PYTHON - An Introduction

Dr RAMANANDA H S[1] Dr SMITA S NAGOUDA[2]
Dr K SUSHAN BAIRY[3] Dr K MADHUKAR[4]

[1] Department of Mathematics, St Joseph Engineering College, Mangaluru, INDIA.
[2] Department of Mathematics,CHRIST(Deemed to be University), Central Campus, Bengaluru, INDIA.
[3] Department of Mathematics, School of Applied Sciences, REVA University, Bengaluru, INDIA.
[4] Department of Mathematics, BMS College of Engineering, Bull Temple Road, Bengaluru, INDIA.

What is PYTHON?

- Python is a computer programming language.

WHAT IS PYTHON?

- Python is a computer programming language.
- High-level general purpose scripting language.

WHAT IS PYTHON?

- Python is a computer programming language.
- High-level general purpose scripting language.
- Beginner-level language that supports development of wide range of applications.

What is PYTHON?

- Python is a computer programming language.
- High-level general purpose scripting language.
- Beginner-level language that supports development of wide range of applications.
- Guido van Rossum (1987) named it after the BBC television show 'Monty Python's Flying Circus.' (Comedy show).

HISTORY

- Created in early 1990's – Guido van Rossum at Stichting Mathematisch Centrum (CWI, see https://www.cwi.nl/), Netherlands as a successor of a language called ABC.

HISTORY

- Created in early 1990's – Guido van Rossum at Stichting Mathematisch Centrum (CWI, see https://www.cwi.nl/), Netherlands as a successor of a language called ABC.
- 1995 – Guido continued work on Python at Corporation for National Research Initiatives (CNRI, see https://www.cnri.reston.va.us/) in Reston, Virginia.

HISTORY

- Created in early 1990's – Guido van Rossum at Stichting Mathematisch Centrum (CWI, see https://www.cwi.nl/), Netherlands as a successor of a language called ABC.
- 1995 – Guido continued work on Python at Corporation for National Research Initiatives (CNRI, see https://www.cnri.reston.va.us/) in Reston, Virginia.
- May 2000 – Guido and the Python core development team move to BeOpen.com to form the BeOpen PythonLabs team.

HISTORY

- Created in early 1990's – Guido van Rossum at Stichting Mathematisch Centrum (CWI, see https://www.cwi.nl/), Netherlands as a successor of a language called ABC.
- 1995 – Guido continued work on Python at Corporation for National Research Initiatives (CNRI, see https://www.cnri.reston.va.us/) in Reston, Virginia.
- May 2000 – Guido and the Python core development team move to BeOpen.com to form the BeOpen PythonLabs team.
- October 2000 – PythonLabs team moved to Digital Creations (now Zope Corporation).

HISTORY

- Created in early 1990's – Guido van Rossum at Stichting Mathematisch Centrum (CWI, see https://www.cwi.nl/), Netherlands as a successor of a language called ABC.
- 1995 – Guido continued work on Python at Corporation for National Research Initiatives (CNRI, see https://www.cnri.reston.va.us/) in Reston, Virginia.
- May 2000 – Guido and the Python core development team move to BeOpen.com to form the BeOpen PythonLabs team.
- October 2000 – PythonLabs team moved to Digital Creations (now Zope Corporation).
- 2001 – Python Software Foundation (PSF) was formed. (Zope corporation is a sponsoring member.)

PYTHON USED FOR...

- Python is used in many places.

PYTHON USED FOR...

- Python is used in many places.
  - Web and Internet Development
  - Desktop GUI Applications
  - Scientific and Numeric
  - Software Development
  - Education
  - Business Applications
  - Games and 3D Graphics
  - Network programming
  - Database Access
- One of the recent growing field of expertise is ‘data science’. Many data scientists use Python for their day-to-day work.

PYTHON'S MAJOR FEATURES

- It is easy to read and write.

Python's major features

- It is easy to read and write.
- Python is an interpreted, interactive, object-oriented programming language

PYTHON'S MAJOR FEATURES

- It is easy to read and write.
- Python is an interpreted, interactive, object-oriented programming language
- It incorporates modules, exceptions, dynamic typing, very high level dynamic data types, and classes.

PYTHON'S MAJOR FEATURES

- It is easy to read and write.
- Python is an interpreted, interactive, object-oriented programming language
- It incorporates modules, exceptions, dynamic typing, very high level dynamic data types, and classes.
- It has interfaces to many system calls and libraries, as well as to various window systems.

PYTHON'S MAJOR FEATURES

- It is easy to read and write.
- Python is an interpreted, interactive, object-oriented programming language
- It incorporates modules, exceptions, dynamic typing, very high level dynamic data types, and classes.
- It has interfaces to many system calls and libraries, as well as to various window systems.
- It is used as an extension language for applications that need a programming interface.

PYTHON'S MAJOR FEATURES

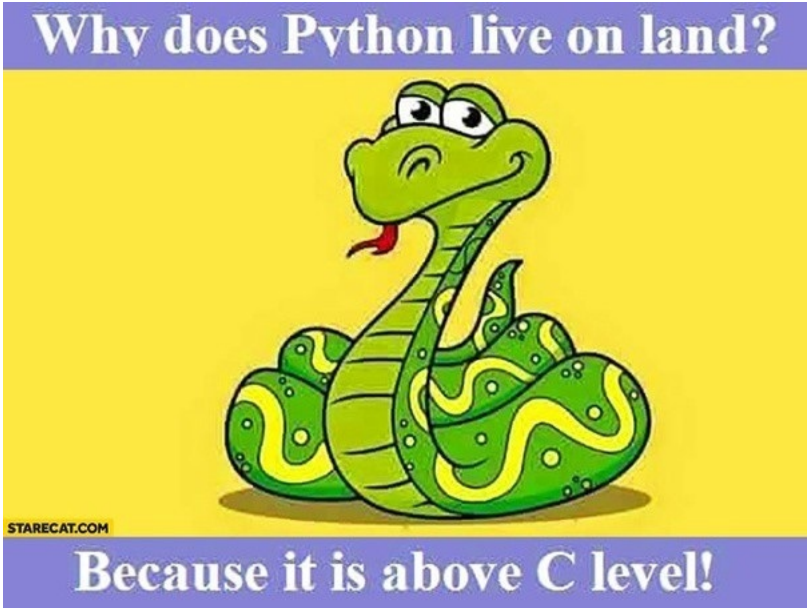
- Python is portable: it runs on many Unix variants including Linux and macOS, and on Windows.

