Python Programming

... a gentle introduction

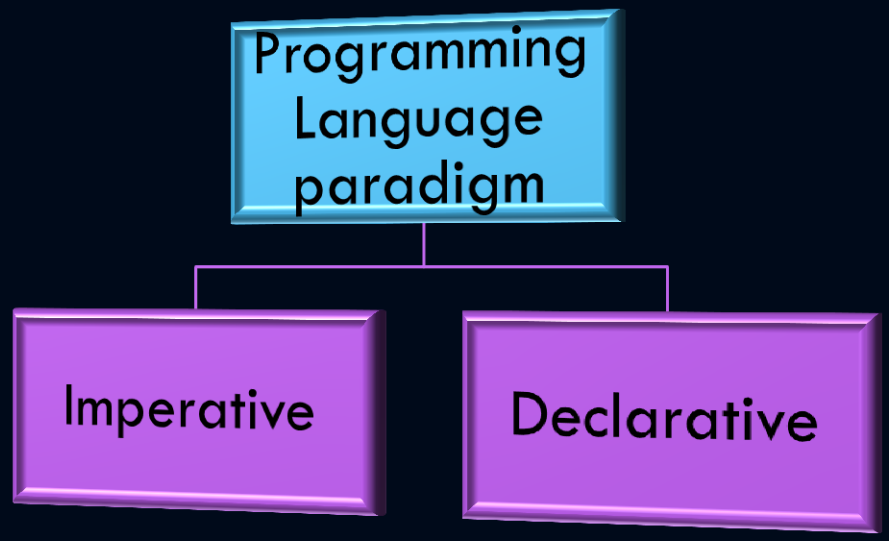
Programming Paradigms

Object-Oriented	Imperative
Functional	Logic

Programming Paradigms

- Programming paradigm is a way to **classify** programming languages based on the **features** of various programming languages

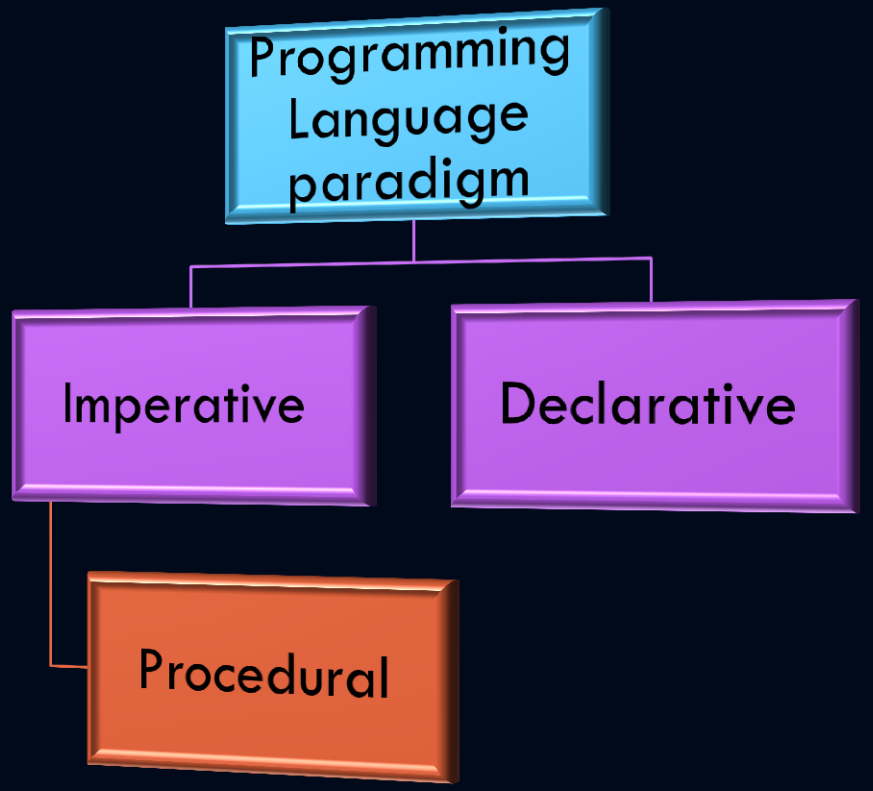
- Programming paradigm is broadly categorized into **Imperative** and **declarative**.



Programming Paradigms - Imperative

- **Procedural** programming uses a list of instructions to tell the computer what to do step-by-step. E.g. C

  Procedural = top-down languages



Programming Paradigms - Imperative

Key features of Procedural Programming

Modularity

Pre-defined functions

Procedures

Parameter Passing

Local variables

Programming libraries

Global Variables

Programming Paradigms - Imperative

Key features of Procedural Programming

Modularity

Modularity is a software technique that shows that separating the functionality into individual, interchangeable modules, each which allows it to execute the specific thing it is designed to do. These all combine as different tasks to achieve an overall goal.

Programming Paradigms - Imperative

Key features of Procedural Programming

Local variables

*Local variables are a variable that can only be **accessed within** the specific chunk/block of code that it was written in, not through the entire script of code (Like a global variable) a local variable is declared to override the same variable name in the larger scope.*

Programming Paradigms - Imperative

Key features of Procedural Programming

Pre-defined functions

Examples of pre-defined function such as print() can be used as a function that is already within a programming language, this grants easy work for programmers.

Programming Paradigms - Imperative

Key features of Procedural Programming

Global Variables

A global variable is a variable that can be viewed throughout the entire program by every other procedure taking place, it is also accessible by every other task running in the program.

Programming Paradigms - Imperative

Key features of Procedural Programming

Procedures

A procedural program follows the procedures step by step, systematically. The program does exactly what it is told to do in the order that has been set by the programmer.

