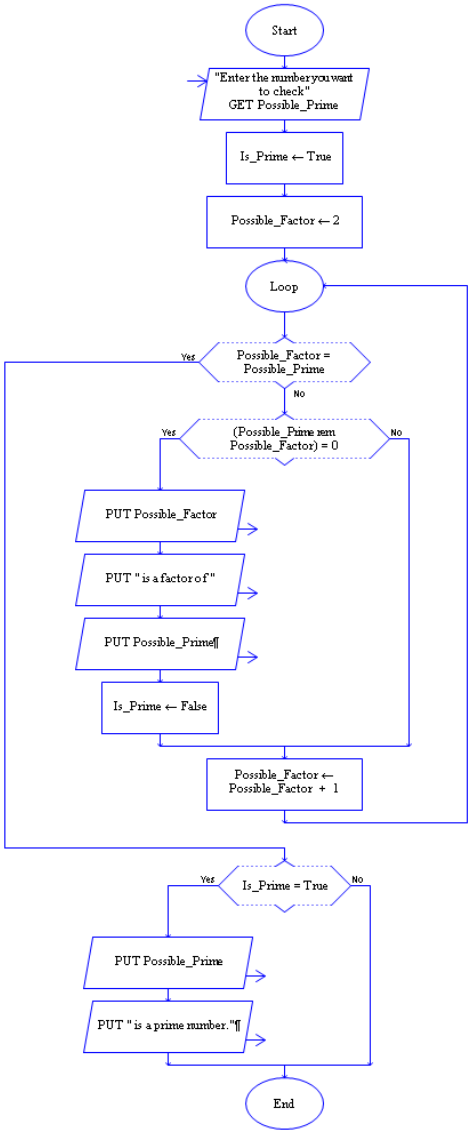
Display all the required text

Set Is_Prime to False indicating that you have found at least one factor

Increment Possible_Factor so you can check if the next number is a factor of the number entered by the user

If Is_Prime is still true, then you didn't find any factors

If so, display the fact that the user's number is a prime number.



3) Graded Lab Tasks

Note: The instructor can design graded lab activities according to the level of difficult and complexity of the solved lab activities. The lab tasks assigned by the instructor should be evaluated in the same lab.

Lab Task 1

A Boutique is having a five-day sale. Each day, starting on Monday, the price will drop 10% of the previous day's price. For example, if the original price of a product is $20.00, the sale price on Monday would be $18.00 (10% less than the original price). On Tuesday the sale price would be $16.20 (10% less than Monday). On Wednesday the sale price would be $14.58; on Thursday the sale price would be $13.12; and on Friday the sale price would be $11.81. Develop a solution that will calculate the price of an item for each of the five days, given the original price. Test the solution for an item costing $10.00.

Lab Task 2

Mary Smith, a student, has borrowed $3,000 to help pay her college expenses. After setting up a budget, $85 was the maximum monthly payment she could afford to make on the loan. Develop a solution to calculate and print the interest, the principal, and the balance on the loan per month. Other information she would like to know is the number of years and months it will take to pay the loan back and the total interest she will pay during that period. The interest rate is 1% per month on the unpaid balance. Keep in mind these formulas:
*interest_normal = balance*interest_rate*
payment = balance – interest
new_balance = balance - payment

Lab 10

Problem Solving with Functions

Objective:

This lab will introduce you to the basic usage of functions. In raptor, function equivalence is established by using sub-routines. Therefore in this lab, you will use raptor to represent functional flow of the problems using subroutines with the help of examples and learning tasks.

Activity Outcomes:

The activities provide hands - on practice on

- How to use subroutines in Raptor

Instructor Note:

As pre-lab activity, read Chapter 8 from the book (Learning Python, Mark Lutz, 5th Edition (2013), O'Reilly Media), and also as given by your theory instructor.

1) Useful Concepts

Many common tasks come up time and time again when programming. Instead of requiring you to constantly reinvent the wheel, programming has a number of built-in features which you can use. Including so much ready to use code is sometimes referred to as a ’batteries included’ philosophy. Python/Raptor comes with just under fifty predefined functions (of which we’ll be only using about a dozen) and the simplest way to use this prewritten code is via function calls.

The syntax of a function call is simply

```
FUNCTION_NAME(ARGUMENTS)
```

Not all functions take an argument, and some take more than one (in which case the arguments are separated by commas).

