

**DEPARTMENT OF INFORMATION TECHNOLOGY**

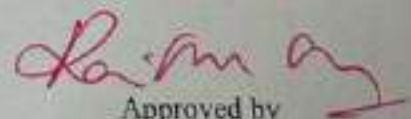
**IT WORKSHOP  
LAB MANUAL**

**II B.Tech-I SEMESTER  
(Academic Year: 2019-20)**



Prepared by

**Ms.D.Vandana**



Approved by

**Dr. I. Ravi Prakash Reddy**

**HOD IT**

## **COURSE OBJECTIVES:**

1. Acquire knowledge of PC & its peripherals and learn installation of OS.
2. Create web pages using HTML + CSS and create spreadsheets using MS Excel.
3. Describe the core syntax and semantics of Python programming language.
4. Illustrate the process of structuring the data using lists, dictionaries, tuples, sets and strings.
5. Handle files and modules in python.
6. Develop the skills of using python libraries.

Department of IT, GNITS

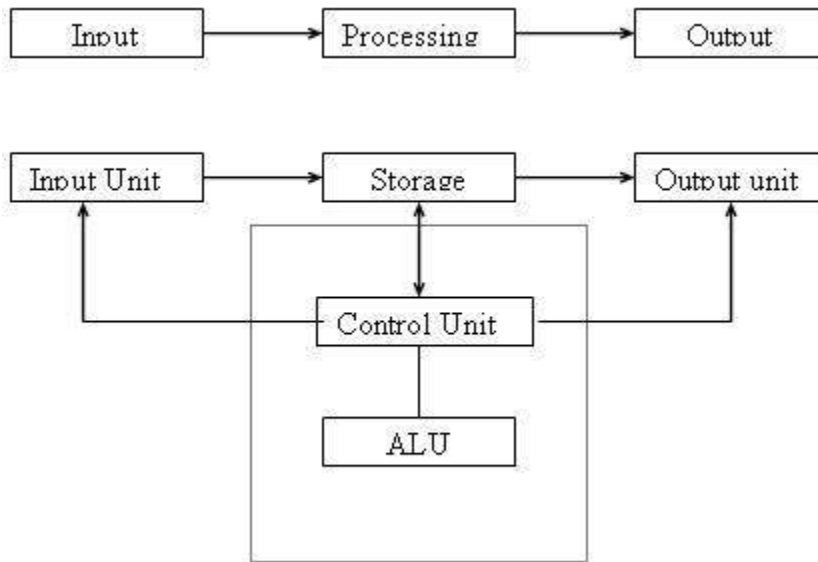


*1. PC HARDWARE*

**AIM:**

- a) Identify the peripherals of a computer, components in a CPU and its functions.
- b) Draw the block diagram of the CPU along with the configuration of each peripheral.

**Block Diagram of Computer:**



A computer can process data, pictures, sound and graphics. They can solve highly complicated problems quickly and accurately.

**InputUnit:**

Computers need to receive data and instruction in order to solve any problem. Therefore we need to input the data and instructions into the computers. The input unit consists of one or more input devices. Keyboard is the one of the most commonly used input device. Other commonly used input devices are the mouse, floppy disk drive, magnetic tape, etc. All the input devices perform the following functions.

1. Accept the data and instructions from the outside world.
2. Convert it to a form that the computer can understand.
3. Supply the converted data to the computer system for further processing.

**StorageUnit:**

The storage unit of the computer holds data and instructions that are entered through the input unit, before they are processed. It preserves the intermediate and final results before these are sent to the output devices. It also saves the data for the later use. The various storage devices of a computer system are divided into two categories.

- **Primary Storage:** Stores and provides very fast. This memory is generally used to hold the program being currently executed in the computer, the data being received

from the input unit, the intermediate and final results of the program. The primary memory is temporary in nature. The data is lost, when the computer is switched off. In order to store the data permanently, the data has to be transferred to the secondary memory. The cost of the primary storage is more compared to the secondary storage.

- **Secondary Storage:** Secondary storage is used like an archive. It stores several programs, documents, data bases etc. The programs that you run on the computer are first transferred to the primary memory before it is actually run. Whenever the results are saved, again they get stored in the secondary memory. The secondary memory is slower and cheaper than the primary memory. Some of the commonly used secondary memory devices are Hard disk, CD, etc.,

#### **Memory Size:**

All digital computers use the binary system, i.e. 0's and 1's. Each character or a number is represented by an 8 bit code.

The set of 8 bits is called a byte. A character occupies 1 byte space.

A numeric occupies 2 byte space.

Byte is the space occupied in the memory.

The size of the primary storage is specified in KB (Kilobytes) or MB (Megabyte). One KB is equal to 1024 bytes and one MB is equal to 1000KB. The size of the primary storage in a typical PC usually starts at 16MB. PCs having 32 MB, 48MB, 128 MB, 256MB memory are quite common.

#### **Output Unit:**

The output unit of a computer provides the information and results of a computation to outside world. Printers, Visual Display Unit (VDU) are the commonly used output devices. Other commonly used output devices are floppy disk drive, hard disk drive, and magnetic tape drive.

#### **Arithmetic Logical Unit:**

All calculations are performed in the Arithmetic Logic Unit (ALU) of the computer. It also does comparison and takes decision. The ALU can perform basic operations such as addition, subtraction, multiplication, division, etc and does logic operations viz, >, <, =, 'etc. Whenever calculations are required, the control unit transfers the data from storage unit to ALU once the computations are done, the results are transferred to the storage unit by the control unit and then it is send to the output unit for displaying results.

**Control Unit:**

It controls all other units in the computer. The control unit instructs the input unit, where to store the data after receiving it from the user. It controls the flow of data and instructions from the storage unit to ALU. It also controls the flow of results from the ALU to the storage unit. The control unit is generally referred as the central nervous system of the computer that control and synchronizes its working.

**Central Processing Unit:**

The control unit and ALU of the computer are together known as the Central Processing Unit (CPU). The CPU is like brain performs the following functions:

- It performs all calculations.
- It takes all decisions.
- It controls all units of the computer.

A PC may have CPU-IC such as Intel 8088, 80286, 80386, 80486, Celeron, Pentium, Pentium Pro, Pentium II, Pentium III, Pentium IV, Dual Core, and AMD etc.

**Introduction to Computer Hardware:**

Hardware is the physical appearance of the devices or tools. It is what which we can touch and feel.

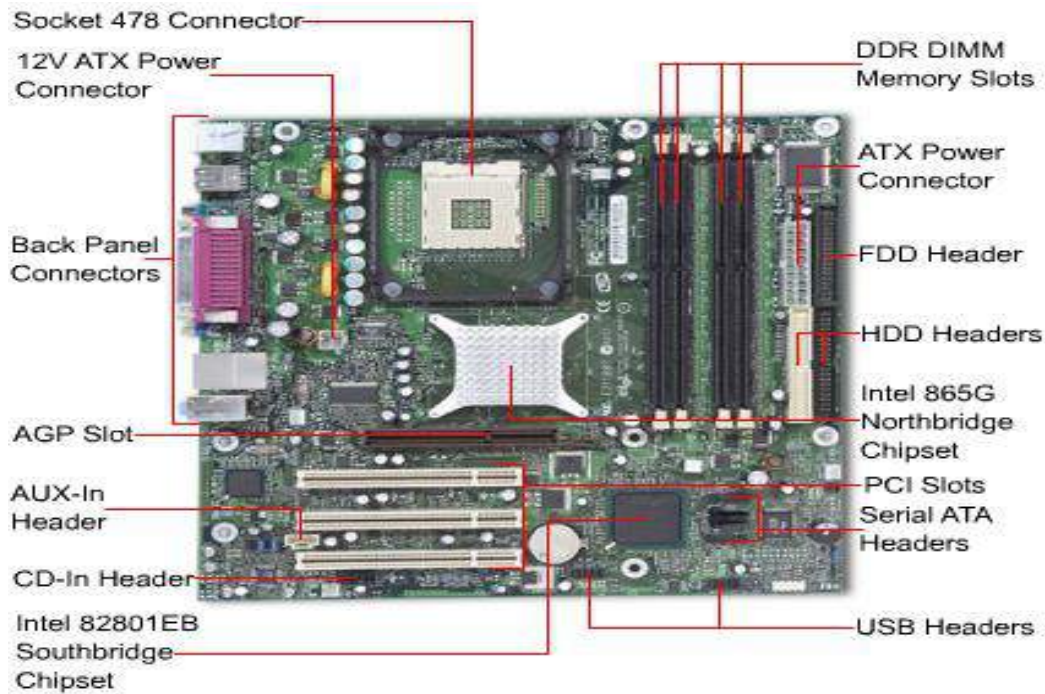
Computer Hardware consists of the Monitor, CPU, Keyboard, Mouse and all other devices connected to the computer either externally or internally.

A typical computer (personal computer, PC) consists of a desktop or tower case (chassis) and the following parts:

- CPU The central processing unit contains the heart of any computer, the processor. The processor is fitted on to a Mother Board. The Mother Board contains various components, which support the functioning of a PC.



- System board/Motherboard which holds the Processor, Random Access Memory and other parts, and has slots for expansion cards



**3. RAM (Random Access Memory)-** For program execution and short term data-storage, so the computer doesn't have to take the time to access the hard drive to find something. More RAM can contribute to a faster PC.

The main memory of the computer is called as Random Access Memory (RAM). The name derives from the fact that data can be stored in and retrieved at random, from anywhere in the electronic main memory chips in approximately the same amount of time, no matter where the data is.

Main memory is in an electronic or volatile state. When the computer is off, main memory is empty, when it is on it is capable of receiving and holding a copy of the software instructions, and data necessary for processing.

Because the main memory is a volatile form of storage that depends on electric power can go off during processing, users save their work frequently on to nonvolatile secondary storage devices such as diskettes or hard disk.

**The main memory is used for the following purposes:**

1. Storage of the copy of the main software program that controls the general operation of the computer. This copy is loaded on to the main memory when the computer is turned on, and it stays there as long as the computer is on.
2. Temporary storage of a copy of application program instruction, to be received by CPU for interpretation and processing or execution.
3. Temporary storage of data that has been input from the key board, until instructions call for the data to be transferred in to CPU for processing.

Temporary storage of data, which is required for further processing or transferred as output to output devices such as screen, a printer, a disk storage device.

### **ROM (Read Only Memory)**

Instructions which are critical to the operation of a computer are stored permanently on Read only Memory. (ROM) chip installed by the manufacturer inside the computer. This ROM chip is also called firm ware, retains instructions in a permanently accessible nonvolatile form. When the power in the computer is turned off, the instructions stored in ROM are not lost. It is necessary and also convenient to have instructions stored in ROM.

The more instructions in ROM, the fewer diskettes you may have to handle. Until recently the process of manufacturing ROM chips and recording data on them was more expensive than the process of producing RAM chips.

As a result the manufacturers tended to record in ROM only those instructions that were crucial to the operation of the computer. Today, due to improvements in the manufacturing process of ROM chips have lowered the cost to the point where manufacturers are beginning to include additional software instructions.

In addition to ROM, three additional categories of nonvolatile memory are used in some computer systems. They are PROMs, EPROMs, and EEPROMs.

PROM stands for programmable read only memory. It works similar to that of ROM. PROM chips are custom made for the user by the manufacturer. The user determines what data and instructions are to be recorded on them. The data on PROM is permanent and cannot be erased. Erasable programmable read only memory (EPROM), developed as an improvement over PROM. The data on the EPROM can be read with the help of a special device that uses ultra violet light. The data or instructions on the EPROM are erasable and new data can be entered in its place. EPROM functions exactly same as PROM. Electronically erasable programmable read only memory (EEPROM) avoids the inconvenience of having to take chips out of the computer to change data and instructions. Changes can be made electrically under software control. These are used in point of sale terminals to records price related data for products. The prices recorded



on them can be easily updated as needed. The only disadvantage of EEPROM is, the regular ROM chips.

### Differences between ROM & RAM

ROM (Read only memory)

1. You can only read the data.
2. Data can't be written every time, to write the data we need PROM, EPROM, OR EEPROM.
3. ROM is nonvolatile in nature. The data stored in ROM is permanent in nature.
4. Size of the ROM has nothing to do with processing.

RAM (Random access memory):

1. You can read and write data on the chip.
2. RAM has volatile memory. It loses its contents when the power is switched off.
3. Size of the RAM makes difference in the processing i.e., bigger the size of the RAM more is the speed of processing.
4. The data can be read and written at any time.

### Dynamic RAM (DRAM)



- Synchronous DRAM (SDRAM)



- Static RAM (SRAM)



**4. Buses: PCI bus, PCI-E bus, ISA bus (outdated), USB, AGP**



**5. Power Supply – a case that holds a transformer, voltage control**



6. Storage controllers, of IDE, SCSI or other type, that control hard disk, floppy disk, CD-ROM and other drives; the controllers sit directly on the motherboard (on-board) or on expansion cards
7. Video display controller that produces the output for the computer display
8. computer bus controllers (parallel, serial, USB, Fire wire) to connect the computer to external peripheral devices such as printers or scanners
9. Some type of a removable media writer:
10. CD - the most common type of removable media, cheap but fragile.

CD-ROM, , CD-RW, CD-R, DVD, DVD-ROM., DVD-RW, DVD-R,



### 11. Floppy disk



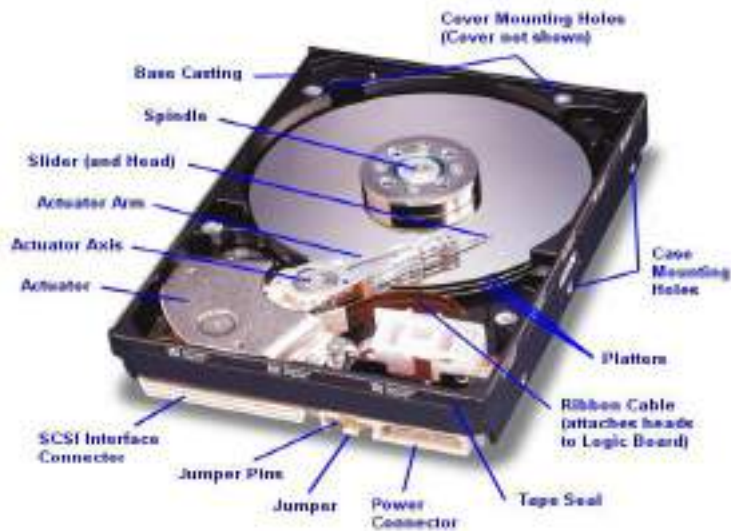
### Floppy Disk Drive



12. Tape Drive - mainly for backup and long-term storage

13. Internal storage - keeps data inside the computer for later use.

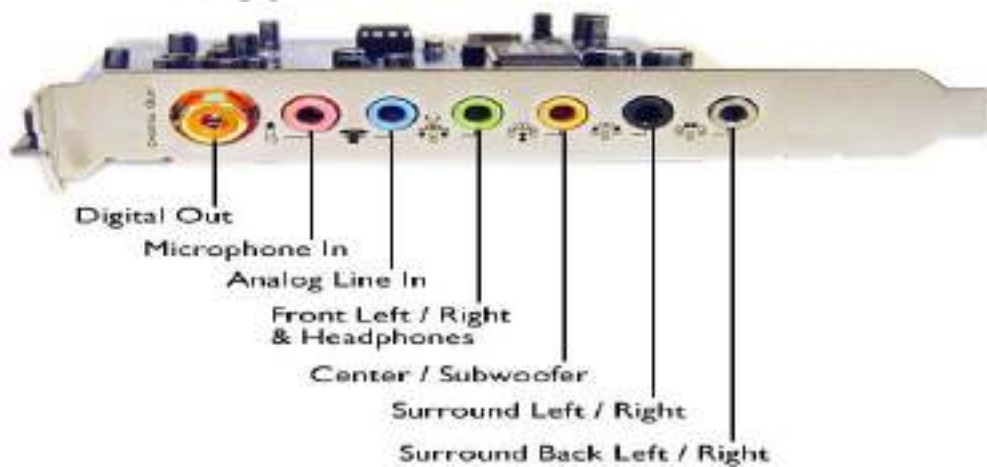
14. Hard disk - for medium-term storage of data.



### 15. Disk array controller



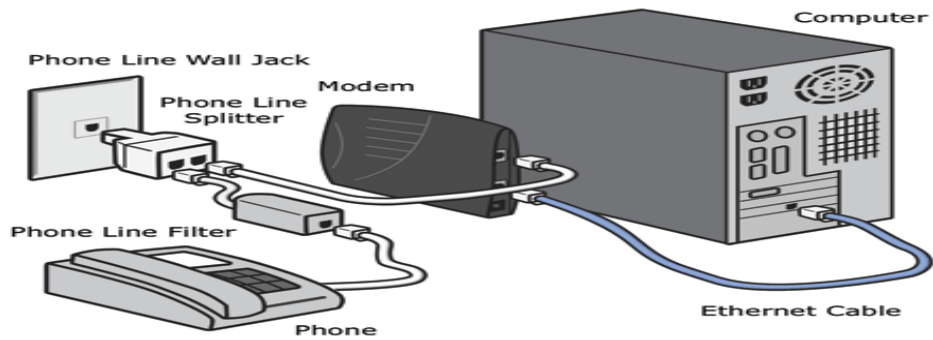
**16. Sound card** - translates signals from the system board into analog voltage levels, and has terminals to plug in speakers.



**17. Networking** - to connect the computer to the Internet and/or other computers



### 18. Modem - for dial-up connections



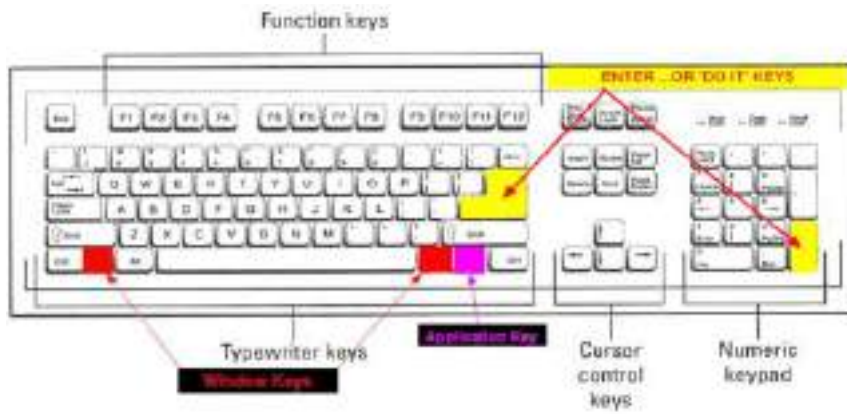
### 19. Network card - for DSL (Digital Subscriber Line)/Cable internet, and/or connecting to other computers.



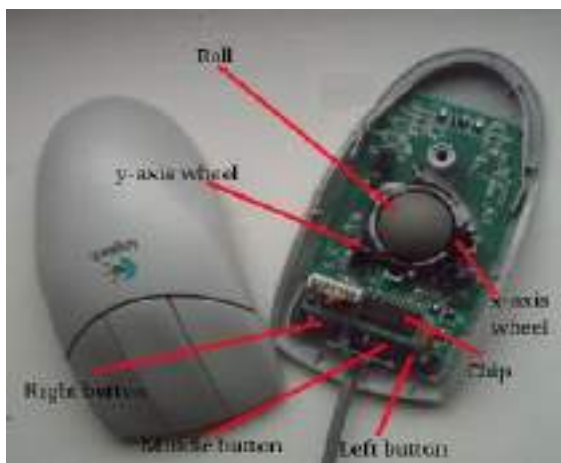
### 20. Other peripherals: In addition, hardware can include external components of a computer system. The following are either standard or very common.

**Standard input devices: Keyboard**





**Mouse**



**Alternate input devices:**

Pens, Touch screens, Game controllers (joy stick), Touch pad, Trackball.