Programming Paradigms - Imperative

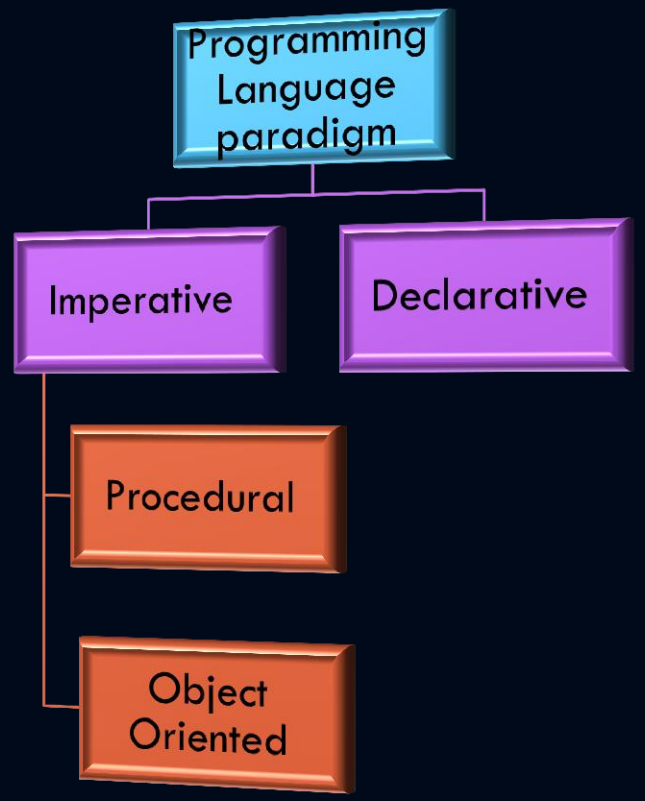
Key features of Procedural Programming

Parameter Passing

Parameter passing allows variable values to be passed through to the program which will handle it with a procedure.

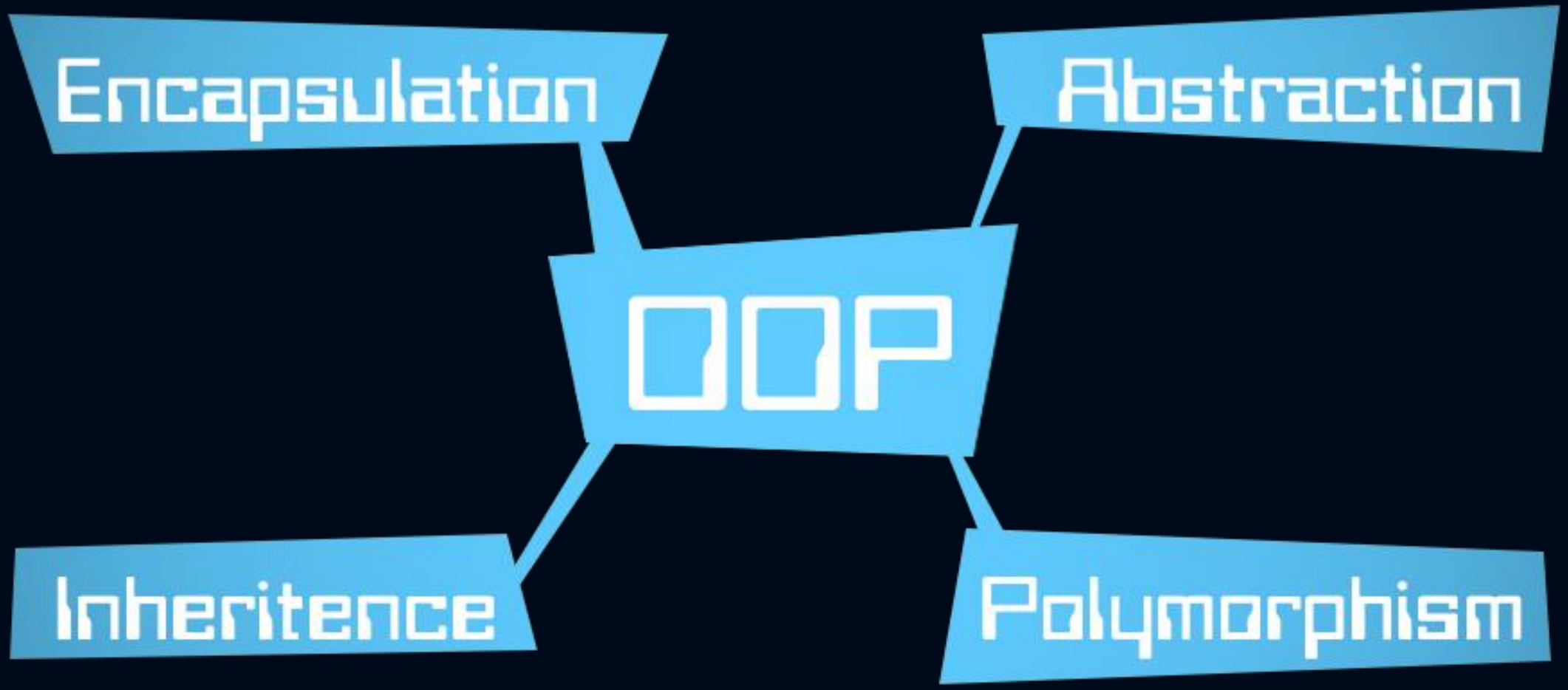
Programming Paradigms - Imperative

- **Object-oriented** programming (OOP) is a programming paradigm based on the concept of "objects".
  E.g. C++, Java, C#, Ruby etc.
  Object = Data + Method