2) Solved Lab Activites

Sr.No	***Allocated Time***	***Level of Complexity***	***CLO Mapping***
1	***30min***	***High***	***CLO-6***
2	***30min***	***High***	***CLO-6***

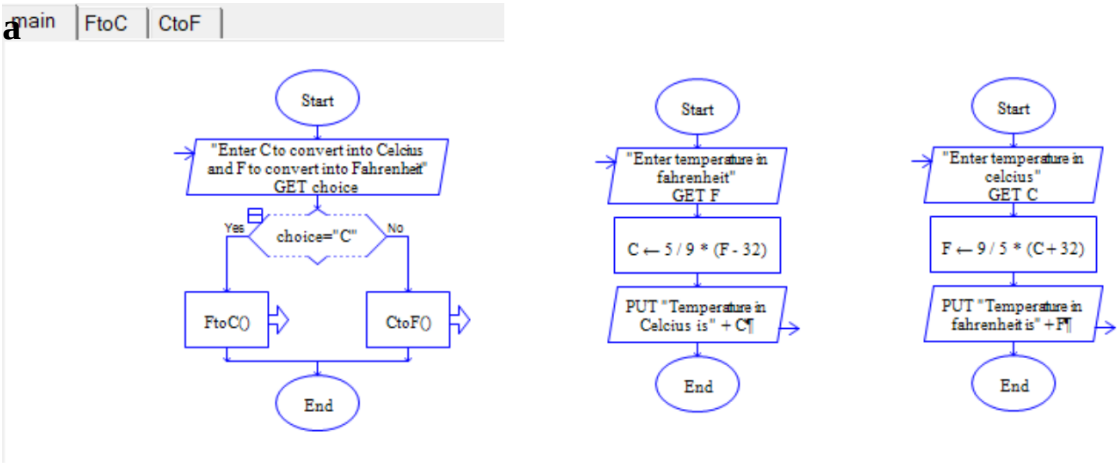
Activity 1:

The following flowchart consists of two subroutines. User will input the choice "C" if he/she wants to convert temperature from Fahrenheit to Celsius and "F" otherwise. Depending on the user's choice subroutine will be called and convert the temperature into required unit.

Solution:

The left most chart is the main program from where the other two subroutines will be called depending on the user's input. The center flowchart is a function to convert the Fahrenheit temperature into Celsius and the right one is to convert from Celsius to Fahrenheit.

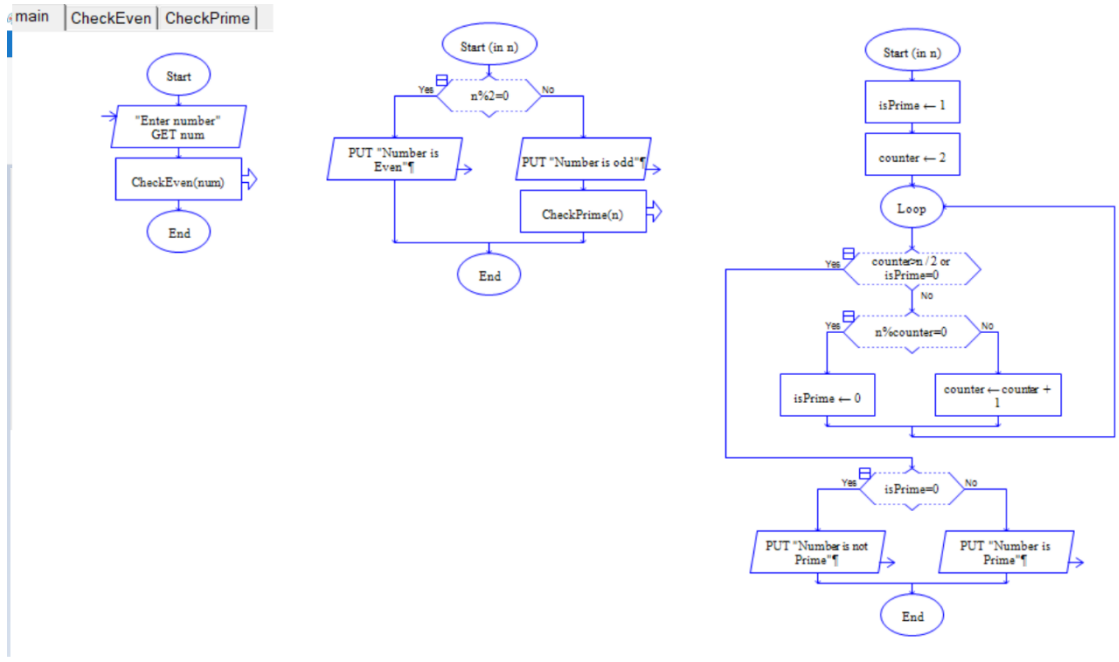
a



Activity 2:

Following flowchart illustrate the call of a function from another function. The main will call a function CheckEven(n). This function will again call the function CheckPrime(n) if the passed argument is odd and print the corresponding ,message otherwise.

Solution:



The left is the main flowchart and center is the CheckEven subroutine and most right is the CheckPrime subroutine.

3) Graded Lab Task

Note: The instructor can design graded lab activities according to the level of difficult and complexity of the solved lab activities. The lab tasks assigned by the instructor should be evaluated in the same lab.

