Unit V

CHAPTER 16

DATA VISUALIZATION USING PYPLOT: LINE CHART, PIE CHART AND BAR CHART

Learning Objectives

RSXUS6

After learning this chapter, the learners will be able to

- Define the term Data Visualization.
- List the types of Data Visualization.
- List the uses of Data Visualization.
- List the types of Visualizations in Matplotlib.
- Practice installing Matplotlib.
- Explore importing Matplotlib.
- Classify the types of Data Visualization plots.
- Practice creating various types of plots using Matplotlib.

16.1 Data Visualization Definition

Data Visualization is the graphical representation of information and data. The objective of Data Visualization is to communicate information visually to users. For this, data visualization uses statistical graphics. Numerical data may be encoded using dots, lines, or bars, to visually communicate a quantitative message.

General types of Data Visualization

- Charts
- Tables
- Graphs
- Maps
- Infographics
- Dashboards

Data visualization - Uses

- Data Visualization help users to analyze and interpret the data easily.

- It makes complex data understandable and usable.
- Various Charts in Data Visualization helps to show relationship in the data for one or more variables.

Infographics → An infographic (information graphic) is the representation of information in a graphic format.

Dashboard → A dashboard is a collection of resources assembled to create a single unified visual display. Data visualizations and dashboards translate complex ideas and concepts into a simple visual format. Patterns and relationships that are undetectable in text are detectable at a glance using dashboard.

Introduction to Matplotlib — Data Visualization in Python

Matplotlib is the most popular data visualization library in Python. It allows you to create charts in few lines of code.

Types of Visualizations in Matplotlib

There are many types of Visualizations under Matplotlib. Some of them are:

- Line plot
- Scatter plot
- Histogram
- Box plot
- Bar chart and
- Pie chart

Scatter plot: A scatter plot is a type of plot that shows the data as a collection of points. The position of a point depends on its two-dimensional value, where each value is a position on either the horizontal or vertical dimension.

Box plot: The box plot is a standardized way of displaying the distribution of data based on the five number summary: minimum, first quartile, median, third quartile, and maximum.

Installing Matplotlib

You can install matplotlib using pip. Pip is a management software for installing python packages.

Note

Detailed installation procedures given in Annexure

16.2 Getting Started

After installing Matplotlib, we will begin coding by importing Matplotlib using the command:

```
import matplotlib.pyplot as plt
```

Now you have imported Matplotlib in your workspace. You need to display the plots. Using Matplotlib from within a Python script, you have to add plt.show() method inside the file to display your plot.

Example

```
import matplotlib.pyplot as plt
plt.plot([1,2,3,4])
plt.show()
```

Output

This window is a matplotlib window, which allows you to see your graph. You can hover the graph and see the coordinates in the bottom right.

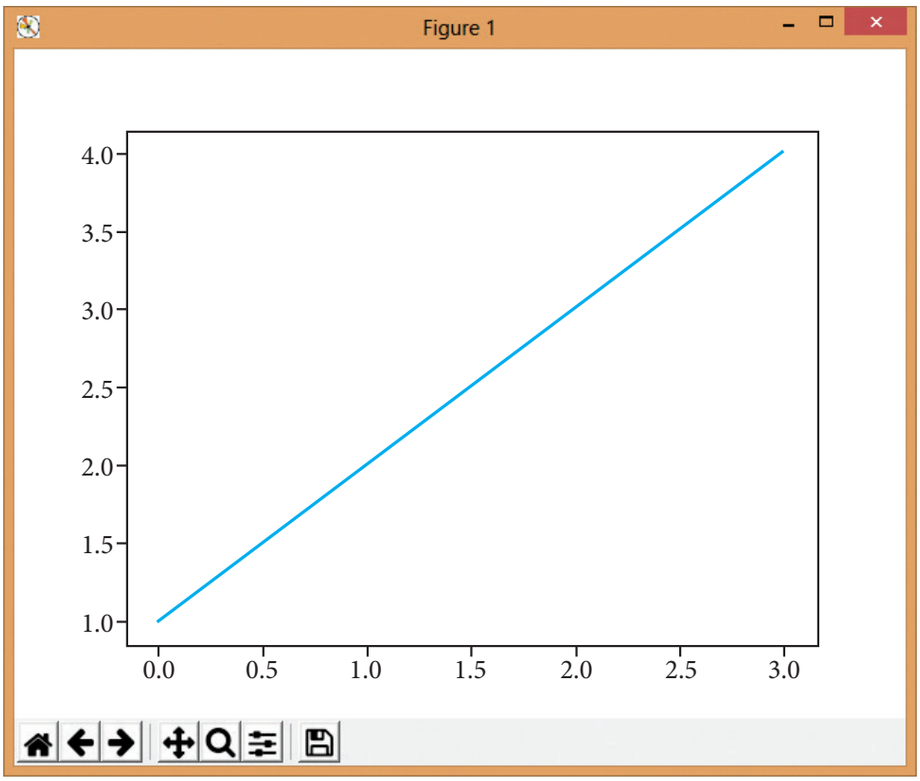


Figure 16.1

You may be wondering why the x-axis ranges from 0-3 and the y-axis from 1-4. If you provide a single list or array to the plot () command, matplotlib assumes it is a sequence of y values, and automatically generates the x values for you. Since python ranges start with 0, the default x vector has the same length as y but starts with 0. Hence the x data are [0, 1, 2, 3].

plot() is a versatile command, and will take an arbitrary number of arguments.

Program

For example, to plot x versus y, you can issue the command:

```
import matplotlib.pyplot as plt
plt.plot([1,2,3,4], [1,4,9,16])
plt.show()
```

This .plot takes many parameters, but the first two here are 'x' and 'y' coordinates. This means, you have 4 co-ordinates according to these lists: (1,1), (2,4), (3,9) and (4,16).

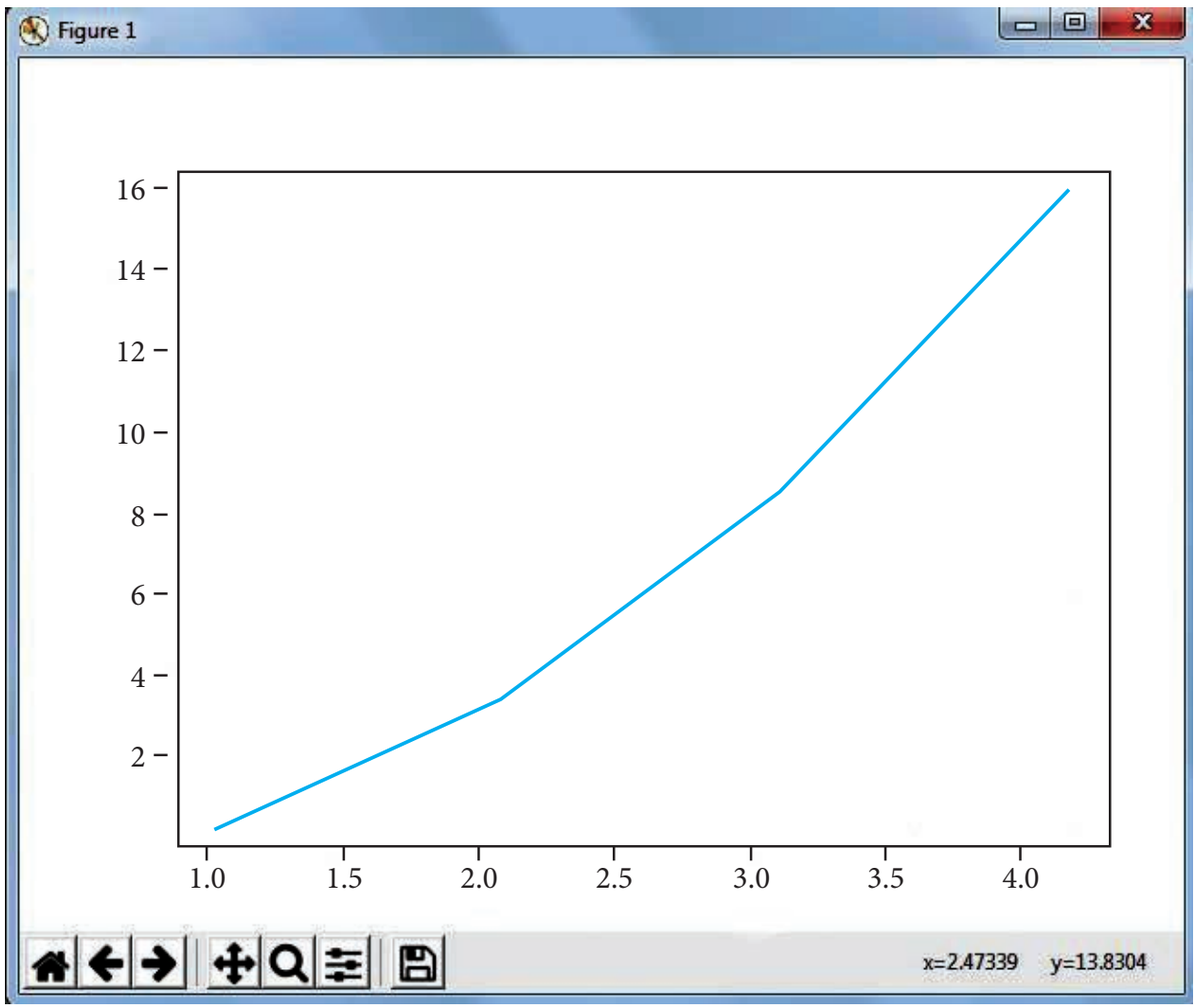


Figure 16.2

Plotting Two Lines

To plot two lines, use the following code:

```
import matplotlib.pyplot as plt
    x = [1,2,3]
    y = [5,7,4]
    x2 = [1,2,3]
    y2 = [10,14,12]
    plt.plot(x, y, label='Line 1')
    plt.plot(x2, y2, label='Line 2')
    plt.xlabel('X-Axis')
    plt.ylabel('Y-Axis')
    plt.title('LINE GRAPH')
    plt.legend()
    plt.show()
```

Output

With plt.xlabel and plt.ylabel, you can assign labels to those respective axis. Next, you can assign the plot's title with plt.title, and then you can invoke the default legend with plt. legend().