**Optical input devices:** Barcode reader, Image scanners.

**Audio visual input devices:** Microphones, Video input, Digital cameras

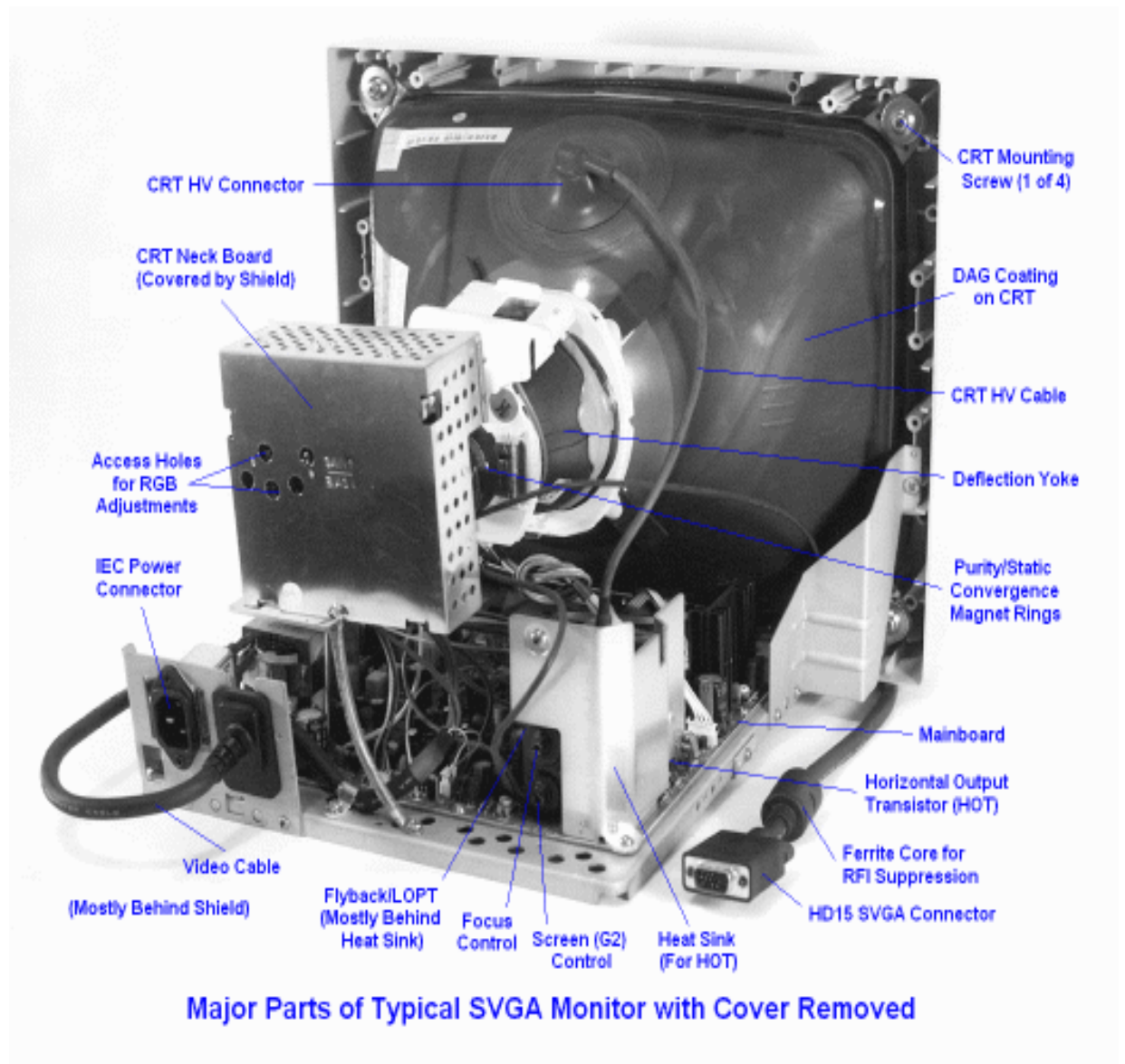


**21. Output:** The output devices are:

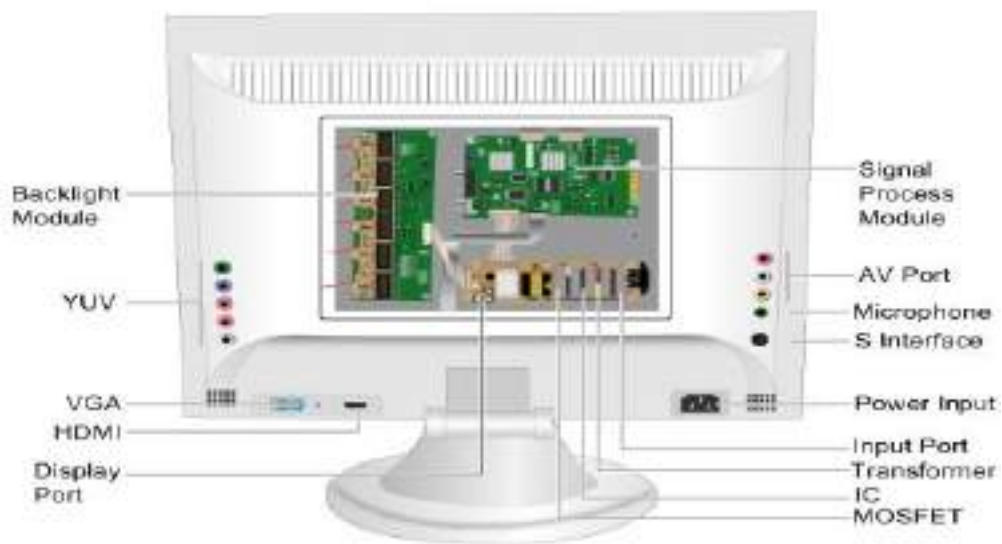
**Standard output devices: Monitor**

**Types of monitors**

## CRT (cathode ray tube) monitors



## LCD (liquid crystal display) monitors



## Printer

### Types of printers

**Impact printers:** An impact printer create an image by using pins or hammers to press an inked ribbon against the paper.ex. Dot matrix printer.

Non-impact printers: This type uses other means to create an image for example in ink jet printers tiny nozzles are used to spray droplets of ink on the page.ex: Ink jet printer

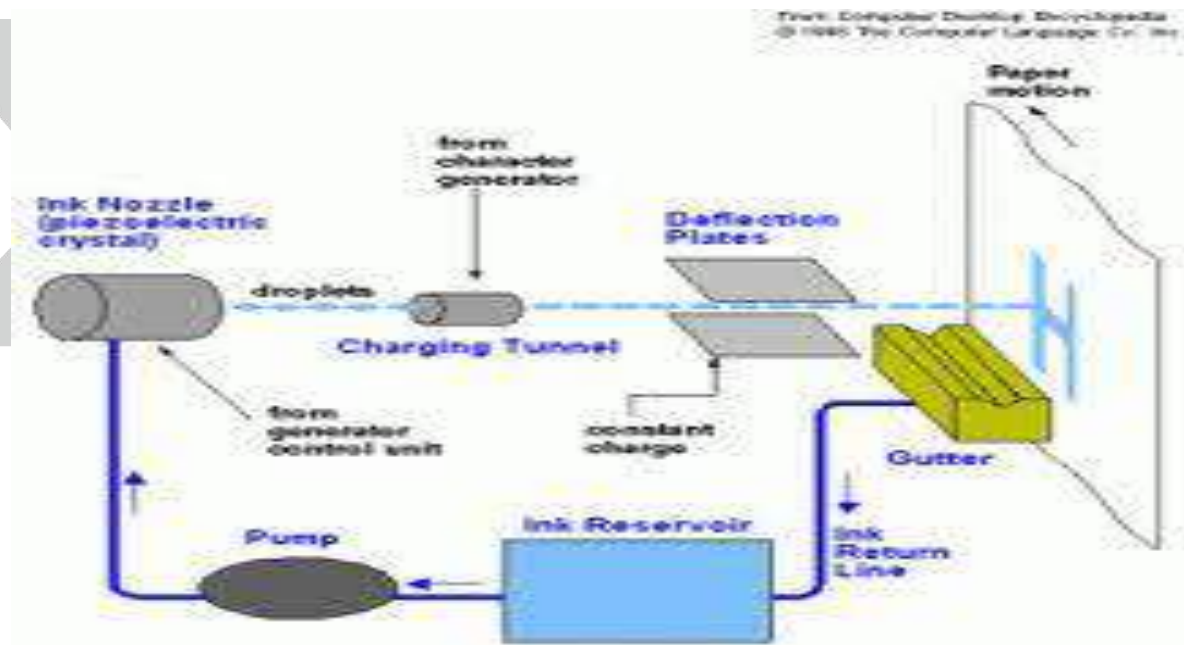
**Laser**

**Printer**

**Dot- Matrix printer:**







## Inkjet printer

### Other output devices:

Speakers, LCD projectors, Networking, Network card

#### Speakers



#### LCD Projectors:



### Viva Questions:

- 1) Define a computer?
- 2) Define hardware and software?
- 3) What are the functional units of a computer?
- 4) Define the following: RAM, ROM, BIOS, BUS, BIT, and PROGRAM.
- 5) What is the use of a mother board?

### **Week1-TASK 3:**

#### **AIM:**

Assembling and disassembling the PC back to working condition.

Safety Precautions:

1. Beware of electrostatic discharge (ESO)
2. Build computer on a hard surface, away from concepts.
3. Wear shoes and the short sleeved cotton wear.
4. Use Phillips, head screw driver.
5. Keep the components away from moisture.
6. Avoid using pressure while installing.

#### **PROCEDURE:**

##### **Steps for Assembling**

- Fix the SMPS on the cabinet of PC using the screws provided.
- Fix the motherboard on the cabinet of PC using the screws provided.
- Connect the power cables from SMPS to motherboard.
- Insert the preprocessor into the slot provided such that the corner with no pin coincide with corner without pinhole on motherboard.
- Apply the appropriate adhesive on the processor for fixing the processor fan.
- Fix the processor fan on the processor and use clips on it to keep it firm.
- Connect the power cable to the processor fan
- Insert the RAM card into the slots provided on the motherboard.
- Set the jumpers setting on the hard disc drive.
- Fix the hard disc drive in the space provided in the PC cabinet using screws provided.
- Fix the FDD in the space provided in the PC cabinet using screws provided.
- Fix the CD-ROM in the space provided in the PC cabinet using screws provided.
- Connect the FDD, HDD, CD-ROM drive to motherboard using flat ribbon.
- Connect power supply to the HDD, FDD, and CD-ROM drive using the cables from the SMPS.
- Connect wires of speakers and lights of cabinet to the motherboard.
- Connect the network interface and other cards to motherboard by inserting in right slots and fix them in cabinet using the screws provided.
- Place the cabinet in right position.

- Fix the doors of the cabinet.
- Connect the data cable of monitor to the CPU.
- Connect the keyboard cable to the CPU.
- Connect the mouse cable to the CPU.
- Connect other devices to CPU.
- Connect the LAN cable to NIC in CPU.
- Connect the power supply to CPU.
- Connect the power supply to Monitor.
- Switch on the computer after giving the power supply.

Getting the Cabinet ready:-

1. Check how to open the cabinet and determine where to fix the components.
2. Determine if the case has the appropriate risers installed.

Preparing to fit the Components:

1. Network adapter drive.
2. Floppy disk drive.
3. Ribbon cables.
4. Hard disk.
5. CD-ROM Drive.
6. RAM
7. CPU
8. Heat sink / cooler / fan.
9. Mother board.
10. Screws.

Fitting the Mother board.

1. Line up the patch on the motherboard ( ps/1, USB, etc ) with the appropriate holes in the block panel I/O shield of the case.
2. Check the points where you and to install
3. Install them and make the mother board sit on them and fix screws if required.

Mother board parts:

1. ACR slot.
2. PCI Slot.
3. AGP Slot.

4. ATX Connectors.
5. CPU Fan.
6. Chipset North Bridge.
7. CPU socket.
8. Floppy.
9. System memory.
10. Chipset south bridge.
11. Panel connector.
12. Power supply.
13. IDE connectors.

ATX Connectors:

1. PS, Mouse.
2. Key board.
3. USB.
4. Parallel ( Prints )
5. Serial COM1.
6. Serial COM 2.
7. Joystick.
8. Sound.

Fitting the processor:

1. Raise the small lever at the side of the socket.
2. Notice that there is a pin missing at one corner, determine the direction to fit in the processor.
3. You should not force the CPU. When inserting it. All pins should slide smoothly into the socket.
4. Lock the lever back down.
5. Install the heat sink over it (Different type for each processor). Heat sink / CPU fan.

Fitting the RAM:

1. The RAM must be suitable for motherboard.
2. There are currently 3 types of RAM available.
  - a) SD RAM.
  - b) DDR SD RAM.
  - c) RD RAM.

3. The mother board's chipset determines which type of RAM may be used.

#### Installing the PCI Cards:

1. Most of the cards are inbuilt these days.
2. NIL, Sound Cards etc. are fitted into PCI slots.

#### Fitting the hard disk and Floppy disk:

1. Place the floppy and hard disks in their slots.
2. Leave some space above HDD to prevent heat building.
3. Check the jumper configuration.
4. Fix the screws.

#### Installing the CD-ROM Drives:

1. CD-ROM drive is similar to installing a hard disk.
2. 1<sup>ST</sup> check that the jumper configuration is correct.
3. Fix the screw.

#### Connecting the ribbon Cables:-

1. Attach the long end of the cable to the IDEU connector on the motherboard first.
2. The red stripe on the IDE cable should be facing the CD Power.

#### Powering the driver and motherboard:

##### Connecting the cables for the case front pane

1. SD, SPK or SPEAK: The loud speakers o/p. it has 4 pins.
2. RS, RE, RS or RESET: Connect the two pin Reset cable here.
3. PWR, PW, PWSW, PS or power SW: Power switch, the pc's on (switch, the plug is two pin).
4. PWLED, PWRLED or Power LED: The light emitting diode on the front panel of the case illuminates when the computer is switched on. It's a 2-pin cable.
5. HD, HDD, and LED: These two pins connect to the cable for the hard disk activity LED.

#### Final Check:-

1. Mother board jumper configurations are the settings for the processor operator.
2. Drive jumper settings, master/ slave correct?
3. Are the processor, RAM modules and plug in cards finally seated in there sockets?
4. Did you plug all the cables in? Do they all fit really?

5. Have you tightened all the screws in plug-in cards or fitted the clips?
6. Are the drives secure?
7. Have you connected the power cables to all drives?

Powering up for the first time:

1. Ensure that no wires are touching the CPU heat sink fan.
2. Plug your monitor, mouse and keyboard.
3. Plug in power card and switch the power supply.
4. If everything is connected as it should be
  - All system fans should start spinning.
  - You should hear a single beep and after about 5-10 sec.
  - Amber light on monitor should go green.
  - You will see computer start to boot with a memory check.
  - Now check front LEDs to see if you plugged them in correctly.
  - Check all other buttons.
  - Power should change any wrong settings.

OUTPUT



#### VIVA QUESTIONS

1. What is SMPS?
2. What is the purpose of heat sink?
3. What is the purpose of USB port?
4. How many slots are there for RAM?

#### Steps for Disassembling

- Switch off the power supply
- Disconnect the power supply cable from monitor.
- Disconnect the power supply cable from CPU.
- Disconnect the LAN cable to NIC in CPU.
- Disconnect the other devices in CPU such as printers.
- Disconnect the mouse cable from CPU.
- Disconnect the keyboard cable from CPU.
- Disconnect data cable of monitor from CPU.

- Remove the doors of cabinet.
- Place the cabinet such that motherboard faces the ceiling.
- Disconnect the NIC and other cards from mother board by removing from slots and unscrewing from cabinet.
- Disconnect the wires of speakers from mother board.
- Remove power supply cables from HDD, FDD, CD-ROM drive etc.
- Disconnect the HDD, FDD, CD-ROM drive from mother board by removing flat ribbon cable.
- Remove CR-ROM from cabinet.
- Remove the FDD from cabinet by unscrewing it.
- Remove the HDD from cabinet by unscrewing it.
- Removing RAM cards from slots on mother board.
- Disconnect the power cables from processor fan.
- Remove the processor fan by unlocking clips on it.
- Disconnect the power cables from SMPS on power cabinet.
- Remove mother board from cabinet by unscrewing it.
- Remove the SMPS from cabinet of PC by unscrewing it.

### **OUTPUT**

System is disassembled.

### **VIVA QUESTIONS**

1. What is SMPS?
2. What is the purpose of heat sink?
3. What is the purpose of USB port?
4. How many slots are there for RAM?
5. What for AGP slots are used?



**d) Installation of OS (Windows and Linux)**

Department of IT, GNITS



Week 2: HTML & CSS

- a) Develop pages using HTML consisting of Text, images, tables, lists, Hyperlinks.
- b) Develop pages using HTML frames and Style sheets.

**AIM**

**Develop pages using HTML consisting of Text, images, tables, lists, Hyperlinks.**

**PROCEDURE:**

Click on start → go to run and type notepad click on ok button and start writing the HTML tags to create a resume

```
<head>  
<style>  
table,th,td  
{
```

```

border: 1px solid black;
border-collapse: collapse;
}
th,td
{
padding: 15px;
text-align: center;
}
table#t01
{
width: 75%;
background-color: pink;
}
</style>
</head>
<body>
<font size="10" color="black"
face="algerian"><caption><center>RESUME</center></caption></font><br>

<b>HRUTHIKA EEDUPUGANTI</B><br>8-3-677/13,<br>S K D
Nagar,<br>Hyderabad-500073.<br>Mobile number: 9494249552<br>email-
id: hruthikaeedupuganti20@gmail.com<br>
<hr>
<mark><b>CAREER OBJECTIVE</b></mark><br>To build a prospering
career in the IT industry by availing every single opportunity that strikes me
and prove myself worthy.
<br><br>
<mark><b>ACADEMIC QUALIFICATIONS</b></mark><br>
<table border="2" style="width:75%">
<tr>
<th>QUALIFICATION</th>
<th>NAME OF INSTITUTE</th>
<th>YEAR OF PASSING</th>
<th>AGGREGATE</th>
</tr>
<tr>
<td>Graduation</td>
<td>G. Narayanamma Institute Of Technology And Science, Hyderabad</td>
<td>2022</td>
<td>90%</td>
</tr>
<tr>

```

Senior Secondary Certification
Sri Chaitanya Junior Kalasala, TS Board
2018
97%

Secondary Certification
A M S P Obul Reddy Public School, CBSE Board
2016
10 CGPA

<b>OTHER QUALIFICATIONS</b>
Basic knowlwdge of computer
NPTEL Certification with Silver Badge in C
<b>EXPERIENCE</b>
Assistant Manager at XYZ Solutions since 2022
<b>PERSONAL SKILL</b>
Team worker
Accepting challenges
<b>PERSONAL PROFILE</b>
<li>Father's name: E Sudhakar</li> <li>Date of birth: 20-10-2000</li> <li>Sex: Female</li> <li>Marital status: Unmarried</li> <li>Languages known: English, Hindi, Telugu</li> <li>Nationality: Indian</li> <li>Hobbies: Gardening, Listening music</li>
<b>DECLARATION</b>
I hereby declare that the above furnished information is true to the best of my knowledge.
<b>Date:</b> 11-07-2023
<b>Place:</b> Hyderabad
HRUTHIKA EEDUPUGANTI

## OUTPUT

## RESUME

### HRUTHIKA EEDUPUGANTI

8-3-677/13,

S K D Nagar,

Hyderabad-500073.

Mobile number: 9494249552

email-id: [hruthikaeedupuganti20@gmail.com](mailto:hruthikaeedupuganti20@gmail.com)



### **CAREER OBJECTIVE**

To build a prospering career in the IT industry by availing every single opportunity that strikes me and prove myself worthy.

### **ACADEMIC QUALIFICATIONS**

QUALIFICATION	NAME OF INSTITUTE	YEAR OF PASSING	AGGREGATE
Graduation	G. Narayanamma Institute Of Technology And Science, Hyderabad	2022	90%
Senior Secondary Certification	Sri Chaitanya Junior Kalasala, TS Board	2018	97%
Secondary Certification	A M S P Obul Reddy Public School, CBSE Board	2016	10 CGPA

### **OTHER QUALIFICATIONS**

- Basic knowlwdge of computer
- NPTEL Certification with Silver Badge in C

### **EXPERIENCE**

Assistant Manager at XYZ Solutions since 2022

### **PERSONAL SKILL**

- Team worker
- Accepting challenges

### **PERSONAL PROFILE**

- Father's name:E Sudhakar
- Date of birth:20-10-2000
- Sex:Female
- Marital status:Unmarried
- Languages known:English, Hindi, Telugu
- Nationality:Indian
- Hobbies:Gardening, Listening music

### **DECLARATION**

I hereby declare that the above furnished information is true to the best of my knowledge.