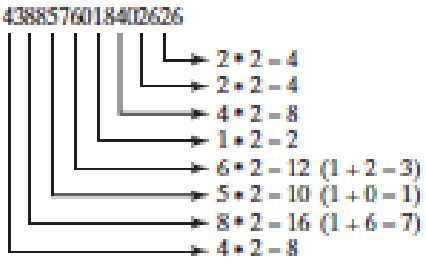
Lab Task 1

Credit card numbers follow certain patterns: It must have between 13 and 16 digits, and the number must start with:

- *4 for Visa cards*
- *5 for MasterCard credit cards*
- *37 for American Express cards*
- *6 for Discover cards*

The Luhn's algorithm is useful to determine whether a card number is entered correctly or whether a credit card is scanned correctly by a scanner. Credit card numbers are generated following this validity check, which can be described as follows (for illustration consider the card number 4388576018402626):

1. *Double every second digit from right to left. If doubling of a digit results in a two-digit number, add up the two digits to get a single-digit number.*
2. *Now add all single-digit numbers from Step 1.*
   4 + 4 + 8 + 2 + 3 + 1 + 7 + 8 = 37
3. *Add all digits in the odd places from right to left in the card number.*
   6 + 6 + 0 + 8 + 0 + 7 + 8 + 3 = 38
4. *Sum the results from Steps 2 and 3*
   37 + 38 = 75
5. *If the result from Step 4 is divisible by 10, the card number is valid; otherwise,it is invalid. For example, the number 4388576018402626 is invalid, but the number 4388576018410707 is valid.*



Draw a flowchart that prompts the user to enter a credit card number as an integer. Display whether the number is valid or invalid. Design your folowchart to use the following functions/subroutines:

```
# Return true if the card number is valid
def isValid(number):

# Get the result from Step 2
def sumOfDoubleEvenPlace(number):

# Return this number if it is a single digit, otherwise, return the sum of the two digits
def getDigit(number):

# Return sum of odd place digits in number
def sumOfOddPlace(number):

# Return true if the digit d is a prefix for number
def prefixMatched(number, d):

# Return the number of digits in d
def getSize(d):

# Return the first k number of digits from number. If the number of digits in number is less than k,
return number.
def getPrefix(number, k):
```

Lab 10

Python: Elementary Programming

Objective:

The objective of this lab will be to learn about variables and the basics of Python with the help of examples and learning tasks.

Activity Outcomes:

The activities provide hands-on practice with the following topics

- Basics of programming
- Variables
- Print & Eval functions

Instructor Note:

As a pre-lab activity, read Chapter 3 from the textbook "Python Basics: A Practical Introduction to Python 3, 2021".

1) Useful Concepts

1. The most important piece of code which comes bundled with Python and the one thing which you will be using the most will be the **print** function. A function is a piece of code that accepts some input and returns an output. The **print** function works like this

```
print(<thing to display>)
```

2. Programming languages use special symbols called **identifiers** to name such programming entities as variables, constants, methods, classes, and packages.
   An identifier is a sequence of characters that consist of letters, digits, underscores (_), and dollar signs ($), must start with a letter, an underscore (_), or a dollar sign ($). It cannot start with a digit. An identifier cannot be a reserved word. An identifier cannot be true, false, or null.
3. **Variables** are used to store values temporarily for the time the code is being executed so that, that value can be reused later. As variables are identifiers as well so they follow the same naming convention. The syntax is given below

```
<var_name> = <value>
```

4. Any line of code that has an effect like creating a variable etc is called a **statement**.
5. The phenomenon of giving a variable a value is called **assignment** and that statement is known as an **assignment statement**. The statement above is known as an assignment

statement. The statement for assigning a value to a variable is called an assignment statement. In Python, the equal sign (=) is used as the assignment operator. The syntax for assignment statements is as follows:

6. Any combination of values, variables, and operators is called an **expression**. An expression represents a computation involving values, variables, and operators that, taken together, evaluate to a value. For example, consider the following code:

```
# Assign the value of the expression to x

variable = expression

x = 5 * (3 / 2) + 3 * 2
```

Numeric Operators

Name	*Meaning*	*Example*	*Result*
`+`	Addition	`34 + 1`	`35`
`-`	Subtraction	`34.0 - 0.1`	`33.9`
`*`	Multiplication	`300 * 30`	`9000`
`/`	Float Division	`1 / 2`	`0.5`
`//`	Integer Division	`1 // 2`	`0`
`**`	Exponentiation	`4 ** 0.5`	`2.0`
`%`	Remainder	`20 % 3`	`2`

7. There are various data types that we can assign to variables. The most basic ones which we encounter a lot are **int**, **float**, **str**(string), and **bool**(boolean).
8. **int** and **float** are the most commonly used types. **int** data type contains all integers and **float** data type contains all float numbers(numbers with decimal values).

```
num1 = 34
num2 = 3.21
```

9. Any text value is a **string**. We wrap these values with inverted commas

```
x = 'This is a tomato'
```

10. **bool** data type contains two values, true and false. These values are very important and are used for making code-related decisions e.g if the email and password entered for a website are correct, then authorize this person to view content.

```
y = True
```

```
z = False
```