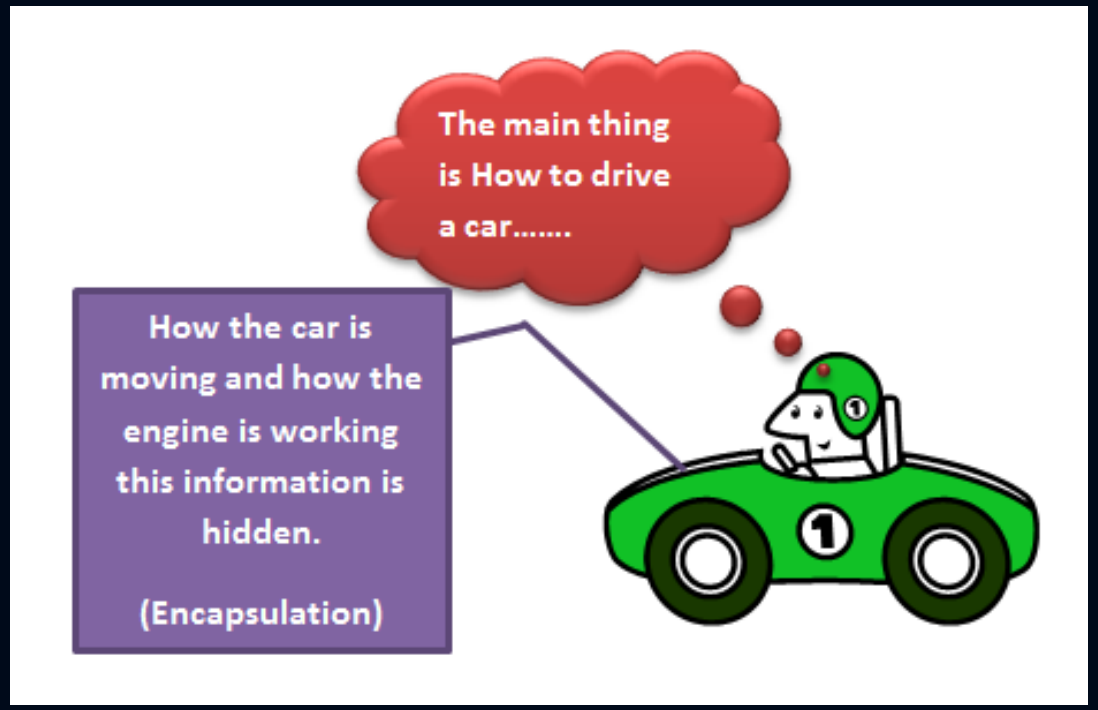
Programming Paradigms - Imperative

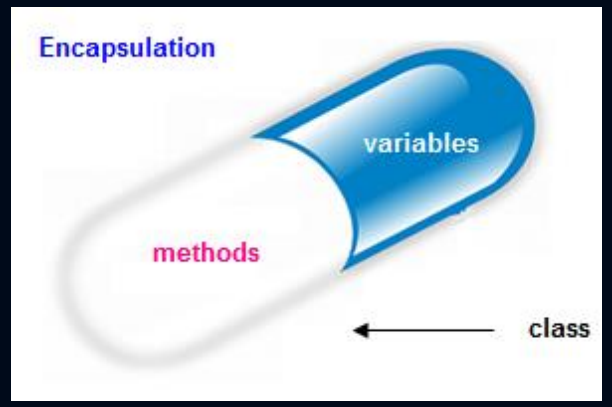
Key features of Object Oriented Programming



Programming Paradigms - Imperative

- Encapsulation is the mechanism that binds together code and the data it manipulates, and keeps both safe from outside interference and misuse.



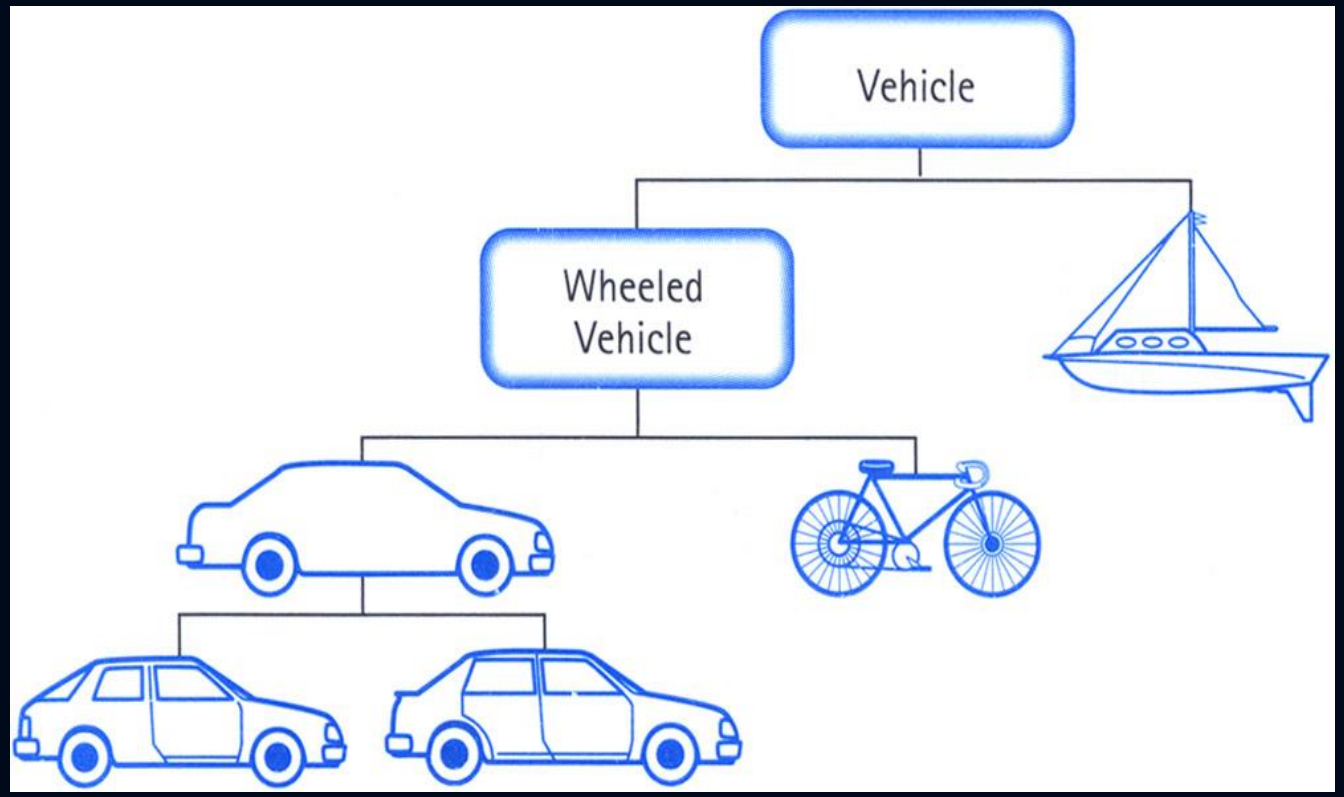


Programming Paradigms - Imperative

- Abstraction is a general concept formed by extracting common features from specific examples or the act of removing something unnecessary.