PYTHON'S MAJOR FEATURES

- Python is portable: it runs on many Unix variants including Linux and macOS, and on Windows.
- Python includes a comprehensive base library.

PYTHON'S MAJOR FEATURES

- Python is portable: it runs on many Unix variants including Linux and macOS, and on Windows.
- Python includes a comprehensive base library.
  Addition to this, One can find hundreds of thousands of external packages contributed by the enormous community. You'll find supporting base libraries and packages for pretty much anything you want to accomplish.


Why does Python live on land?
STARECAT.COM
Because it is above C level!

Introduction to Anaconda

- Anaconda is a package manager, environment manager, and Python distribution with a collection of 1,500+ open source packages with free community support. Anaconda is free and easy to install and can be used on Windows, macOS, or Linux.

Introduction to Anaconda

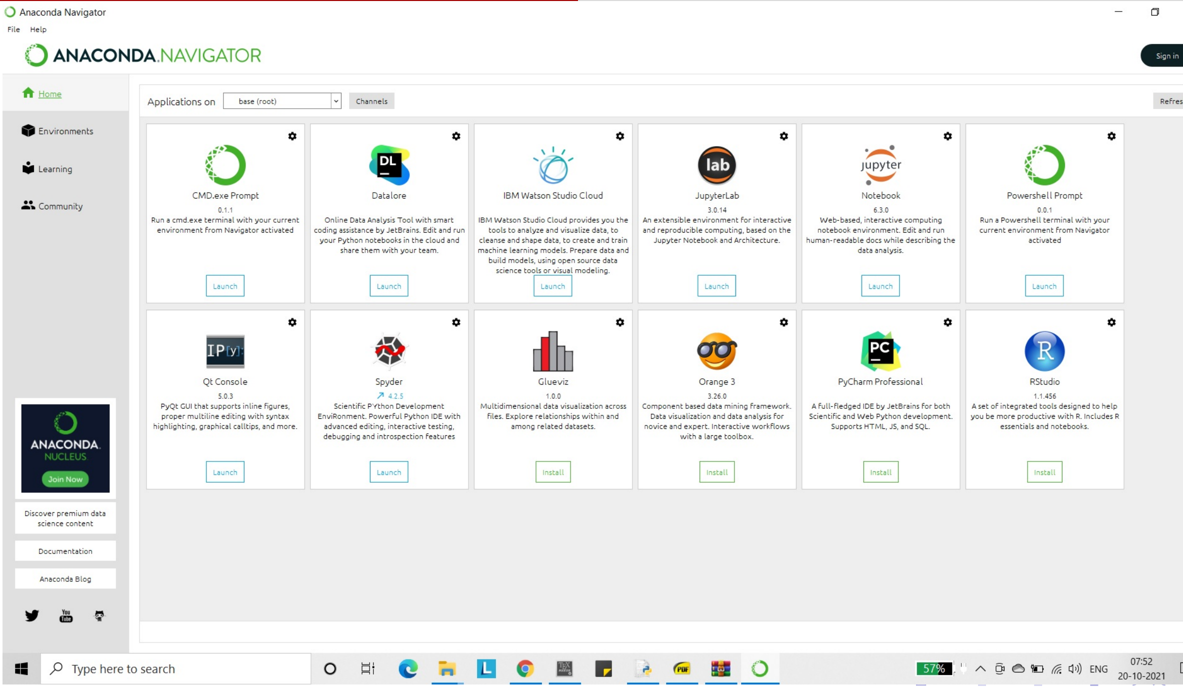
- Anaconda is a package manager, environment manager, and Python distribution with a collection of 1,500+ open source packages with free community support. Anaconda is free and easy to install and can be used on Windows, macOS, or Linux.
- Anaconda can be downloaded from
  `https://www.anaconda.com/products/individual`

Introduction to Anaconda

- Anaconda is a package manager, environment manager, and Python distribution with a collection of 1,500+ open source packages with free community support. Anaconda is free and easy to install and can be used on Windows, macOS, or Linux.
- Anaconda can be downloaded from `https://www.anaconda.com/products/individual`
- After installing Anaconda, we use Anaconda Navigator to launch applications and easily manage packages, environments and channels without using command-line commands.

INTRODUCTION TO ANACONDA

- Anaconda is a package manager, environment manager, and Python distribution with a collection of 1,500+ open source packages with free community support. Anaconda is free and easy to install and can be used on Windows, macOS, or Linux.
- Anaconda can be downloaded from `https://www.anaconda.com/products/individual`
- After installing Anaconda, we use Anaconda Navigator to launch applications and easily manage packages, environments and channels without using command-line commands.
- Navigator is an easy, point-and-click way to work with packages and environments without needing to type conda commands in the terminal window.



Anaconda Navigator

File Help

ANACONDA.NAVIGATOR

Home

Environments

Learning

Community

ANACONDA. NUCLEUS

Join Now

Discover premium data science content

Documentation

Anaconda Blog

Applications on base (root) Channels

CMD.exe Prompt 0.1.1
Run a cmd.exe terminal with your current environment from Navigator activated
Launch

Datalore
Online Data Analysis Tool with smart coding assistance by JetBrains. Edit and run your Python notebooks in the cloud and share them with your team.
Launch

IBM Watson Studio Cloud
IBM Watson Studio Cloud provides you the tools to analyze and visualize data, to cleanse and shape data, to create and train machine learning models. Prepare data and build models, using open source data science tools or visual modeling.
Launch

JupyterLab 3.0.14
An extensible environment for interactive and reproducible computing, based on the Jupyter Notebook and Architecture.
Launch

Notebook 6.3.0
Web-based, interactive computing notebook environment. Edit and run human-readable docs while describing the data analysis.
Launch

Powershell Prompt 0.0.1
Run a Powershell terminal with your current environment from Navigator activated
Launch

Qt Console 5.0.3
PyQt GUI that supports inline figures, proper multiline editing with syntax highlighting, graphical calltips, and more.
Launch

Spyder 4.2.5
Scientific PYthon Development EnviRonment. Powerful Python IDE with advanced editing, interactive testing, debugging and introspection features
Launch

Glueviz 1.0.0
Multidimensional data visualization across files. Explore relationships within and among related datasets.
Install

Orange 3 3.26.0
Component based data mining framework. Data visualization and data analysis for novice and expert. Interactive workflows with a large toolbox.
Install

PyCharm Professional
A full-fledged IDE by JetBrains for both Scientific and Web Python development. Supports HTML, JS, and SQL.
Install

RStudio 1.1.456
A set of integrated tools designed to help you be more productive with R. Includes R essentials and notebooks.
Install

Type here to search

57% ENG 07:52 20-10-2021

Applications in Navigator

The following applications are available by default in Navigator:

- JupyterLab
- Jupyter Notebook
- Spyder
- PyCharm
- VSCode
- Glueviz
- Orange 3 App
- RStudio
- Anaconda Prompt (Windows only)
- Anaconda PowerShell (Windows only)

JUPYTER NOTEBOOK

The notebook extends the console-based approach to interactive computing in a qualitatively new direction, providing a web-based application suitable for capturing the whole computation process: developing, documenting, and executing code, as well as communicating the results.