11. You may sometimes want to write some notes in your code so that you know what you did in the code. This can be achieved by writing **comments.** Comments can be written by using the # and writing comment after it or surrounding your comment with ‘’’ triple commas.

```
#                                                                    Comment

'''
Multiline                                                            Comment
'''
```

12. What is eval () in python and what is its syntax?
    eval() is a built-in- function used in python, eval function parses the expression argument and evaluates it as a python expression. In simple words, the eval function evaluates the “String” like a python expression and returns the result as an integer.

```
Syntax
The syntax of the eval function is as shown below:

eval(expression, [globals[, locals]])
```

Operator Precedance to follow

Precedence	*Operator*
↓	+, - (Unary plus and minus)
	** (Exponentiation)
	not
	*, /, //, % (Multiplication, division, integer division, and remainder)
	+, - (Binary addition and subtraction)
	<, <=, >, >= (Comparison)
	==, != (Equality)
	and
	or
	=, +=, -=, *=, /=, //=, %= (Assignment operators)

2) Solved Lab Activities

Sr.No	*Allocated Time*	*Level of Complexity*	*CLO Mapping*
1	*10*	*Low*	*CLO-6*
2	*10*	*Low*	*CLO-6*
3	*10*	*Low*	*CLO-6*
4	*15*	*Medium*	*CLO-6*
5	*15*	*Low*	*CLO-6*

Activity 1:

Python program to do some math operations on numbers

Solution:

```
x = 2
y = 3
z = 1
result = 2 + 3 - 1
print(result)
```

Output

```
4
```

Activity 2:

Python program to illustrate a string

```
x               =               "This               is               Python"
y                =                "This                 is                 Java"
print(x)
print(y)
```

Output

```
This is Python

This is Java
```

Activity 3:

Write a program that takes three numbers and prints their sum. Every number is given on a separate line

```
# Addition Function
# This program reads two numbers and prints their sum:
a = int(input())
b = int(input())
c = int(input())
print(a + b + c)
```

Input

2

3

5

Output: 10

Activity 4:

Write a program that reads an integer number and prints its previous and next numbers.

```
# Addition Subtraction Function

# Read an integer:
a = int(input())
# Read a interger:
#Print a value:
print(a-1, a+1)
```

Input: 179

Output

```
The next number for the number 179 is 180.
The previous number for the number 179 is 178.
```

Activity 5:

Python program to illustrate comments

```
# Addition Function

'''
Addition Function
This function takes in two numbers and returns the sum
@param1 number
@param2 number
@returns number
'''
```

Output

```
No output
```

3) Graded Lab Tasks

Note: The instructor can design graded lab activities according to the level of difficulty and complexity of the solved lab activities. The lab tasks assigned by the instructor should be evaluated in the same lab.

Lab Task 1

Write a program that greets the person by printing the word "Hi" and the name of the person.

Lab Task 2

Write a program that takes a number and print its square.

Lab Task 3

Write a program that reads the length of the base and the height of a right-angled triangle and prints the area. Every number is given on a separate line.

Lab 11

Conditional Structure

Objective:

The objective of this lab will be to learn about conditional statements with the help of examples and learning tasks.

Activity Outcomes:

The activities provide hands - on practice with the following topics

- Implement an if statement.
- Implement an if-else statement.
- Implement an if-elif statement
- Nest if-else statements.

Instructor Note:

As a pre-lab activity, read Chapter 8 from the textbook “Python Basics: A Practical Introduction to Python 3, 2021”.

1) Useful Concepts

Condition statements allow us to write code that behaves differently in different scenarios. e.g let's consider a calculator. A calculator would not always calculate the sum of two numbers. It would calculate the result based on the operation selected by us. This is made possible using conditional statements.

1. The most basic conditional statement is an if statement. The syntax of an if statement is given below. The code inside the if statement would only execute if the condition is fulfilled i.e the condition inside the round brackets returns true.

```
if(<condition>):
    // some code
```

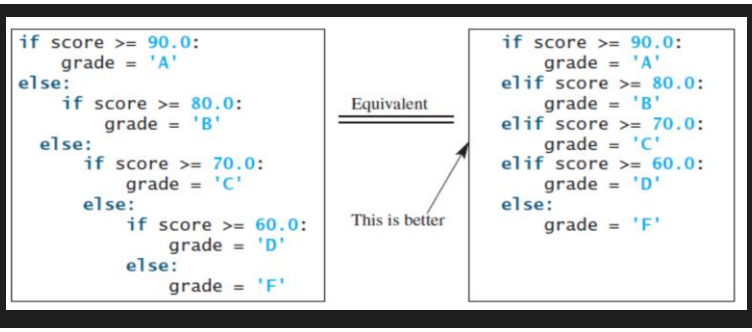
2. We can have another scenario in which in one condition we want to do one thing but in another condition, we want to do something else. This can be done by using if-else. The syntax is given under. The else statement runs only when the condition corresponding to the if block returns false.

```
if(<condition>):
    //some                                                                code
else:
    //some other code
```

3. When we have multiple conditions and we want to write different code for each of them, we can use if elif else. If we have 3 conditions lets say, the first condition would be checked using an if statement, the second with the elif statement and then third can either be checked with an elif again or just else as there is no other scenario.

```
if(<condition_1>):
    //some                                                                code
elif(<condition_2>):
    //                      condition                     2                 code
elif(<condition_3>):
    //                      condition                     3                 code
else:
    //some other code
```

4. The nested if statement can be used to implement multiple alternatives. The statement given in below Figure, for instance, assigns a letter value to the variable grade according to the score, with multiple alternatives.