Programming Paradigms - Imperative

- Inheritance enables new objects to take on the properties of existing objects. A class that is used as the basis for inheritance is called a superclass or base class.

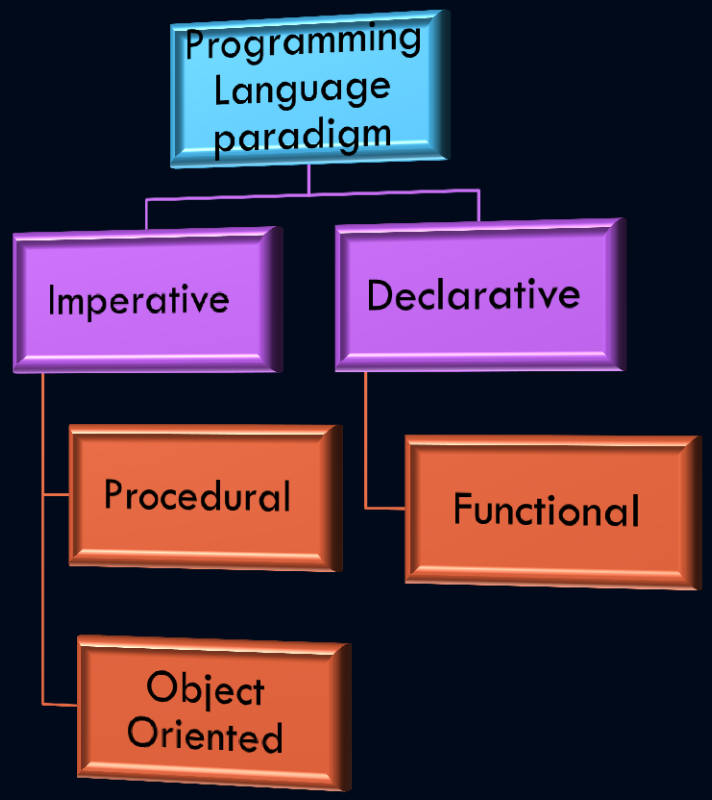


Programming Paradigms - Imperative

- Polymorphism refers to a programming language's ability to process objects differently depending on their data type or class. More specifically, it is the ability to redefine methods for derived classes.

Programming Paradigms - Declarative

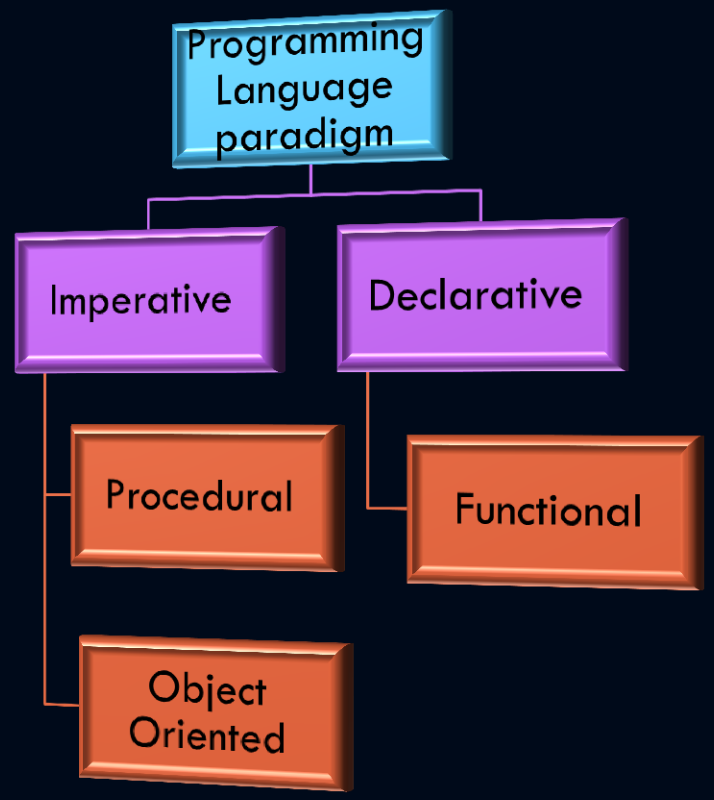
- **Functional** programming is when functions, not objects or procedures, are used as the fundamental building blocks of a program. E.g. Haskell Language