JUPYTER NOTEBOOK

The notebook extends the console-based approach to interactive computing in a qualitatively new direction, providing a web-based application suitable for capturing the whole computation process: developing, documenting, and executing code, as well as communicating the results.
The Jupyter notebook combines two components:

JUPYTER NOTEBOOK

The notebook extends the console-based approach to interactive computing in a qualitatively new direction, providing a web-based application suitable for capturing the whole computation process: developing, documenting, and executing code, as well as communicating the results.
The Jupyter notebook combines two components:
A web application: a browser-based tool for interactive authoring of documents which combine explanatory text, mathematics, computations and their rich media output.

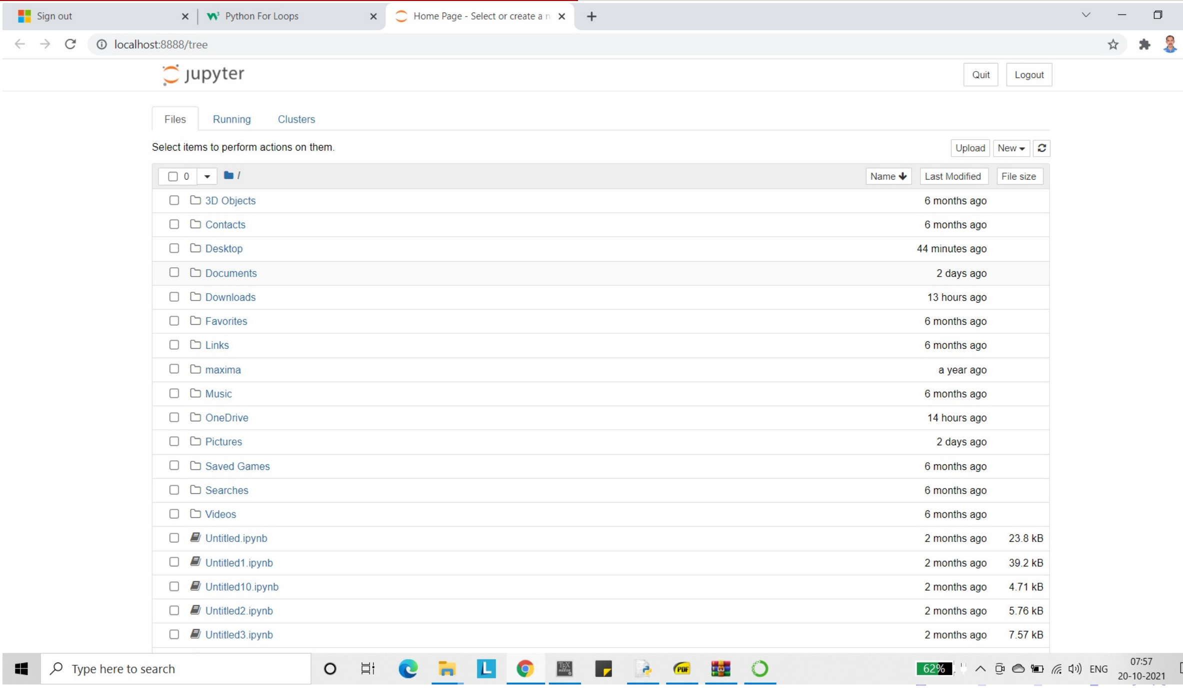
JUPYTER NOTEBOOK

The notebook extends the console-based approach to interactive computing in a qualitatively new direction, providing a web-based application suitable for capturing the whole computation process: developing, documenting, and executing code, as well as communicating the results.

The Jupyter notebook combines two components:

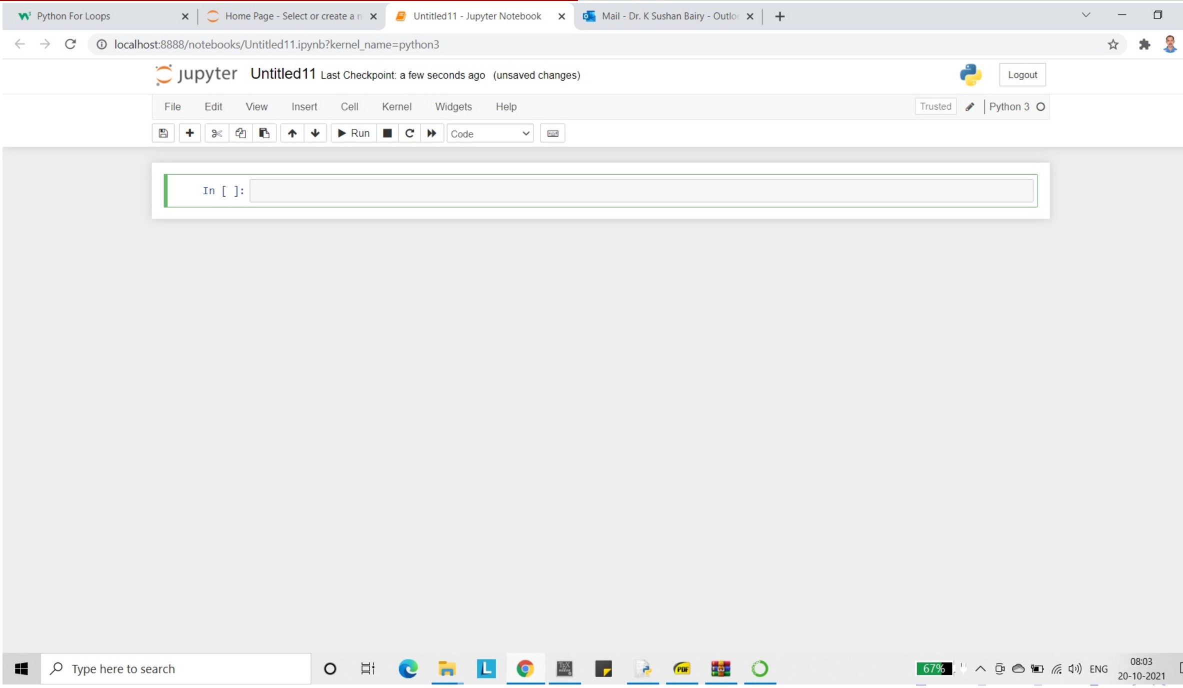
A web application: a browser-based tool for interactive authoring of documents which combine explanatory text, mathematics, computations and their rich media output.