2) Solved Lab Activites

Sr.No	*Allocated Time*	*Level of Complexity*	*CLO Mapping*
1	*15*	*Low*	*CLO-7*
2	*15*	*Medium*	*CLO-7*
3	*15*	*Medium*	*CLO-7*
4	*15*	*Medium*	*CLO-7*

Activity 1:

Python program to illustrate an if statement

Solution:

```
x = 4
if(x==3):
    print('Lions are the king of the jungle')

if(x==4):
    print("Bears eat honey")
```

Output

```
Bears eat honey
```

Activity 2:

Python program to illustrate an if else statement

```
x = 4
if(x==3):
    print('Lions are the king of the jungle')
else:
    print("Bears eat honey")
```

Output

```
Bears eat honey
```

Activity 3:

Python program to illustrate a function definition with return

```
x = 4
```

```
if(x==3):
    print('Lions are the king of the jungle')
elif(x==4):
    print('Canberra is the capital of Australia')
else:
    print("Bears eat honey")
```

Output

```
Canberra is the capital of Australia
```

Activity 4:

Python program to illustrate nested if else statements

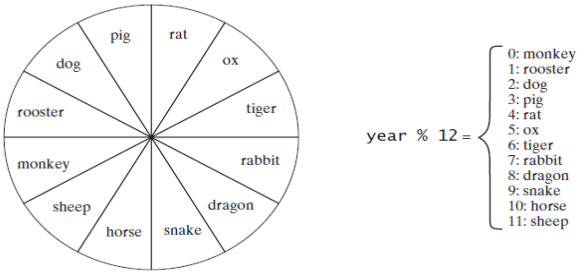
```
if(x<=2 and y<20):
    print('The numbers x and y fall under the criteria')
    sum = x + y
    if(sum<50):
        print('The sum of x and y is:',sum)
else:
    print('The numbers x and y dont fall under the criteria')
```

Output

```
The numbers x and y fall under the criteria
The sum of x and y is: 12
```

Activity 5:

Write a program to find out the Chinese zodiac sign for a given year. The Chinese zodiac sign is based on a 12-year cycle, and each year in this cycle is represented by an animal—monkey, rooster, dog, pig, rat, ox, tiger, rabbit, dragon, snake, horse, and sheep



```
year = eval(input("Enter a year: "))
2 zodiacYear = year % 12
3 if zodiacYear == 0:
4 print("monkey")
5 elif zodiacYear == 1:
6 print("rooster")
7 elif zodiacYear == 2:
8 print("dog")
9 elif zodiacYear == 3:
10 print("pig")
11 elif zodiacYear == 4:
12 print("rat")
13 elif zodiacYear == 5:
14 print("ox")
15 elif zodiacYear == 6:
16 print("tiger")
17 elif zodiacYear == 7:
18 print("rabbit")
19 elif zodiacYear == 8:
20 print("dragon")
21 elif zodiacYear == 9:
22 print("snake")
23 elif zodiacYear == 10:
24 print("horse")
25 else:
26 print("sheep")
```

Output

```
Enter a year: 1963
rabbit
Enter a year: 1877
Ox
```

3) Graded Lab Tasks

Note: The instructor can design graded lab activities according to the level of difficulty and complexity of the solved lab activities. The lab tasks assigned by the instructor should be evaluated in the same lab.

Lab Task 1

Write a program to check whether an integer is positive, negative, or zero

Lab Task 2

Write a program to input marks of five subjects Physics, Chemistry, Biology, Mathematics, and Computer. Calculate percentage and grade according to following:
Percentage >= 90% : Grade A
Percentage >= 80% : Grade B
Percentage >= 70% : Grade C
Percentage >= 60% : Grade D
Percentage >= 40% : Grade E
Percentage < 40% : Grade F

Lab Task 3

Write a program to check whether the triangle is equilateral, isosceles or scalene triangle.

Lab Task 4

Write a dummy authentication system program in which you accept user inputs for email and password. Let's say the correct email and password are abc@gmail.com and abc respectively. If the email and password entered are correct it should display "User is logged in". If the email is correct, then prompt the user that the password is not correct. If the password is correct then prompt the user to enter the correct email. If both are incorrect then display the corresponding message

Lab Task 5:

Write a program to check whether a year is a leap year or not.

Lab 12

Strings

Objective:

The objective of this lab will be to learn about strings and with the help of examples and learning tasks.