Programming Paradigms - Declarative

Key features of Functional Programming

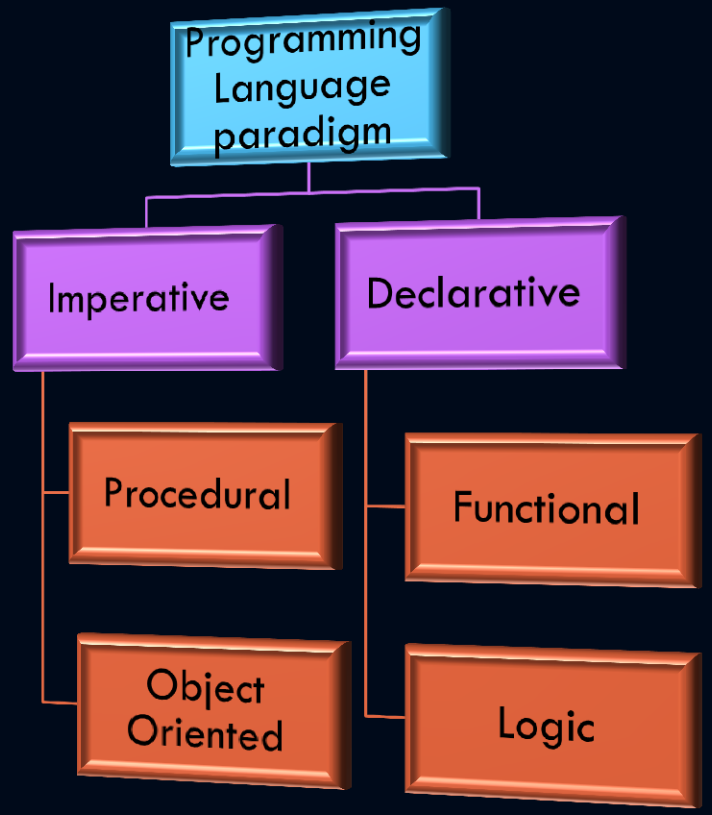
- All programs and procedures are functions
- There are no variables or assignments - only input parameters
- There are no loops - only recursive functions
- The value of a function depends only on the value of its parameters



Programming Paradigms - Declarative

- **Logic** programming (OOP) is a set of sentences in logical form, expressing facts and rules about some problem domain.
  E.g. Prolog, Parlog, Polka, Mercury.

If A and B are true, then C is true

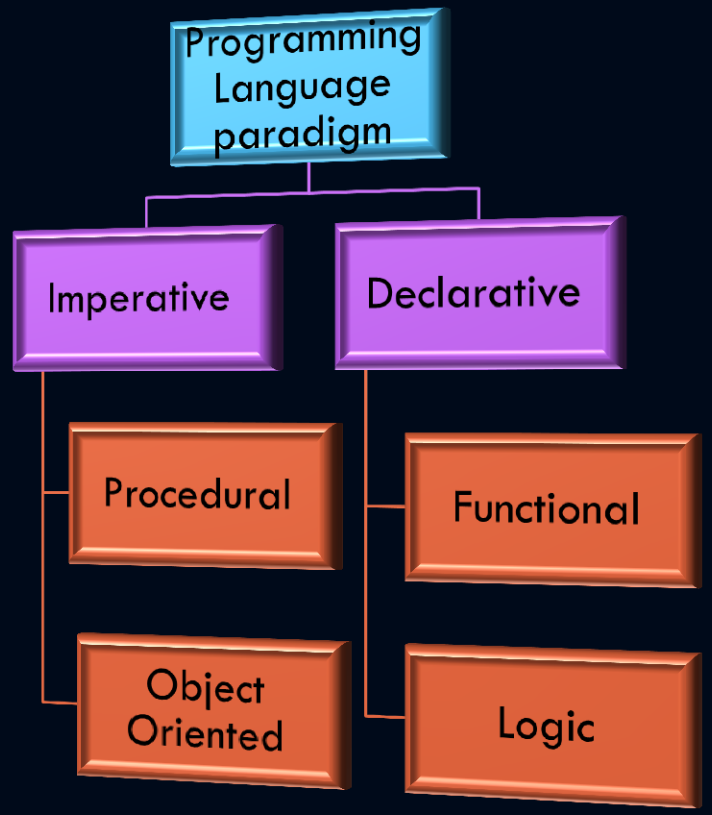


Programming Paradigms - Declarative

Key features of Logic Programming

- Programmer is responsible for specifying the basic logical relationships and does not specify the manner in which the inference rules are applied

Logic + Control = Algorithms



Programming Paradigms - Declarative

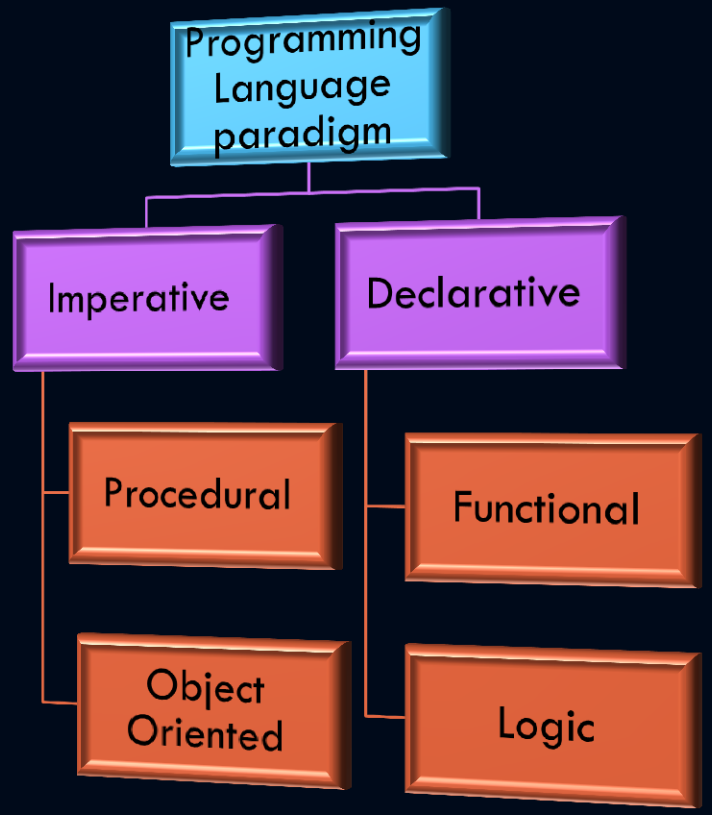
Key features of Logic Programming

- Logic programming is based on tuples.