Notebook documents: a representation of all content visible in the web application, including inputs and outputs of the computations, explanatory text, mathematics, images, and rich media representations of objects.



Select items to perform actions on them.

Name	Last Modified	File size
3D Objects	6 months ago	
Contacts	6 months ago	
Desktop	44 minutes ago	
Documents	2 days ago	
Downloads	13 hours ago	
Favorites	6 months ago	
Links	6 months ago	
maxima	a year ago	
Music	6 months ago	
OneDrive	14 hours ago	
Pictures	2 days ago	
Saved Games	6 months ago	
Searches	6 months ago	
Videos	6 months ago	
Untitled.ipynb	2 months ago	23.8 kB
Untitled1.ipynb	2 months ago	39.2 kB
Untitled10.ipynb	2 months ago	4.71 kB
Untitled2.ipynb	2 months ago	5.76 kB
Untitled3.ipynb	2 months ago	7.57 kB


Python For Loops
Home Page - Select or create a n
Untitled11 - Jupyter Notebook
Mail - Dr. K Sushan Bairy - Outloo
localhost:8888/notebooks/Untitled11.ipynb?kernel_name=python3
jupyter
Untitled11 Last Checkpoint: a few seconds ago (unsaved changes)
Logout
File
Edit
View
Insert
Cell
Kernel
Widgets
Help
Trusted
Python 3
Run
Code
In [ ]:
Type here to search
67%
ENG
08:03
20-10-2021

DOWNLOAD AND DOCUMENTATION

Python home - `https://www.python.org/`

Download area - `https://www.python.org/downloads/`

Documentation and Help - `https://www.python.org/doc/`

INSTALLING PYTHON

NOTE:

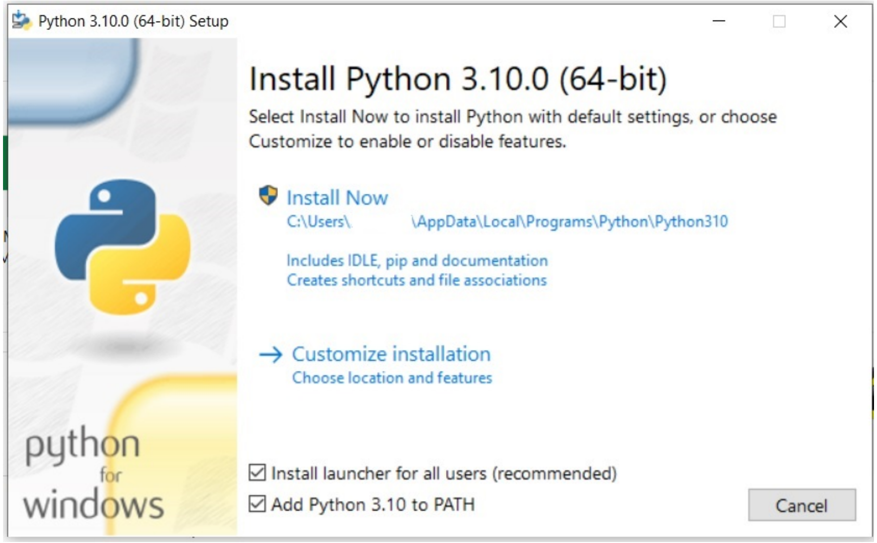
Python 3.10 supports Windows 8.1 and newer. If you require Windows 7 support, please install Python 3.8.

INSTALLING PYTHON

NOTE:

Python 3.10 supports Windows 8.1 and newer. If you require Windows 7 support, please install Python 3.8.

- For full installation: download "Python 3.10" installer available for download.
- The following dialogue box appears.



Python 3.10.0 (64-bit) Setup

Install Python 3.10.0 (64-bit)

Select Install Now to install Python with default settings, or choose Customize to enable or disable features.

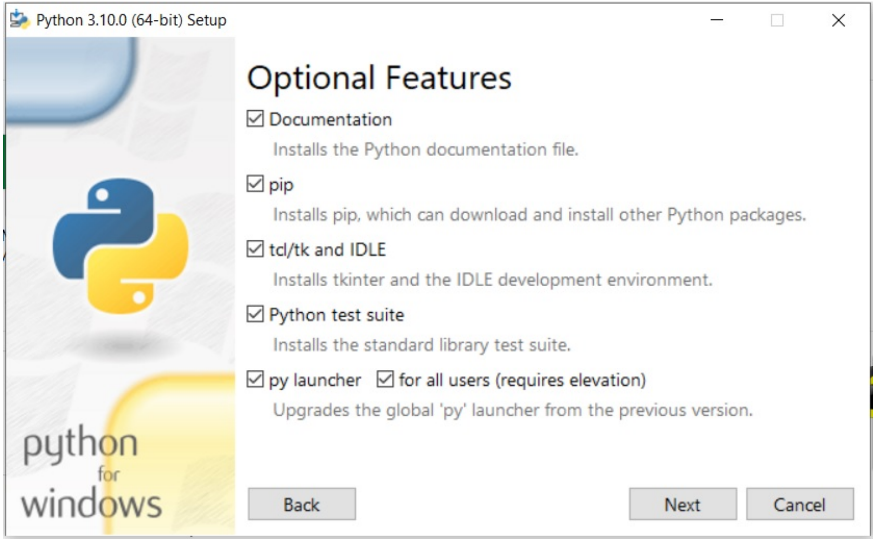
Install Now
C:\Users\ [redacted] \AppData\Local\Programs\Python\Python310

Includes IDLE, pip and documentation
Creates shortcuts and file associations

Customize installation
Choose location and features

☑ Install launcher for all users (recommended)
☑ Add Python 3.10 to PATH

- Select any one of the option and continue.