Activity Outcomes:

The activities provide hands - on practice with the following topics

- Implement a string and its related operations

Instructor Note:

As a pre-lab activity, read Chapter 4 and 9 from the textbook “Python Basics: A Practical Introduction to Python 3, 2021”.

1) Useful Concepts

A string is a collection of text. Anything in Python surrounded by ‘’ or “” is called a string.

1. Single line strings are surrounded by single or double commas while we can write multiline strings as well using triple commas.

```
x = 'single line string'
x = "single line string"

y = '''multiline string
              multiline string '''
```

2. We can access any string character by specifying its index in the string. Indexes define the position of any character in the string and they start from 0. So the first character would be present on the 0th index. The syntax is as follows

```
<string_name>[index]
```

3. We can access any string Length by using the len() function and it will return the total no of characters in the string Suppose S1=’Hello’

```
Len(< S1> )
5
```

4. We can also take out a chunk of a string. This is known as slicing. This is done by using the index operator and specifying two indexes separated by a : . The first is the starting index and the other is the ending index.

```
<string_name>[start_index:end_index]
```

5. We can join two strings together by using the + operator. This is known as concatenation.

```
result = <string1> + <string2>
```

6. Repetition as the name suggest means to multiply the items or Repeat Sequence s1 n times. n copies of sequence s are concatenated The syntax is follows

```
s1 = s1 x n
Or
s1 = n x s1
```

7. Membership testing: is done by using ***in*** or ***not in*** operator to find the presence of an element in the sequence

```
x in s # True if element x is in sequence s.
x not in  s # True if element x is not in sequence s.
```

8. Suppose you want to print a message with quotation marks in the output. Can you write a statement like this?
print("He said, "John's program is easy to read"")

No, this statement has an error. Python thinks the second quotation mark is the end of the string and does not know what to do with the rest of the characters. To overcome this problem, Python uses a special notation to represent special characters, as shown in below Table 3.3. This special notation, which consists of a backslash (\) followed by a letter or a combination of digits, is called an escape sequence.

Python Escape Sequences

Character Escape Sequence	*Name*	*Numeric Value*
`\b`	Backspace	8
`\t`	Tab	9
`\n`	Linefeed	10
`\f`	Formfeed	12
`\r`	Carriage Return	13
`\\`	Backslash	92
`\'`	Single Quote	39
`\"`	Double Quote	34

Operation	**Description**
○ len(s)	Length of sequence s, i.e., the number of elements in s.
○ min(s)	Smallest element in sequence s.
○ max(s)	Largest element in sequence s.
○ sum(s)	Sum of all elements in sequence s.
○ <, <=, >, >=, =, !=	Compares two sequences.

2) Solved Lab Activities

Sr.No	***Allocated Time***	***Level of Complexity***	***CLO Mapping***
1	***10***	***Low***	***CLO-7***
2	***10***	***Low***	***CLO-7***
3	***10***	***Low***	***CLO-7***
4	***10***	***Medium***	***CLO-7***
5	***10***	***Medium***	***CLO-7***
6	***10***	***Medium***	***CLO-7***

Activity 1:

Python program to tell the number of characters in a string.

Solution:

```
x                    =                    "This                    is                    Python"
print(len(x))
```

Output

```
14
```

```
The value of x is: 2
```

```
The value of x is: 3
```

Activity 2:

Python program to illustrate using a string with escaped characters

```
x = "\"This is Python\", said Amanda"
print(x)
```

Output

```
"This is Python", said Amanda
```

Activity 3:

Python program to illustrate string concatenation, indexing, and slicing.

```
string1 = 'Australia'
string2 = 'Pakistan won the world cup in 2012'

result = string1 + ' vs ' + string2[:8] + ' Match ' + string2[30]
print(result)
```

Output

```
Australia vs Pakistan match 2
```

Activity 4:

Python program to illustrate membership testing .

```
>>> s1 = "Welcome"
>>> "come" in s1
True
>>> "come" not in s1
False
```

Here is another

```
s = input("Enter a string: ")
if "Python" in s:
print("Python", "is in", s)
else:
print("Python", "is not in", s)
```

Output

```
python is in Welcome to Python.
```

Activity 5:

A string is a palindrome if it reads the same forward and backward. The words "mom," "dad," and "noon," for instance, are all palindromes

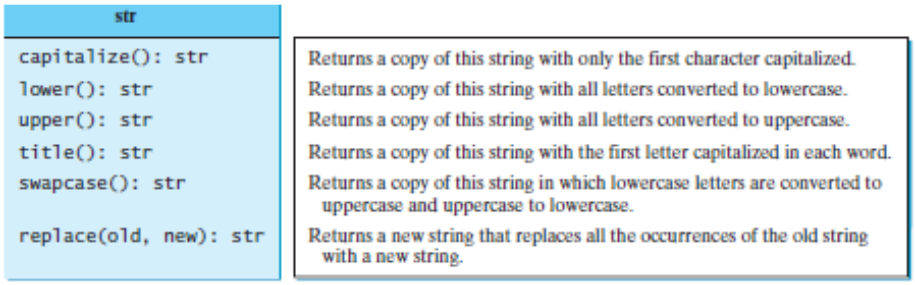
```
# Prompt the user to enter a string
s = input("Enter a string: ")
s = s.strip()            # Remove white spaces

low = 0                  # The index of the first character in the string
high = len(s)-1          # The index of the last character in the string

while low < high:
    print(low, "    ", high)
    if s[low] == s[high]:
        low += 1
        high -= 1
    else:
        print("Not Palindrome")
        break

if low >= high:
    print("Its a Palindrome")
```