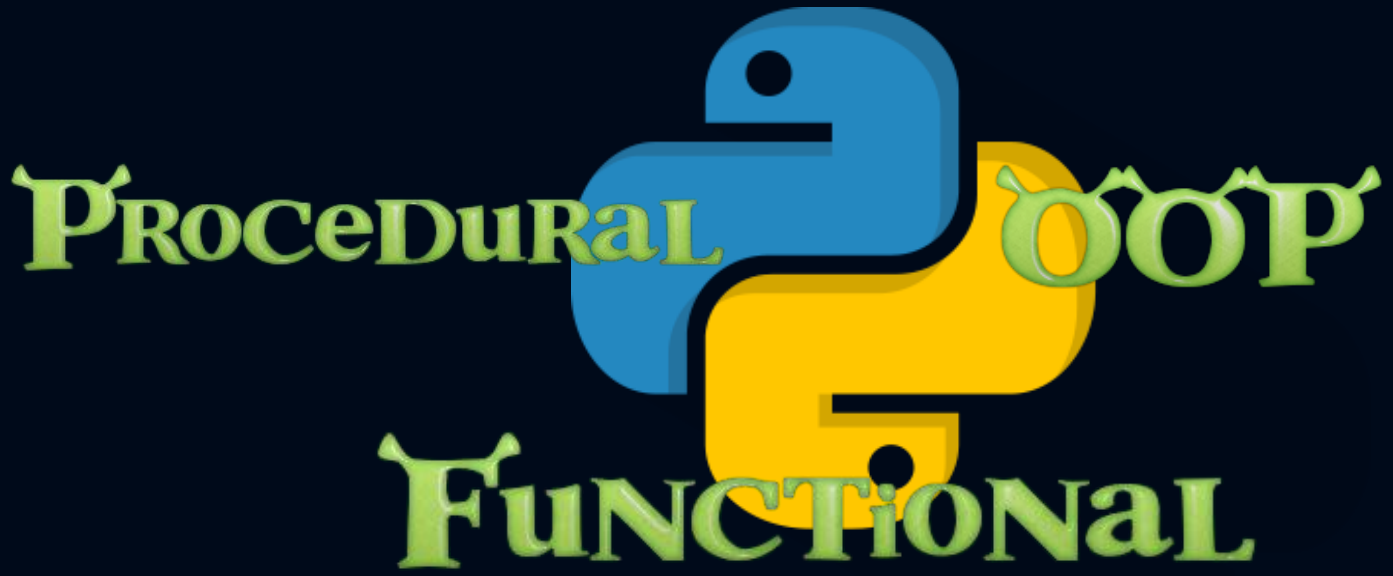
The squaring function for natural numbers may be written as a set of tuples as follows:

{(0,0), (1,1), (2,4) ...}

Such a set of tuples is called a relation and in this case the tuples define the squaring relation.



Introduction to Python Programming



Introduction to Python Programming

- Python 1.0 was released in January 1994. The major new features included in this release were the functional programming tools lambda, map, filter and reduce.
- Python 2.0 was released in October 16, 2000. It introduced the following:
  - ✓ Unicode objects (other than alphanumeric characters)
  - ✓ List comprehensions (differentiates between local & global list)
  - ✓ Augmented assignment (+=)
  - ✓ Cyclic garbage collection (Deadlock situation)
  - ✓ *args and **kwargs argument unpacking (unspecified param)

Introduction to Python Programming

- Python 3.0 (also called "Python 3000" or "Py3K") was released on December 3, 2008. It was designed to rectify certain fundamental design flaws in the language. The guiding principle of Python 3 was: “reduce feature duplication by removing old ways of doing things”.

Important differences between Python 2.x and Python 3.x

	Python 2.x	Python 3.x
Division Operator	print 7 / 5 Output: 1	print (7 / 5) Output: 1.4
print function	print 'Hello, Geeks‘ print('Hope You like these ')	print('Hope You like these facts')
Unicode	print(type('default string ')) print(type(u'string with b ')) Output: <type 'str'> <type 'unicode'>	print(type('default string ')) print(type(u'string with b ')) Output: <type 'str'> <type ‘str’>

Important differences between Python 2.x and Python 3.x

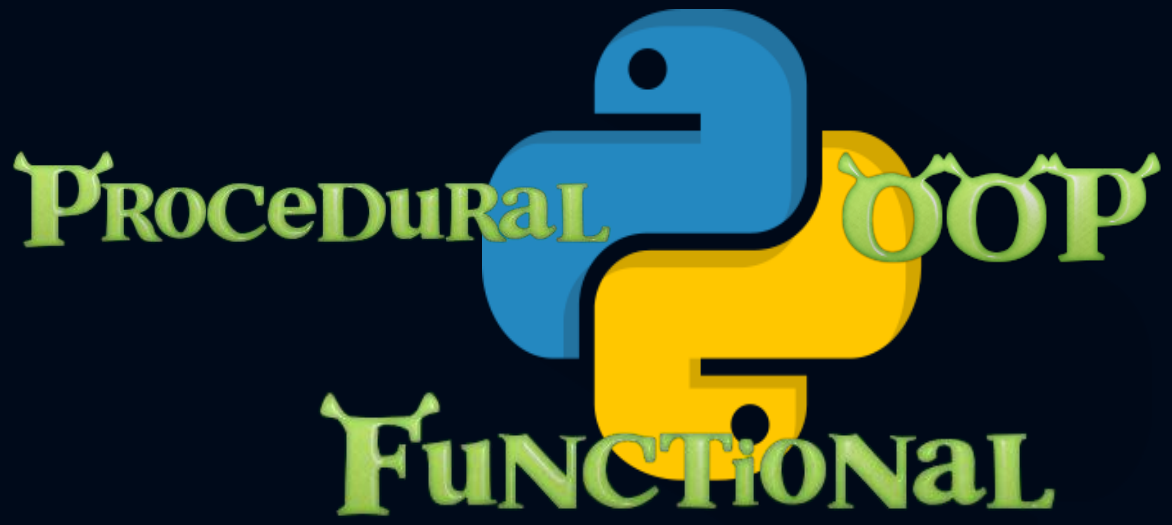
	Python 2.x	Python 3.x

xrange	for x in xrange(1, 5):
	  print(x)
	for x in range(1, 5):
	  print(x)
		for x in range(1, 5):
		  print(x)

Error Handling	try:
	  trying_to_check_error
	except NameError, err:
	  print err, 'Error Caused'
		try:
		  trying_to_check_error
		except NameError as err:
		# 'as' is needed in Python 3.x
		  print (err, 'Error Caused')

Multi-Paradigm (Python)

Multi-paradigm language is a programming language that supports more than one programming paradigm.