**Date:**11-07-2023

**Place:**Hyderabad HRUTHIKA EEDUPUGANTI

## WEEK 2 TASK 2:

### AIM:

Develop pages using HTML frames and Stylesheets

### PROCEDURE:

#### FRAMES1.HTML

```
<HTML>
<HEAD>
<TITLE>A simple frameset document</TITLE>
</HEAD>
<FRAMESET cols="20%, 80%">
  <FRAMESET rows="100, 200">
    <FRAME src="contents_of_frame1.html">
    <FRAME src="contents_of_frame2.gif">
  </FRAMESET>
  <FRAME src="contents_of_frame3.html">
</NOFRAMES>
  <P>This frameset document contains:
  <UL>
    <LI><A href="contents_of_frame1.html">Some neat contents</A>
    <LI><IMG src="contents_of_frame2.gif" alt="A neat image">
    <LI><A href="contents_of_frame3.html">Some other neat contents</A>
  </UL>
</NOFRAMES>
</FRAMESET>
</HTML>
```

that might create a frame layout something like this:

FRAME1	
--------	--

<b>FRAME2</b>	<b>FRAME3</b>
---------------	---------------

### FRAMES2.HTML

```

<HTML>
<HEAD>
<TITLE>A well-designed frameset document</TITLE>
</HEAD>
<FRAMESET cols="20%, 80%">
  <FRAME src="table_of_contents.html">
  <FRAME src="ostrich-container.html">
</FRAMESET>
</HTML>
<!-- In ostrich-container.html: -->
<HTML>
<HEAD>
<TITLE>The fast and powerful ostrich</TITLE>
</HEAD>
<P>
<OBJECT data="ostrich.gif" type="image/gif">
  These ostriches sure taste good!
</OBJECT>
</HTML>

```

### FRAMES3.HTML

```

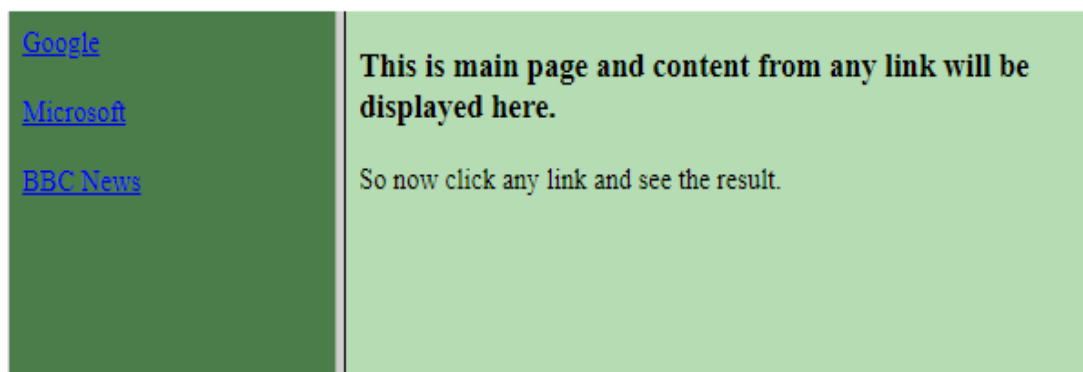
<!DOCTYPE html>
<html>

  <body bgcolor = "#b5dcb3">
    <h3>This is main page and content from any link will be displayed here.</h3>
    <p>So now click any link and see the result.</p>
  </body>

</html>

```

When we load **test.htm** file, it produces following result –



## Styling HTML with CSS

CSS stands for Cascading Style Sheets.

CSS describes **how HTML elements are to be displayed on screen, paper, or in other media.**

CSS **saves a lot of work.** It can control the layout of multiple web pages all at once.

CSS can be added to HTML elements in 3 ways:

- **Inline** - by using the style attribute in HTML elements
- **Internal** - by using a <style> element in the <head> section
- **External** - by using an external CSS file

### **Inline CSS**

An inline CSS is used to apply a unique style to a single HTML element.

An inline CSS uses the style attribute of an HTML element.

This example sets the text color of the <h1> element to blue:

#### **EXAMPLE**

```
<!DOCTYPE html>
<html>
<body>
<h1 style="color:blue;">This is a Blue Heading</h1>
</body>
</html>
```

### **Internal CSS**

An internal CSS is used to define a style for a single HTML page.

An internal CSS is defined in the <head> section of an HTML page, within a <style> element:

#### **Example**

```
<!DOCTYPE html>
<html>
<head>
<style>
body {background-color: powderblue;}
h1 {color: blue;}
p {color: red;}
</style>
</head>
<body>
```



```
<h1>This is a heading</h1>
<p>This is a paragraph.</p>
```

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

## External CSS

An external style sheet is used to define the style for many HTML pages.

**With an external style sheet, you can change the look of an entire web site, by changing one file!**

To use an external style sheet, add a link to it in the `<head>` section of the HTML page:

### Example

```
<!DOCTYPE html>
<html>
<head>
  <link rel="stylesheet" href="styles.css">
</head>
<body>
```

```
<h1>This is a heading</h1>
<p>This is a paragraph.</p>
```

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

### Styles.css

```
body {background-color: powderblue;}
h1  {color: blue;}
p   {color: red;}
```



Department of IT, GNITS

***2: OFFICE EXCEL***

Week 3: MS Office - Excel Spreadsheet Orientation: Accessing, overview of toolbars, saving spreadsheet files, Using help and resources

Creating a Scheduler: Features to be covered: Gridlines, Format Cells, Freezing Rows and Columns, Selecting Ranges, Summation, auto fill, Formatting Text.

## STARTING EXCEL

1. Locate Excel on your computer.
2. Click Microsoft Excel to launch the Excel application and present you with workbook options.
3. Click the first option; “Blank Workbook”.

## THE EXCEL WORKBOOK

Once Excel is started, a blank workbook will open on your screen.

A workbook is an Excel file that contains one or more worksheets (sometimes referred to as spreadsheets). Excel will assign a file name to the workbook, such as **Book1**, **Book2**, **Book3**, and so on, depending on how many new workbooks are opened. **Figure 1.2** shows a blank workbook after starting Excel. Take some time to familiarize yourself with this screen. Your screen may be slightly different based on the version you're using.

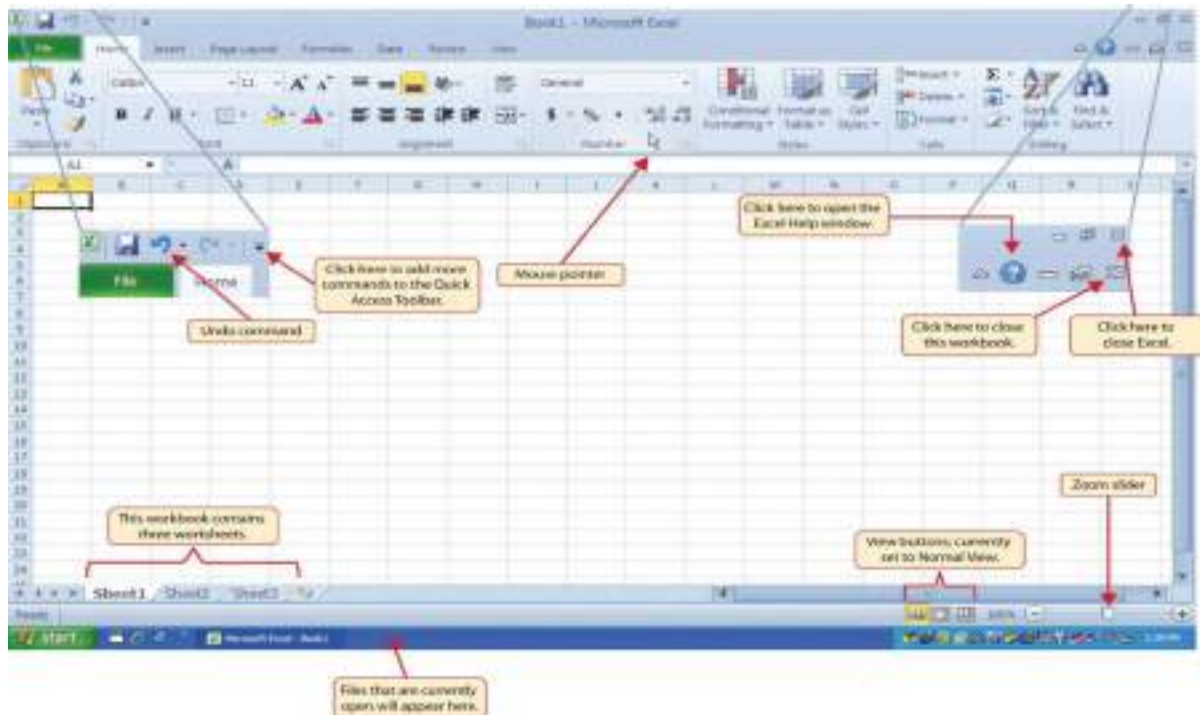


Figure 1.2 Blank Workbook

Your workbook should already be maximized (or shown at full size) once Excel is started, as shown in **Figure 1.2**. However, if your screen looks like **Figure 1.3** after starting Excel, you should click the Maximize button, as shown in the figure.

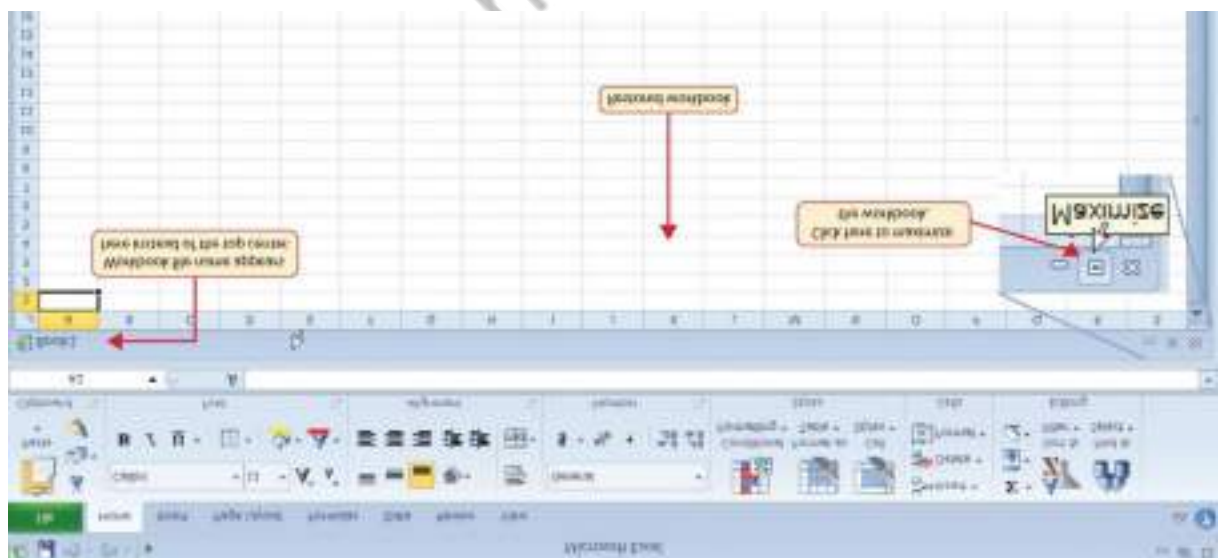


Figure 1.3 Restored Worksheet

## NAVIGATING WORKSHEETS

Data are entered and managed in an Excel worksheet. The worksheet contains several rectangles called cells for entering numeric and nonnumeric data. Each cell in an Excel worksheet contains an address, which is defined by a column letter followed by a row number. For example, the cell that is

currently activated in **Figure 1.3** is A1. This would be referred to as cell location A1 or cell reference A1. The following steps explain how you can navigate in an Excel worksheet:

1. Place your mouse pointer over cell D5 and left click.
2. Check to make sure column letter D and row number 5 are highlighted, as shown in **Figure 1.4**.

*Note: Your highlighted column letter and row number may be different than figure shown.*

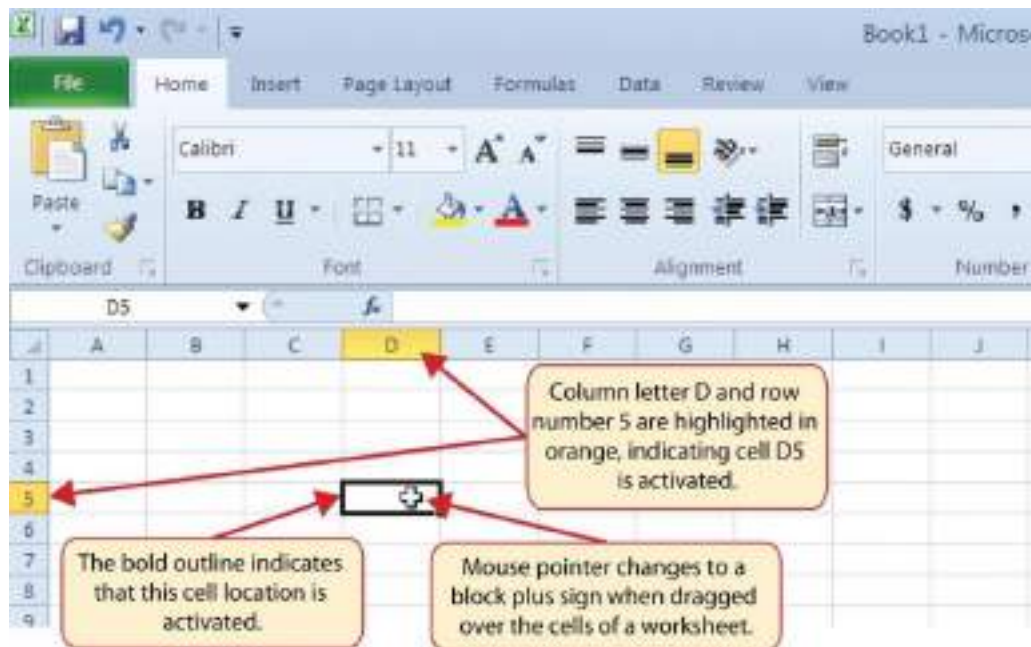


Figure 1.4 Activating a Cell Location

1. Move the mouse pointer to cell A1.
2. Click and hold the left mouse button and drag the mouse pointer back to cell D5.
3. Release the left mouse button. You should see several cells highlighted, as shown in **Figure 1.5**.

This is referred to as a *cell range* and is documented as follows: **A1:D5**. Any two cell locations separated by a colon are known as a cell range. The first cell is the top left corner of the range, and the second cell is the lower right corner of the range.

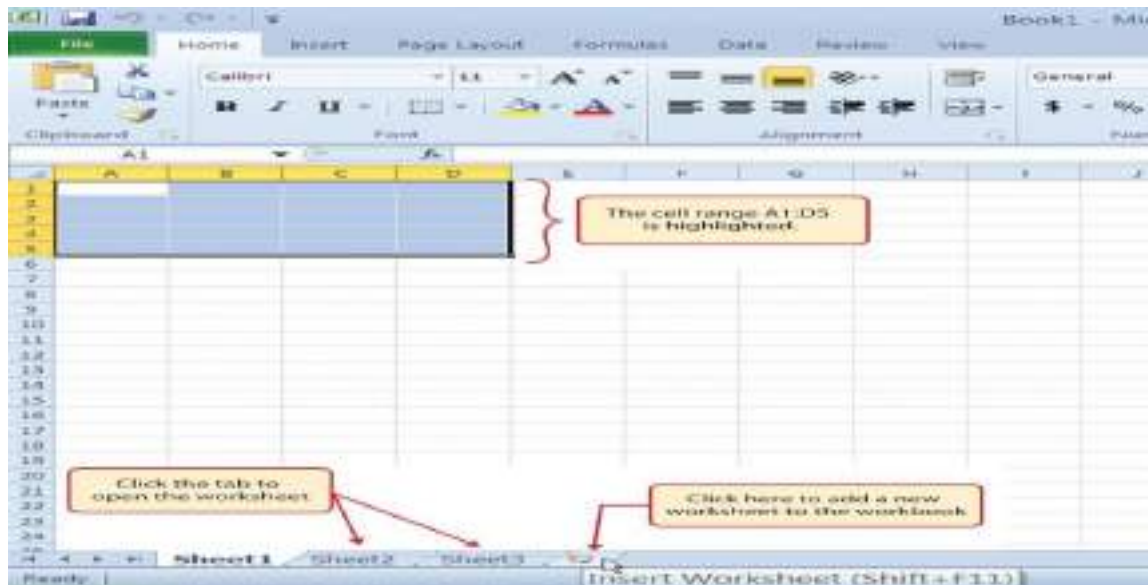


Figure 1.5 Highlighting a Range of Cells

1. At the bottom of the screen, you'll see worksheets. Depending on your version of Excel, you will see either three as displayed above or just one. If you only have one sheet, click the "Insert Worksheet" to add a worksheet. Depending on your version, you instead may have a + sign; a click on the + adds an additional worksheet as well. This is how you open or add a worksheet within a workbook. Add another worksheet so that you now have three sheets displaying here.
2. Click the Sheet1 worksheet tab at the bottom of the worksheet to return to the worksheet shown in **Figure 1.5**.

### Keyboard Shortcuts

### Basic Worksheet Navigation

- Use the arrow keys on your keyboard to activate cells on the worksheet.
- Hold the SHIFT key and press the arrow keys on your keyboard to highlight a range of cells in a worksheet.
- Hold the CTRL key while pressing the PAGE DOWN or PAGE UP keys to open other worksheets in a workbook.

## THE EXCEL RIBBON

Excel's features and commands are found in the Ribbon, which is the upper area of the Excel screen that contains several tabs running across the top. Each tab provides access to a different set of Excel commands. **Figure 1.6** shows the commands available in the Home tab of the Ribbon. **Table 1.1**

“**Command Overview for Each Tab of the Ribbon**” provides an overview of the commands that are found in each tab of the Ribbon.



Figure 1.6 Home Tab of Ribbon

Table 1.1 Command Overview for Each Tab of the Ribbon

Tab Name	Description of Commands
<b>File</b>	Also known as the Backstage view of the Excel workbook. Contains all commands for opening, closing, saving, and creating new Excel workbooks. Includes print commands, document properties, e-mailing options, and help features. The default settings and options are also found in this tab.
<b>Home</b>	Contains the most frequently used Excel commands. Formatting commands are found in this tab along with commands for cutting, copying, pasting, and for inserting and deleting rows and columns.
<b>Insert</b>	Used to insert objects such as charts, pictures, shapes, PivotTables, Internet links, symbols, or text boxes.
<b>Page Layout</b>	Contains commands used to prepare a worksheet for printing. Also includes commands used to show and print the gridlines on a worksheet.
<b>Formulas</b>	Includes commands for adding mathematical functions to a worksheet. Also contains tools for auditing mathematical formulas.
<b>Data</b>	Used when working with external data sources such as Microsoft® Access®, text files, or the Internet. Also contains sorting commands and access to scenario tools.
<b>Review</b>	Includes Spelling and Track Changes features. Also contains protection features to password protect worksheets or workbooks.

Tab Name	Description of Commands
<b>View</b>	Used to adjust the visual appearance of a workbook. Common commands include the Zoom and Page Layout view.

The Ribbon shown in **Figure 1.6** is full, or maximized. The benefit of having a full Ribbon is that the commands are always visible while you are developing a worksheet. However, depending on the screen dimensions of your computer, you may find that the Ribbon takes up too much vertical space on your worksheet. If this is the case, you can minimize the Ribbon by clicking the button shown in **Figure 1.6**. When minimized, the Ribbon will show only the tabs and not the command buttons. When you click on a tab, the command buttons will appear until you select a command or click anywhere on your worksheet.

### Keyboard Shortcuts

#### Minimizing or Maximizing the Ribbon

- Hold down the CTRL key and press the F1 key.
- Hold down the CTRL key and press the F1 key again to maximize the Ribbon.

## QUICK ACCESS TOOLBAR AND RIGHT-CLICK MENU

The Quick Access Toolbar is found at the upper left side of the Excel screen above the Ribbon, as shown in **Figure 1.7**. This area provides access to the most frequently used commands, such as Save and Undo. You also can customize the Quick Access Toolbar by adding commands that you use on a regular basis. By placing these commands in the Quick Access Toolbar, you do not have to navigate through the Ribbon to find them. To customize the Quick Access Toolbar, click the down arrow as shown in **Figure 1.7**. This will open a menu of commands that you can add to the Quick Access Toolbar. If you do not see the command you are looking for on the list, select the More Commands option.



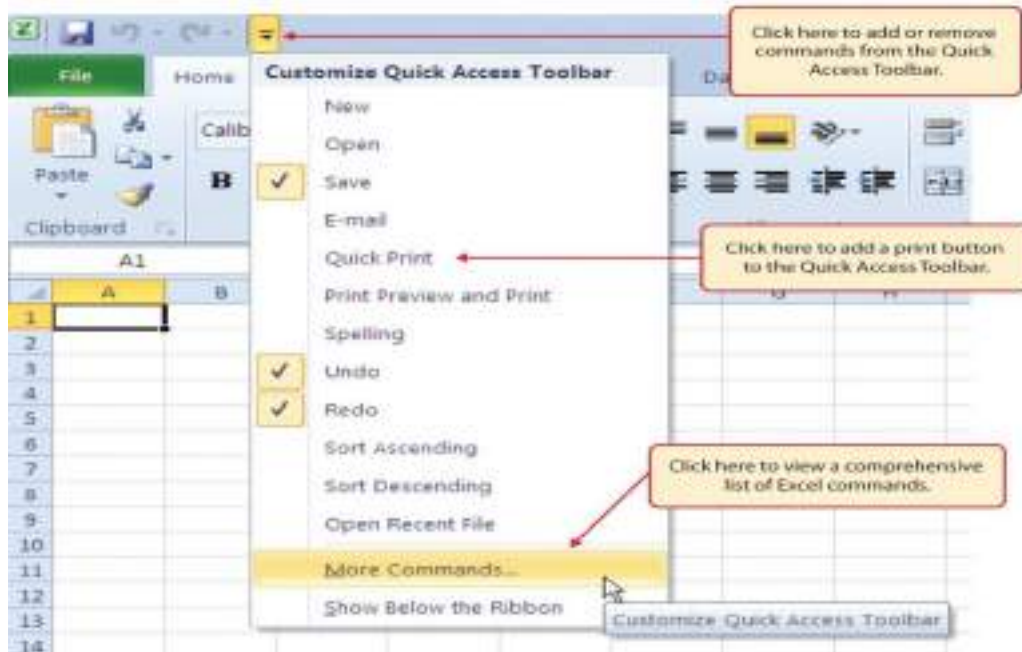


Figure 1.7 Customizing the Quick Access Toolbar

In addition to the Ribbon and Quick Access Toolbar, you can also access commands by right clicking anywhere on the worksheet. **Figure 1.8** shows an example of the commands available in the right-click menu.