Output

```
Enter a string: MoM
```

```
It’s a Palindrome
```

Activity 6:



Output

```
1 >>> s = "welcome to python"
2 >>> s1 = s.capitalize()
3 >>> s1
4 'Welcome to python'
5 >>> s2 = s.title()
6 >>> s2
7 'Welcome To Python'
8 >>> s = "New England"
9 >>> s3 = s.lower()
10 >>> s3
11 'new england'
12 >>> s4 = s.upper()
13 >>> s4
14 'NEW ENGLAND'
15 >>> s5 = s.swapcase()
16 >>> s5
17 'nEW eNGLAND'
18 >>> s6 = s.replace("England", "Haven")
19 >>> s6
20 'New Haven'
21 >>> s
22 'New England'
23 >>>
```

3) Graded Lab Tasks

Note: The instructor can design graded lab activities according to the level of difficulty and complexity of the solved lab activities. The lab tasks assigned by the instructor should be evaluated in the same lab.

Lab Task 1

Write a program to which replaces the first character of a string with the character 'z'

Lab Task 2

Write a Python program to remove the characters which have odd index values of a given string

Lab Task 3

Write a Python program to get a single string from two given strings, separated by a space and swap the first two characters of each string
Sample String : 'abc', 'xyz'
Expected Result : 'xyc abz'

Lab Task 4

Write a Python program to print a specified list after removing the 0th, 4th and 5th elements.

Lab 13

Loops + Nested Loops

Objective:

The objective of this lab will be to learn about loops and nested loops with the help of examples and learning tasks.

Activity Outcomes:

The activities provide hands - on practice with the following topics

- Implement a while loop
- Implement a for-in loop
- Implement nested loops

Instructor Note:

As a pre-lab activity, read Chapter 6 from the textbook "Python Basics: A Practical Introduction to Python 3, 2021".

1) Useful Concepts

A loop is a block of code that gets repeated over and over again either a specified number of times or until some condition is met which is why a loop runs only a finite number of times. If we specify a condition that does not get fulfilled ever during loop iterations then it would go on forever and the program would freeze.

1. One kind of a loop is a while loop. The syntax is given below. We specify a condition in the round brackets after a while just like we do for the if statement. This loop keeps on running while the statement passed is true(hence the name while).

```
while(<condition>):
    // some code to repeat
```

2. Another kind of a loop is a for-in loop. The syntax is given below. A for-in loop iterates over a sequence(list, string or range). The variable we use after **for**, gets the value of the next item in the sequence in each iteration while the variable after **in** is a sequence which we want iterated on.

```
for                    <item_name>                    in                    <sequence_name>:
    //                 some                 code                 to                 repeat
```

3. Sometimes we would want to nest loops. This can be done as well. This will look something like this

```
for <item_name> in <sequence_name>:
    // some code to repeat
    while(<some_condition>):
        // some nested code to repeat
```

2) Solved Lab Activites

Sr.No	*Allocated Time*	*Level of Complexity*	*CLO Mapping*
1	*15*	*Low*	*CLO-7*
2	*15*	*Low*	*CLO-7*
3	*15*	*Low*	*CLO-7*
4	*15*	*Low*	*CLO-7*

Activity 1:

Python program to illustrate an while loop

Solution:

```
x = 0
while(x<4):
    print("The value of x is:",x)
    x = x + 1
```

Output

```
The value of x is: 0
The value of x is: 1
The value of x is: 2
The value of x is: 3
```

Activity 2:

Python program to illustrate an for in loop.

```
subjects = ['Maths','English','Urdu']
for subject in subjects:
    print("The name of the subject is:",subject)
```

Output

```
The name of the subject is: Maths
```

```
The name of the subject is: English
The name of the subject is: Urdu
```

Activity 3:

Python program to illustrate a nested loop.

```
subjects = ['Maths','English','Urdu']
grades = ['A','B']
for subject in subjects:
    print("The name of the subject is:",subject)
    for grade in grades:
        print("The name of the grade is:", grade)
```

Output

```
The name of the subject is: Maths
The name of the grade is: A
The name of the grade is: B
The name of the subject is: English
The name of the grade is: A
The name of the grade is: B
The name of the subject is: Urdu
The name of the grade is: A
The name of the grade is: B
```

3) Graded Lab Tasks

Note: The instructor can design graded lab activities according to the level of difficult and complexity of the solved lab activities. The lab tasks assigned by the instructor should be evaluated in the same lab.