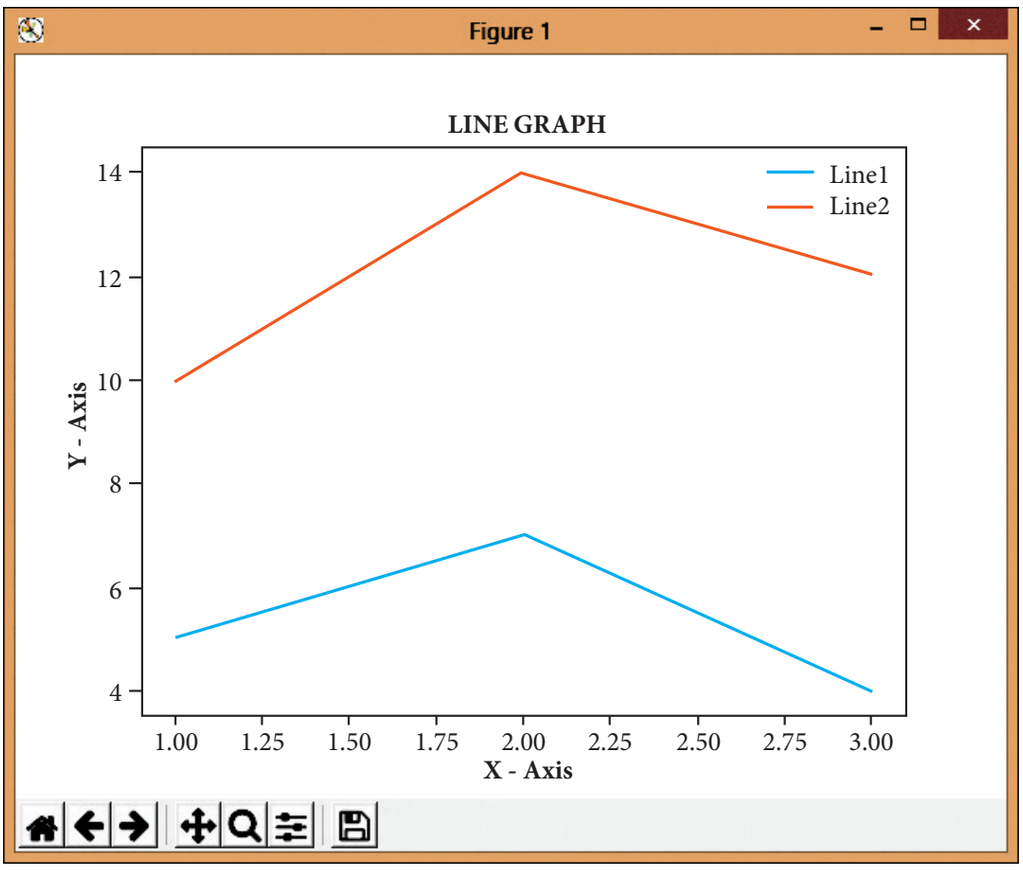


Figure 16.3

Buttons in the output

In the output figure, you can see few buttons at the bottom left corner. Let us see the use of these buttons.

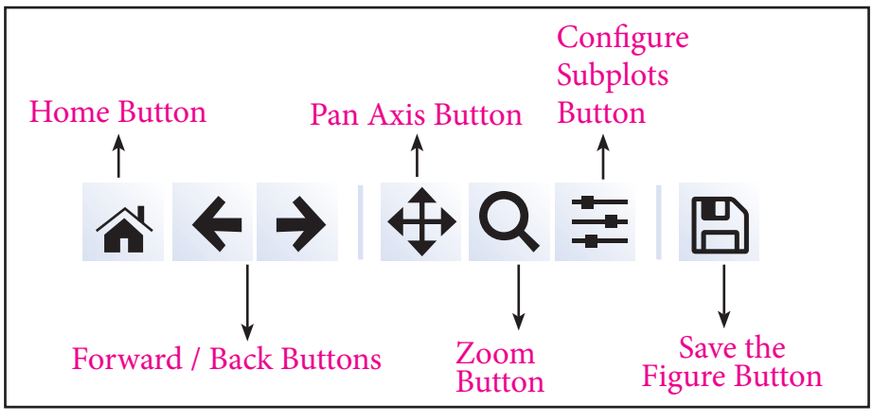


Figure 16.4

Home Button → The Home Button will help once you have begun navigating your chart. If you ever want to return back to the original view, you can click on this.

Forward/Back buttons → These buttons can be used like the Forward and Back buttons in your browser. You can click these to move back to the previous point you were at, or forward again.

Pan Axis → This cross-looking button allows you to click it, and then click and drag your graph around.

Zoom → The Zoom button lets you click on it, then click and drag a square that you would like to zoom into specifically. Zooming in will require a left click and drag. You can alternatively zoom out with a right click and drag.

Configure Subplots → This button allows you to configure various spacing options with your figure and plot.

Save Figure → This button will allow you to save your figure in various forms.

16.3 Special Plot Types

Matplotlib allows you to create different kinds of plots ranging from histograms and scatter plots to bar graphs and bar charts.

Line Chart

A Line Chart or Line Graph is a type of chart which displays information as a series of data points called 'markers' connected by straight line segments. A Line Chart is often used to visualize a trend in data over intervals of time – a time series – thus the line is often drawn chronologically.

Example: Line plot

```
import matplotlib.pyplot as plt
years = [2014, 2015, 2016, 2017, 2018]
total_populations = [8939007, 8954518, 8960387, 8956741, 8943721]
plt.plot (years, total_populations)
plt.title ("Year vs Population in India")
plt.xlabel ("Year")
plt.ylabel ("Total Population")
plt.show()
```

In this program,

Plt.title() → specifies title to the graph

Plt.xlabel() → specifies label for X-axis

Plt.ylabel() → specifies label for Y-axis

Output

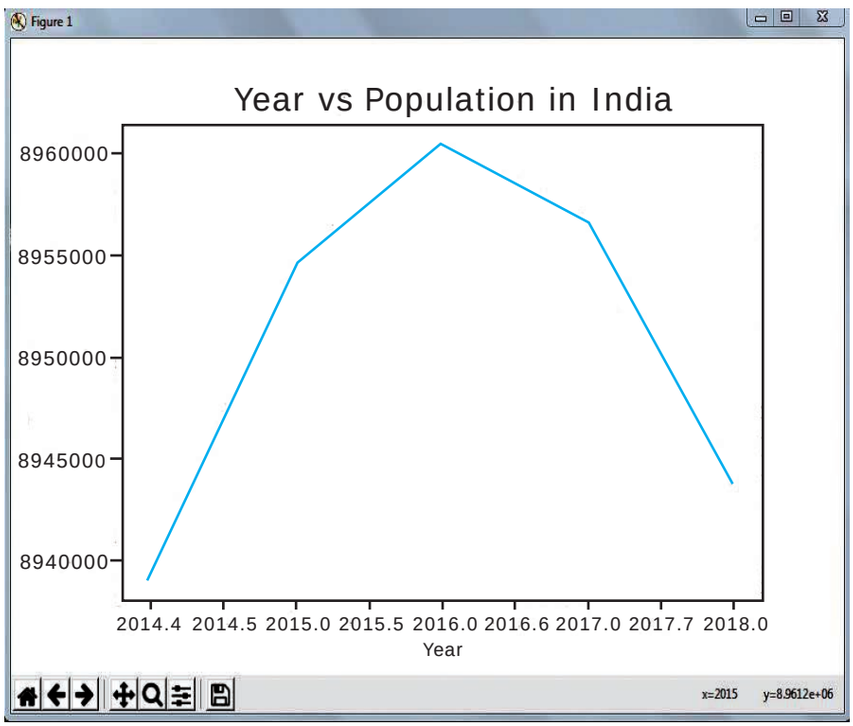


Figure 16.5

Bar Chart

A BarPlot (or BarChart) is one of the most common type of plot. It shows the relationship between a numerical variable and a categorical variable.

Bar chart represents categorical data with rectangular bars. Each bar has a height corresponds to the value it represents. The bars can be plotted vertically or horizontally. It's useful when we want to compare a given numeric value on different categories. To make a bar chart with Matplotlib, we can use the plt.bar() function.