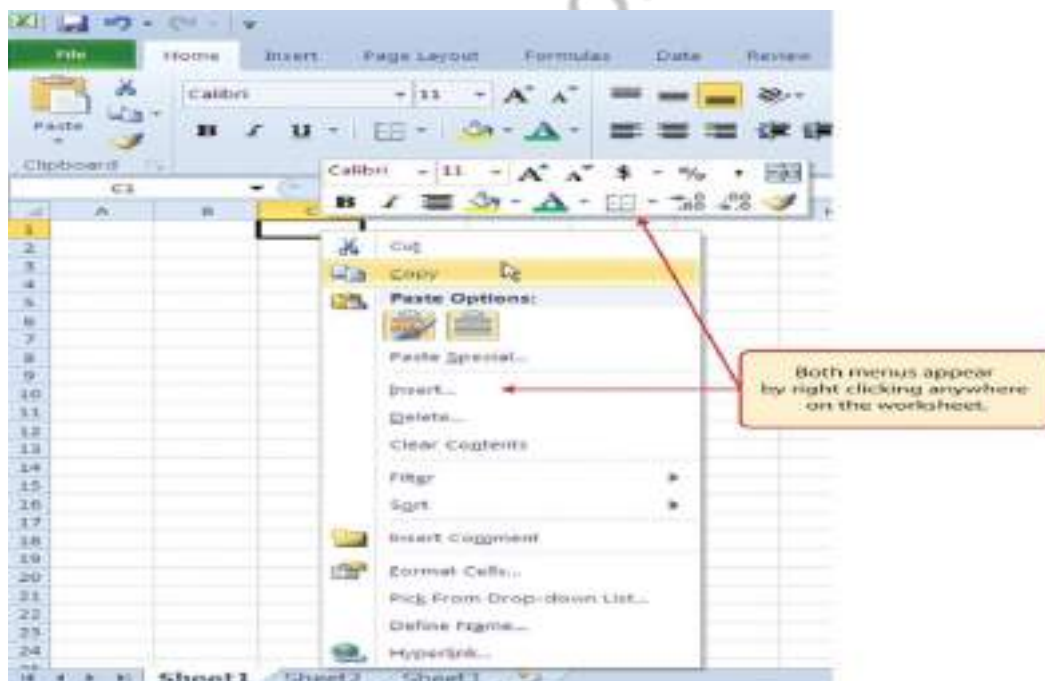


Figure 1.8 Right-Click Menu

## THE FILE TAB

The File tab is also known as the **Backstage** view of the workbook. It contains a variety of features and commands related to the workbook that is currently open, new workbooks, or workbooks stored in other locations on

your computer or network. **Figure 1.9** shows the options available in the File tab or Backstage view. To leave the Backstage view and return to the worksheet, click the arrow in the upper left-hand corner as shown below.

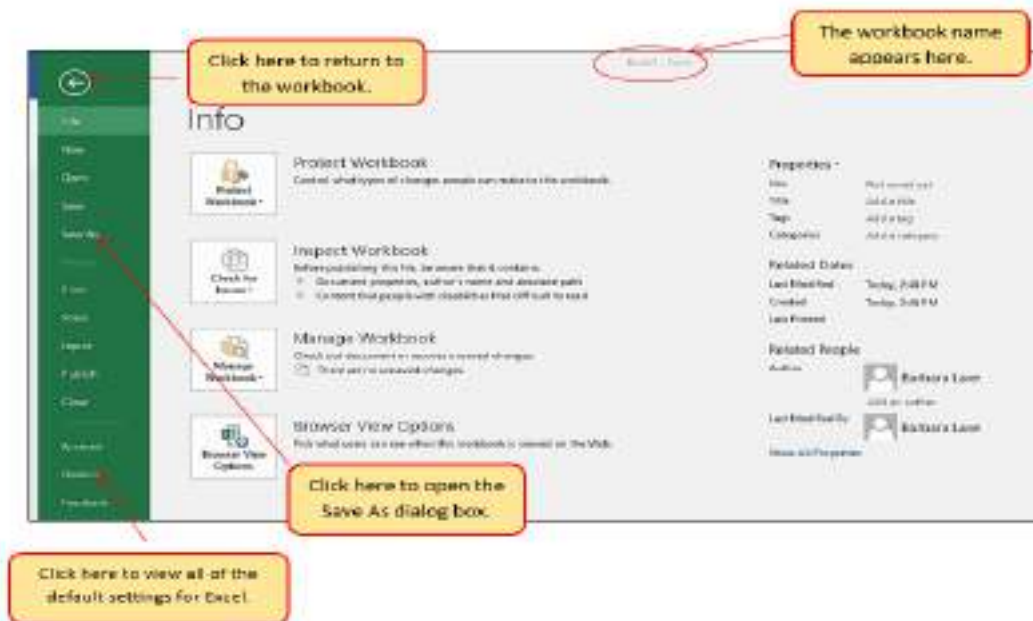


Figure 1.9 File Tab or Backstage View of a Workbook

Included in the File tab are the default settings for the Excel application that can be accessed and modified by clicking the Options button. **Figure 1.10** shows the Excel Options window, which gives you access to settings such as the default font style, font size, and the number of worksheets that appear in new workbooks.

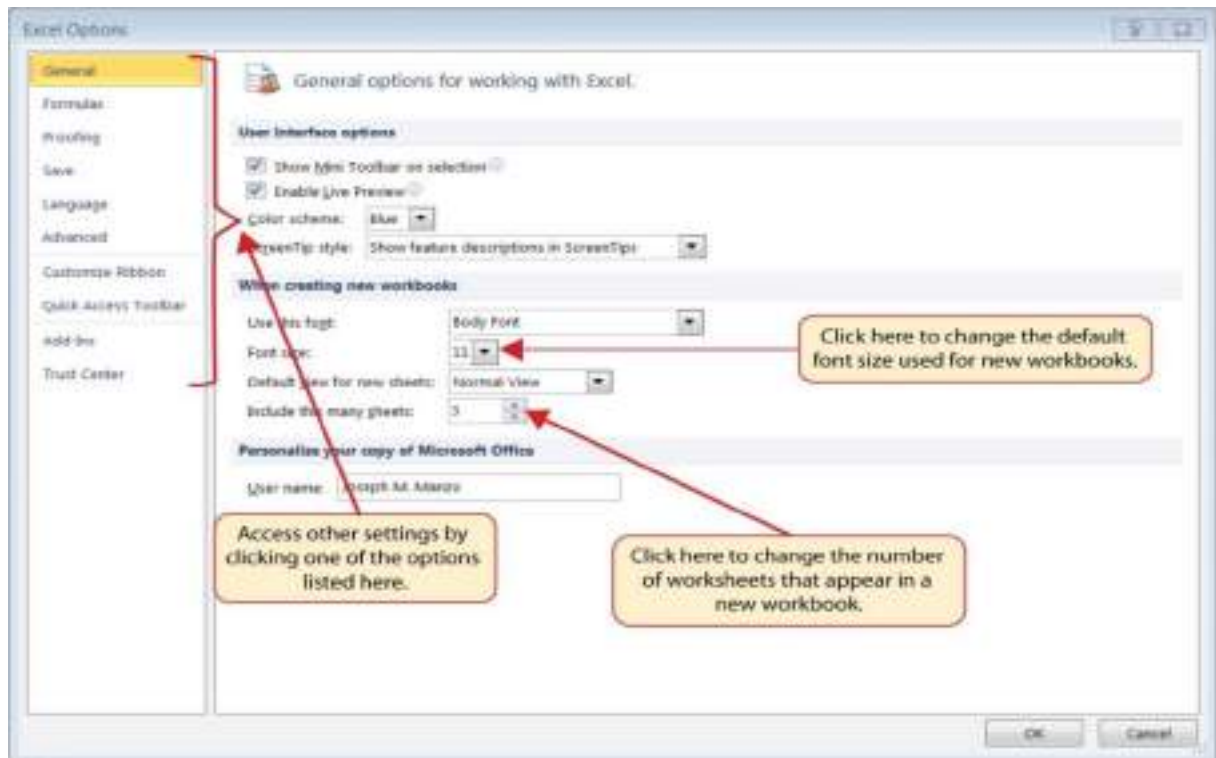


Figure 1.10 Excel Options Window

## SAVING WORKBOOKS (SAVE AS)

Once you create a new workbook, you will need to change the file name and choose a location on your computer or network to save that file. It is important to remember where you save this workbook on your computer or network as you will be using this file in the **Section 1.2 “Entering, Editing, and Managing Data”** to construct the workbook shown in **Figure 1.1**. The process of saving can be different with different versions of Excel. Please be sure you follow the steps for the version of Excel you are using. The following steps explain how to save a new workbook and assign it a file name.

### SAVING WORKBOOKS IN EXCEL 2013

1. If you have not done so already, open a blank workbook in Excel.
2. When saving your workbook for the *first* time, click the File tab.
3. Click the **Save As** button in the upper left side of the Backstage view window. This will open the **Save As** dialog box, as shown in **Figure 1.11**.
4. Click in the File Name box at the bottom of the **Save As** dialog box and use the BACKSPACE key to remove the current default name of the workbook.
5. Type the file name: **CH1 GMW Sales Data**.
6. Click the Desktop button on the left side of the **Save As** dialog box if you wish to save this file on your desktop. If you want to save this

workbook in a different location, such as a USB drive, select your preferred location.

7. Click the Save button on the lower right side of the **Save As** dialog box.
8. As you continue to work on your workbook, you will want to Save frequently by click either the Save button on the Home ribbon; or by selecting the Save option from the File menu.

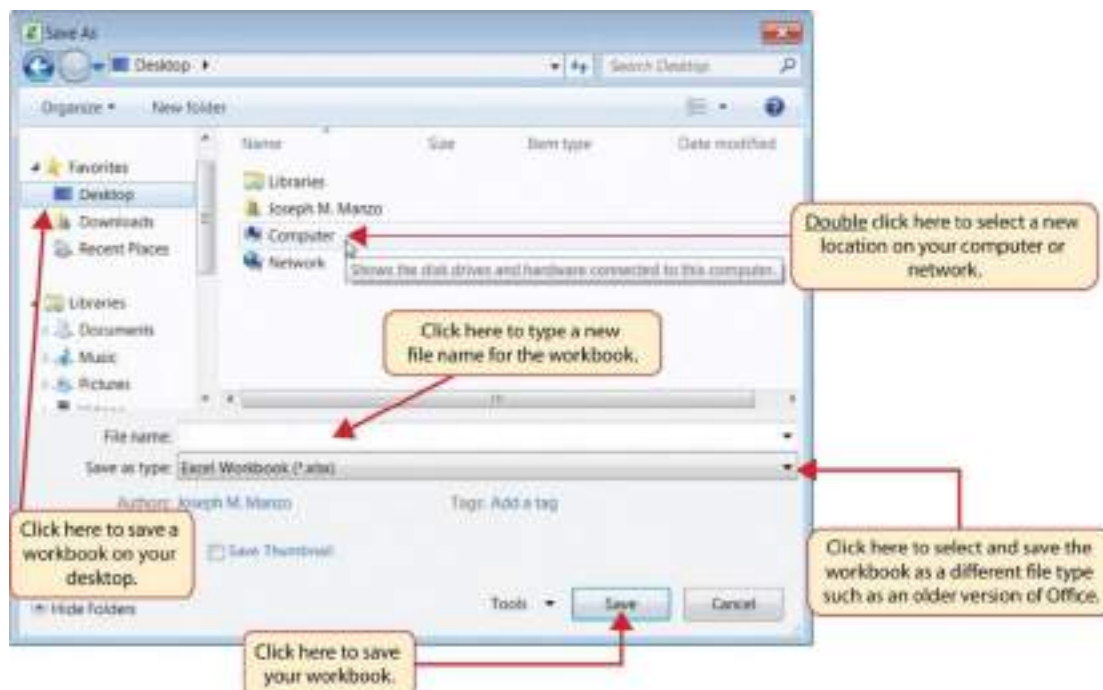


Figure 1.11 Save As Dialog Box in Excel 2013

## SAVING WORKBOOKS IN EXCEL 2016

1. If you have not done so already, open a blank workbook in Excel.
2. Click the File tab and then the **Save As** button in the left side of the Backstage view window. This will open the **Save As** dialog box.
3. Determine a location for saving on your computer by clicking **Browse** on the left side to open the **Save As** dialog box.
4. Click in the File Name box near the bottom of the **Save As** dialog box. Type the new file name: **CH1 GMW Sales Data**
5. Review the settings in the screen for correctness and click the Save button.



Figure 1.12 Save As Dialog in 2016  
Keyboard Shortcuts

### Save As

- Press the F12 key and use the tab and arrow keys to navigate around the Save As dialog box. Use the ENTER key to make a selection.
- Or press the ALT key on your keyboard. You will see letters and numbers, called Key Tips, appear on the Ribbon. Press the F key on your keyboard for the File tab and then the A key. This will open the Save As dialog box.

### Skill Refresher

#### Saving Workbooks (Save As)

1. Click the File tab on the Ribbon.
2. Click the Save As option.
3. Select a location on your PC.
4. Click in the File name box and type a new file name if needed.
5. Click the down arrow next to the “Save as type” box and select the appropriate file type if needed.
6. Click the Save button.



## THE STATUS BAR

The Status Bar is located below the worksheet tabs on the Excel screen (see Figure 1.13). It displays a variety of information, such as the status of certain keys on your keyboard (e.g., CAPS LOCK), the available views for a workbook, the magnification of the screen, and mathematical functions that can be performed when data are highlighted on a worksheet. You can customize the Status Bar as follows:

1. Place the mouse pointer over any area of the Status Bar and right click to display the “Customize Status Bar” list of options (see **Figure 1.13**).
2. Select the Caps Lock option from the menu (see **Figure 1.13**).
3. Press the CAPS LOCK key on your keyboard. You will see the Caps Lock indicator on the lower right side of the Status Bar.
4. Press the CAPS LOCK on your keyboard again. The indicator on the Status Bar goes away.

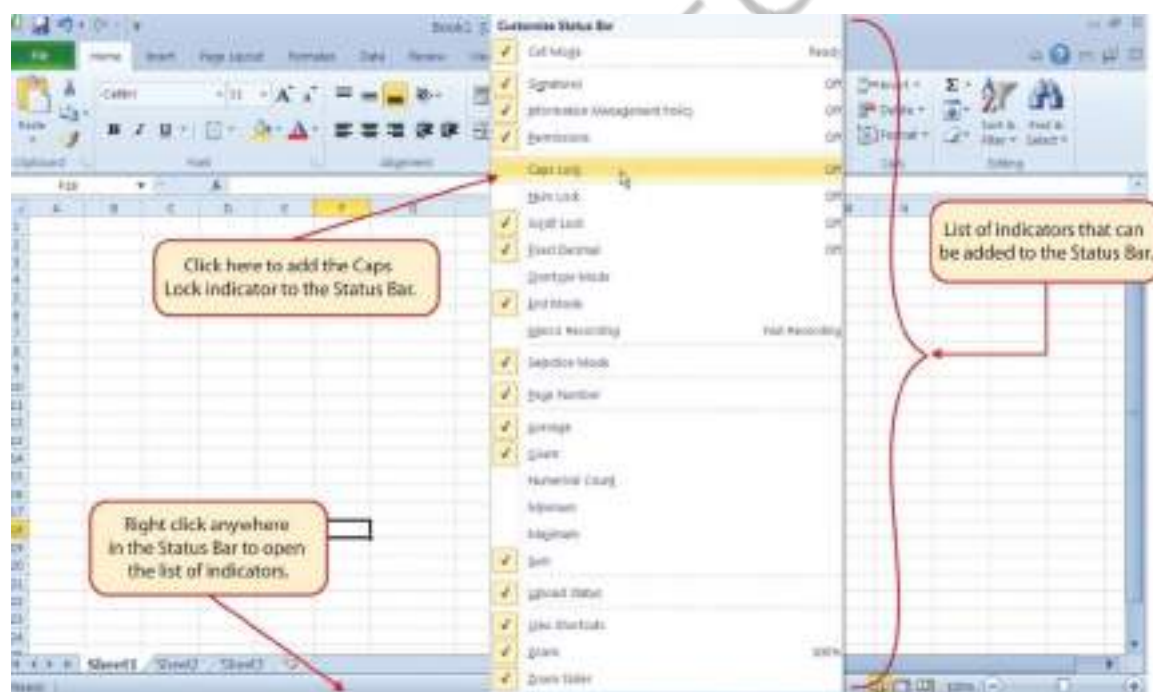


Figure 1.13 Customizing the Status Bar

## EXCEL HELP

The Help feature provides extensive information about the Excel application. Although some of this information may be stored on your computer, the Help window will automatically connect to the Internet, if you have a live connection, to provide you with resources that can answer most of your questions. You can open the Excel Help window by clicking the question mark in the upper right area of the screen or ribbon. With newer

versions of Excel, use the query box to enter your question and select from helpful option links or select the question mark from the dropdown list to launch Excel Help windows.



Figure 1.14 Excel Help Window  
Keyboard Shortcuts

## Excel Help

- Press the F1 key on your keyboard.

### Key Takeaways

- Excel is a powerful tool for processing data for the purposes of making decisions.
- You can find Excel commands throughout the tabs in the Ribbon.
- You can customize the Quick Access Toolbar by adding commands you frequently use.
- You can add or remove the information that is displayed on the Status Bar.
- The Help window provides you with extensive information about Excel.

## ATTRIBUTION

Adapted by Barbara Lave from How to Use Microsoft Excel: The Careers in Practice Series, adapted by The Saylor Foundation without attribution as requested by the work's original creator or licensee, and licensed under CC BY-NC-SA 3.0.

### Media Attributions

- figure\_1-6\_ribbon\_for\_excel
- figure-1-9
- figure\_1-12\_save\_as\_dialog\_box\_in\_2016
- 1-14\_excel\_help\_window

## WEEK 3: TASK2

### AIM:

**Creating a Scheduler:** Features to be covered:-Gridlines, Format Cells, Summation, auto fill, Formatting Text.

### PROCEDURE:

Steps to create a scheduler:

1. Create and save a new document named scheduler.
2. Insert tables from insert menu by choosing required number of rows and columns.
3. Fill the details inside the table by choosing required font size and style.
4. To merge cells, select the cells which you want to merge and right click, then select merge cells option.
5. To split cells, select the cell and right click, then select split cells option after which enter the number of rows and columns into which you want to split the cell.
6. 'lunch breaks' in the scheduler - select the cell and choose. Text direction from layout menu.
7. Choose fill bucket option from the home menu and choose the color.
8. Click on save button to save the document

### OUTPUT:

GNITS		IT / CTSW/04/700								
TIMETABLE SEMESTER		DEPARTMENT - IT								
Year/II		Semester - I			Section - A		Class Room No: F-3		w.s.e. 05/07/19	
1	Time/Day	1	2	3	4	5	6			
2		9:10 - 10:10	10:10 - 11:10	11:20-12:20	1:00-2:00	2:00-3:00	3:00-4:00			
3	MON	DS(CC-3)B1/OOP(CC-4)B2/ITW(CC-2)B3 LAB			DS	GS				
4	TUE	EM		PS	DS(CC-3)B2/OOP(CC-3)B3/ITW(CC-2)B1 LAB					
5	WED	DS	OOP	DLD	PL LAB		MBC			
6	THU	DS(CC-3)B3/OOP(CC-4)B1/ITW(CC-2)B2 LAB				EM	PS			
7	FRI	DLD	DS	OOP	Lecture/Workshop (12:30 - 1:30) Lab/SD	PS	MBC			
8	SAT	OOP	DLD	DS		5/L/I				

<b>SUBJECTS</b> Probability and Statistics Digital Logic Design Engineering Mechanics Data Structures Object Oriented Programming Gender Sensitization	<b>FACULTY</b> Dr. M. Madhavilata Ms. L. Smitha Mr. M. Venkata Ramana Reddy Dr. I. Ravi Prakash Reddy Ms. S. Vaisnavi Ms. V. Jaganthi	<b>LABS</b> IT Workshop Lab Data Structures through C++ Lab Object Oriented Programming through Java Lab Programming Lab(CC-4) Maths Bridge Course	<b>FACULTY</b> Ms. D. Vandana Dr. I. Ravi Prakash Reddy Ms. S. Vaisnavi Mr. B. Vijay Kumar Dr. M. Madhavilata
--	---	---	--

S- Sports	L- Library	I- Internet	TB- Technoboots
Batch1: 18251A1201 to 18251A1220	Batch2: 18251A1221 to 18251A1240	Batch3: 18251A1241 to 18251A1260	

DEPT. TIMETABLE COORDINATOR

TIMETABLE COORDINATOR

HOD

PRINCIPAL



## Week4 -Task1

### AIM:

#### **Calculation of Grade Point Average:**

Features to be covered:- Cell Referencing, Formulae in spreadsheet – average, std. deviation, Charts, Renaming and Inserting worksheets, Hyper linking, Count function, Sorting, Conditional formatting.