Python 3.10.0 (64-bit) Setup

Optional Features

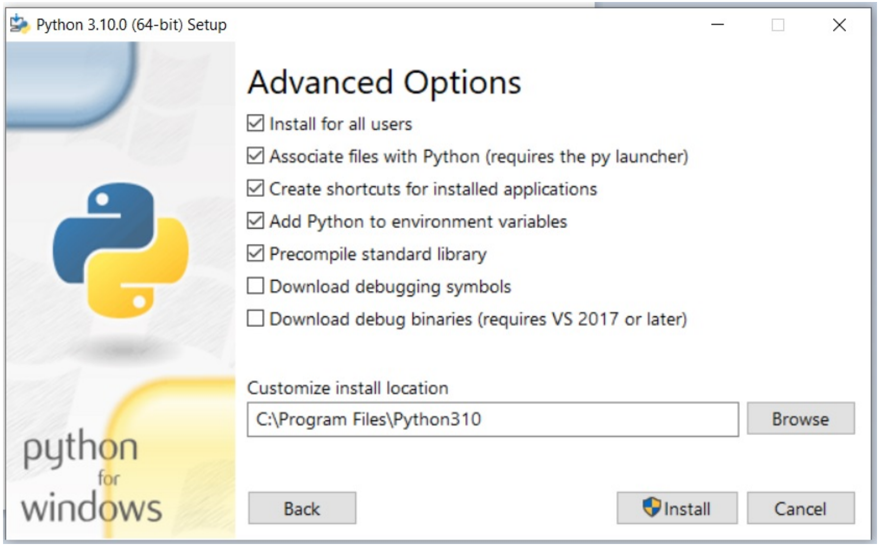
[✓] Documentation
Installs the Python documentation file.

[✓] pip
Installs pip, which can download and install other Python packages.

[✓] tcl/tk and IDLE
Installs tkinter and the IDLE development environment.

[✓] Python test suite
Installs the standard library test suite.

[✓] py launcher [✓] for all users (requires elevation)
Upgrades the global 'py' launcher from the previous version.

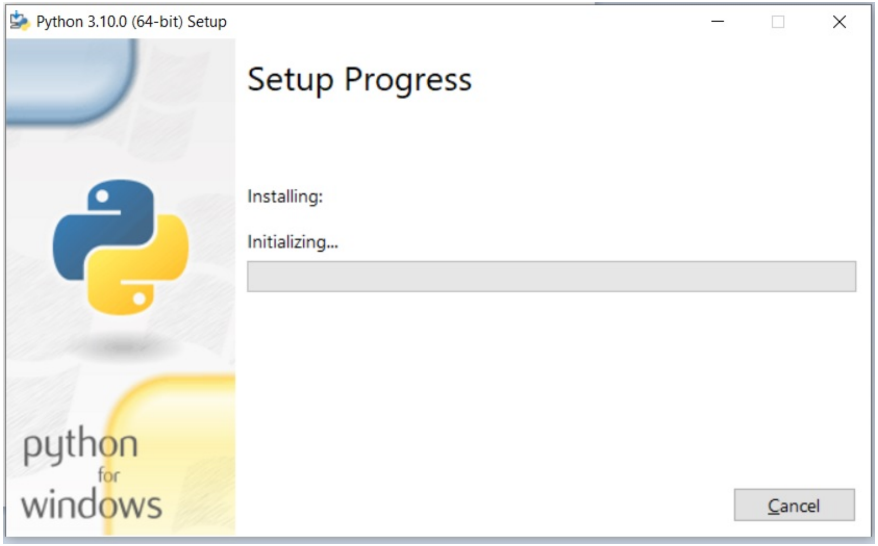

Python 3.10.0 (64-bit) Setup

Advanced Options

- [✓] Install for all users
- [✓] Associate files with Python (requires the py launcher)
- [✓] Create shortcuts for installed applications
- [✓] Add Python to environment variables
- [✓] Precompile standard library
- [ ] Download debugging symbols
- [ ] Download debug binaries (requires VS 2017 or later)

Customize install location

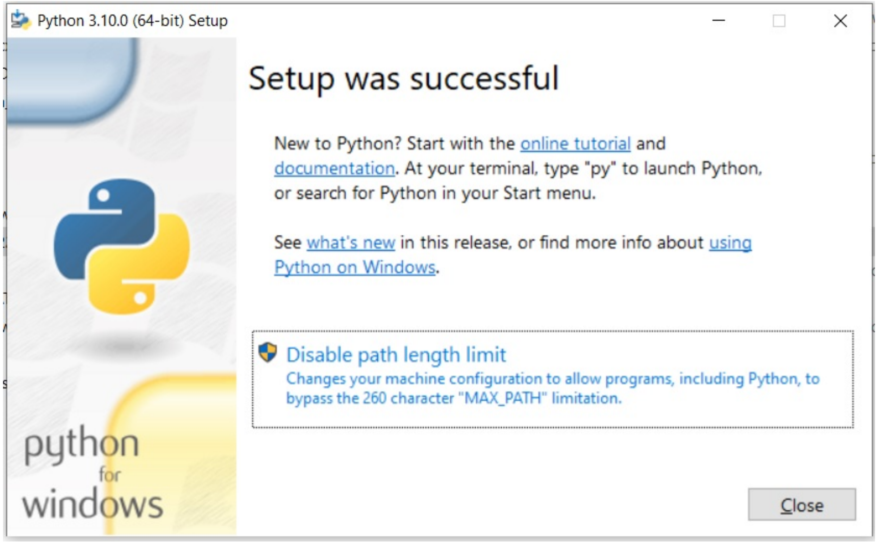
C:\Program Files\Python310


Python 3.10.0 (64-bit) Setup

Setup Progress

Installing:

Initializing...



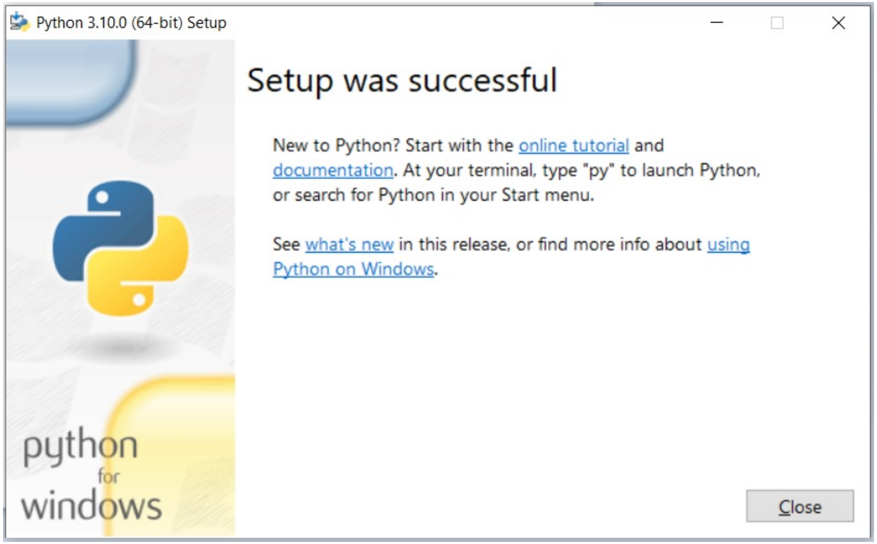
Python 3.10.0 (64-bit) Setup

Setup was successful

New to Python? Start with the online tutorial and documentation. At your terminal, type "py" to launch Python, or search for Python in your Start menu.

