Introduction To Python

**Python** is a widely used [general-purpose](http://en.wikipedia.org/wiki/General-purpose_programming_language), [high-level programming language](http://en.wikipedia.org/wiki/High-level_programming_language). Its design philosophy emphasizes code [readability](http://en.wikipedia.org/wiki/Readability), and its syntax allows programmers to express concepts in fewer [lines of code](http://en.wikipedia.org/wiki/Lines_of_code) than would be possible in languages such as [C](http://en.wikipedia.org/wiki/C_(programming_language)). The language provides constructs intended to enable clear programs on both a small and large scale.

Python supports multiple [programming paradigms](http://en.wikipedia.org/wiki/Programming_paradigm), including [object-oriented](http://en.wikipedia.org/wiki/Object-oriented_programming), [imperative](http://en.wikipedia.org/wiki/Imperative_programming) and [functional programming](http://en.wikipedia.org/wiki/Functional_programming) or [procedural](http://en.wikipedia.org/wiki/Procedural_programming) styles. It features a [dynamic type](http://en.wikipedia.org/wiki/Dynamic_type) system and automatic [memory management](http://en.wikipedia.org/wiki/Memory_management) and has a large and comprehensive [standard library](http://en.wikipedia.org/wiki/Standard_library) .

Python was conceived in the late 1980s and its implementation was started in December 1989[ by [Guido van Rossum](http://en.wikipedia.org/wiki/Guido_van_Rossum) at [CWI](http://en.wikipedia.org/wiki/Centrum_Wiskunde_%26_Informatica) in the[Netherlands](http://en.wikipedia.org/wiki/Netherlands) as a successor to the [ABC language](http://en.wikipedia.org/wiki/ABC_(programming_language)) (itself inspired by [SETL](http://en.wikipedia.org/wiki/SETL)) capable of [exception handling](http://en.wikipedia.org/wiki/Exception_handling) and interfacing with the [Amoeba operating system](http://en.wikipedia.org/wiki/Amoeba_(operating_system)).

About Python

Python is a remarkably powerful dynamic programming language that is used in a wide variety of [application domains](http://www.python.org/about/apps). Python is often compared to Tcl, Perl, Ruby, Scheme or Java. Some of its key distinguishing features include:

* very clear, readable syntax
* strong introspection capabilities
* intuitive object orientation
* natural expression of procedural code
* full modularity, supporting hierarchical packages
* exception-based error handling
* very high level dynamic data types
* extensive standard libraries and third party modules for virtually every task
* extensions and modules easily written in C, C++ (or Java for Jython, or .NET languages for IronPython)
* embeddable within applications as a scripting interface

Python Paradigms:

* Procedural
* Object-Oriented
* Functional

2 Ways to Interact with Python:

**Interactive mode:** In interactive mode, you type instructions to Python one line at a time—much the same way that an operating system (shell) accepts instructions from a command line. You can also write short multiline programs or import code from text files or from Python's builtin modules.

Interactive mode includes extensive help, too. With interactive mode, you can explore Python's abilities.

**IDLE:** The IDLE development environment includes Python's interactive mode and more—tools for writing and running programs and for keeping track of names.

IDLE is written in Python and shows off Python's considerable abilities

Basic Structure :

**Comments**

* Non-computational parts of the program that textually describe the behavior of the program.
* Comments begin with #, everything to right of the hash is ignored by Python.
* Comments should be frequent so you and others can understand the code.
* Lecture examples - [CodeSkulptor](http://www.codeskulptor.org/)
* More examples - [Comments, Strings, and Print](http://www.codeskulptor.org/#examples-more-1a_codeskulptor-prints_strings_comments.py)

**Strings**

* Sequence of characters enclosed by a pair of single or double quotes
* Examples are "cats hate dogs" and 'Strings are fun!'.
* Strings are one kind of data in Python. Their data type is denoted str.
* Lecture examples - [Hello World](http://www.codeskulptor.org/#examples-hello_world.py)
* More examples - [Comments, Strings, and Print](http://www.codeskulptor.org/#examples-more-1a_codeskulptor-prints_strings_comments.py)

**Numbers — Arithmetic Expressions**

* There are two kinds of numerical data in Python: integers and decimal numbers.
* Integers correspond to the data type int. Decimal numbers are represented by floating-point numbers corresponding to the data type float.
* Floating-point numbers have around 15 decimal digits of accuracy.
* In CodeSkulptor, all numbers (even integers) are represented internally as floating-point numbers.
* Lecture examples - [Arithmetic Expressions](http://www.codeskulptor.org/#examples-arithmetic_expressions.py)
* More examples - [Floats and Ints](http://www.codeskulptor.org/#examples-more-1a_arithmetic_expressions-floats_and_ints.py)

**Arithmetic Operators — Arithmetic Expressions**

* Five basic arithmetic operators; addition (+), subtraction (-), multiplication (\*), division (/) and exponentiation (\*\*)
* CodeSkulptor implements a subset of Python 2. In Python 2, the division operator (/) returns a float approximation to the exact answer if either of the operands is a  float.  If both operands are integers, division returns the exact answer round down to the nearest integer.
* The integer division operator // returns the quotient of two numbers..
* Lecture examples - [Arithmetic Expressions](http://www.codeskulptor.org/#examples-arithmetic_expressions.py)
* More examples - [Arithmetic Operations](http://www.codeskulptor.org/#examples-more-1a_arithmetic_expressions-operations.py), [Division](http://www.codeskulptor.org/#examples-more-1a_arithmetic_expressions-division.py)

**Arithmetic Expressions — Arithmetic Expressions**

* An arithmetic expression is either a number or an operator applied to two arithmetic expressions.
* Arithmetic expressions are entered as a sequence of numbers and arithmetic operators.
* Expressions are formed by grouping operators and operands via precedence: "Please excuse my dear Aunt Sallie"; parentheses, exponentiation, multiplication, division, addition, subtraction.
* Lecture examples - [Arithmetic Expressions](http://www.codeskulptor.org/#examples-arithmetic_expressions.py)
* More examples - [Order of Operations for Arithmetic Expressions](http://www.codeskulptor.org/#examples-more-1a_arithmetic_expressions-order_of_operations.py), [Possible Errors for Arithmetic Expressions](http://www.codeskulptor.org/#examples-more-1a_arithmetic_expressions-errors.py)

**Variables — Variables**

* Variable names consist of a sequence of letters, number and underscores (\_).
* Variable names start with a letter or underscore and are case sensitive.
* Single equals (=) is used for assignment to variables. Double equals (==) is used for testing equality.
* Lecture examples - [Variables](http://www.codeskulptor.org/#examples-variables.py)
* More examples - [Variable Naming](http://www.codeskulptor.org/#examples-more-1a_variables-names.py), [Vabiable Assignment](http://www.codeskulptor.org/#examples-more-1a_variables-assignment.py), [Variable Operations](http://www.codeskulptor.org/#examples-more-1a_variables-operations.py), [Formulas](http://www.codeskulptor.org/#examples-more-1a_variables-formulas.py)

A Simple Program, illustrating the above mentioned attributes of Python , goes hereunder:

#This is your First code:

Var1=1

var2=”Hello World!”

print var2

Standard Data Types:

The data stored in memory can be of many types. For example, a person's age is stored as a numeric value and his or her address is stored as alphanumeric characters. Python has various standard types that are used to define the operations possible on them and the storage method for each of them.

Python has five standard data types:

1. Numbers
2. String
3. List
4. Tuple
5. Dictionary