#### **PROCEDURE**

- **Cell Referencing:** Cell referencing is used to see the cell. With the help of row and columns, a cell can be named. For ex: D5- this cell is located in the 5<sup>th</sup> row of D column.
- **Formulae In Spreadsheet** - average, standard deviation, charts, renaming and inserting worksheets, hyper linking, count function, sorting, and conditional formatting.
- **Average:** we can take the average of a column or row using this function. Average of large numbers can be calculated using this formula. Click on the cell, and enter =AVERAGE (num1, num2....etc).or we can also use the option present in the ribbon”auto average”.
- **Standard deviation:** we can take the standard deviation of numbers using Formula =STDEV (num, num2.....etc). We use standard deviation to show the amount / range of difference from the mean.
- **Charts:** charts can be included in Ms-Excel for varied purposes. We can use it for presentations or to show the difference and statistics. Go to insert and press on charts option on the ribbon. We can choose which type of chart is desired. Ex: pie, bar, graph...etc.
- **Renaming and inserting worksheets:** worksheets can be found at the bottom left side of the desktop. Different worksheets are present in book. We can insert worksheets by clicking right click on the mouse pointing at a sheet where the new sheet has to be entered. A pop up is observed showing an option “insert”. Click on it and a new sheet is popped. In the same pop up, rename is observed. One can rename the desired sheet by clicking on that option.
- **Hyper linking:** we can use hyperlink in excel by using the option hyper link present in insert. Choose the address required and enter.
- **Count function:** excel provides us with a function called count function. With the help of this function we can see the number of alphabets or operators used. This is present at left corner of the page.

- **Sorting:** this feature is present in the home. Sorting is present In the ribbon. With the help of this we can sort the sheet in an alphabetical order. From A to Z or Z to A. custom sort helps us sort according to us.
- **Conditional formatting:** this is a very useful feature which helps us highlight or differentiate a cell from others. This makes the sheet more presentable and expressive. We can highlight different cells as required, i.e., dates in between or duplicate values. Values lesser than or equal to ...etc.

### **GPA PROCEDURE:**

1. Open a new Excel sheet and save as GPA.xlsx.
2. Create a GRADE table consisting of grade and points
3. Create a SUBJECT table consisting of subject and units
4. Create a STUDENT table consisting of rollno, maths, physics, chem,Eng and GPA.
5.  $GPA (Grade Point Average) = \frac{\sum \text{grade points} * \text{No. units of a sub}}{\text{total no. of units in all subjects}}$
6. To get grade points and No.of units VLOOKUP( ) function is used.
7. To name the tables Insert → Name → Define option is used.
8. GPA is calculated for one cell and it is copied to all other cells.

### **GPA-OUTPUT:**

ROLL NO.	CHEMISTRY	PHYSICS	ENGLISH	MATHS	G-CHEM	G-PHY	G-ENG	G-MATH	GPA	RANK
1201	A	B	A	B	10	9	10	9	9.53846	1
1202	B	C	A	B	9	8	10	9	9	3
1203	C	A	A	B	8	10	10	9	9.38462	2
1204	D	C	A	B	7	8	10	9	8.53846	5
1205	A	B	C	B	10	9	8	9	8.92308	4

GRADE TABLE		CREDIT TABLE	
A	10	CHEMISTRY	3
B	9	PHYSICS	4
C	8	ENGLISH	4
D	7	MATHS	2
E	6		13
F	5		

**OUTPUT**

## VIVA QUESTIONS

1. Mathematical expression/Formula consists of:-Operands, Values, Variables and Symbols.
2. Entering an equal (=) sign will bring up which Toolbar?
3. What is displayed in the Formula bar?
4. What are arguments in VLOOKUP ( )?
5. In HLOOKUP ( ) 'H' stands for?
6. What happens if a wrong argument is given in a LOOKUP function?
7. By default, what is the extension for Excel Workbook?
8. What is countif function?
9. How many rows and columns does an Excel worksheet have?
10. What is the procedure to insert chart in excel?

## WEEK 4 TASK 2

**AIM** Creating Charts: Understand chart terminology, select appropriate chart types for a specific set of data, create basic chart types, including column, pie, line, XY Scatter, and bar charts

### The Pie Chart

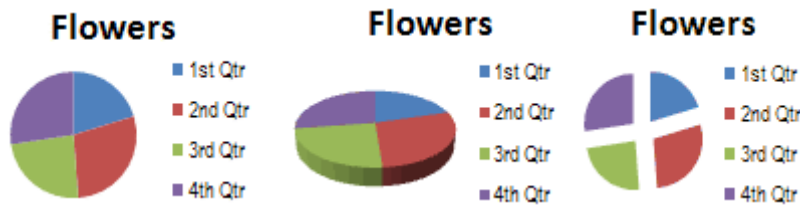
---

A Pie Chart displays *one series of data*. A data series is a row or column of numbers used for charting. In the worksheet below, we have outlined a single **data series**. If we had selected multiple series for the Pie Chart, Excel would ignore all but the first.  
Single Data Series

Excel uses the row heading (series identifier) for the chart title and displays the data as proportional slices of a pie (1st image below). One can customize the design of the pie chart so either numeric values or their percentages display on top of the slices of pie.

	A	B	C	D	E	F
1		1st Qtr	2nd Qtr	3rd Qtr	4th Qtr	Year
2	Flowers	\$170	\$240	\$200	\$230	\$840
3	Shrubs	\$220	\$280	\$250	\$290	\$1,040
4	Trees	\$260	\$340	\$200	\$320	\$1,120

The legend can be moved to the top, bottom, left, right, or top right ("corner" in older versions) of the chart. Legend names can be changed by changing the column headings in the sheet or editing the chart directly in new Excel versions.



Popular Pie Chart sub-types include Pie Chart in 3-D (2nd chart above), Exploded Pie Chart (3rd chart above), and Exploded Pie in 3-D. Other sub-types include the **Pie of Pie** and **Bar of Pie** - where a second pie or bar is created from certain values in the first pie. To customize, right-click on the segment in the first pie and select "Format Data Point."

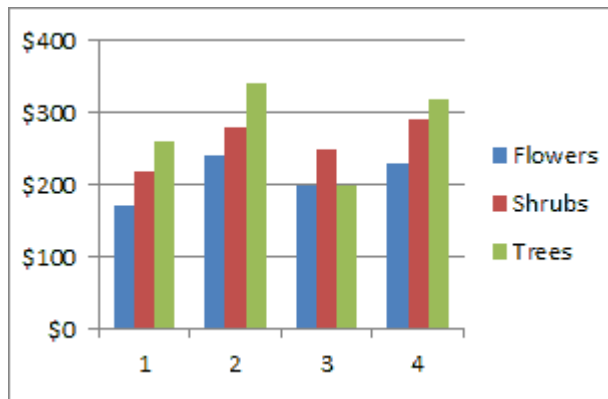
### The Column Chart

The Column Chart effectively compares a single set of data points, but it shines when comparing multiple series. Outlined in red, the image below shows the three data series we're charting. Notice that we do not include the totals in our data series.

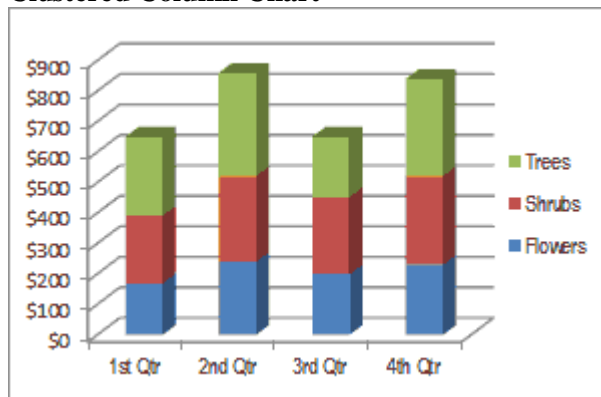
	A	B	C	D	E	F
1		1st Qtr	2nd Qtr	3rd Qtr	4th Qtr	Year
2	Flowers	\$170	\$240	\$200	\$230	\$840
3	Shrubs	\$220	\$280	\$250	\$290	\$1,040
4	Trees	\$260	\$340	\$200	\$320	\$1,120

### Multiple Data Series

Because Excel uses a different color for each data series, it's easy to see how a single series changes over time, or compare multiple series over a given time period. The Clustered Column Chart is especially popular.

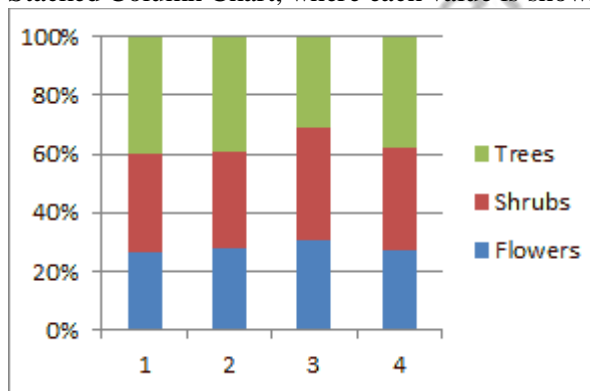


**Clustered Column Chart**



**Stacked Column Chart**

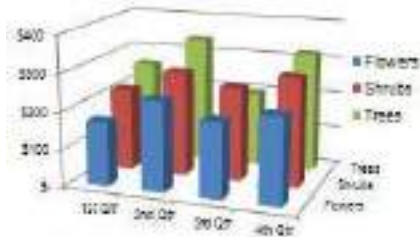
With the Stacked Column Chart, data points for each time period are "stacked" atop each other. This chart type shows each data point's **percentage of the total**. Also available is the 100% Stacked Column Chart, where each value is shown as a portion of 100%.



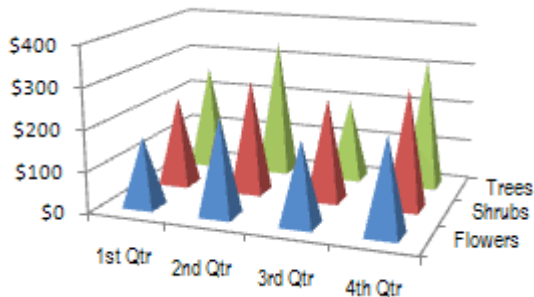
**100% Stacked Column Chart**

In a Column Chart, the vertical axis always displays numeric values, and the horizontal axis displays time, names, or other category. By default, Excel plots whichever has the most entries, row or column data, on the horizontal axis. For example, if plotting five rows and two columns, row headings would reside along the horizontal axis. This can be flipped around by customizing the Excel chart.

All Column Charts have three-dimension versions—the Stacked Column Chart is 3-D. But the "3-D Column Chart" is special because the chart itself is three-dimensional using the X, Y, and Z axes. The first chart below is a 3-D Column Chart of our data series.



**3-D Column Chart**

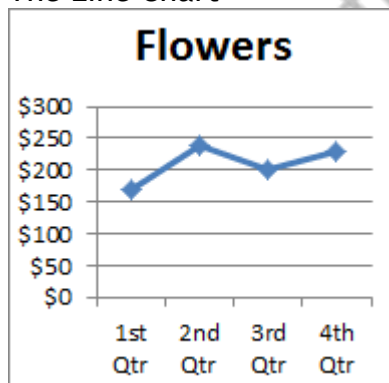


**3-D Pyramid Chart**

In newer versions of Excel, cylinders, pyramids, and cones can be used instead of bars for most of the Column charts. The second chart above shows a 3-D Pyramid Chart.

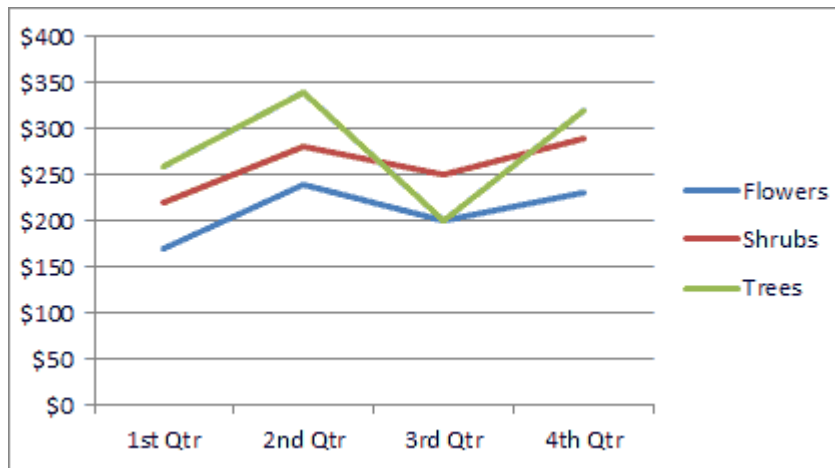
[A KeynoteSupport.com Tutorial](http://KeynoteSupport.com)

### The Line Chart



The Line Chart is especially effective in displaying trends. The vertical axis (Y-axis) always displays numeric values and the horizontal axis (X-axis) displays time or other category.

The first image shows the Line with Markers chart of our single data series. Markers—the circles, squares, triangles, or other shapes which mark the data points—are optional, and Excel displays a unique marker in shape and/or color for each data series.



The Line Chart is equally effective in displaying trends for multiple series as shown in the above Line Chart without markers. Notice that each line is a different color.

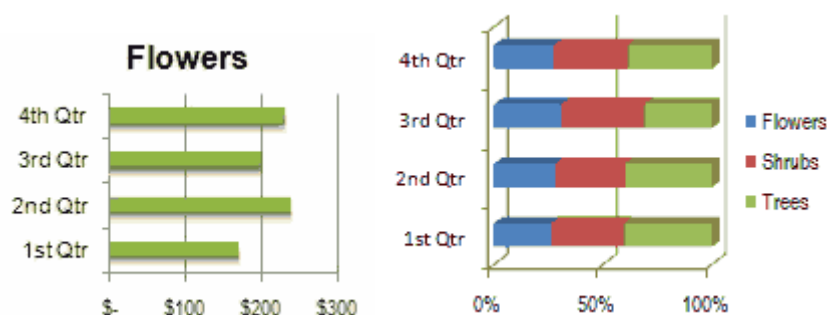
Though not as colorful as the other charts, it is easy to see how effective the Line Chart in showing a trend for a single series, and comparing trends for multiple series of data values.

Besides the Line Chart, we have the Stacked Line Chart and the 100% Stacked Line Chart - with or without markers. A 3-D Line Chart is available, but the Line Chart does not display data well in three dimension.

### The Bar Chart

The Bar Chart is like a Column Chart lying on its side. The horizontal axis of a Bar Chart contains the numeric values. The first chart below is the Bar Chart for our single series, Flowers.

When to use a Bar Chart versus a Column Chart depends on the type of data and user preference. Sometimes it is worth the time to create both charts and compare the results. However, Bar Charts do tend to display and compare a large number of series better than the other chart types.

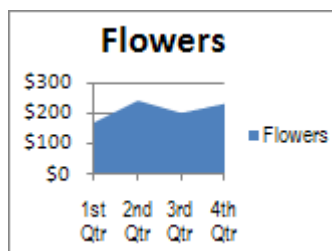


All Bar Charts are available in 2-D and 3-D formats. Excel provides the Stacked Bar Chart and 100% Stacked Bar Chart. The second chart above is our 100% Stacked Bar Chart in 3-D. This sub-type allows us to see what portion each data point has of 100%.

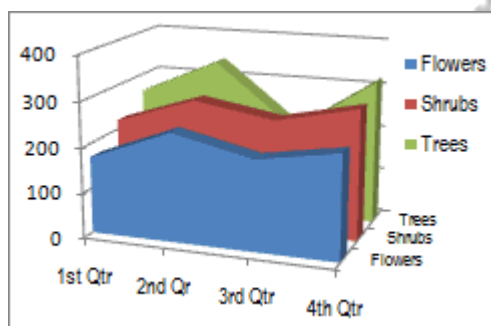
As with the other chart types, new versions of Excel provide the option of using cylinders, pyramids, or cones instead of bars.

[A KeynoteSupport.com Tutorial](http://www.keynotesupport.com)

### The Area Chart



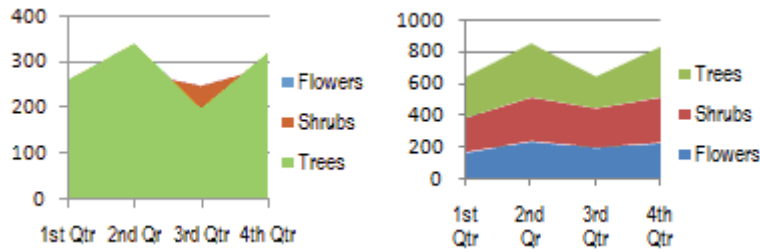
Area Charts are like Line Charts except that the area below the plot line is solid. And like Line Charts, Area Charts are used primarily to show trends over time or other category. The chart at left is an Area Chart for our single series.



Also available are the Stacked Area Chart and 100% Stacked Area Chart. Each comes in 2-D format and in true 3-D format with X, Y, and Z axes. The chart at right is our 3-D Area Chart and effectively displays our three series.

In many cases, the 2-D version of the Area Chart displays multiple series of data poorly as series with lesser values may be completely hidden. In the first chart below, Flowers is totally hidden, and just a wee bit of Trees peaks through. Not an effective chart!





This problem does not occur in the Stacked Area Chart shown above or the 100% Stacked Area Chart.

[A KeynoteSupport.com Tutorial](http://AKeynoteSupport.com)

### The Scatter Chart

The purpose of a Scatter Chart is to observe how the values of two series compares over time or other category. To illustrate the Scatter Chart, we will use the worksheet values shown below:

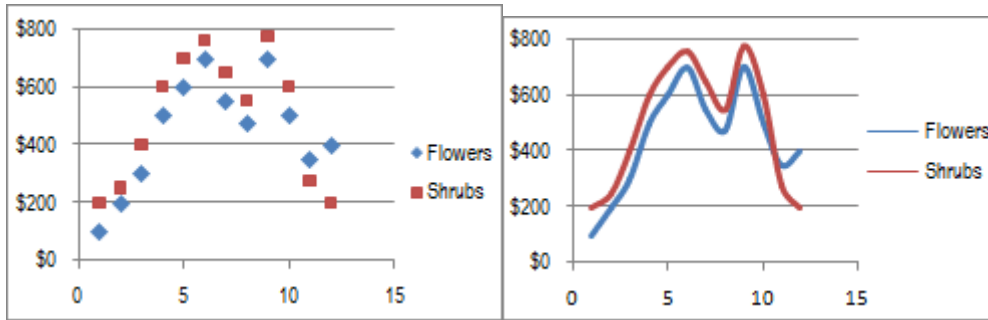
	A	B	C	D	E	F	G	H	I	J	K	L	M
1		Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
2	Flowers	\$100	\$200	\$300	\$500	\$600	\$700	\$550	\$475	\$700	\$500	\$350	\$400
3	Shrubs	\$200	\$250	\$400	\$600	\$700	\$760	\$650	\$550	\$775	\$600	\$275	\$200

According to [Scatter Plots \(U. of Illinois\)](#), "Scatter plots are similar to line graphs in that they use horizontal and vertical axes to plot data points. However, they have a very specific purpose. Scatter plots show how much one variable is affected by another. The relationship between two variables is called their correlation."

The series pair has a Positive Correlation if they increase similarly, and a Negative Correlation if they both decrease in like manner. Otherwise, they have No Correlation.

Excel does not use labels from the worksheet to label the horizontal axis. It numbers the X-axis chronologically.

The Scatter Chart comes in several different formats: markers can indicate the data points, and the points can be unconnected or connected with smooth or straight lines. The first chart below is a Scatter Chart with Only Markers, and the second chart is a Scatter Chart with Smooth Lines.