Example

```
import matplotlib.pyplot as plt
# Our data
labels = ["TAMIL", "ENGLISH", "MATHS", "PHYSICS", "CHEMISTRY", "CS"]
usage = [79.8, 67.3, 77.8, 68.4, 70.2, 88.5]
# Generating the y positions. Later, we'll use them to replace them with labels.
y_positions = range (len(labels))
# Creating our bar plot
plt.bar (y_positions, usage)
plt.xticks (y_positions, labels)
plt.ylabel ("RANGE")
plt.title ("MARKS")
plt.show()
```

Output

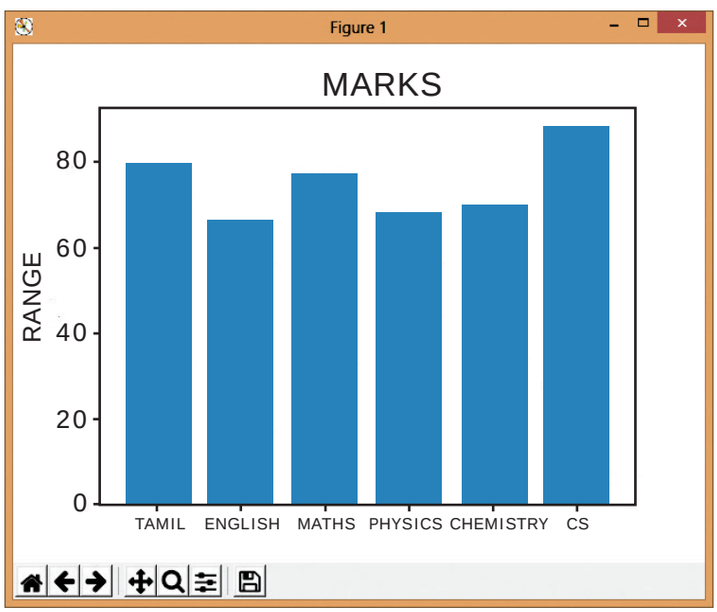


Figure 16.6

The above code represents the following:

Labels → Specifies labels for the bars.

Usgae → Assign values to the labels specified.

Xticks → Display the tick marks along the x-axis at the values represented. Then specify the label for each tick mark.

Range → Create sequence of numbers.

DO YOU KNOW?

Bar Graph and Histogram are the two ways to display data in the form of a diagram.

Key Differences Between Histogram and Bar Graph

The differences between Histogram and bar graph are as follows

1. Histogram refers to a graphical representation; that displays data by way of bars to show the frequency of numerical data. A bar graph is a pictorial representation of data that uses bars to compare different categories of data.
2. A histogram represents the frequency distribution of continuous variables. Conversely, a bar graph is a diagrammatic comparison of discrete variables.
3. Histogram presents numerical data whereas bar graph shows categorical data.
4. The histogram is drawn in such a way that there is no gap between the bars. On the ot her hand, there is proper spacing between bars in a bar graph that indicates discontinuity.
5. Items of the histogram are numbers, which are categorised together, to represent ranges of data. As opposed to the bar graph, items are considered as individual entities.
6. In the case of a bar graph, it is quite common to rearrange the blocks, from highest to lowest. But with histogram, this cannot be done, as they are shown in the sequence of classes.
7. The width of rectangular blocks in a histogram may or may not be same while the width of the bars in a bar graph is always same.

Pie Chart

Pie Chart is probably one of the most common type of chart. It is a circular graphic which is divided into slices to illustrate numerical proportion. The point of a pie chart is to show the relationship of parts out of a whole.

To make a Pie Chart with Matplotlib, we can use the ***plt.pie()*** function. The autopct parameter allows us to display the percentage value using the Python string formatting.

Example

```
import matplotlib.pyplot as plt

sizes = [89, 80, 90, 100, 75]

labels = ["Tamil", "English", "Maths", "Science", "Social"]

plt.pie (sizes, labels = labels, autopct = "%.2f")

plt.axes().set_aspect ("equal")

plt.show()
```

Output

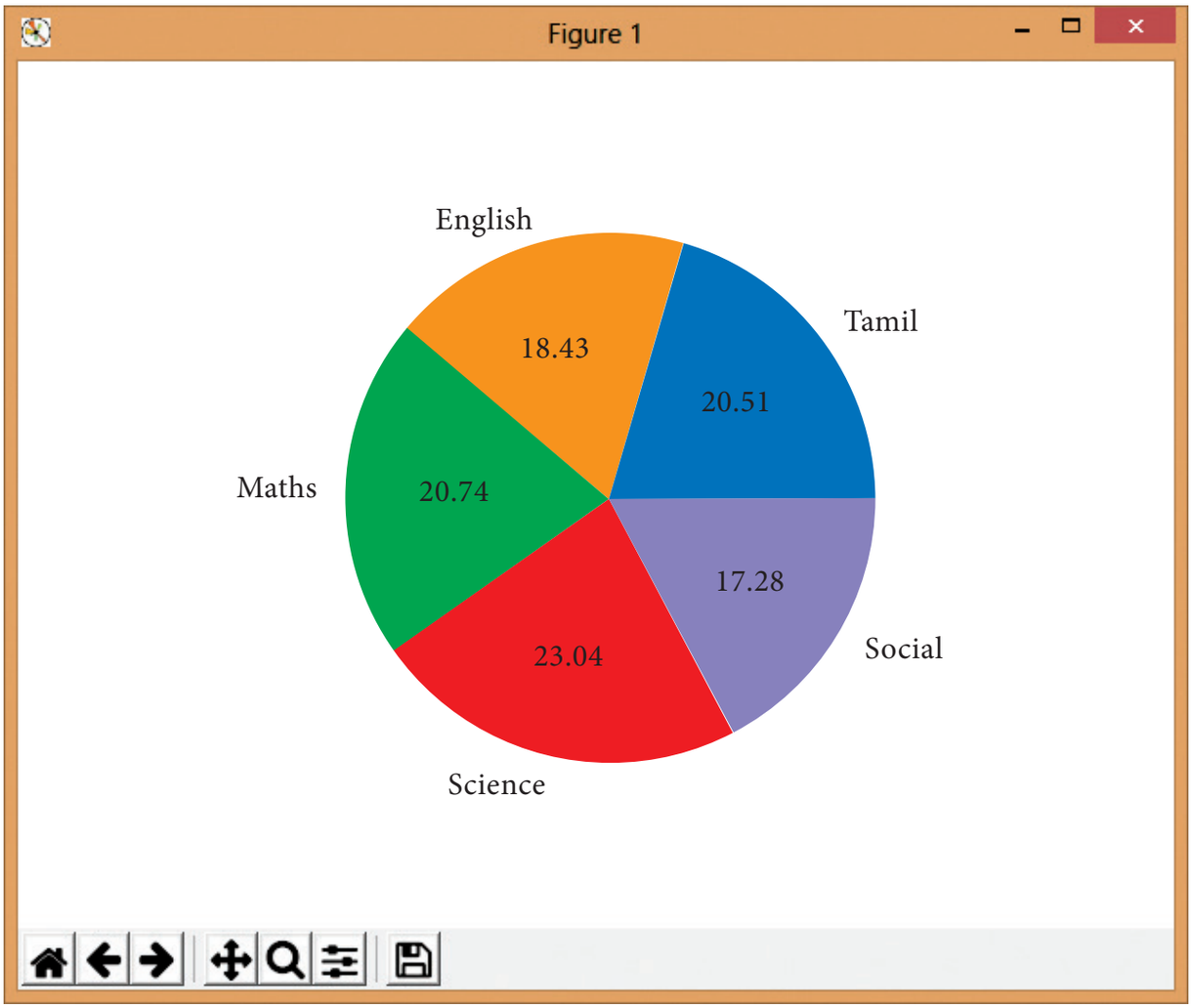


Figure 16.7

Hands on Practice

1. Create a plot. Set the title, the *x* and *y* labels for both axes.
2. Plot a pie chart for your marks in the recent examination.
3. Plot a line chart on the academic performance of Class 12 students in Computer Science for the past 10 years.
4. Plot a bar chart for the number of computer science periods in a week.

Evaluation

T39IUH

Part - I

1. Which is a python package used for 2D graphics?

   a. matplotlib.pyplot b. matplotlib.pip

   c. matplotlib.numpy d. matplotlib.plt

2. Identify the package manager for Python packages, or modules.

   a. Matplotlib b. PIP c. plt.show() d. python package

3. Read the following code: Identify the purpose of this code and choose the right option from the following.

   C:\Users\YourName\AppData\Local\Programs\Python\Python36-32\Scripts>pip –version

   a. Check if PIP is Installed b. Install PIP

   c. Download a Package d. Check PIP version

4. Read the following code: Identify the purpose of this code and choose the right option from the following.

   C:\Users\Your Name\AppData\Local\Programs\Python\Python36-32\Scripts>pip list

   a. List installed packages b. list command

   c. Install PIP d. packages installed

5. To install matplotlib, the following function will be typed in your command prompt. What does "-U"represents?

   Python –m pip install –U pip

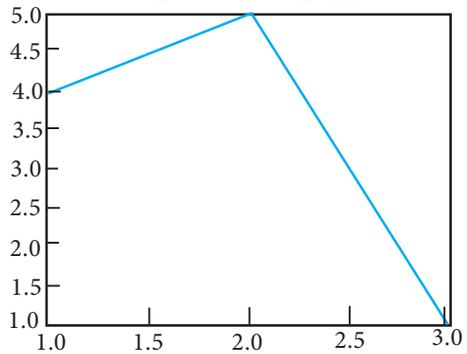
   a. downloading pip to the latest version

   b. upgrading pip to the latest version

   c. removing pip

   d. upgrading matplotlib to the latest version

6. Observe the output figure. Identify the coding for obtaining this output.



a. import matplotlib.pyplot as plt
plt.plot([1,2,3],[4,5,1])
plt.show()

b. import matplotlib.pyplot as plt
plt.plot([1,2],[4,5])
plt.show()

c. import matplotlib.pyplot as plt
plt.plot([2,3],[5,1])
plt.show()

d. import matplotlib.pyplot as plt
plt.plot([1,3],[4,1])
plt.show()

7. Read the code:

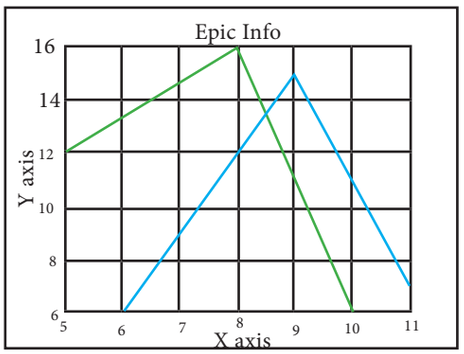
a. import matplotlib.pyplot as plt

b. plt.plot(3,2)

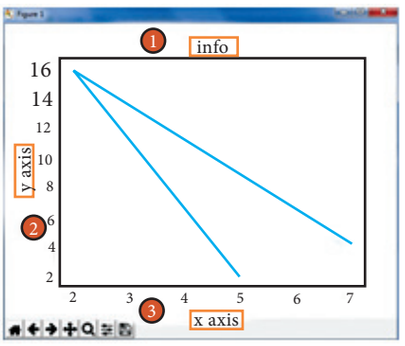
c. plt.show()

Identify the output for the above coding.