Lab Task 1

Write a program to print all natural numbers from 1 to n.

Lab Task 2

Write a program to print all even numbers between 1 to 100.

Lab Task 3

Write a program to ask user input for a number. Check if it is a prime number. If it is, end the program but if it is not ask for the user input again till the user enters a prime number.

Lab Task 4

Write a program to print a table of numbers from 1 to 8.

Lab Task 5

Write a nested for loop that prints the following output:

```
    1
  1 2 1
1 2 4 2 1
1 2 4 8 4 2 1
```

Lab 14

Functions

Objective:

The objective of this lab will be to learn about functions with the help of examples and learning tasks.

Activity Outcomes:

The activities provide hands - on practice with the following topics

- Write a function definition
- Write a function definition with default params
- Write a function definition with return statement.
- Call a function
- Can a function with params

Instructor Note:

As pre-lab activity, read Chapter 6 from the text book “Python Basics: A Practical Introduction to Python 3, 2021”.

1) Useful Concepts

Functions break code into smaller chunks and are great for tasks that a program uses repeatedly. Instead of writing the same code each time the program needs to perform the task, just call the function!

1. First, we have to define a function. A function definition is actually the code that runs when we call the function. This would be the reusable code described above which we want to use in multiple places.
   A function does not always perform a single thing. It can perform a bunch of things in a certain category. e.g we want to write a function that converts the temperature to celsius when Fahrenheit temperature is passed and vice versa. We can do this using function parameters/arguments. We can use these to change the function behavior based on what parameters are passed.

```
def                 <function_name>(param1,              param2,              ....):
    // function body
```

2. After we have defined a function, we want to use it. Using a function is known as function calls. We do this by writing the function name and then using a pair of opening and closing round brackets.
   If we want to pass some function parameters we can write them inside the round brackets separated by commas.
   It is not mandatory to pass parameters if your function definition didnt specify them.

```
<function_name>(param1, param2, ....)
```

2) Solved Lab Activities

Sr.No	*Allocated Time*	*Level of Complexity*	*CLO Mapping*
1	*15*	*Low*	*CLO-7*
2	*15*	*Low*	*CLO-7*
3	*15*	*Low*	*CLO-7*
4	*15*	*Low*	*CLO-7*

Activity 1:

Python program to illustrate a function definition

Solution:

```
def example_function():
    x = 5
    print("The value of x is:", x)
example_function()
```

Output

```
The value of x is:5
```

Activity 2:

Python program to illustrate a function definition with params

```
def example_function(num1,num2):
    x = num1 + num2
    print("The sum is:", x)
example_function(1,2)
example_function(2,10)
```

Output

```
The sum is:3
```

```
The sum is:12
```

Activity 3:

Python program to illustrate a function definition with return

```
def example_function(num1,num2):
    return num1 + num2

print(example_function(1,2))
```

```
print(example_function(3,23))
```

Output

```
3
26
```

Activity 4:

Python program to illustrate a function definition with default/optional params

```
Def example_function(num1,num2=10):
    return num1 + num2

print(example_function(1,2))
print(example_function(1))
print(example_function(8))
```

Output

```
3
11
18
```

3) Graded Lab Tasks

Note: The instructor can design graded lab activities according to the level of difficulty and complexity of the solved lab activities. The lab tasks assigned by the instructor should be evaluated in the same lab.

Lab Task 1

Write a calculator function that takes in as a param two numbers and a string specifying the operation and returns the result. It should be able to do addition, subtraction, multiplication, and division.

Lab Task 2

Write a program that takes in 4 numbers as params and returns the largest number of them all.

Lab Task 3

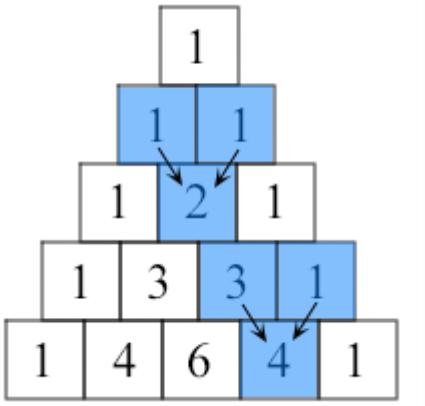
Write a function that takes in a number and returns the Fibonacci sequence till that number.
The Fibonacci Sequence is the series of numbers:
0, 1, 1, 2, 3, 5, 8, 13, 21, 34, ...
The next number is found by adding up the two numbers before it

Lab Task 4

Write a function that takes in params for different shape measurements and returns the area of that shape depending upon those measurements and the type of shape. If the type of shape is not specified it should by default calculate the area of a rectangle. It should be able to calculate the area of square, rectangle, circle, and triangle.

Lab Task 5

Write a Python function that prints out the first n rows of Pascal's triangle
Sample Pascal's triangle:



Each number is the two numbers above it added together