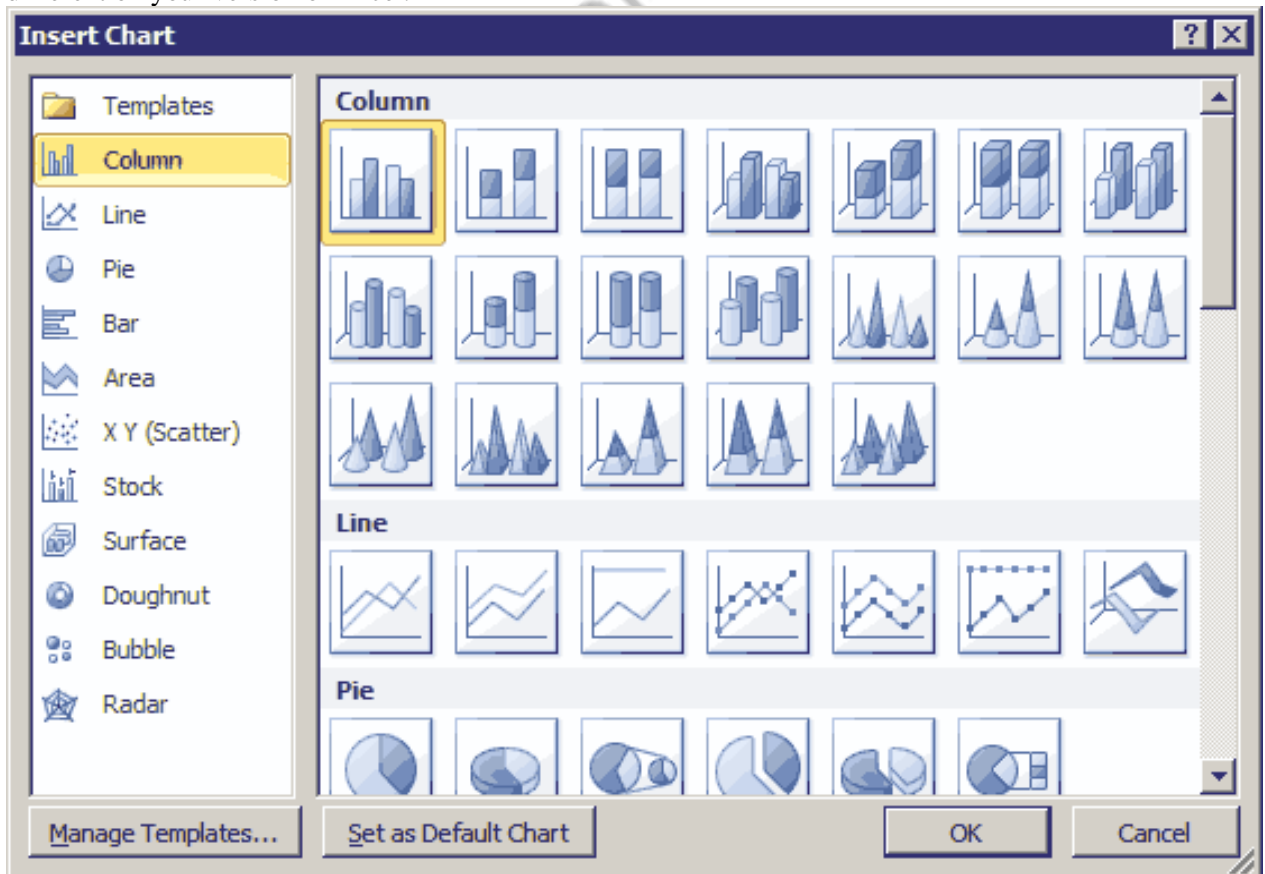


In general, markers work well when the number of data points is small, and smooth lines without markers are often used when the number of data points is large. But it is best to try the different sub-types to see which one best presents your data. For a good discussion on Scatter Plots, see [Scatter Plots - U. of Illinois](#).

[A KeynoteSupport.com Tutorial](#)

### Other Chart Types

Excel offers other chart types, such as Stock, Surface, Doughnut, Bubble, and Radar. To locate a menu of all available chart types in newer Excel versions, begin to insert any chart type and click **All Chart Types ...** The image below shows the menu that displays—menu may look different on your version of Excel.



### **Week 5: Python Programming**

Introduction to Python, variables, number data types and operators in Python.

- a) Write a program to demonstrate different number data types in Python.
- b) Write a program to perform different Arithmetic Operations on numbers.

Department of IT, GNITS

## Week 6: Control Flow Statements

if statement, if...else statement, if...elif...else statement, nested if statement, while loop, for loop, continue and break statements

# Python if...else Statement

In this article, you will learn to create decisions in a Python program using different forms of if..else statement.

What is if...else statement in Python?

Decision making is required when we want to execute a code only if a certain condition is satisfied.

The `if...elif...else` statement is used in Python for decision making.

## Python if Statement Syntax

```
if test expression:  
    statement(s)
```

Here, the program evaluates the `test expression` and will execute `statement(s)` only if the test expression is `True`.

If the test expression is `False`, the `statement(s)` is not executed.

In Python, the body of the `if` statement is indicated by the indentation. The body starts with an indentation and the first unindented line marks the end.

Python interprets non-zero values as `True`. `None` and `0` are interpreted as `False`.

## Python if Statement Flowchart

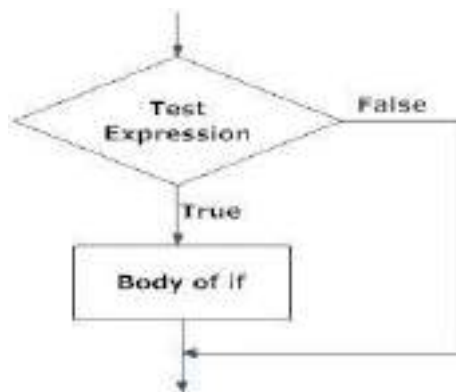


Fig: Operation of if statement.

Flowchart of if statement in Python programming

## Example: Python if Statement

```
# If the number is positive, we print an appropriate message

num = 3
if num > 0:
    print(num, "is a positive number.")
print("This is always printed.")

num = -1
if num > 0:
    print(num, "is a positive number.")
print("This is also always printed.")
```

When you run the program, the output will be:

```
3 is a positive number
This is always printed
This is also always printed.
```

In the above example, `num > 0` is the test expression.

The body of `if` is executed only if this evaluates to `True`.

When the variable `num` is equal to 3, test expression is true and statements inside the body of `if` are executed.

If the variable `num` is equal to -1, test expression is false and statements inside the body of `if` are skipped.

The `print()` statement falls outside of the `if` block (unindented). Hence, it is executed regardless of the test expression.

## Python if...else Statement

### Syntax of if...else

```
if test expression:  
    Body of if  
else:  
    Body of else
```

The `if..else` statement evaluates `test expression` and will execute the body of `if` only when the test condition is `True`.

If the condition is `False`, the body of `else` is executed. Indentation is used to separate the blocks.

### Python if..else Flowchart

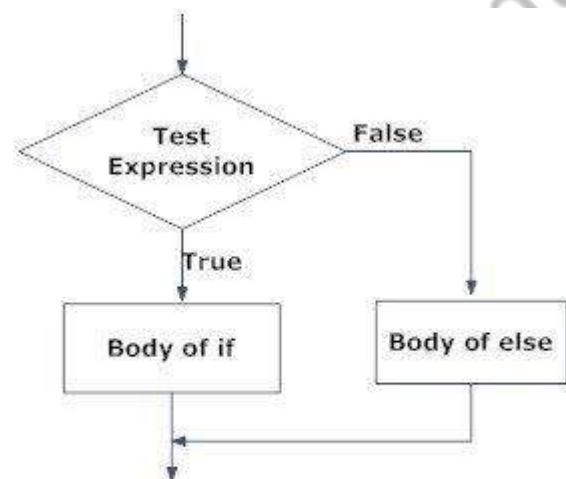


Fig: Operation of if...else statement

Flowchart of if...else statement in Python

### Example of if...else

```
# Program checks if the number is positive or negative  
# And displays an appropriate message
```

```
num = 3

# Try these two variations as well.
# num = -5
# num = 0

if num >= 0:
    print("Positive or Zero")
else:
    print("Negative number")
```

## Output

```
Positive or Zero
```

In the above example, when `num` is equal to 3, the test expression is true and the body of `if` is executed and the `body` of `else` is skipped.

If `num` is equal to -5, the test expression is false and the body of `else` is executed and the body of `if` is skipped.

If `num` is equal to 0, the test expression is true and body of `if` is executed and `body` of `else` is skipped.

## Python if...elif...else Statement

### Syntax of if...elif...else

```
if test expression:
    Body of if
elif test expression:
    Body of elif
else:
    Body of else
```

The `elif` is short for else if. It allows us to check for multiple expressions.

If the condition for `if` is `False`, it checks the condition of the next `elif` block and so on.

If all the conditions are `False`, the body of `else` is executed.

Only one block among the several `if...elif...else` blocks is executed according to the condition.

The `if` block can have only one `else` block. But it can have multiple `elif` blocks.

### Flowchart of `if...elif...else`

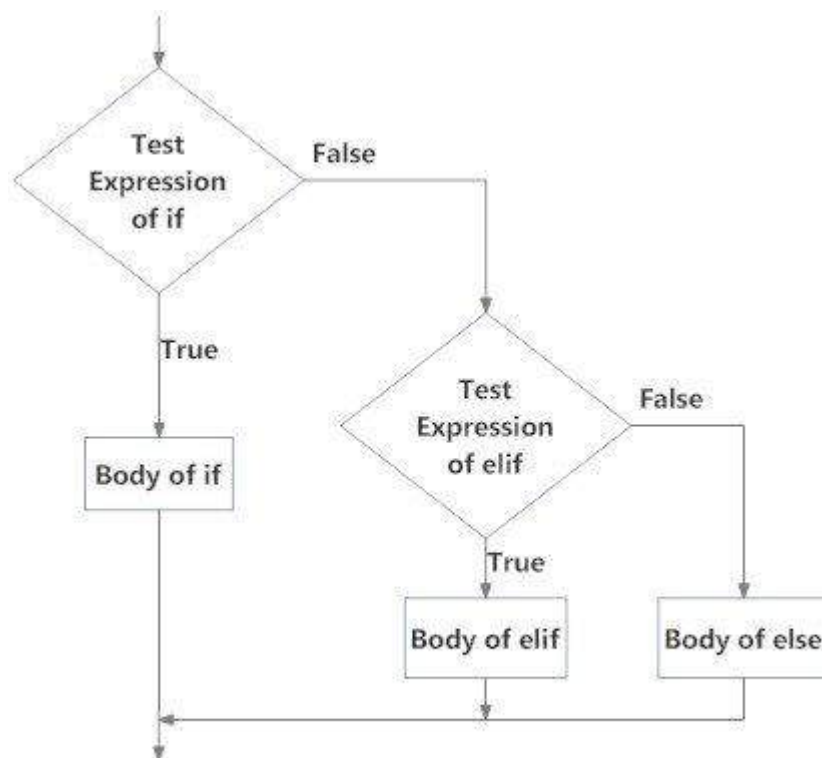


Fig: Operation of `if...elif...else` statement

### Flowchart of `if...elif...else` statement in Python

#### Example of `if...elif...else`

```
'''In this program,  
we check if the number is positive or  
negative or zero and  
display an appropriate message'''  
  
num = 3.4  
  
# Try these two variations as well:
```



```

# num = 0
# num = -4.5

if num > 0:
    print("Positive number")
elif num == 0:
else:
    print("Negative number")

```

When variable `num` is positive, `Positive number` is printed.

If `num` is equal to 0, `Zero` is printed.

If `num` is negative, `Negative number` is printed.

### Python Nested if statements

We can have a `if...elif...else` statement inside another `if...elif...else` statement. This is called nesting in computer programming.

Any number of these statements can be nested inside one another.

Indentation is the only way to figure out the level of nesting. They can get confusing, so they must be avoided unless necessary.

### Python Nested if Example

```

'''In this program, we input a number
check if the number is positive or
negative or zero and display
an appropriate message
This time we use nested if statement'''

num = float(input("Enter a number: "))
if num >= 0:
    if num == 0:
        print("Zero")
    else:
        print("Positive number")
else:
    print("Negative number")

```

### Output 1

```
Enter a number: 5  
Positive number
```

### **Output 2**

```
Enter a number: -1  
Negative number
```

### **Output 3**

```
Enter a number: 0  
Zero
```

Department of IT, GNIT

a) Write a python program to find largest of three numbers

## Source Code:

```
1. # Python program to find the largest number among the three input
   numbers
2. # take three numbers from user
3. num1 = float(input("Enter first number: "))
4. num2 = float(input("Enter second number: "))
5. num3 = float(input("Enter third number: "))
6.
7. if (num1 > num2) and (num1 > num3):
8.     largest = num1
9. elif (num2 > num1) and (num2 > num3):
10.    largest = num2
11. else:
12.    largest = num3
13.
14. print("The largest number is",largest)
```

## Output 1:

```
Enter first number: 10
Enter second number: 12
Enter third number: 14
The largest number is 14.0
```

## Output 2:

```
Enter first number: -1
Enter second number: 0
Enter third number: -3
The largest number is 0.0
```

## Explanation

In this program, we ask the user to input three numbers. We use the if...elif...else ladder to find the largest among the three and display it.

b) Write a Python program to convert temperatures to and from Celsius, Fahrenheit. [

Formula:  $c/5 = f-32/9$ ]

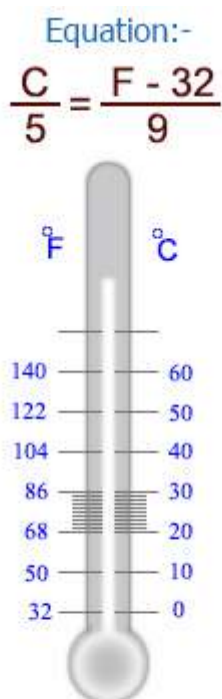
### Python: Centigrade and Fahrenheit Temperatures :

The centigrade scale, which is also called the Celsius scale, was developed by Swedish astronomer Andres Celsius. In the centigrade scale, water freezes at 0 degrees and boils at 100 degrees. The centigrade to Fahrenheit conversion formula is:

Fahrenheit and centigrade are two temperature scales in use today. The Fahrenheit scale was developed by the German physicist Daniel Gabriel Fahrenheit . In the Fahrenheit scale, water freezes at 32 degrees and boils at 212 degrees.

$$C = (5/9) * (F - 32)$$

where F is the Fahrenheit temperature. You can also use this Web page to convert Fahrenheit temperatures to centigrade. Just enter a Fahrenheit temperature in the text box below, then click on the Convert button.



$$C = (5 ( F - 32 )) / 9$$

$$F = (9C + (32 * 5)) / 5$$

© w3resource.com

**Sample Solution:-**

## Python Code:

```
temp = input("Input the temperature you like to convert?
(e.g., 45F, 102C etc.) : ")

degree = int(temp[:-1])

i_convention = temp[-1]

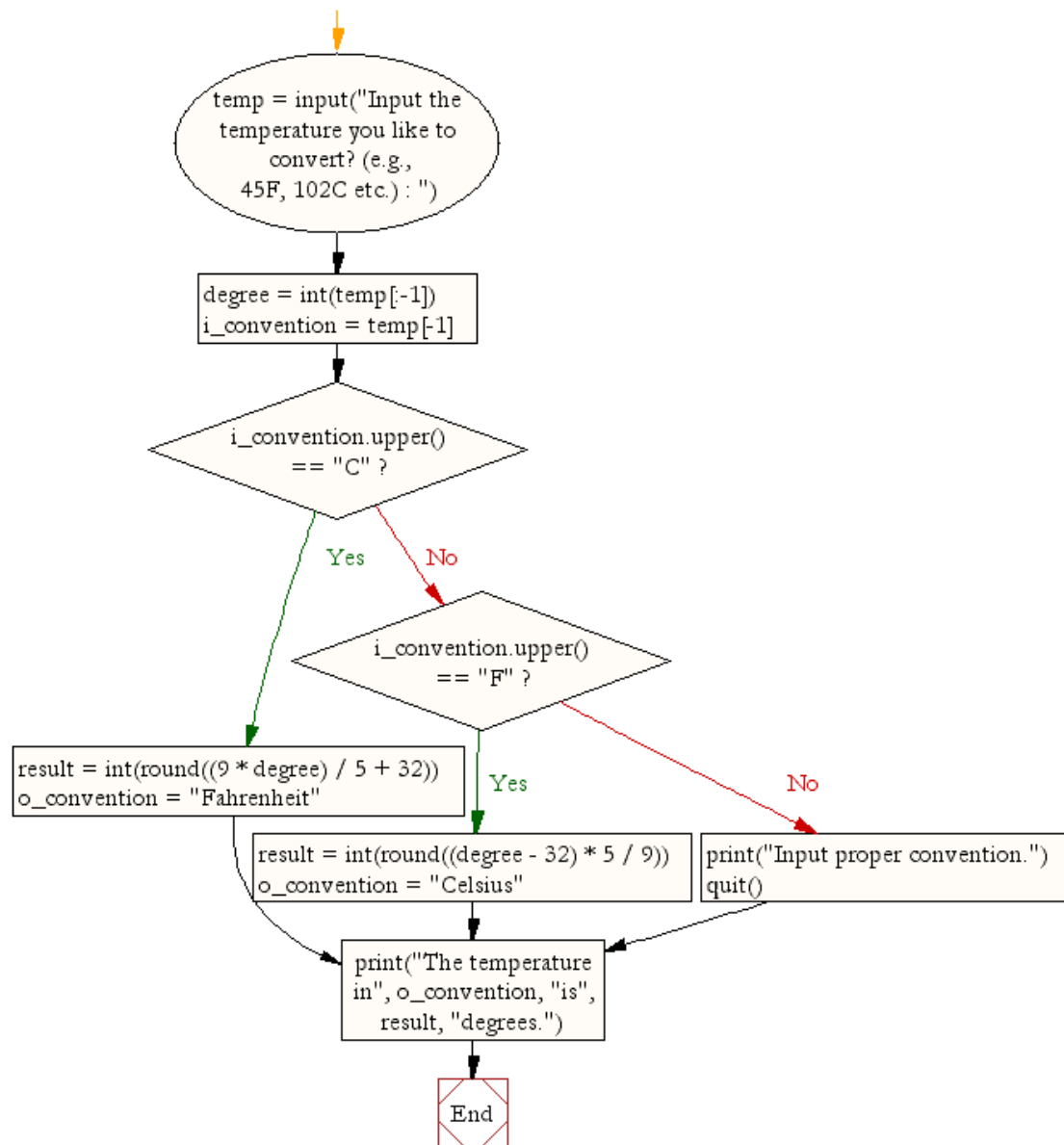
if i_convention.upper() == "C":
    result = int(round((9 * degree) / 5 + 32))
    o_convention = "Fahrenheit"
elif i_convention.upper() == "F":
    result = int(round((degree - 32) * 5 / 9))
    o_convention = "Celsius"
else:
    print("Input proper convention.")
    quit()

print("The temperature in", o_convention, "is", result,
"degrees.")
```

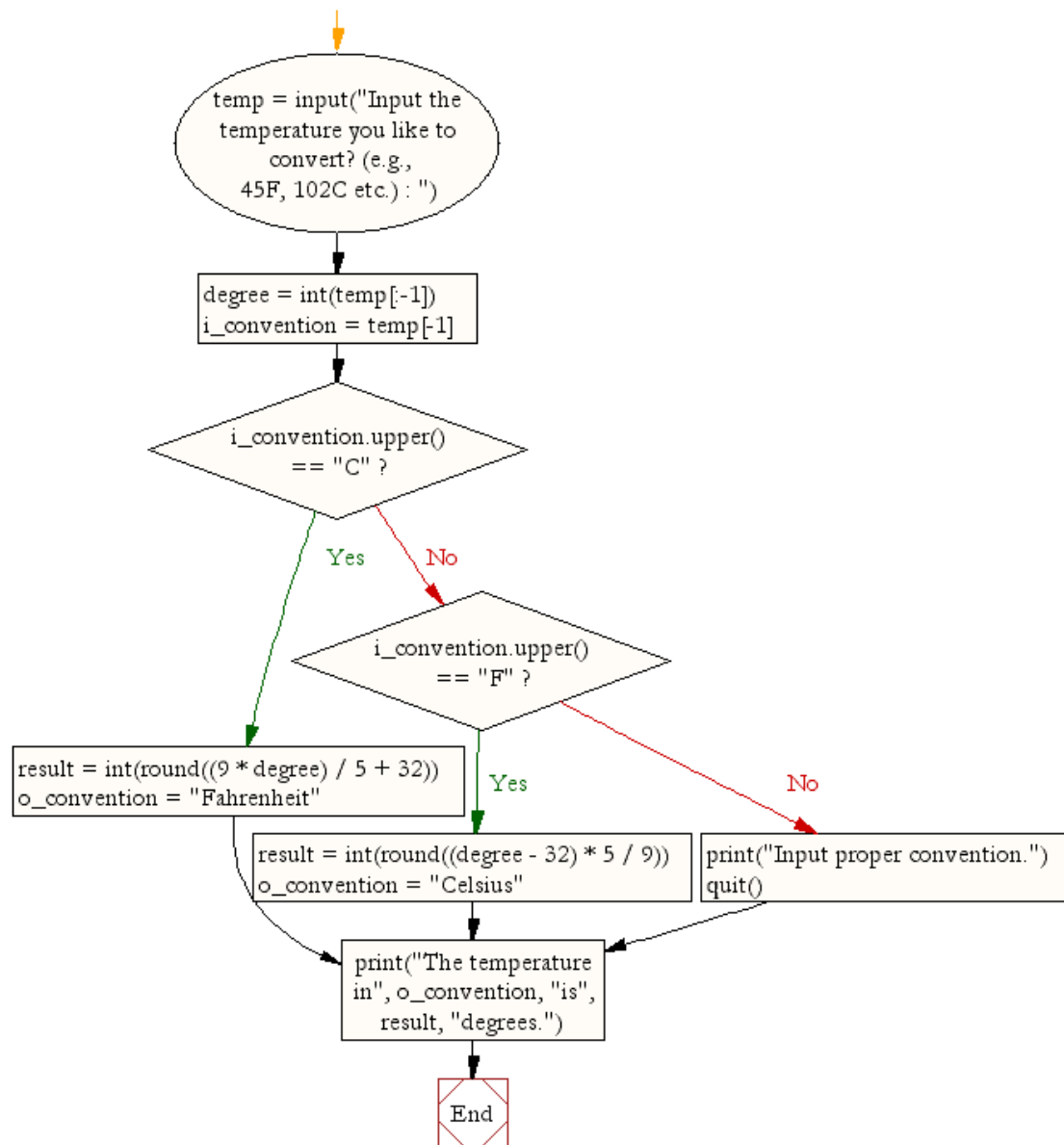
## Sample Output:

```
Input the temperature you like to convert? (e.g., 45F,
102C etc.) : 104f
The temperature in Celsius is 40 degrees.
```