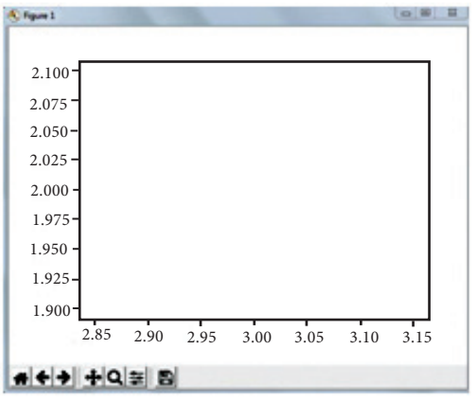
a.



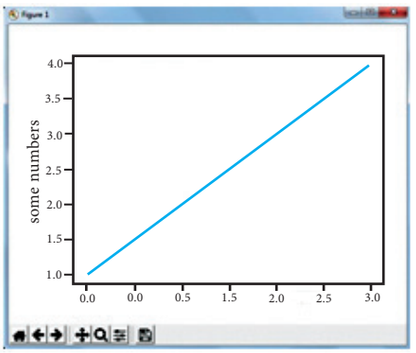
b.



c.



d.



8. Which key is used to run the module?

   a. F6 b. F4 c. F3 d. F5

9. Identify the right type of chart using the following hints.

   Hint 1: This chart is often used to visualize a trend in data over intervals of time.

   Hint 2: The line in this type of chart is often drawn chronologically.

   a. Line chart b. Bar chart c. Pie chart d. Scatter plot

10. Read the statements given below. Identify the right option from the following for pie chart.

    Statement A: To make a pie chart with Matplotlib, we can use the plt.pie() function.

    Statement B: The autopct parameter allows us to display the percentage value using the Python string formatting.

    a. Statement A is correct b. Statement B is correct

    c. Both the statements are correct d. Both the statements are wrong

Part - II

1. Define: Data Visualization.
2. List the general types of data visualization.
3. List the types of Visualizations in Matplotlib.
4. How will you install Matplotlib?

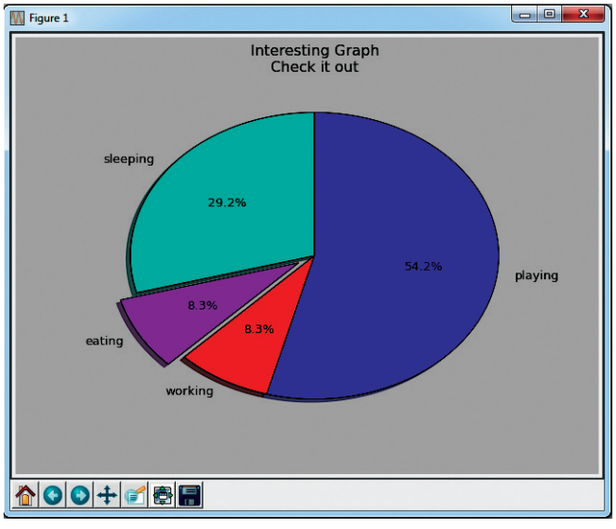
5. Write the difference between the following functions: plt.plot([1,2,3,4]), plt.plot([1,2,3,4], [1,4,9,16]).

Part - III

1. Draw the output for the following data visualization plot.

```
import matplotlib.pyplot as plt
plt.bar([1,3,5,7,9],[5,2,7,8,2], label="Example one")
plt.bar([2,4,6,8,10],[8,6,2,5,6], label="Example two", color='g')
plt.legend()
plt.xlabel('bar number')
plt.ylabel('bar height')
plt.title('Epic Graph\nAnother Line! Whoa')
plt.show()
```

2. Write any three uses of data visualization.
3. Write the coding for the following:
   a. To check if PIP is Installed in your PC.
   b. To Check the version of PIP installed in your PC.
   c. To list the packages in matplotlib.
4. Write the plot for the following pie chart output.



Part - IV

1. Explain in detail the types of pyplots using Matplotlib.
2. Explain the various buttons in a matplotlib window.
3. Explain the purpose of the following functions:
   a. plt.xlabel

b. plt.ylabel

c. plt.title

d. plt.legend()

e. plt.show()

Reference

1. https://towards datascience.com / data - science - with - python - intro -to- data -visualization-and-matplotlib-5f799b7c6d82.
2. https://heartbeat.fritz.ai/introduction-to-matplotlib-data-visualization-in-python-d9143287ae39.
3. https://python programming.net / legends - titles - labels - matplotlib - tutorial/?completed=/matplotlib-intro-tutorial/.
4. https://keydifferences.com/difference-between-histogram-and-bar-graph.html.

GLOSSARY

THSDVT

Terminology	Meaning
Access control	*security technique that regulates who or what can view or use resources in a computing environment*
Access modifiers	*Private , Protected and Public*
Alternative	*One of two choices*
append()	*Used to add an element in a list*
Argument	*Argument is the actual value of this variable that gets passed to function.*
argv	*An array containing the values passed through command line argument*
Attribute	*Data items that makes up an object*
Authorization	*Giving permission or access*
Block	*Set of Statements*
Boolean	*means Logical*
break	*Exit the control*
c = sqlite3.connect('test. db')	*create a database connection to the SQLite database 'test.db'. You can also supply the special name :memory: to create a database in RAM.*
c.close()	*To release the connection of the database*
c.commit()	*To save the changes made in the table*
c.execute()	*Executes all SQL commands .Accepts two kinds of placeholders: question marks ? ("qmark style") and named placeholders :name ("named style").*
Cartesian product	*Cartesian operation is helpful to merge columns from two relations*
cd	*cd command refers to change directory*
Class	*Template of creating objects.*
Class variable	*An ordinary variable declared inside a class*
cls	*To clear the screen in command window*
Comma(,)	*Comma is used to separate each data in a csv file*

compiler	*Scans the entire program and translates it as a whole into machine code. It generates the error message only after scanning the whole program. Hence debugging is comparatively hard.*
Conjunction	*Concurrence, coincidence*
Constraint	*Restriction or limitation*
Constructor	*A special function get execution automatically when an object enter into scope.*
continue	*To skip the remaining part and start with next iteration.*
CRUD	*Create, Read, Update and Delete*
csv.reader()	*The reader function is designed to take each line of the file and make a list of all columns*
csv.register_dialect()	*A dialect describes the format of the csv file that is to be read*
CsvQuote All	*If quoting is set to csvquote all, then writerow() will quote all fields.*
cur = c.cursor()	*Creating cursor object*
cur.fetchall()	*method to get a list of the matching rows.*
cur.fetchmany()	*method that returns the next number of rows (n) of the result set*
cur.fetchone()	*method to retrieve a single matching row*
CWI	*Centrum Wiskunde & Informatica*
DBA	*DataBase Adminstrator*
DBMS	*Database Management System*
def	*This keyword is used to define function.*
Destructor	*A special function get execution automatically when an object exit from its scope.*
dict()	*It is used to print the data in dictionary format without orderdict*
Dictionary	*Collection of Key-Value pairs*
DictReader()	*Works by reading in the first line of the CSV and using each column comma separated value in this line as a dictionary key.*
Dictwriter()	*Write dictionary data into a CSV file*
elif	*else…if*
Embedded	*Firmly attached*
Enter key	*Enter key or newline is used to create rows in a csv file*

eval()	*This function is used to evaluate the value of a string.*
Father of Relational Database	*Dr. Edgar Frank Codd*
g++	*compiler to compile c++ program*
GIS	*Geographic Information System*
global Scope	*A variable, with global scope can be used anywhere in the program.*
Glue language	*You do not write the complete application in the language, but rather, you use the language to orchestrate(organize) modules written in (possibly many different) other languages, making them work together to form the application. A glue language makes it easy to do that (convenient syntax, good support for inter-process communication and data managing, no compilation step etc).*
id ()	*It returns the memory address of the given object.*
IDLE	*Integrated Development Environment*
immutable	*unchangeable*
Implementation	*Implementation carries out the operation declared in the interfac*
import	*Import in python is similar to #include header_file in C++. Python modules can get access to code from another module by importing the file/function using "import" statement.*
Impure Functions	*Any function that changes the internal state of one of its arguments or the value of some external variable is an impure function.*
Instantiation	*Process of creating an object*
Integrity	*Whole and undivided*
Interactive Mode	*A way of using the Python interpreter by typing command and expressions at the prompt.*
Interface	*Interface defines what an object can do, but doesn't actually do it*
interpreter	*Translates program one statement at a time. It continues translating the program until the first error is met, in which case it stops. Hence debugging is easy.*
Intersection	*Intersection defines a relation consisting of a set of all tuple that are in both A and B.*
Key	*Data that is mapped to a value in a dictionary*

lambda	*Lambda function is mostly used for creating small and one- time anonymous function.*
LEGB rule	*Local → Enclosed → Global → Built-in scope*
List	*Mutable ordered collection of values*
local Scope	*A variable declared inside the function's body or in a block is called local scope.*
Looping	*Repetition*
Mapping	*The process of binding a variable name with an object*
Method	*A function declared and defined inside a class.*
module	*A module is a file containing Python definitions and statements. The file name is the module name with the suffix .py appended. Within a module, the module's name (as a string) is available as the value of the global variable name .*
Namespaces	*containers for mapping names of variables to objects*
Nested Block	*A block within a block is called nested block.*
next()	*The next() function returns the next item from the iterator. It can also be used to skip a row of the csv file*
Object	*Collection of Data and Functions.*
Object Oriented Programming	*Computer Programming concept based on real world objects.*
operator.itemgetter(col_ no)	*To sort by more than one column from a csv file*
os.system()	*Used to execute system command and here in our python program is used to compile the c++ program using g++*
Parameter	*Parameter is variable in the declaration of function definition.*
parse	*To split an input into pieces of data that can be easily stored or manipulated.*
pass	*Can be used as placeholder in functions and loops.*
Projection (π)	*The projection eliminates all attributes of the input relation but those mentioned in the projection list*
Prompt	*Character (<<<) displayed by the interpreter to indicate that it is ready to take input from the user.*
Pure Functions	*Pure functions always returns the same result if the same arguments are passed in*
Python prompt	>>>