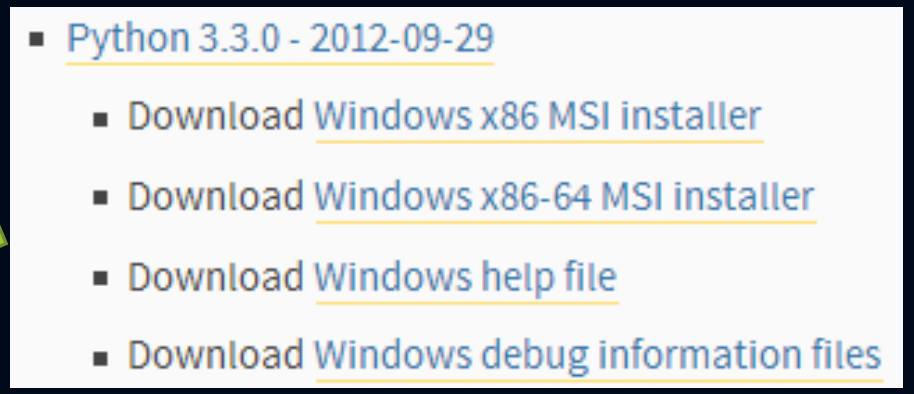


Python is a multi-paradigm programming language. We shall use python 3.3 for the purpose of this training.

Python Installation

Python 3.3 Installation Procedure

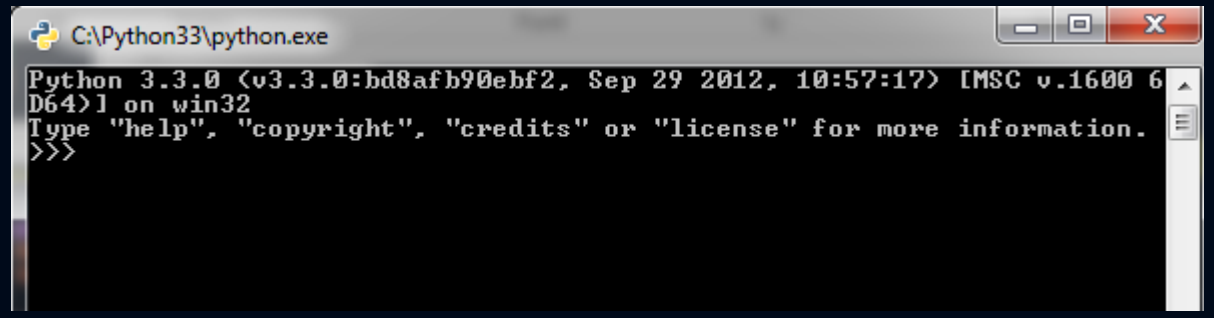
1. Go to www.python.org and select ‘Windows’ from 'Download' menu choice