## Flowchart:



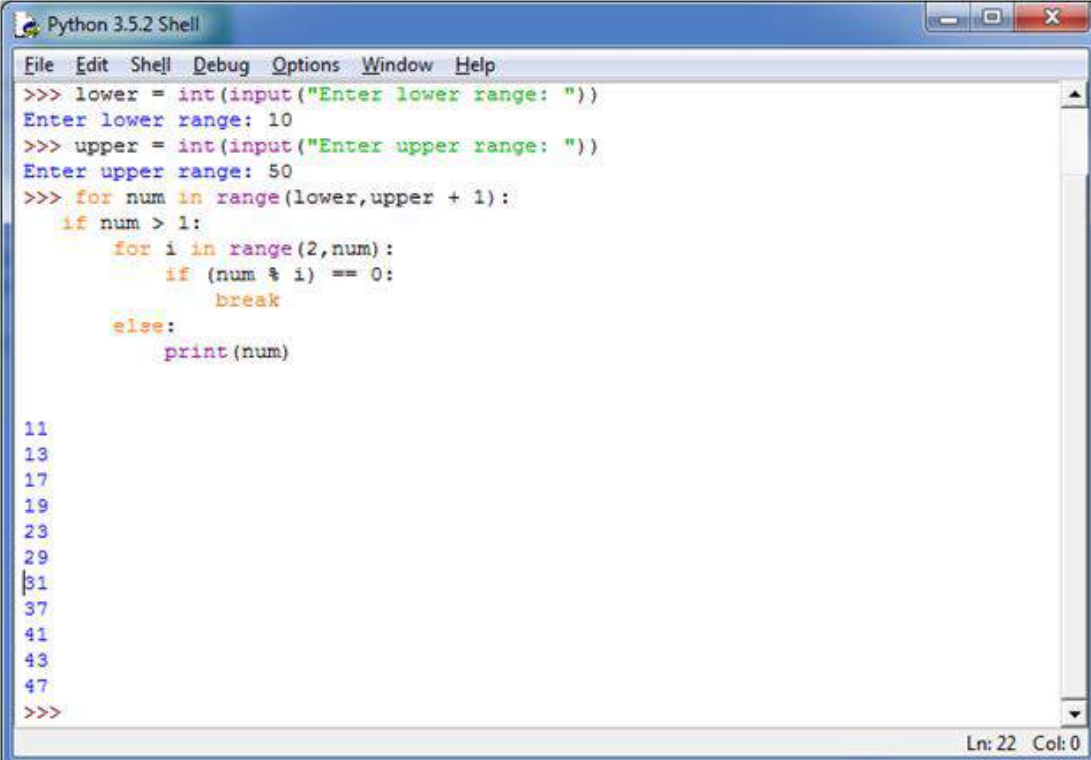
Dev



Der

c) Write a Python program that prints prime numbers less than 20(using for-else).

1. #Take the input from the user:
2. lower = int(input("Enter lower range: "))
3. upper = int(input("Enter upper range: "))
- 4.
5. for num in range(lower,upper + 1):
6. if num > 1:
7. for i in range(2,num):
8. if (num % i) == 0:
9. break
10. else:
11. print(num)



```
Python 3.5.2 Shell
File Edit Shell Debug Options Window Help
>>> lower = int(input("Enter lower range: "))
Enter lower range: 10
>>> upper = int(input("Enter upper range: "))
Enter upper range: 50
>>> for num in range(lower,upper + 1):
    if num > 1:
        for i in range(2,num):
            if (num % i) == 0:
                break
        else:
            print(num)

11
13
17
19
23
29
31
37
41
43
47
>>>
```

Ln: 22 Col: 0



d) Write a Python program to construct the following pattern, using a nested for loop

```
1
2 2
3 3 3
4 4 4 4
5 5 5 5 5
4 4 4 4
3 3 3
2 2
1
```

## Python Programs to Print Pattern – Print Number, Pyramid, Star, Triangle, Diamond, and Alphabets Patterns

### Steps to Print Pattern in Python

Use the below steps to print pattern in Python

#### 1. **Decide the number of rows and columns**

There is a typical structure to print any pattern, i.e., the number of rows and columns. We need to use two loops to print any pattern, i.e., use [nested loops](#).

The outer loop tells us the number of rows, and the inner loop tells us the column needed to print the pattern.

Accept the number of rows from a user using the [input\(\)](#) function to decide the size of a pattern.

## 2. Iterate rows

Next, write an outer loop to iterate the number of rows using a [for loop](#) and [range\(\)](#) function.

## 3. Iterate columns

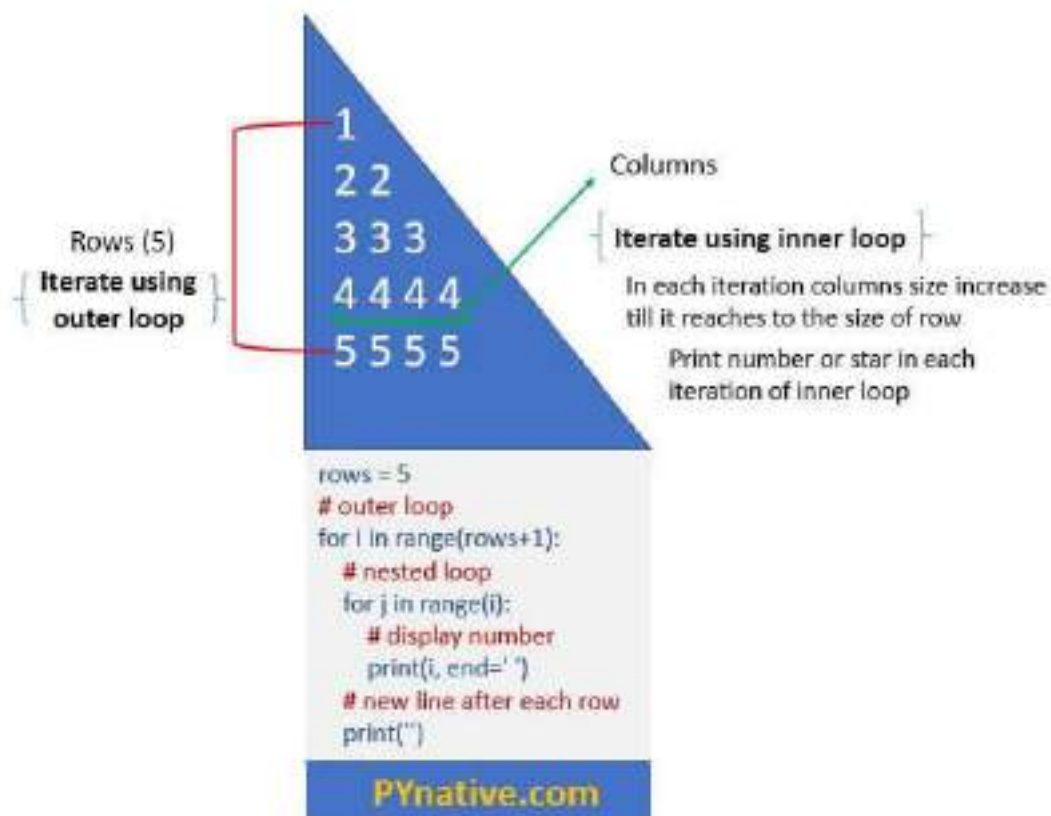
Next, write the inner loop or nested loop to handle the number of columns. The internal loop iteration depends on the values of the outer loop.

## 4. Print star or number

Use the `print()` function in each iteration of nested `for` loop to display the symbol or number of a pattern (like a star (asterisk `*`) or number).

## 5. Add new line after each iteration of outer loop

Add a new line using the `print()` function after each iteration of the outer loop so that the pattern display appropriately



## Python program to display half diamond pattern of numbers with star border

Given a number n, the task is to write a Python program to print a half diamond pattern of numbers with a star border.

### Examples:

Input: n = 5

Output:

```
*
*1*
*121*
*12321*
*1234321*
*123454321*
*1234321*
*12321*
*121*
*1*
*
```

Input: n = 3

Output:

```
*
*1*
*121*
*12321*
*121*
*1*
*
```

### Approach:

- Two for loops will be run in this program in order to print the numbers as well as stars.
- First print \* and then run for loop from 1 to (n+1) to print up to the rows in ascending order.
- In this particular for loop \* will be printed up to i and then one more for loop will run from 1 to i+1 in order to print the numbers in ascending order.
- Now one more loop will run from i-1 to 0 in order to print the number in the reverse order.
- Now one star will be printed and this for loop will end.
- Now second for loop will run from n-1 to 0 to print the pattern as in the middle in which the numbers are in a reverse manner.

- In this for loop also the same work will be done as in first for loop.
- The required pattern will be displayed.

**Below is the implementation of the above pattern:**

Python3

```
# function to display the pattern up to n

def display(n):

    print("**")

    for i in range(1, n+1):

        print("**", end="")

        # for loop to display number up to i

        for j in range(1, i+1):

            print(j, end="")

        # for loop to display number in reverse
direction

        for j in range(i-1, 0, -1):

            print(j, end="")

        print("**", end="")

        print()

    # for loop to display i in reverse direction

    for i in range(n-1, 0, -1):

        print("**", end="")

        for j in range(1, i+1):
```

```

        print(j, end="")

    for j in range(i-1, 0, -1):

        print(j, end="")

    print("*", end="")

    print()

    print("*")

# driver code

n = 5

print('\nFor n =', n)

display(n)

n = 3

print('\nFor n =', n)

display(n)

```

**Output:**

```

For n = 5
*
*1*
*121*
*12321*
*1234321*
*123454321*
*1234321*
*12321*
*121*
*1*

```

\*

For  $n = 3$

\*

\*1\*

\*121\*

\*12321\*

\*121\*

\*1\*

\*

Department of IT, GNITS

e) Write a program to get the binary form of a given number.

		REMAINDER
2	15	1
2	7	1
2	3	1
	1	

Given a positive number N, the task here is to print the binary value of numbers from 1 to N. For this purpose various approaches can be used.

The binary representation of a number is its equivalent value using 1 and 0 only. Example for  $k = 15$ , binary value is 1 1 1 1

*WORKING FOR METHOD 1*

**Method 1:** Using the Elementary method with recursion.

**Approach**

- Divide k by 2.
- Recursive call on the function and print remainder while returning from the recursive call.
- Repeat the above steps till the k is greater than 1.
- Repeat the above steps till we reach N

**Program:**

Python3

```
# code to print binary values of first 5 numbers
```

```
# recursive function

def Print_Binary_Values(num):

    # base condition

    if(num > 1):

        Print_Binary_Values(num // 2)

    print(num % 2, end=" ")

# driver code

if __name__ == "__main__":

    N = 5

    # looping N times

    for i in range(1, N+1):

        Print_Binary_Values(i)

    print(end=" ")
```

## Output

## Method 2: Using Bitwise Operator.

### Approach

- Check if  $k > 1$
- Right shift the number by 1 bit and perform a recursive call on the function



- Print the bits of number
- Repeat the steps till we reach N

### Program:

### Python3

```
# code to print binary values of first 5 numbers
```

```
# recursive function
```

```
def Print_Binary_Values(num):
```

```
    # base condition
```

```
    if(num > 1):
```

```
        Print_Binary_Values(num >> 1)
```

```
    print(num & 1, end="")
```

```
# driver code
```

```
if __name__ == "__main__":
```

```
    N = 5
```

```
    # looping N times
```

```
for i in range(1, N+1):  
  
    Print_Binary_Values(i)  
  
    print(end=" ")
```

## Output

### Method 3: Using Inbuilt Library of Python

bin() is an inbuilt python function that can convert any decimal number given to it as input to its equivalent binary.

#### Syntax:

here number is the decimal number that gets converted to binary

### Program

#### Python3

```
# code to print first 5 binary number using builtIn library  
  
def Print_Binary_Number(num):  
  
    for i in range(1, num+1):  
  
        # using bin to print binary value  
  
        print(int(bin(i).split('0b')[1]), end=" ")  
  
# driver code  
  
if __name__ == "__main__":  
  
    num = 5  
  
    Print_Binary_Number(num)
```

## Output

**1 10 11 100 101**

## Week 7: Lists

Creating Lists, Basic List operations, Indexing and Slicing in Lists, Built-in functions used on Lists, List methods, List comprehension.

### How to create a list?

In Python programming, a list is created by placing all the items (elements) inside square brackets `[]`, separated by commas. It can have any number of items and they may be of different types (integer, float, string etc.).

```
# empty list
my_list = []

# list of integers
my_list = [1, 2, 3]

# list with mixed data types
my_list = [1, "Hello", 3.4]
```

A list can also have another list as an item. This is called a nested list.

```
# nested list
my_list = ["mouse", [8, 4, 6], ['a']]
```

### Access List Elements

There are various ways in which we can access the elements of a list.

### List Index

We can use the index operator `[]` to access an item in a list. In Python, indices start at 0. So, a list having 5 elements will have an index from 0 to 4.

Trying to access indexes other than these will raise an `IndexError`.  
The index must be an integer. We can't use float or other types, this will result in `TypeError`.

Nested lists are accessed using nested indexing.

```
# List indexing

my_list = ['p', 'r', 'o', 'b', 'e']

# Output: p
print(my_list[0])

# Output: o
print(my_list[2])

# Output: e
print(my_list[4])

# Nested List
n_list = ["Happy", [2, 0, 1, 5]]

# Nested indexing
print(n_list[0][1])

print(n_list[1][3])

# Error! Only integer can be used for indexing
print(my_list[4.0])
```

## Output

```
p
o
e
a
5
Traceback (most recent call last):
  File "<string>", line 21, in <module>
TypeError: list indices must be integers or slices, not float
```

## Negative indexing

Python allows negative indexing for its sequences. The index of -1 refers to the last item, -2 to the second last item and so on.

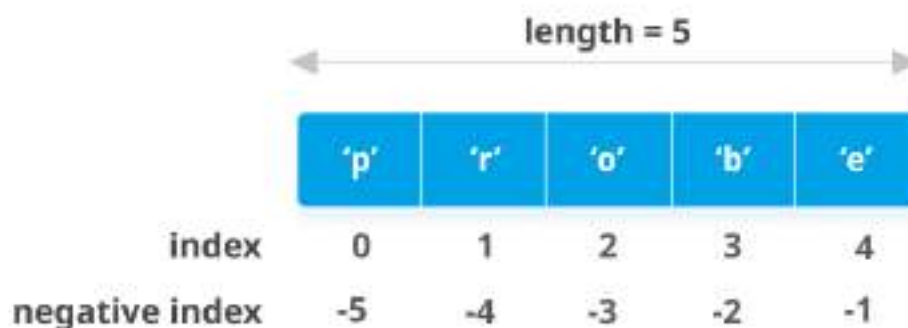
```
# Negative indexing in lists
my_list = ['p','r','o','b','e']

print(my_list[-1])

print(my_list[-5])
```

When we run the above program, we will get the following output:

```
e
p
```



List

indexing in Python

## How to slice lists in Python?

We can access a range of items in a list by using the slicing operator `:` (colon).

```
# List slicing in Python

my_list = ['p','r','o','g','r','a','m','i','z']
```

```
# elements 3rd to 5th
print(my_list[2:5])

# elements beginning to 4th
print(my_list[:-5])

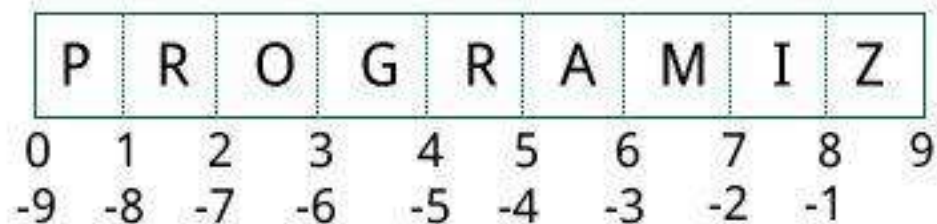
# elements 6th to end
print(my_list[5:])

# elements beginning to end
print(my_list[:])
```

## Output

```
['o', 'g', 'r']
['p', 'r', 'o', 'g']
['a', 'm', 'i', 'z']
['p', 'r', 'o', 'g', 'r', 'a', 'm', 'i', 'z']
```

Slicing can be best visualized by considering the index to be between the elements as shown below. So if we want to access a range, we need two indices that will slice that portion from the list.



Element Slicing from a list in Python

---

## Add/Change List Elements

Lists are mutable, meaning their elements can be changed unlike [string](#) or [tuple](#).

We can use the assignment operator `=` to change an item or a range of items.

```
# Correcting mistake values in a list
odd = [2, 4, 6, 8]

# change the 1st item
odd[0] = 1

print(odd)

# change 2nd to 4th items
odd[1:4] = [3, 5, 7]

print(odd)
```

### Output

```
[1, 4, 6, 8]
[1, 3, 5, 7]
```

We can add one item to a list using the `append()` method or add several items using `extend()` method.

```
# Appending and Extending lists in Python
odd = [1, 3, 5]

odd.append(7)

print(odd)

odd.extend([9, 11, 13])

print(odd)
```

### Output

```
[1, 3, 5, 7]
[1, 3, 5, 7, 9, 11, 13]
```

We can also use `+` operator to combine two lists. This is also called concatenation.

The `*` operator repeats a list for the given number of times.

```
# Concatenating and repeating lists
odd = [1, 3, 5]

print(odd + [9, 7, 5])

print(["re"] * 3)
```

## Output

```
[1, 3, 5, 9, 7, 5]
['re', 're', 're']
```

Furthermore, we can insert one item at a desired location by using the method `insert()` or insert multiple items by squeezing it into an empty slice of a list.

```
# Demonstration of list insert() method
odd = [1, 9]
odd.insert(1,3)

print(odd)

odd[2:2] = [5, 7]

print(odd)
```

## Output

```
[1, 3, 9]
[1, 3, 5, 7, 9]
```

---

## Delete/Remove List Elements

We can delete one or more items from a list using the keyword `del`. It can even delete the list entirely.

```
# Deleting list items
my_list = ['p', 'r', 'o', 'b', 'l', 'e', 'm']
```



```

# delete one item
del my_list[2]

print(my_list)

# delete multiple items
del my_list[1:5]

print(my_list)

# delete entire list
del my_list

# Error: List not defined
print(my_list)

```

## Output

```

['p', 'r', 'b', 'l', 'e', 'm']
['p', 'm']
Traceback (most recent call last):
  File "<string>", line 18, in <module>
NameError: name 'my_list' is not defined

```

We can use `remove()` method to remove the given item

or `pop()` method to remove an item at the given index.

The `pop()` method removes and returns the last item if the index is not provided. This helps us implement lists as stacks (first in, last out data structure).

We can also use the `clear()` method to empty a list.

```

my_list = ['p','r','o','b','l','e','m']
my_list.remove('p')

# Output: ['r', 'o', 'b', 'l', 'e', 'm']
print(my_list)

# Output: 'o'
print(my_list.pop(1))

# Output: ['r', 'b', 'l', 'e', 'm']

```

```
print(my_list)

# Output: 'm'
print(my_list.pop())

# Output: ['r', 'b', 'l', 'e']
print(my_list)

my_list.clear()

# Output: []
print(my_list)
```

## Output

```
['r', 'o', 'b', 'l', 'e', 'm']
o
['r', 'b', 'l', 'e', 'm']
m
['r', 'b', 'l', 'e']
[]
```

Finally, we can also delete items in a list by assigning an empty list to a slice of elements.

```
>>> my_list = ['p', 'r', 'o', 'b', 'l', 'e', 'm']
>>> my_list[2:3] = []
>>> my_list
['p', 'r', 'b', 'l', 'e', 'm']
>>> my_list[2:5] = []
>>> my_list
['p', 'r', 'm']
```

## Python List Methods

Methods that are available with list objects in Python programming are tabulated below.

They are accessed as `list.method()`. Some of the methods have already been used above.

## Python List Methods

**append()** - Add an element to the end of the list

**extend()** - Add all elements of a list to the another list

**insert()** - Insert an item at the defined index

**remove()** - Removes an item from the list

**pop()** - Removes and returns an element at the given index

**clear()** - Removes all items from the list

**index()** - Returns the index of the first matched item

**count()** - Returns the count of the number of items passed as an argument

**sort()** - Sort items in a list in ascending order

**reverse()** - Reverse the order of items in the list

**copy()** - Returns a shallow copy of the list

Some examples of Python list methods:

```
# Python list methods
my_list = [3, 8, 1, 6, 0, 8, 4]

# Output: 1
print(my_list.index(8))

# Output: 2
print(my_list.count(8))

my_list.sort()

# Output: [0, 1, 3, 4, 6, 8, 8]
print(my_list)

my_list.reverse()
```

```
# Output: [8, 8, 6, 4, 3, 1, 0]
print(my_list)
```

## Output

```
1
2
[0, 1, 3, 4, 6, 8, 8]
[8, 8, 6, 4, 3, 1, 0]
```

---

## List Comprehension: Elegant way to create Lists

List comprehension is an elegant and concise way to create a new list from an existing list in Python.