Queue	*Queue is an abstract data structure, somewhat similar to Stacks. Unlike stacks, a queue is open at both its ends. One end is always used to insert data(enqueue) and the other is used to remove data(dequeue). Queue follows First-In-First-Out methodology, i.e., the data item stored first will be accessed first.*
RDBMS	*Relational Database Management System*
recursion	*When a function calls itself is known as recursion.*
Redundant	*Duplication of data*
Routines	*routines are otherwise called as functions or methods. In Python it is also called as definition*
Schema	*Structure or model*
Scope	*Visibility of variables, parameters and functions in one part of a program to another part of the same program.*
Script	*A Python program stored in a file.*
Script Mode	*A way of using the Python interpreter by typing command and expressions at the prompt.*
Select (σ)	*The SELECT operation is used for selecting a subset of the tuples according to a given selection condition*
Selectors	*Functions that retrieve information from the data type.*
Sequential	*One after another*
Set difference(-)	*(-) symbol denotes it. The result of A-B is a relation which includes all tuples that are in A but not in B.*
skipinitialspace=true	*When true, whitespace immediately following the delimeter is ignored. The default is false*
slicing	*cut*
Stack	*A stack (sometimes called a "push-down stack") is an ordered collection of items where the addition of new items and the removal of existing items always takes place at the same end. This end is commonly referred to as the "top." The end opposite the top is known as the "base.This ordering principle is sometimes called LIFO, last- in first-out.*
stride	*a long step*
string	*sequence of letters, numbers or symbols*
subscript	*an index number*

Syntax	*The structure of a program*
Syntax Error	*An error in a program that makes it impossible to parse.*
Token	*One of the basic elements of the syntactic structure of a program.*
Tuple	*It is a sequence of immutable(not changeable) objects. Tuples are sequences, just like lists.Tuples are defined by having values between parentheses ( ).*
Union operation(U)	*Union is symbolized by symbol. It includes all tuples that are in tables A or in B.*
variable	*Memory box to store values*
writer ow()	*Method to write a single row of data in a file*
writer ows()	*Method to write multiple rows of data in a file*
FALSE	*Logical value 0*
TRUE	*Logical value 1*

ANNEXURE - 1

List of Python Functions

I. Built-in Functions

Function	Description
abs()	returns absolute value of a number
all()	returns true when all elements in iterable is true
any()	Checks if any Element of an Iterable is True
ascii()	Returns String Containing Printable Representation
bin()	converts integer to binary string
bool()	Converts a Value to Boolean
bytearray()	returns array of given byte size
bytes()	returns immutable bytes object
callable()	Checks if the Object is Callable
chr()	Returns a Character (a string) from an Integer
classmethod()	returns class method for given function
compile()	Returns a Python code object
complex()	Creates a Complex Number
delattr()	Deletes Attribute From the Object
dir()	Tries to Return Attributes of Object
divmod()	Returns a Tuple of Quotient and Remainder
enumerate()	Returns an Enumerate Object
eval()	Runs Python Code Within Program
exec()	Executes Dynamically Created Program
filter()	constructs iterator from elements which are true
float()	returns floating point number from number, string
format()	returns formatted representation of a value
getattr()	returns value of named attribute of an object
globals()	returns dictionary of current global symbol table
hasattr()	returns whether object has named attribute
hash()	returns hash value of an object
help()	Invokes the built-in Help System
hex()	Converts to Integer to Hexadecimal
id()	Returns Identify of an Object
isinstance()	Checks if a Object is an Instance of Class
issubclass()	Checks if a Object is Subclass of a Class
iter()	returns iterator for an object
len()	Returns Length of an Object
locals()	Returns dictionary of a current local symbol table
map()	Applies Function and Returns a List
max()	returns largest element
memoryview()	returns memory view of an argument
min()	returns smallest element
next()	Retrieves Next Element from Iterator
object()	Creates a Featureless Object
oct()	converts integer to octal

open()	Returns a File object
ord()	returns Unicode code point for Unicode character
pow()	returns x to the power of y
print()	Prints the Given Object
property()	returns a property attribute
range()	return sequence of integers between start and stop
repr()	returns printable representation of an object
reversed()	returns reversed iterator of a sequence
round()	rounds a floating point number to ndigits places.
set()	returns a Python set
setattr()	sets value of an attribute of object
slice()	creates a slice object specified by range()
sorted()	returns sorted list from a given iterable
staticmethod()	creates static method from a function
str()	returns informal representation of an object
sum()	Add items of an Iterable
super()	Allow you to Refer Parent Class by super
type()	Returns Type of an Object
vars()	Returns __dict__ attribute of a class
__import__()	Advanced Function Called by import

II. String Functions

Function	Description
capitalize()	Converts first character to Capital Letter
casefold()	converts to casefolded strings
center()	Pads string with specified character
count()	returns occurrences of substring in string
encode()	returns encoded string of given string
endswith()	Checks if String Ends with the Specified Suffix
expandtabs()	Replaces Tab character With Spaces
find()	Returns the index of first occurrence of substring
format()	formats string into nicer output
format_map()	Formats the String Using Dictionary
index()	Returns Index of Substring
input()	reads and returns a line of string
int()	returns integer from a number or string
isalnum()	Checks Alphanumeric Character
isalpha()	Checks if All Characters are Alphabets
isdecimal()	Checks Decimal Characters
isdigit()	Checks Digit Characters
isidentifier()	Checks for Valid Identifier
islower()	Checks if all Alphabets in a String are Lowercase
isnumeric()	Checks Numeric Characters
isprintable()	Checks Printable Character
isspace()	Checks Whitespace Characters

istitle()	Checks for Titlecased String
isupper()	returns if all characters are uppercase characters
join()	Returns a Concatenated String
ljust()	returns left-justified string of given width
lower()	returns lowercased string
lstrip()	Removes Leading Characters
maketrans()	returns a translation table
partition()	Returns a Tuple
replace()	Replaces Substring Inside
rfind()	Returns the Highest Index of Substring
rindex()	Returns Highest Index of Substring
rjust()	returns right-justified string of given width
rpartition()	Returns a Tuple
rsplit()	Splits String From Right
rstrip()	Removes Trailing Characters
slice()	creates a slice object specified by range()
split()	Splits String from Left
splitlines()	Splits String at Line Boundaries
startswith()	Checks if String Starts with the Specified String
strip()	Removes Both Leading and Trailing Characters
swapcase()	swap uppercase characters to lowercase; vice versa
title()	Returns a Title Cased String
translate()	returns mapped charactered string
upper()	returns uppercased string
zfill()	Returns a Copy of The String Padded With Zeros

III. List Functions

Function	Description
append()	Add Single Element to The List
clear()	Removes all Items from the List
copy()	Returns Shallow Copy of a List
count()	returns occurrences of element in a list
extend()	Add Elements of a List to Another List
index()	returns smallest index of element in list
insert()	Inserts Element to The List
list() Function	creates list in Python
pop()	Removes Element at Given Index
remove()	Removes Element from the List
reverse()	Reverses a List
slice()	creates a slice object specified by range()
sort()	sorts elements of a list

IV. Tuple Functions

Function	Description
count()	returns occurrences of element in a tuple
index()	returns smallest index of element in tuple
slice()	creates a slice object specified by range()
tuple() Function	Creates a Tuple
zip()	Returns an Iterator of Tuples

V. Set Functions

Function	Description
add()	adds element to a set
clear()	remove all elements from a set
copy()	Returns Shallow Copy of a Set
difference()	Returns Difference of Two Sets
difference_update()	Updates Calling Set With Intersection of Sets
discard()	Removes an Element from The Set
frozenset()	returns immutable frozenset object
intersection()	Returns Intersection of Two or More Sets
intersection_update()	Updates Calling Set With Intersection of Sets
isdisjoint()	Checks Disjoint Sets
issubset()	Checks if a Set is Subset of Another Set
issuperset()	Checks if a Set is Superset of Another Set
pop()	Removes an Arbitrary Element
remove()	Removes Element from the Set
set()	returns a Python set
symmetric_difference()	Returns Symmetric Difference
symmetric_difference_update()	Updates Set With Symmetric Difference
union()	Returns Union of Sets
update()	Add Elements to The Set.