See what's new in this release, or find more info about using Python on Windows.

Disable path length limit
Changes your machine configuration to allow programs, including Python, to bypass the 260 character "MAX_PATH" limitation.


Python 3.10.0 (64-bit) Setup

Setup was successful

New to Python? Start with the online tutorial and documentation. At your terminal, type "py" to launch Python, or search for Python in your Start menu.

See what's new in this release, or find more info about using Python on Windows.

- This completes the successful installation of “Python 3.10.0”.

INSTALLING PACKAGES FOR PYTHON

- Open command prompt by searching *cmd*
- pip (package installer for Python) is used to install packages from Python Package Index and other indexes. First we update the ‘pip’ to latest version and then use pip to install the packages.

INSTALLING PACKAGES FOR PYTHON

- Open command prompt by searching *cmd*
- pip (package installer for Python) is used to install packages from Python Package Index and other indexes. First we update the 'pip' to latest version and then use pip to install the packages.
- Type: python -m pip install - -upgrade pip
- pip install numpy
- pip install sympy
- pip install matplotlib
- pip install statistics

SHELL/EDITOR

SHELL/EDITOR

- Open IDLE (Integrated Development and Learning Environment). IDLE shell opens and one can start working

SHELL/EDITOR

SHELL/EDITOR

- Open IDLE (Integrated Development and Learning Environment). IDLE shell opens and one can start working
- Select “New File” from “File” menu.

SHELL/EDITOR

SHELL/EDITOR

- Open IDLE (Integrated Development and Learning Environment). IDLE shell opens and one can start working
- Select “New File” from “File” menu.
  The Editor will open. We type Python program here and then use the interpreter to execute the content from the file.
- Files to be saved with extension .py

Keywords and Identifiers

- **Keywords** are the reserved words in Python used by Python interpreter to recognize the structure of the program.

Keywords and Identifiers

- **Keywords** are the reserved words in Python used by Python interpreter to recognize the structure of the program.
- **Identifiers** are the name given to entities like class, functions, variables etc. It helps to differentiate one entity from another.

KEYWORDS AND IDENTIFIERS

- **Keywords** are the reserved words in Python used by Python interpreter to recognize the structure of the program.
- **Identifiers** are the name given to entities like class, functions, variables etc. It helps to differentiate one entity from another.
- Identifiers can be a combination of letters from “a” to “z”, from “A” to “Z” and digits from “0” to “9” or special character _,.

Keywords and Identifiers

- **Keywords** are the reserved words in Python used by Python interpreter to recognize the structure of the program.
- **Identifiers** are the name given to entities like class, functions, variables etc. It helps to differentiate one entity from another.
- Identifiers can be a combination of letters from “a” to “z”, from “A” to “Z” and digits from “0” to “9” or special character _,.
- Keywords cannot be used as identifiers.

Keywords and Identifiers

- **Keywords** are the reserved words in Python used by Python interpreter to recognize the structure of the program.
- **Identifiers** are the name given to entities like class, functions, variables etc. It helps to differentiate one entity from another.
- Identifiers can be a combination of letters from “a” to “z”, from “A” to “Z” and digits from “0” to “9” or special character _,.
- Keywords cannot be used as identifiers.
- Only special character used in identifiers is _.

Keywords and Identifiers

- **Keywords** are the reserved words in Python used by Python interpreter to recognize the structure of the program.
- **Identifiers** are the name given to entities like class, functions, variables etc. It helps to differentiate one entity from another.
- Identifiers can be a combination of letters from “a” to “z”, from “A” to “Z” and digits from “0” to “9” or special character _,.
- Keywords cannot be used as identifiers.
- Only special character used in identifiers is _.
- Identifier can be of any length.

Most important Python keywords

False	True	and	or
not	break	continue	class
def	if	elif	else
for	while	in	is
None	lambda	return	

Data types

- Variables can store data of different types, and different types can do different things.

DATA TYPES

- Variables can store data of different types, and different types can do different things.
- Python has the following data types built-in by default, in these categories:

Data types

- Variables can store data of different types, and different types can do different things.
- Python has the following data types built-in by default, in these categories:

Text Type	str
Numeric Types	int, float, complex
Sequence Types	list, tuple, range
Mapping Type	dict
Set Types	set, frozenset
Boolean Type	bool
Binary Types	bytes, bytearray, memoryview

Examples:

```
a=2.4
x=3
y='a'
z="Hello"
print(type(a))  #type() gives the data type of the variable.
print(type(x))
print(type(y))
print(type(z))
```

Example for Numeric types:

```
x = 1     # int
y = 2.8   # float
z = 1j    # complex
X = 35e3 # e indicates the power of 10
Y = 12E4
Z = -87.7e100
```

Example for Text type:

Strings in python are surrounded by either single quotation marks, or double quotation marks.

Example for Text type:

Strings in python are surrounded by either single quotation marks, or double quotation marks.

```
a = "Hello"
print(a)

b = """Hello all, good morning,
welcome to the introduction to Python,
will discuss very briefly how to use
Python editor using IDLE; Jupyter Notebook (Anaconda)."""
print(b)
```

Operators

Python divides the operators in the following groups:

- Arithmetic operators
- Assignment operators
- Comparison operators
- Logical operators
- Identity operators
- Membership operators
- Bitwise operators

ARITHMETIC OPERATORS

All the common algebraic operators presented in the following table are available in Python.