A list comprehension consists of an expression followed by [for statement](#) inside square brackets.

Here is an example to make a list with each item being increasing power of 2.

```
pow2 = [2 ** x for x in range(10)]
print(pow2)
```

## Output

```
[1, 2, 4, 8, 16, 32, 64, 128, 256, 512]
```

This code is equivalent to:

```
pow2 = []
for x in range(10):
    pow2.append(2 ** x)
```

A list comprehension can optionally contain more `for` or `if` [statements](#). An optional `if` statement can filter out items for the new list. Here are some examples.

```
>>> pow2 = [2 ** x for x in range(10) if x > 5]
>>> pow2
[64, 128, 256, 512]
>>> odd = [x for x in range(20) if x % 2 == 1]
>>> odd
[1, 3, 5, 7, 9, 11, 13, 15, 17, 19]
>>> [x+y for x in ['Python ', 'C '] for y in ['Language', 'Programming']]
['Python Language', 'Python Programming', 'C Language', 'C
Programming']
```

---

## Other List Operations in Python

### List Membership Test

We can test if an item exists in a list or not, using the keyword `in`.

```
my_list = ['p', 'r', 'o', 'b', 'l', 'e', 'm']

# Output: True
print('p' in my_list)

# Output: False
print('a' in my_list)

# Output: True
print('c' not in my_list)
```

### Output

```
True
False
True
```

---

## Iterating Through a List

Using a `for` loop we can iterate through each item in a list.

```
for fruit in ['apple', 'banana', 'mango']:  
    print("I like", fruit)
```

### Output

```
I like apple  
I like banana  
I like mango
```

## Built-in List Functions

There are some built-in functions in Python that you can use on python lists.



### a. len()

It calculates the length of the list.

```
>>> len(even)
```

### Output

```
3
```

## b. max()

It returns the item from the list with the highest value.

```
>>> max(even)
```

### Output

```
18
```

If all the items in your list are strings, it will compare.

```
>>> max(['1', '2', '3'])
```

### Output

```
'3'
```

But it fails when some are numeric, and some are strings in python.

```
>>> max([2, '1', '2'])
```

### Output

```
Traceback (most recent call last):File "<pyshell#116>", line 1, in <module>
```

```
max([2,'1','2'])
```

```
TypeError: '>' not supported between instances of 'str' and 'int'
```

## c. min()

It returns the item from the Python list with the lowest value.

```
>>> min(even)
```

### Output

```
6
```

## d. sum()

It returns the sum of all the elements in the list.

```
>>> sum(even)
```

### Output

```
36
```

However, for this, the Python list must hold all numeric values.

```
>>> a=['1','2','3']
```

```
>>> sum(a)
```

### Output

```
Traceback (most recent call last):File "<pyshell#112>", line 1, in <module>
```

```
sum(a)
```

```
TypeError: unsupported operand type(s) for +: 'int' and 'str'
```

It works on floats.

```
>>> sum([1.1,2.2,3.3])
```

### Output

```
6.6
```



## e. sorted()

It returns a sorted version of the list, but does not change the original one.

```
>>> a=[3,1,2]

>>> sorted(a)
```

### Output

```
[1, 2, 3]
```

```
>>> a
```

### Output

```
[3, 1, 2]
```

If the Python list members are strings, it sorts them according to their ASCII values.

```
>>> sorted(['hello', 'hell', 'Hello'])
```

### Output

```
['Hello', 'hell', 'hello']
```

Here, since H has an ASCII value of 72, it appears first.

## f. list()

It converts a different data type into a list.

```
>>> list("abc")
```

### Output

```
['a', 'b', 'c']
```

It can't convert a single int into a list, though, it only converts iterables.

```
>>> list(2)
```

### Output

```
Traceback (most recent call last):File "<pyshell#122>", line 1, in <module>
```

```
list(2)
```

```
TypeError: 'int' object is not iterable
```

## g. any()

It returns True if even one item in the Python list has a True value.

```
>>> any(['', '', '1'])
```

### Output

```
True
```

It returns False for an empty iterable.

```
>>> any([])
```

### Output

```
False
```

## h. all()

It returns True if all items in the list have a True value.

```
>>> all(['', '', '1'])
```

### Output

```
False
```

It returns True for an empty iterable.

```
>>> all([])
```

## Output

```
True
```

a) Write a program to demonstrate various list methods in python.

<https://www.programiz.com/python-programming/methods/list>  
<https://www.geeksforgeeks.org/python-list-exercise/>

```
# Python List Functions
```

```
# Python List Append Function
```

```
number = [150, 200, 130, 340]
```

```
print("Original List Items are      :", number)
```

```
number.append(520)
```

```
print("List Items after 520 appended are :", number)
```

```
number.append(650)
```

```
print("List Items after 650 appended are :", number)
```

```
number.append(-70)
```

```
print("List Items after -70 appended are :", number)
```

List append function output

```
Original List Items are      : [150, 200, 130, 340]
List Items after 520 appended are : [150, 200, 130, 340, 520]
List Items after 650 appended are : [150, 200, 130, 340, 520, 650]
List Items after -70 appended are : [150, 200, 130, 340, 520, 650, -70]
```

# Python List extend

The Python extend function adds the items in New\_List at the end of a list. In this example, we declared the number, a, b, and c lists. Next, we used this function to add a, b, c list items to the number list.

```
# Python List Functions
# Python List Extend Function

number = [10, 200, 630, 90]

print("Original List Items are      :", number)

a = [222, 333]
number.extend(a)
print("List Items after extending to a :", number)

b = [5, 9]
number.extend(b)
print("List Items after extending to b :", number)

c = [-12, 73]
number.extend(c)
print("List Items after extending to c :", number)
```

List extend function output

```
Original List Items are      : [10, 200, 630, 90]

List Items after extending to a : [10, 200, 630, 90, 222, 333]

List Items after extending to b : [10, 200, 630, 90, 222, 333,
5, 9]

List Items after extending to c : [10, 200, 630, 90, 222, 333,
5, 9, -12, 73]
```

# Python List insert function

Python insert function inserts the given item at a specified index position. The first statement inserts 100 at index position 2, and the second statement inserts 500 at position 4.

```
# Python List Functions
# Python List Insert Function

number = [5, 10, 15, 22, 19, 90]

print("Original List Items are      :", number)
```

```
number.insert(2, 100)
print("List Items after Inserting 100 at 2 :", number)
```

```
number.insert(4, 500)
print("List Items after Inserting 500 at 4 :", number)
```

```
number.insert(8, 700)
print("List Items after Inserting 700 at 8 :", number)
```

List insert function output

```
Original List Items are           : [5, 10, 15, 22, 19, 90]
List Items after Inserting 100 at 2 : [5, 10, 100, 15, 22, 19, 90]
List Items after Inserting 500 at 4 : [5, 10, 100, 15, 500, 22, 19, 90]
List Items after Inserting 700 at 8 : [5, 10, 100, 15, 500, 22, 19, 90, 700]
```

## Python list del

The Python del function deletes the value at a specified index. This example deletes items at index positions 5, 0, and 3.

```
# Python List Functions
# Python List Del Function
number = [9, 17, 10, 18, 55, 120, 90]
print("Original List Items are           :", number)
del number[5]
print("List Items after Deleting Item at Index 5 :", number)
del number[0]
print("List Items after Deleting Item at Index 0 :", number)
del number[3]
print("List Items after Deleting Item at Index 3 :", number)
```

List delete function output

```
Original List Items are           : [9, 17, 10, 18, 55, 120, 90]
```

```
List Items after Deleting Item at Index 5 : [9, 17, 10, 18, 55, 90]
```

```
List Items after Deleting Item at Index 0 : [17, 10, 18, 55, 90]
```

```
List Items after Deleting Item at Index 3 : [17, 10, 18, 90]
```

## Python List pop function

The Python pop function removes the items at the user given index and displays the removed element. After removing, the remaining values adjust to fill the index gap. The below program remove and displays the items at index position 6, 0, and 4.

```
# Python List Functions

# Python List Pop Function

number = [17, 6, 10, 18, 120, 220, 90, 119]

print("Original List Items are      :", number)

a = number.pop(6)
print("\nList Items after Deleting Item at Index 6 :", number)
print("Items Extracted by the Pop Function   :", a)

b = number.pop(0)
print("\nList Items after Deleting Item at Index 0 :", number)
print("Items Extracted by the Pop Function   :", b)

c = number.pop(4)
print("\nList Items after Deleting Item at Index 4 :", number)
print("Items Extracted by the Pop Function   :", c)
```

List pop function output

```
Original List Items are      : [17, 6, 10, 18, 120, 220, 90, 119]
```

```
List Items after Deleting Item at Index 6 : [17, 6, 10, 18, 120, 220, 119]
```

```
Items Extracted by the Pop Function   : 90
```

```
List Items after Deleting Item at Index 0 : [6, 10, 18, 120, 220, 119]
```

```
Items Extracted by the Pop Function   : 17
```

```
List Items after Deleting Item at Index 4 : [6, 10, 18, 120, 119]
```

```
Items Extracted by the Pop Function : 220
```

## Python List remove method

If we know the List item, we can use python remove function to remove a list item. The below program removes number list items 22, 19, and 5.

```
# Python List Functions
```

```
# Python List Remove Function
```

```
number = [55, 98, 10, 18, 22, 162, 170, 90]
```

```
print("Original List Items are : ", number)
```

```
number.remove(22)
```

```
print("List Items after Removing 22 are : ", number)
```

```
number.remove(98)
```

```
print("List Items after Removing 98 are : ", number)
```

```
number.remove(162)
```

```
print("List Items after Removing 162 are : ", number)
```

List remove function output

```
Original List Items are      : [55, 98, 10, 18, 22, 162,
170, 90]

List Items after Removing 22 are : [55, 98, 10, 18, 162, 170,
90]

List Items after Removing 98 are : [55, 10, 18, 162, 170, 90]

List Items after Removing 162 are : [55, 10, 18, 170, 90]
```

## Python list copy

The Python List copy method shallow copies the list items into a completely new list.

```
# Python List Copy Function

numbers = [6, 10, 18, 220, 90, 119]

print("List Items are      :", numbers)

new_list = numbers.copy()
print("\nNew List Items are :", new_list)
```

List copy function output

```
List Items are      : [6, 10, 18, 220, 90, 119]

New List Items are  : [6, 10, 18, 220, 90, 119]
```

## Python list clear

Python List clear method helps you to clear all the existing list items. After executing this method, if you call or print the same list, it returns an empty list.

```
# Python List Clear Function

numbers = [6, 10, 18, 220, 90, 119]

print("List Items are      :", numbers)

new_list = numbers.clear()
print("\nNew List Items are :", new_list)
```

List clear function output



```
List Items are      : [6, 10, 18, 220, 90, 119]
```

```
New List Items are : None
```

## Python list count

Python List count function counts the number of times a specified value repeated in a list. Here, we are counting how many times 22, 6, and 19 repeated in numbers list.

```
# Python List Count Function
```

```
numbers = [22, 6, 15, 19, 22, 90, 19, 22, 6, 19, 22]
```

```
print("List Items are      :", numbers)
```

```
a = numbers.count(22)
print("Number of Times 22 was repeated :", a)
```

```
b = numbers.count(6)
print("Number of Times 6 was repeated  :", b)
```

```
c = numbers.count(19)
print("Number of Times 19 was repeated :", c)
```

List count function output

```
List Items are      : [22, 6, 15, 19, 22, 90, 19, 22, 6, 19, 22]
```

```
Number of Times 22 was repeated : 4
```

```
Number of Times 6 was repeated  : 2
```

```
Number of Times 19 was repeated : 3
```

## Python list index

Python List index returns the index position of a user given value in a list. Here, we are finding the index position of a numbers list items 12, -9, and -19

```
# Python List Index Function
```

```
numbers = [22, 6, 12, 15, 19, 16, -9, 4]
```

```
print("List Items are      :", numbers)
```

```
a = numbers.index(12)
print("Index Position of 12 in this List :", a)
```

```
b = numbers.index(-9)
print("Index Position of -9 in this List :", b)
```

```
c = numbers.index(19)
print("Index Position of 19 in this List :", c)
```

List index function output

```
List Items are      : [22, 6, 12, 15, 19, 16, -9, 4]

Index Position of 12 in this List : 2

Index Position of -9 in this List : 6

Index Position of 19 in this List : 4
```

## Python list reverse

The Python List reverse method helps to reverse the items in a list. This code reverses the numbers list.

```
# Python List Reverse Function

numbers = [22, 6, 12, 15, 19, 16, -9, 4]

print("List Items are      :", numbers)

numbers.reverse()
print("\nNew List Items are :", numbers)
```

List reverse function output

```
List Items are      : [22, 6, 12, 15, 19, 16, -9, 4]

New List Items are  : [4, -9, 16, 19, 15, 12, 6, 22]
```

## Python list sort

The Python List sort method sorts the list items in Ascending order.

```
# Python List Sort Function
```

```
numbers = [2, 6, 0, 12, 15, -2, 19, 16, -9, 4]

print("List Items are    :", numbers)

numbers.sort()
print("\nNew Sorted List Items are :", numbers)
```

List sort function output

```
List Items are    : [2, 6, 0, 12, 15, -2, 19, 16, -9, 4]

New Sorted List Items are : [-9, -2, 0, 2, 4, 6, 12, 15, 16, 19]
```

## Python List sum, min, max

The Python List sum returns the sum of all items available in a given list. Next, the Python List min returns the minimum value among the given list items and the List max returns the maximum value.

```
# Python List Sum, Min, Max Function

numbers = [2, 6, 17, 12, 15, -2, 25, 16, -9, 4]

print("List Items are    :", numbers)

# Python List Max Function
maximum = max(numbers)
print("The Maximum Value in this List :", maximum)

# Python List MinFunction
minimum = min(numbers)
print("The Minimum Value in this List :", minimum)

# Python List Sum Function
total = sum(numbers)
print("The Sum of all Value in this List :", total)
```

b) Write a program to get a list of even numbers from a given list of numbers. (use only list comprehensions)



<https://www.datacamp.com/community/tutorials/python-list-comprehension>

<https://www.geeksforgeeks.org/python-program-to-print-even-numbers-in-a-list/>

<https://www.programiz.com/python-programming/list-comprehension>

## Python List Comprehension

With the recap of the Python lists fresh in mind, you can easily see that defining and creating lists in Python can be a tiresome job. Typing in all of the values separately can take quite some time, and you can easily make mistakes.

List comprehension in Python is also surrounded by brackets, but instead of the list of data inside it, you enter an expression followed by `for` loop and `if-else` clauses.

A most basic form of List comprehensions in Python are constructed as follows: `list_variable = [expression for item in collection]` The first expression generates elements in the list followed by a `for` loop

over some collection of data which would evaluate the expression for every item in the collection.

But how do you get to this formula-like way of building and using these constructs in Python? Let's dig a little bit deeper.

### The Mathematics

Luckily, Python has the solution for you: it offers you a way to implement a mathematical notation to do this: list comprehension.

Remember in maths, the common ways to describe lists (or sets, or tuples, or vectors) are:  $S = \{x^2 : x \text{ in } \{0 \dots 9\}\}$   $V = (1, 2, 4, 8, \dots, 2^{12})$   $M = \{x \mid x \text{ in } S \text{ and } x \text{ even}\}$  In other words, you'll find that the above definitions tell you the following:

- $S$  is a sequence that contains values between 0 and 9 included, and each value is raised to the power of two.
- The sequence  $V$ , on the other hand, contains the value 2 that is raised to a certain power  $x$ . The power  $x$  starts from 0 and goes till 12.
- Lastly, the sequence  $M$  contains only the even elements from the sequence  $S$ .

If the above definitions look a little cryptic to you, then take a look at the actual lists that these definitions would produce:  $S = \{0, 1, 4, 9, 16, 25, 36, 49, 64, 81\}$   $V = \{1, 2, 4, 8, 16, 32, 64, 128, 256, 512, 1024, 2048, 4096\}$   $M = \{0, 4, 16, 36, 64\}$  You see the result of each list and the operations that were described in them!

Now that you've understood some of the maths behind lists, you can translate or implement the mathematical notation of constructing lists in Python using list comprehensions! Take a look at the following lines of code:

```
S = [x**2 for x in range(10)]
V = [2**i for i in range(13)]
S
[0, 1, 4, 9, 16, 25, 36, 49, 64, 81]
V
[1, 2, 4, 8, 16, 32, 64, 128, 256, 512, 1024, 2048, 4096]
```

If you only want to extract expression for a specific set of data from the entire collection, you can add on an `if` clause after the `for` loop. The expression will be added to the list only if the `if` clause is True. You can even have more than one `if` clause, and the expression will be in the list only if all the `if` clauses return True.

Similarly, to select only an `even` number from the collection `S`, you will have an `if` clause which will check whether the given element is even or not. If it is even then that element will be added to the list.

```
M = [x for x in S if x % 2 == 0]
M
[0, 4, 16, 36, 64]
```

This all looks very similar to the mathematical definitions that you just saw, right?

No worries if you're a bit at lost at this point; Even if you're not a math genius, these list comprehensions are quite easy if you take your time to study them. Take a second, closer look at the Python code that you see in the code chunk above.

You'll see that the code tells you that:

- The list `S` is built up with the square brackets that you read above in the first section. In those brackets, you see that there is an element `x`, which is raised to the power of 10. Now, you just need to know for how many values (and which values!) you need to raise to the power of 2. This is determined in `range(10)`.

Considering all of this, you can derive that you'll raise all numbers, going from 0 to 9, to the power of 2.

- The list `v` contains the base value 2, which is raised to a certain power. Just like before, now you need to know which power or `i` is exactly going to be used to do this. You see that `i`, in this case, is part of `range(13)`, which means that you start from 0 and go until 12. All of this means that your list is going to have 13 values - those values will be 2 raised to the power 0, 1, 2, ... up to 12.
- Lastly, the list `M` contains elements that are part of `S` if -and only if- they can be divided by 2 without having any leftovers. The modulo needs to be 0. In other words, the list `M` is built up with the equal values that are stored in list `S`.

<https://data-flair.training/blogs/python-list-examples/>

Department

### **Week 8: Tuples and Sets**

Tuples: Creating Tuples, Basic Tuple operations, Indexing and Slicing in Tuples, Built-in Functions used on Tuples, Relation between Tuples and Lists. Sets: Set Methods, operations of sets.

- a) Write a program to add an item in a tuple without converting into a list.
- b) Write a program to count the elements in a list until an element is a tuple.
- c) Write a Python program to demonstrate set operations.

### **Week 9: Strings and Dictionaries**

Strings: Basic String operations, String slicing and joining, String methods. Dictionaries: Creating Dictionary, Accessing and Modifying key-value pairs in Dictionaries, Built - in functions used on Dictionaries, Dictionary methods.

- a) Write a program to access a sub string from a given string (Use slicing)
  - Get the first 5 characters of a string
  - Get a substring of length 4 from the 3rd character of the string
    - Get the last 5 characters of a string
  - Get a substring which contains all characters except the last 4 characters and the 1st character
  - Get every other character from a string
- b) Get a string from a given string where all occurrences of its first char have been changed to '\$', except the first char itself Eg: restart output: resta\$t
- c) Write a program to sort a dictionary by a value.
- b) Write a program to display the count of individual vowels in the input string using dictionary. (Ex: Input String: "welcome" Output: {'a':0,'e':2,'i':0,'o':1,'u':0})

### **Week 10: Functions and Modules**

**Functions:** Built-In Functions, Function definition and calling the function, The return statement and void function, recursion. Classes and objects.

**Modules:** Importing Modules, Importing Module Attributes.

- a) Write a python program to find N largest element from given list of integers using functions
- b) Write a python program to find sum of elements of nested list using recursion. (Input: [9, 1, [3,4], [5,2]], Output:24)
- c) Write a program to implement stack data structure using class.
- d) Write a python program to define a module to find Fibonacci Numbers and import the module to another program.



e) Define a module that consist of factorial and sum of individual digits of a number as functions. Write a program to find ncr by importing only factorial function from the above module.

#### **Week 11: Exception Handling and Files**

**Exception Handling:** Catching exceptions using try and except statement, user defined exceptions

**Files:** Creating files, File input/output methods

- a) Write a program to handle exceptions using try..except..finally...else
- b) Write a program to sort words in a file and put them in another file. The output file should have only lower-case words, so any upper-case words from source must be lowered. (Handle exceptions)
- c) Write a program that inputs a text file. The program should print all of the unique words in the file in alphabetical order.

#### **Week 12: Visualization of Data using Python Libraries**

**Python Libraries:** Introduction to python libraries and exploring Numpy, Pandas, matplotlib, seaborn.

- a) Write a python program to demonstrate array operations using Numpy library.
- b) Write a python program to plot a bar graph on any data set using pandas library.
- c) Write a python program to plot a scatter plot on any data set using matplotlib library.
- d) Write a python program to plot a box plot on any data set using seaborn library.