VI. Dictionary Functions

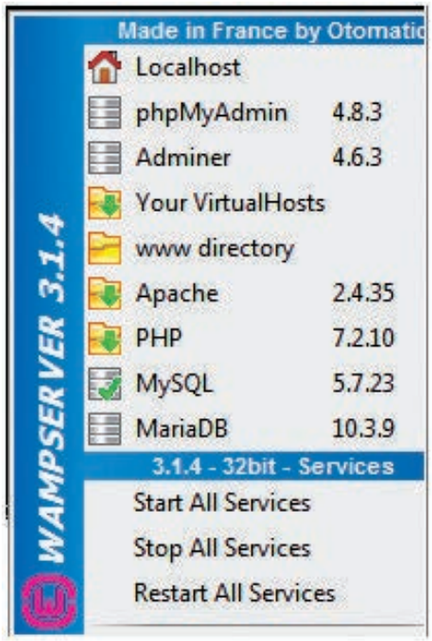
Function	Description
clear()	Removes all Items
copy()	Returns Shallow Copy of a Dictionary
dict()	Creates a Dictionary
fromkeys()	creates dictionary from given sequence
get()	Returns Value of The Key
items()	returns view of dictionary's (key, value) pair
keys()	Returns View Object of All Keys
pop()	removes and returns element having given key
popitem()	Returns & Removes Element From Dictionary
setdefault()	Inserts Key With a Value if Key is not Present
update()	Updates the Dictionary
values()	returns view of all values in dictionary

ANNEXURE - 2

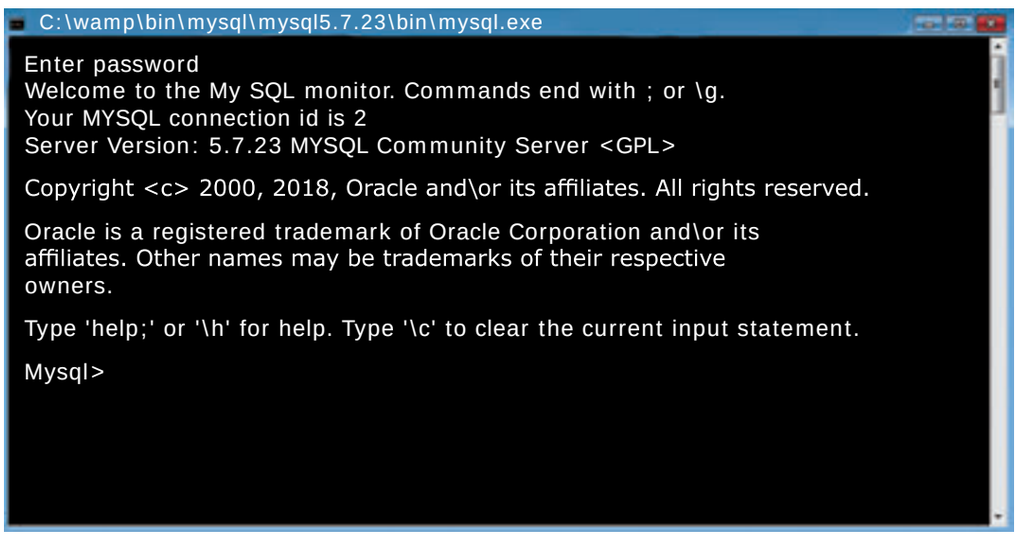
UCK7WT

Installation of MySQL and Creating a database

- Download WAMP based on system configuration and install it in your computer.



- log on to MySQL console as user (root).
- Enter password appears (No need to enter password), press Enter key.
- MySQL prompt appears from where you can enter the SQL commands.



ANNEXURE-3

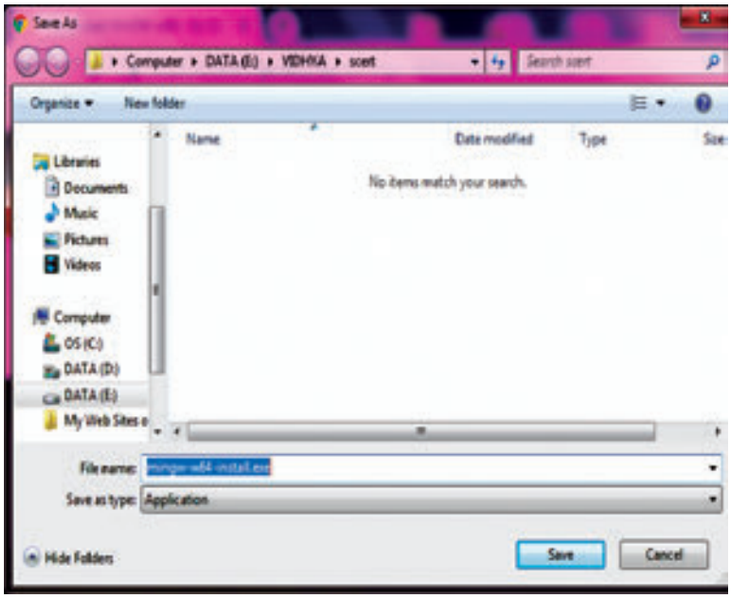
Installation of MinGW

US42Y5

STEPS TO INSTALL MinGW-w64 - for 32 and 64 bit Windows

STEP1: Type https://sourceforge.net/projects/mingw-w64 in any search engine(www.google.in) and double click it.

Step 2 : Click the Download button which appears in the home page. The file should start downloading in your standard download folder or it prompts the save as dialog box

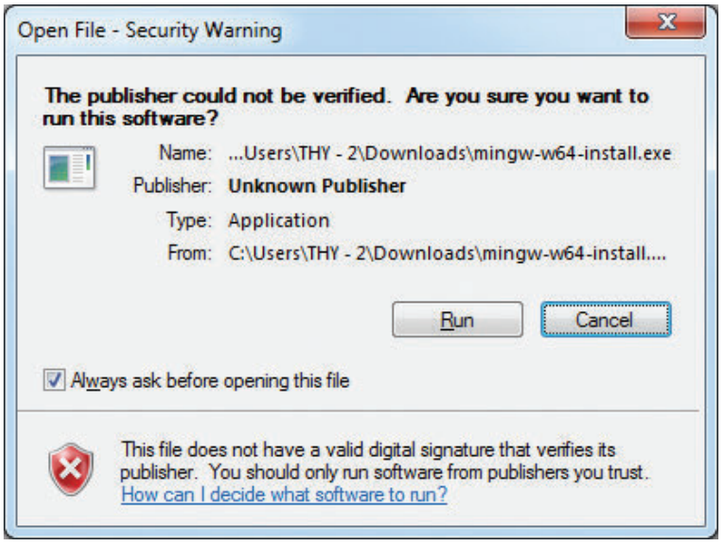


Step 3 : The file should appear as mingw-get-setup.exe . Terminate the window browsing the SourceForge web site. Move this file to a more permanent location, so that you can install MinGW (and reinstall it later, if necessary).

step 4: Start the Installation as per the instructions given bellow.

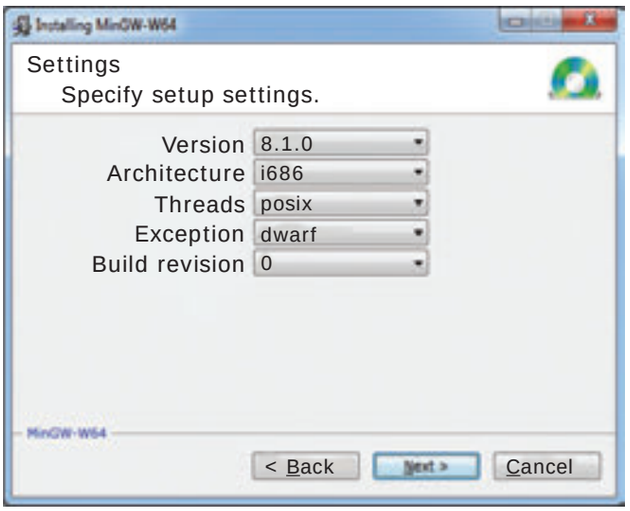
Installing:

1. Double-click the mingw-get-setup.exe . icon. The following pop-up window will appear.

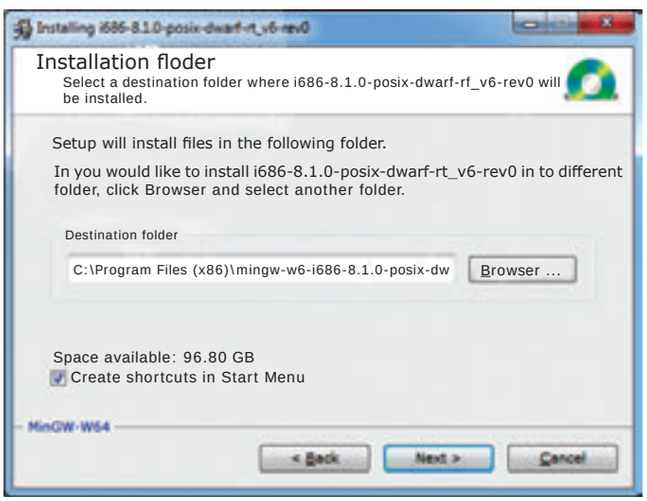


2. Click Run. The following pop-up window will appear. Click next button in the following window

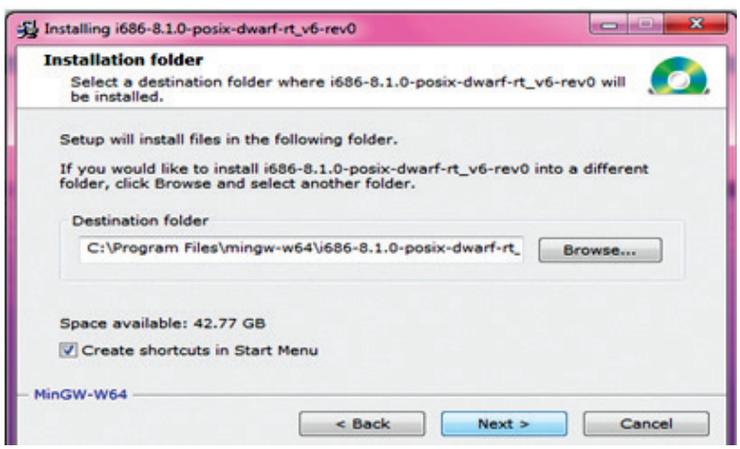
3. The following pop-up window will appear, which specify the Setup settings