Addition	+
Subtraction	−
Multiplication	*
Division	/
Integer Division	//
Power	**
Remainder	%

Comparison Operators

Comparison operators:

$<$	less than
$>$	greater than
$<=$	less than or equal to
$>=$	greater than or equal to
$==$	equal to
$! =$	not equal to

Logical operators

Logical operators:

and	logical and
or	logical or
not	logical not

Assignment Operator

- $=$, is used as assignment operator. For example $x = 5$, means 5 is stored to variable x.
- $+=$. The expression $a+ = 3$, give the result of $a = a + 3$.
- $-=$. The expression $x- = 3$, give the result of $x = x - 3$.
- $*=$. The expression $a* = 3$, give the result of $a = a * 3$.
- $/=$. The expression $a/ = 3$, give the result of $a = a/3$.
- $//=$. The expression $a// = 3$, give the result of $a = a//3$.
- $\% =$. The expression $a\% = 3$, give the result of $a = a\%3$.
- $** =$. The expression $a* = 3$, give the result of $a = a * *3$.

Identity Operators

Identity Operators

is	Returns True if both variables are the same object
is not	Returns True if both variables are not the same object

Membership Operators

Membership Operators

in	Returns True if a sequence with the specified value is present in the object
not in	Returns True if a sequence with the specified value is not present in the object

EXERCISE:

Type the following statements and observe the output:

```
32+56−74*2+65/5−2**3
32+(56−74)*2+65/5−2**3
32+(56−74)*2+(65/5−2)**3
1254//9
487**2%3
int(22/6)
round(22/6)
round(22/6,2)
round(22/6,6)
round(2.49)
round(2.5)
```

Exercise:

Type the following statements and observe the output:

```
round(3.5)
round(3.51)
round(2.52)
x=23145621.456872194
x
round(x)
round(x,1)
round(x,2)
round(x,4)
round(x,6)
```

Exercise:

Type the following statements and observe the output:

```
round(x,-1)
round(x,-2)
round(x,-4)
round(x,-7)
round(x,-8)
x,y,z=23,45,67
x+y,x+y+z,x*y
print(x+y,x+y+z,x*y)
```

EXERCISE:

Type the following statements and observe the output:

```
x
x+=2
x
x-=4
x
x*=2
x
x%=2
x
x//=5
x
```

Lists, Tuples, Sets

- Lists
- Tuples
- Sets

These are used to store collections of data.

Lists, Tuples

- are used to store multiple items in a single variable.

Lists, Tuples

- are used to store multiple items in a single variable.
- Lists are created using square brackets: for example a=[1,2,3]

Lists, Tuples

- are used to store multiple items in a single variable.
- Lists are created using square brackets: for example a=[1,2,3]
- Tuples are created using round brackets: for example a=(1,2,3)

Lists, Tuples

- are used to store multiple items in a single variable.
- Lists are created using square brackets: for example a=[1,2,3]
- Tuples are created using round brackets: for example a=(1,2,3)
- items are ordered, allow duplicate values.

Lists, Tuples

- are used to store multiple items in a single variable.
- Lists are created using square brackets: for example a=[1,2,3]
- Tuples are created using round brackets: for example a=(1,2,3)
- items are ordered, allow duplicate values.
- List items are changeable, whereas; Tuple items are unchangeable.

Lists, Tuples

- are used to store multiple items in a single variable.
- Lists are created using square brackets: for example a=[1,2,3]
- Tuples are created using round brackets: for example a=(1,2,3)
- items are ordered, allow duplicate values.
- List items are changeable, whereas; Tuple items are unchangeable.
- items are indexed, the first item has index [0], the second item has index [1] etc.

Lists (contd...)

- ordered, it means that the items have a defined order, and that order will not change.

Lists (contd...)

- ordered, it means that the items have a defined order, and that order will not change.
- If you add new items to a list, the new items will be placed at the end of the list.

LISTS (CONTD...)

- ordered, it means that the items have a defined order, and that order will not change.
- If you add new items to a list, the new items will be placed at the end of the list.
- The list is changeable, meaning that we can change, add, and remove items in a list after it has been created.

Lists (contd...)

- ordered, it means that the items have a defined order, and that order will not change.
- If you add new items to a list, the new items will be placed at the end of the list.
- The list is changeable, meaning that we can change, add, and remove items in a list after it has been created.
- Since lists/tuples are indexed, lists can have items with the same value.

LISTS (CONTD...)

- ordered, it means that the items have a defined order, and that order will not change.
- If you add new items to a list, the new items will be placed at the end of the list.
- The list is changeable, meaning that we can change, add, and remove items in a list after it has been created.
- Since lists/tuples are indexed, lists can have items with the same value.
- List/tuple items can be of any data type.

Lists (contd...)

- ordered, it means that the items have a defined order, and that order will not change.
- If you add new items to a list, the new items will be placed at the end of the list.
- The list is changeable, meaning that we can change, add, and remove items in a list after it has been created.
- Since lists/tuples are indexed, lists can have items with the same value.
- List/tuple items can be of any data type.
- A list/tuple can contain different data types

Lists (contd...)

- len() function is used to find the number of items in the list/tuple

Lists (contd...)

- len() function is used to find the number of items in the list/tuple
- list(())/tuple(()) constructor can be used to create a list/tuple

Lists (contd...)

- len() function is used to find the number of items in the list/tuple
- list(())/tuple(()) constructor can be used to create a list/tuple
- List/tuple items are indexed and can be accessed them by referring to the index number. The first item has index 0.

Lists (contd...)

- len() function is used to find the number of items in the list/tuple
- list(())/tuple(()) constructor can be used to create a list/tuple
- List/tuple items are indexed and can be accessed them by referring to the index number. The first item has index 0.
- Negative indexing means start from the end
  -1 refers to the last item, -2 refers to the second last item etc.

- insert() is used to insert an item to the list

- insert() is used to insert an item to the list
- append() is used to add an item to the end of the list

- insert() is used to insert an item to the list
- append() is used to add an item to the end of the list
- extend() is used to append elements from another list to the current list

- insert() is used to insert an item to the list
- append() is used to add an item to the end of the list
- extend() is used to append elements from another list to the current list
- remove() used to remove specified item.

- insert() is used to insert an item to the list
- append() is used to add an item to the end of the list
- extend() is used to append elements from another list to the current list
- remove() used to remove specified item.
- pop() used to remove specified indes.

- insert() is used to insert an item to the list
- append() is used to add an item to the end of the list
- extend() is used to append elements from another list to the current list
- remove() used to remove specified item.
- pop() used to remove specified indes.
- del / clear() is used to delete the complete list.

- insert() is used to insert an item to the list
- append() is used to add an item to the end of the list
- extend() is used to append elements from another list to the current list
- remove() used to remove specified item.
- pop() used to remove specified indes.
- del / clear() is used to delete the complete list.
- sort() is used to sort the items in the list.

- insert() is used to insert an item to the list
- append() is used to add an item to the end of the list
- extend() is used to append elements from another list to the current list
- remove() used to remove specified item.
- pop() used to remove specified indes.
- del / clear() is used to delete the complete list.
- sort() is used to sort the items in the list.

These functions can be used for a list. Tuple items cannot be changed using any of the above. First, convert tuple to a list, apply changes and then again convert list to tuple.

Sets

- Sets are used to store multiple items in a single variable.

Sets

- Sets are used to store multiple items in a single variable.
- A set is a collection which is unordered, unchangeable, and unindexed.

SETS

- Sets are used to store multiple items in a single variable.
- A set is a collection which is unordered, unchangeable, and unindexed.
- Sets are written with curly brackets.

Sets

- Sets are used to store multiple items in a single variable.
- A set is a collection which is unordered, unchangeable, and unindexed.
- Sets are written with curly brackets.
- Set items do not allow duplicate values.

Sets

- Sets are used to store multiple items in a single variable.
- A set is a collection which is unordered, unchangeable, and unindexed.
- Sets are written with curly brackets.
- Set items do not allow duplicate values.
- Once a set is created, one cannot change its items, but can add new items.

SETS

- Sets are used to store multiple items in a single variable.
- A set is a collection which is unordered, unchangeable, and unindexed.
- Sets are written with curly brackets.
- Set items do not allow duplicate values.
- Once a set is created, one cannot change its items, but can add new items.
- A set can contain different data types

Sets

- Sets are used to store multiple items in a single variable.
- A set is a collection which is unordered, unchangeable, and unindexed.
- Sets are written with curly brackets.
- Set items do not allow duplicate values.
- Once a set is created, one cannot change its items, but can add new items.
- A set can contain different data types
- set() constructor can be used to make a set.

EXERCISE:

Type the following statements and observe the output:

```
a=[23,46,56,21,13]
a
len(a)
a[0],a[1],a[5]
a[6]
a[-1]
a[-4]
a.append(23)
a
b=[78,58,74]
```

EXERCISE:

Type the following statements and observe the output:

```
a.append(b)
a
a.append(b[2])
a
a.insert(4,34)
a
a.insert(0,87)
a
a.pop()
a
```

EXERCISE:

Type the following statements and observe the output:

```
a.pop(2)
a
a.pop(-1)
a
a.remove(34)
a
a.remove(a[2])
a
a=[12,36,9,23,8,5,3]
a
a.sort()
a
```

EXERCISE:

Type the following statements and observe the output:

```
b=(12,36,9,23,8,5,3)
b
len(b)
c=list(b)
c
c.sort()
c
c.sort(reverse=True)
c
b=tuple(c)
c.clear()
```

EXERCISE:

Type the following statements and observe the output:

```
a={3,12,36,9,23,8,5,3}
a
b=set([12,16,19,25,35,36,39])
b
c=a.union(b)
c
d=a.intersection(b)
d
e=a.symmetric_difference(b)
e
e.clear()
e
```

EXERCISE:

Type the following statements and observe the output:

```
a={3,12,36,9,23,8,5,3}
c=a.copy()
b={3,56,23}
d=a.difference(b)
c=a.union(b)
a.isdisjoint(b)
a.issubset(b)
a.remove(3)
a
```

INPUT STATEMENT

The *input()* function allows user input.

SYNTAX:

input(*prompt*)

prompt– A String, representing a default message before the input.

For example:

```
x = input('Enter your name:')
print('Hello, ' +x)
print('Hello, ', x)
print(type(x))
```

INPUT STATEMENT

- To read integer then the following to be used.

```
x = int(input('Enter an integer:'))
print(type(x))
```

INPUT STATEMENT

- To read integer then the following to be used.

  ```
  x = int(input('Enter an integer:'))
  print(type(x))
  ```

- To read float then the following to be used.

  ```
  x = float(input('Enter an float:'))
  print(type(x))
  ```

OUTPUT STATEMENTS

Python provides the *print()* function to display output to the standard output devices.

SYNTAX:

print(value(s),sep= ' ', end = '\n', file=file, flush=flush)

value(s) – Any value, and as many as you like. Will be converted to string before printed
sep='separator' – (Optional) Specify how to separate the objects, if there is more than one. Default :' '
end='end' – (Optional) Specify what to print at the end.Default : '\n'
file – (Optional) An object with a write method. Default :sys.stdout
flush – (Optional) A Boolean, specifying if the output is flushed (True) or buffered (False). Default: False

Examples for print()

Type the following commands and note the difference in output:

Codes:

```
print("REVA")
print('R', 'E', 'V', 'A')
print("REVA", end = "@")
print('R', 'E', 'V', 'A', sep="#")
```

Formatting output

Code 1:

```
name = "REVA"
print(f'Welcome to {name}!')
```

CODE 2:

```
a = 20
b = 10
# addition
sum = a + b
# subtraction
sub = a- b
# Output
print('The value of a is {} and b is {}'.format(a,b))
print('{2} is the sum of {0} and {1}'.format(a,b,sum))
print('{sub_value} is the subtraction of {value_a} and {value_b}'
      .format(value_a = a, value_b = b,sub_value =sub))
```

Using % Operator

We can use ‘%’ operator. % values are replaced with zero or more value of elements. The formatting using % is similar to that of ‘printf’ in the C programming language.

- %d – integer
- %f – float
- %s – string
- %x – hexadecimal
- %o – octal

Example

Code

```
# Taking input from the user
num = int(input("Enter a value: "))
add = num + 5
# Output
print("The sum is %d" %add)
```

Example

Code

```
# Taking input from the user
num = int(input("Enter a value: "))
add = num + 5
# Output
print("The sum is %d" %add)
```

Code:

```
x = 12.3456789
print('The value of x is %3.2f' %x)
print('The value of x is %3.4f' %x)
```

Output statements

display() command can be used to get the output printed on the screen. For example

```
x = 12.3456789
display('The value of x is %3.2f' %x)
display('The value of x is ',x)
```

OUTPUT STATEMENTS

display() command can be used to get the output printed on the screen. For example

```
x = 12.3456789
display('The value of x is %3.2f' %x)
display('The value of x is ',x)
```

- *display()* works similar to *print()*.

EXERCISES:

- Read a list of elements using input function
- Read a set from user input
- Print the type of variable which were read in previous two commands

Thank You