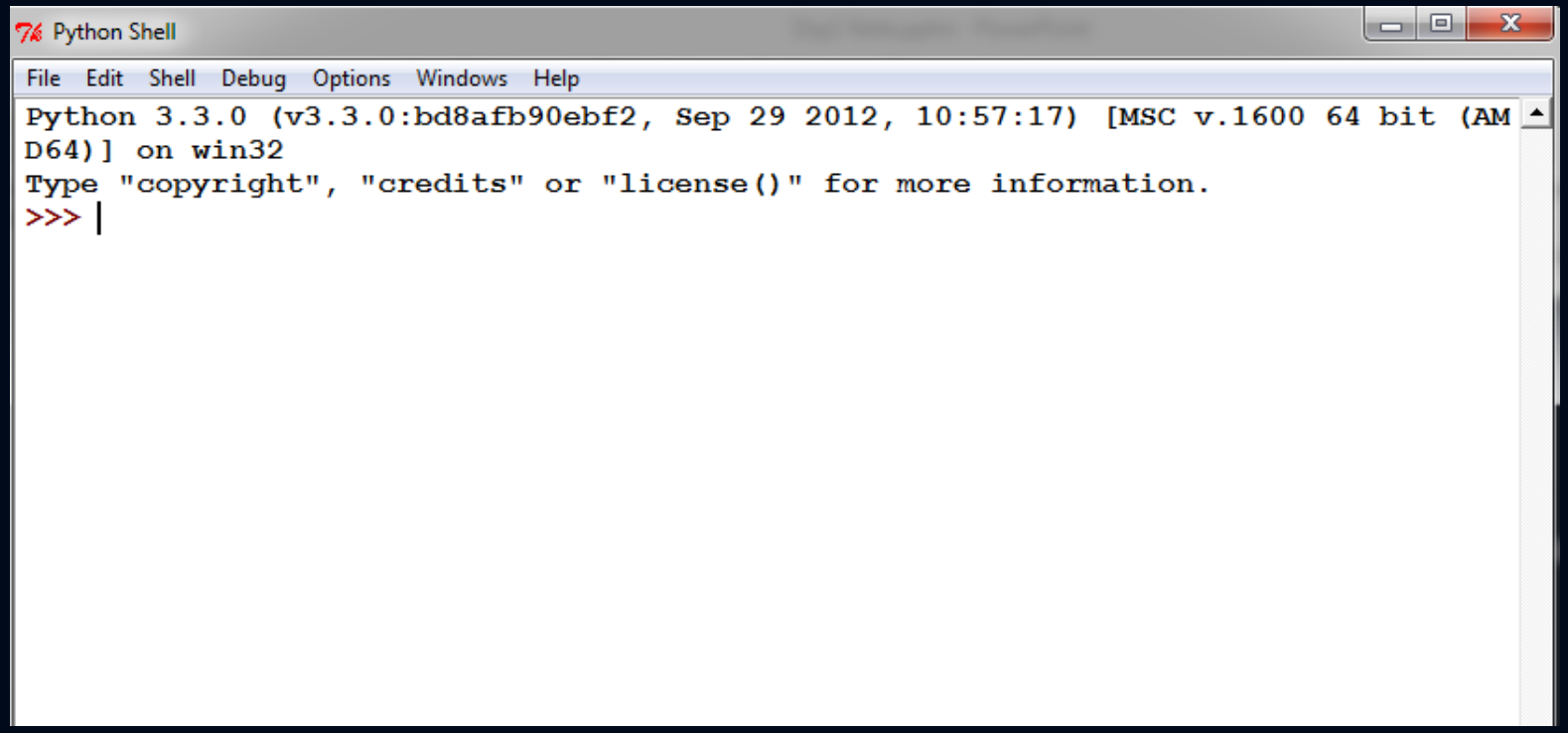


2. Click on “Windows x86-64 MSI installer” to download python interpreter setup file

Python Installation

After Python installation we have access to both python shell and idle (GUI)



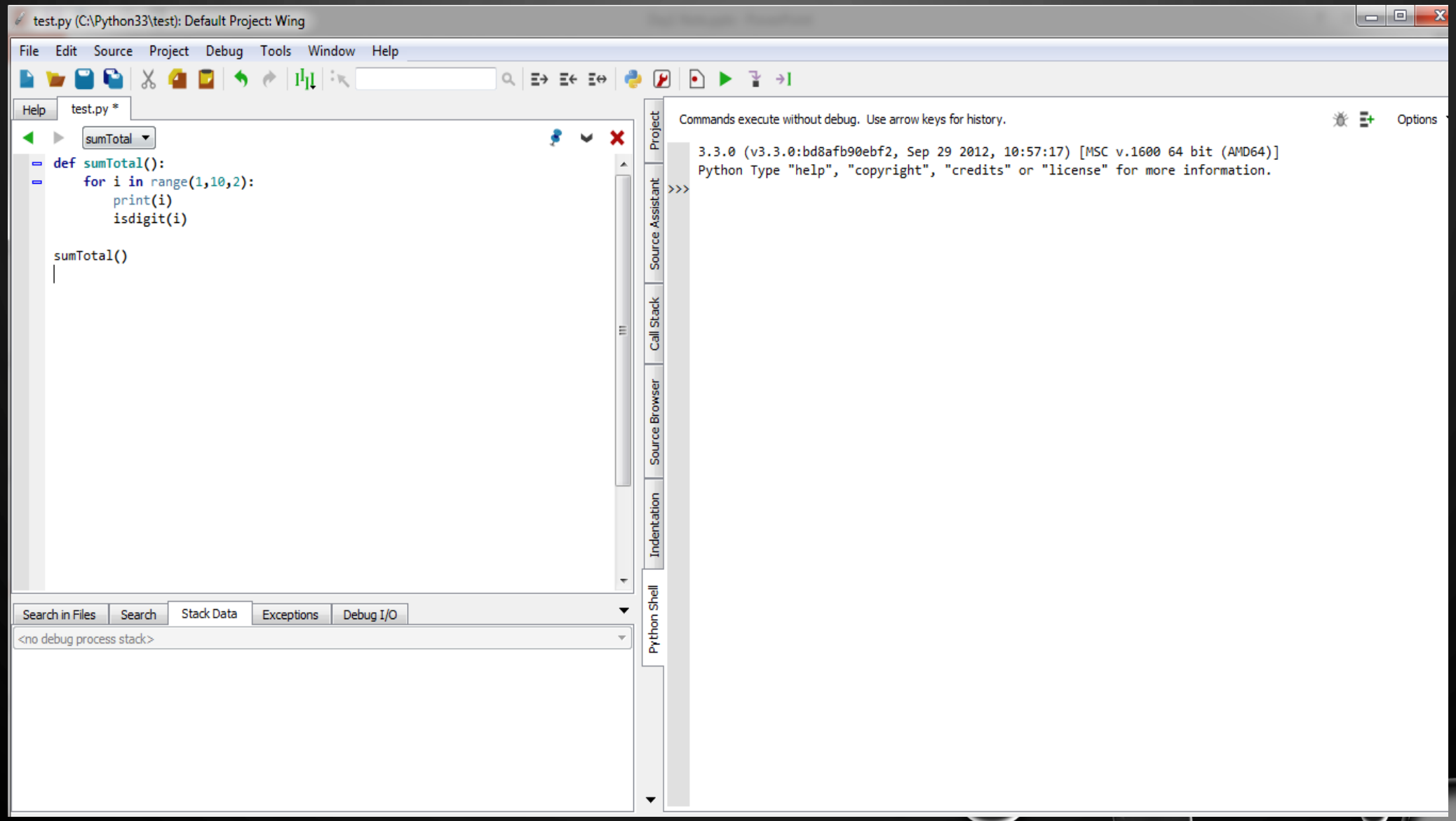


Code Editor for Python

There are lots of editors that can be used for writing and editing python codes. The choice of which to use largely boils down to personal preference. Most Python programmers use complex but extremely powerful IDEs (Integrated Development Environments), such as PyCharm. However, we shall use a powerful yet simple editor which is WingIDE Personal Edition. Visit the site below to download the editor.

https://wingware.com/downloads/wing-personal

WingIDE Development Studio



Exercises

1. Write a Python program to print the following *Sample string:*
   - "Hello World"
   - "Hello World Python"

   (Use Python shell)

Exercises

2. Write a Python program to get the Python version you are using. (Use Python shell)

Exercises

3. Write a Python program to display the current date and time. (Use WingIDE)

Exercises

4. Write a Python program to find the operating system name, platform and platform release date. (Use WingIDE)

Next Lecture ...

Python  Programming

Day 2: Data Types, Identifiers and Operators