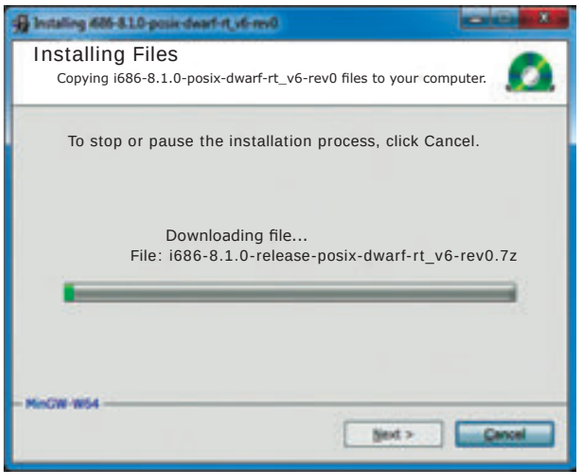
4. Select the destination folder in the following window to install MinGW-W64. Click next button



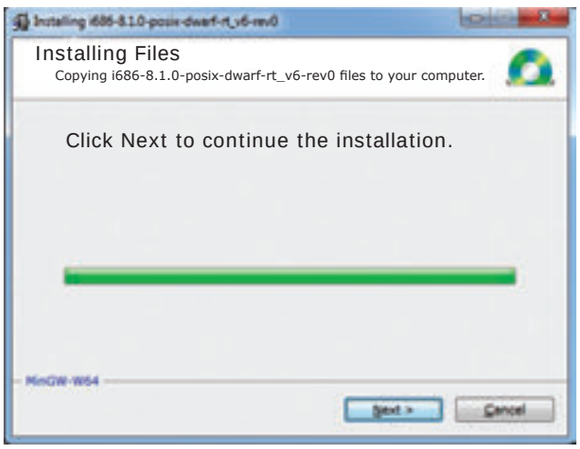
5. Select the destination folder in the following window to install MinGW-W64.



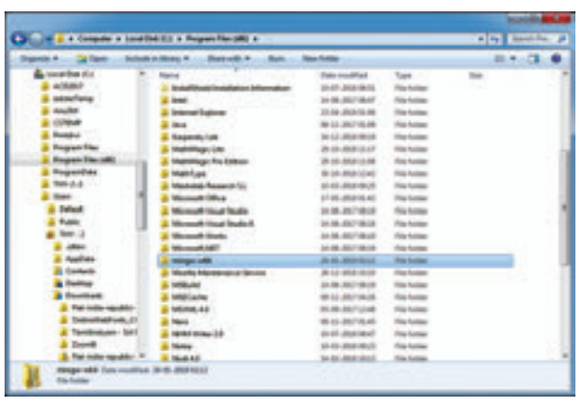
6. Wait for the completion of installation



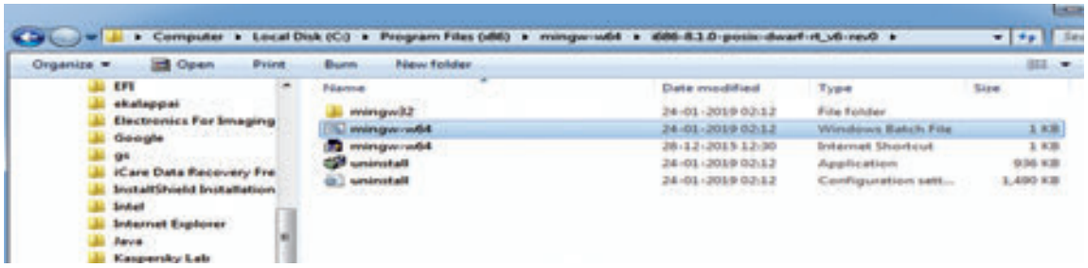
7 Click the button once it gets highlighted.



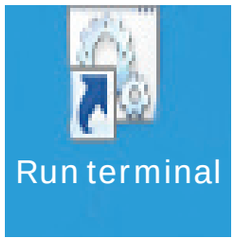
8. Locate the folder in your PC. For example here the mingw64 is present in the following path ***c:\Program Files\mingw-w64.***



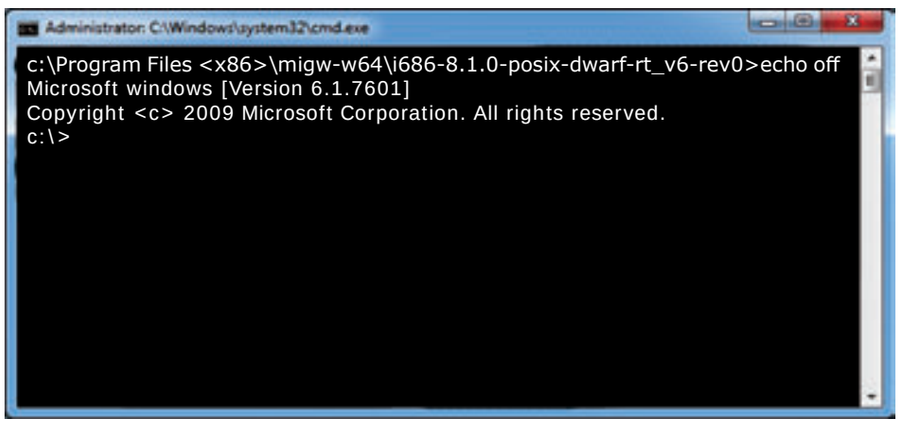
9. Open the folder and double click the batch file. The program will get executed.



10 A shortcut for run will be created in the desktop. Double click and open the "command window".



11 Through this command **window only we have to execute the Python program** because it contains the other programming language program ie is C++. This command window dynamically invoke the g++ compiler to compile and execute C++ program using Python program



ANNEXURE-4

Installation of pip

W2ER1G

First of all you need to identify whether pip is installed in your PC. If so, upgrade the pip in your system. To do this, you need to launch the command prompt. Before trying to install or upgrade the pip, the command will work only if you have appended the path of python directory in the path variable of the windows.

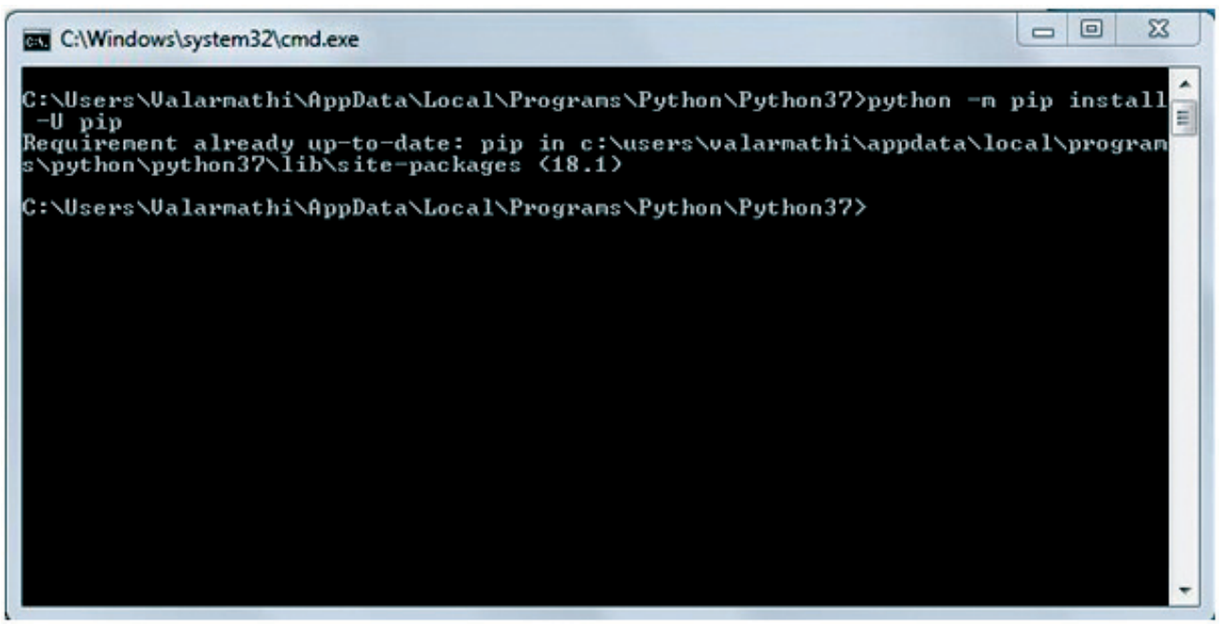
Check if pip is Installed

To check if pip is already installed in your system, navigate your command line to the location of Python's script directory.

You can install the latest version of pip from your command prompt using the following command:

Python –m pip install –U pip

-U represents upgrading pip to the latest version.

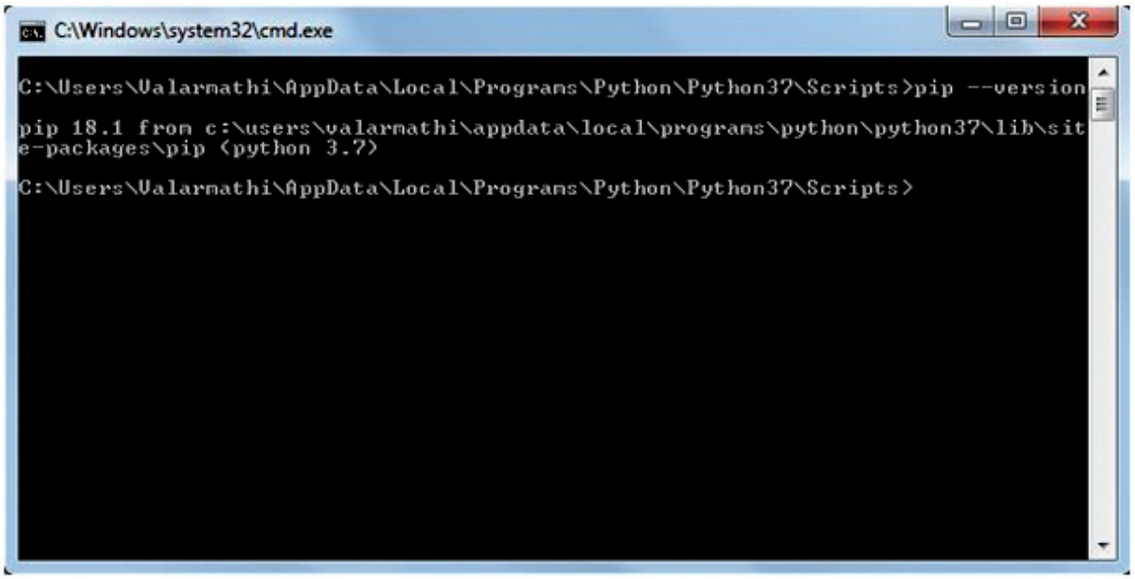


To Check pip version:

To check the version of pip in your system, type the following command:

C:\Users\Your Name\AppData\Local\Programs\Python\Python36-32\Scripts>pip --version

The output in command prompt will look like this:

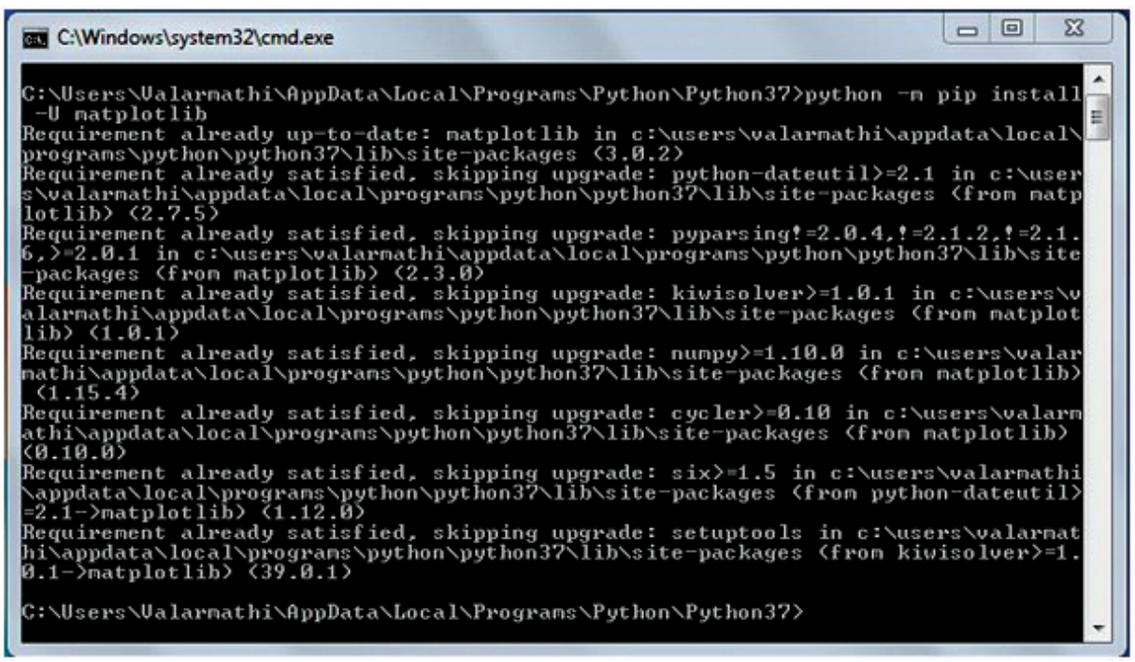


You can see the version of pip installed from the output screen.

To install matplotlib, type the following in your command prompt:

Python –m pip install –U matplotlib

This command will download matplotlib from the source library. If it is already installed the screen will look like the following:

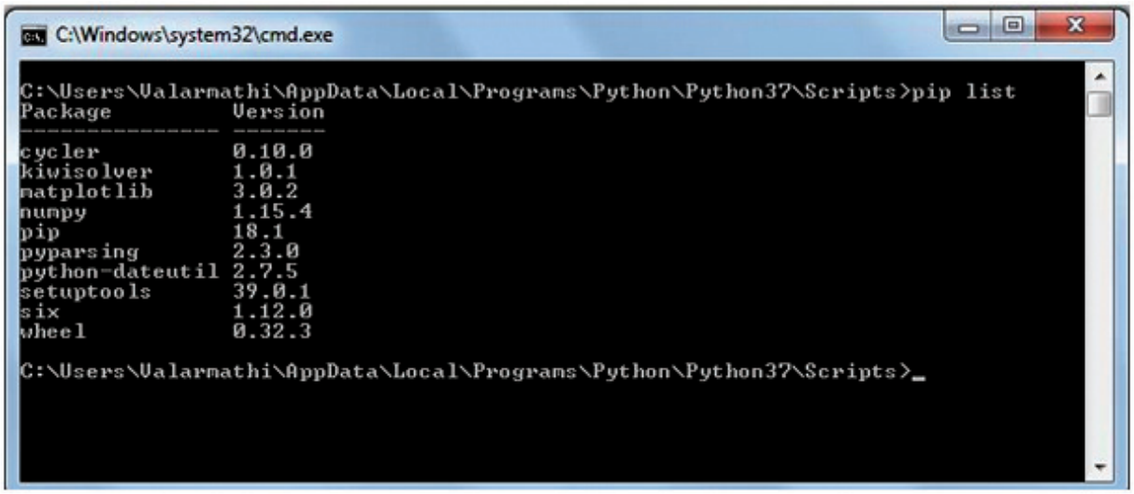


List Packages

To view the list of installed packages on your system, use the List command:

C:\Users\YourName\AppData\Local\Programs\Python\Python36-32\Scripts>pip list The

screen will display the list of all the packages installed on your system.



COMPUTER SCIENCE – XII
List of Authors and Reviewers

Domain Expert

Dr. T.V.Gopal
Professor, Dept of Computer Science and Technology,
College of Engineering, Guindy, Anna University, Chennai.

Reviewers

Dr. Ranjani Parthasarathi
Professor, Dept of Info Sci and Tech, College of Engineering, Guindy,
Anna University, Chennai

Content Expert

Mr. Malaiarasu P
Associate Professor, Dept of Computer Science,
Govt. Arts College, Perumpakkam, Chennai

Dr. Ramesh Kumar M
Assistent Professor and Head of the Department,
Dept of Computer Science,
Govt. Arts College, Nandhanam, Chennai

Dr. Radha P
Assistant Professor, Dept of Information Technology,
Govt. Arts & Science College (A), Coimbatore

Dr. Nester Jeyakumar M
Associate Professor and Head of the Department,
Dept of Computer Science, Loyola College, Chennai

Experts Co-ordinator

Mr. Ravikumar Arumugam
Principal, District Institute of Education and Training,
Mayannur, Karur Dt.

Academic Coordinators

Mrs. Tamil Selvi R
B.T. Assistant,
Government High School, Poonampalayam, Trichy District

Authors

Mr. Kannan K
Post Graduate Teacher, Chennai Girls Hr Sec School,
Rotler street , Chennai

Mr. Ramakrishnan V G
Post Graduate Teacher, Karnataka Sangha Hr Sec School,
T Nagar, Chennai

Mrs. Dr. Vidhya H
Post Graduate Teacher,
DAV Boys Senior Secondary School,
Gopalapuram, Chennai.

Mr. Sreenivasan R
Post Graduate Teacher, Santhome Hr Sec School, Mylapore, Chennai

Mr. Gowrisankar N.V
Post Graduate Teacher, Chennai Girls Hr Sec School,
Nungambakkam, Chennai

Mr. Lenin K
Post Graduate Teacher, Chennai Girls Hr Sec School, Saidapet, Chennai

Mrs. Bindhu Mohandas
Post Graduate Teacher, Vijayanta Model Hr Sec School,
H.V.F Estate , Avadi, Chennai

Mrs. Gajalakshmi R
Post Graduate Teacher, Jaigopal Garodia Hindu Vidyalaya Hr Sec School,
West Mambalam, Chennai

Dr. Valarmathi K E
Post Graduate Teacher, Velammal Vidhyashram, Surapet, Chennai

QR Code Management Team

R. Jaganathan,
SGT, PUMS - Ganesapuram,
Polur , Thiruvannamalai.

J.F. Paul Edwin Roy,
B.T.Asst. PUMS -Rakkipatty, Salem.

M. Saravanan,
B.T.Asst, G.G.H.S.S, Puthupalayam,
Vazhapadi, Salem.

Typist

Mrs. Meena T
SCERT, Chennai.

Art and Design Team

Layout

THY designers and computers
Chennai

In-House

QC - Rajesh Thangapan
- Mathan Raj R
- Balasubramani. R
- Wrapper Design - Kathir Arumugam

Co-ordination

Ramesh Munisamy

This book has been printed on 80 G.S.M.
Elegant Maplitho paper.

Printed by offset at: