Workshop on Python (Day 1)

By Suriya G
Organized by Suresh Sir, UPNM

TABLE OF CONTENTS

01 Python

- What is python?
- Use cases

02 Installation

- Downloading
- Installation

03 Basics

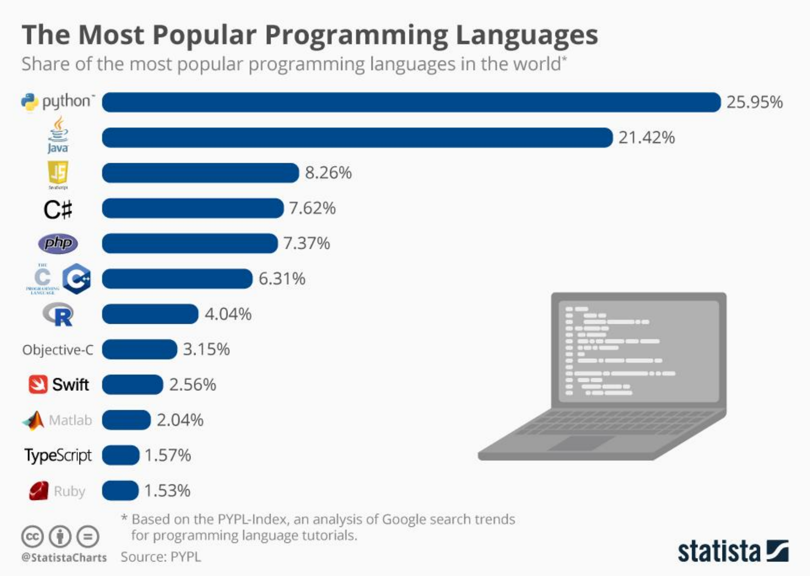
- Variables
- Data Types
- Operators

04 Execution

- Inputs/Outputs
- Comments
- Debugging

History of Python

- Developed by **Guido van Rossum (early 90's)**
- Open source
- Very popular today.



The Most Popular Programming Languages

Share of the most popular programming languages in the world*

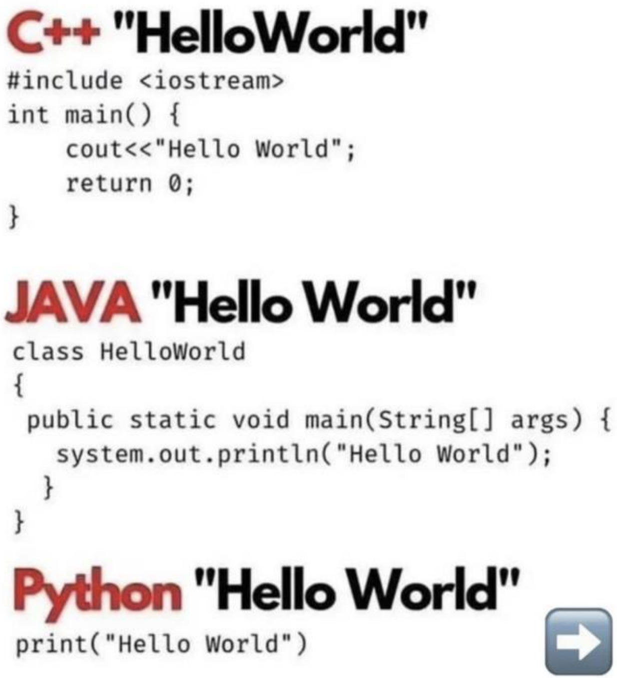
* Based on the PYPL-Index, an analysis of Google search trends for programming language tutorials.

@StatistaCharts Source: PYPL

statista

Advantages of Python

- Simple & Easy to understand.
- Combines well with cloud services.
- Variety of libraries available for AI development



C++ "HelloWorld"

```
#include <iostream>
int main() {
    cout<<"Hello World";
    return 0;
}
```

JAVA "Hello World"

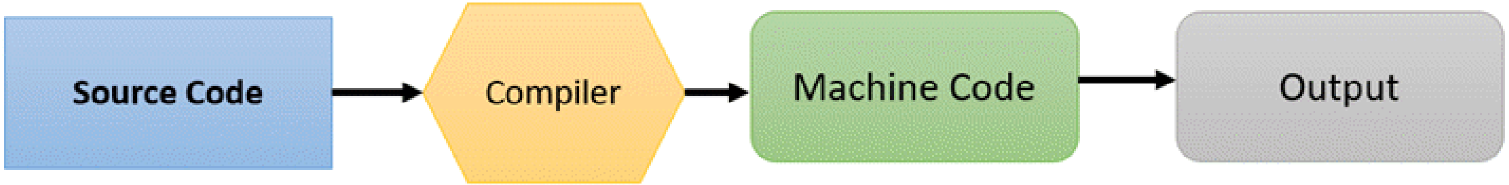
```
class HelloWorld
{
 public static void main(String[] args) {
   system.out.println("Hello World");
  }
}
```

Python "Hello World" ➡️

```
print("Hello World")
```

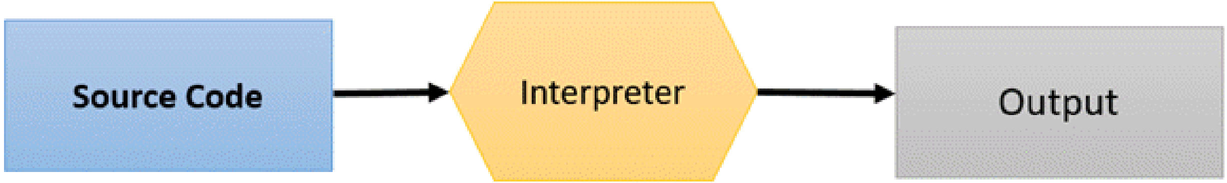
Compiler vs Interpreter

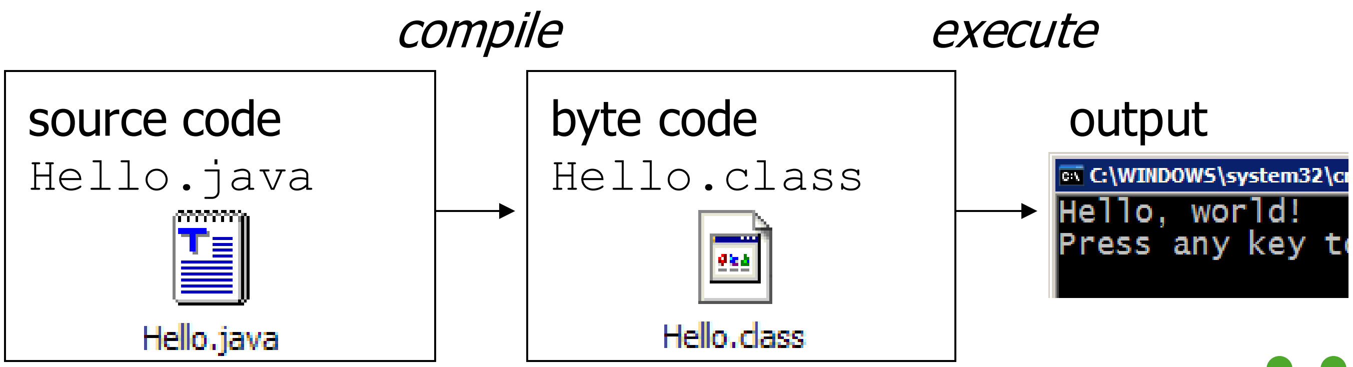
How Compiler Works



© guru99.com

How Interpreter Works





compile

execute

source code

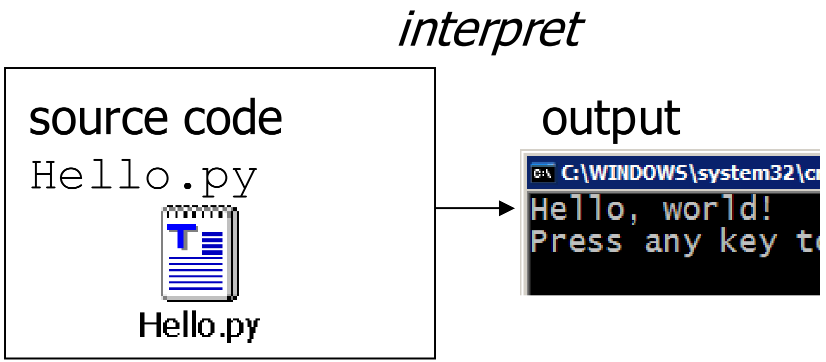
`Hello.java`

byte code

`Hello.class`

output

```
Hello, world!
Press any key to
```



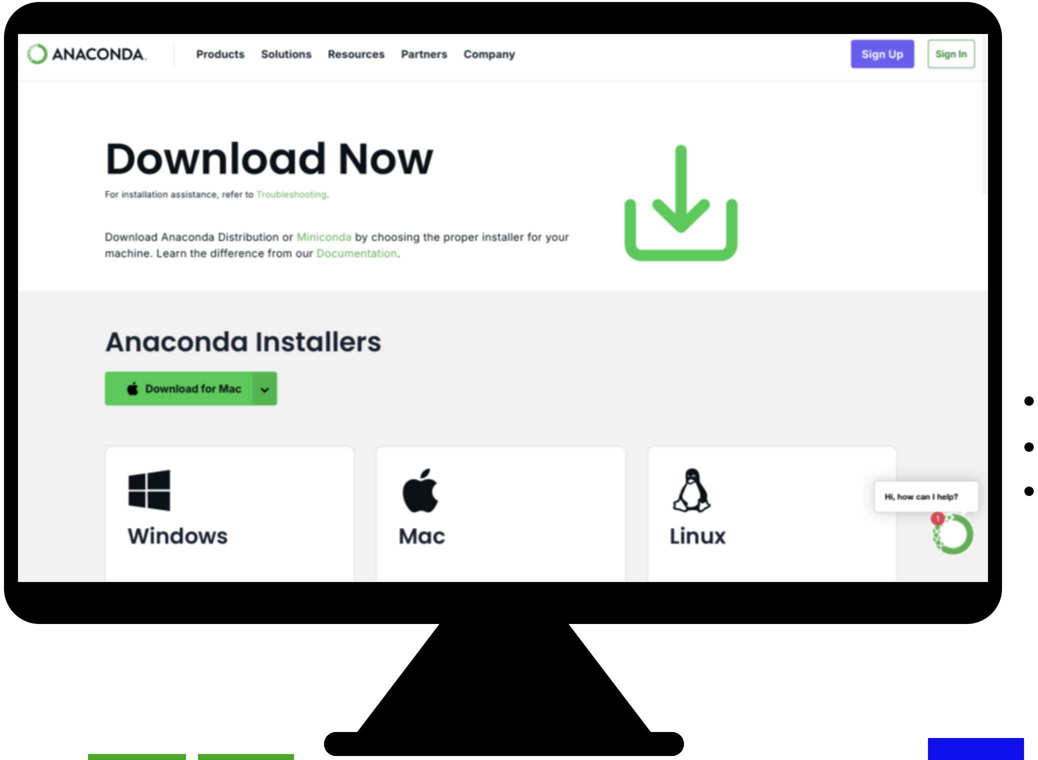
interpret

source code

`Hello.py`

output

```
Hello, world!
Press any key to
```



Download Now

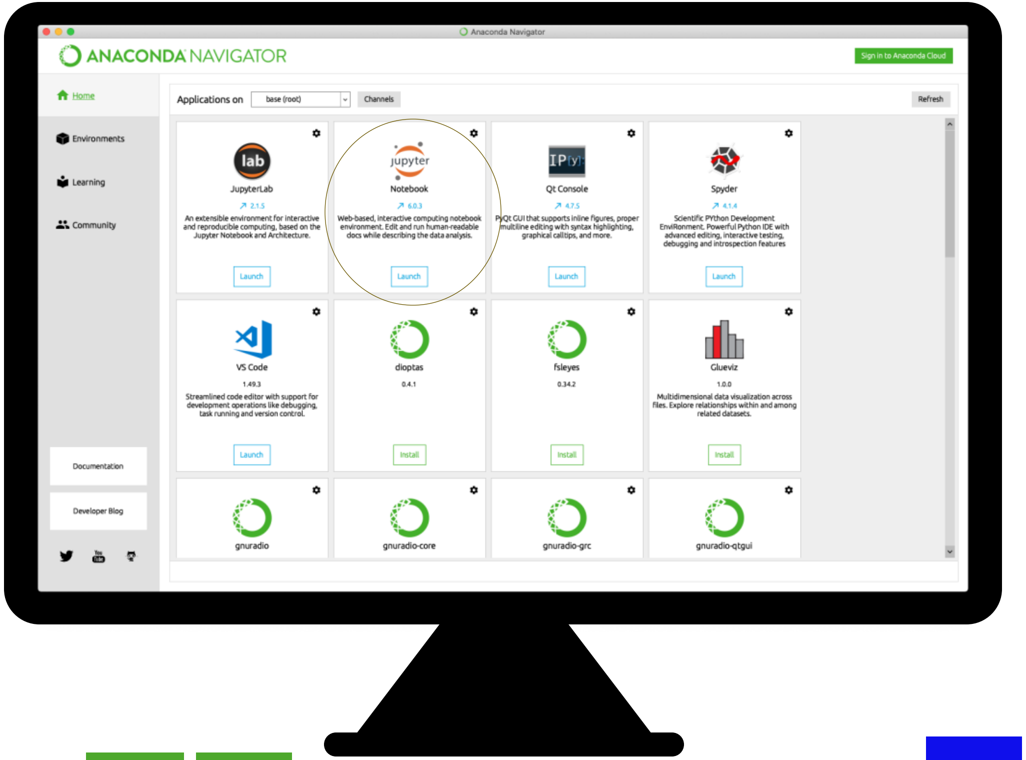
For installation assistance, refer to Troubleshooting.

Download Anaconda Distribution or Miniconda by choosing the proper installer for your machine. Learn the difference from our Documentation.

Anaconda Installers

Installation!!!

- https://www.anaconda.com/download
- Click Download for Mac/Windows
- Install in your PC



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

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

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

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

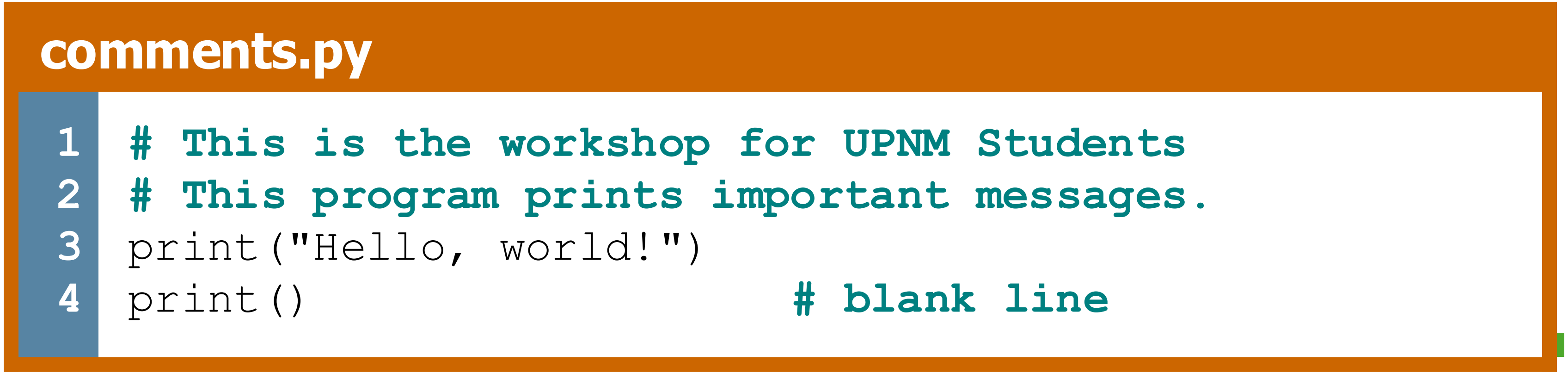
VS Code: Streamlined code editor with support for development operations like debugging, task running and version control.

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

- **After successful installation, open Jupyter Notebook.**
- **It will be redirected to your web browser.**

Comments

- Comments can be used to explain Python code.
- Helps the code to be more readable.

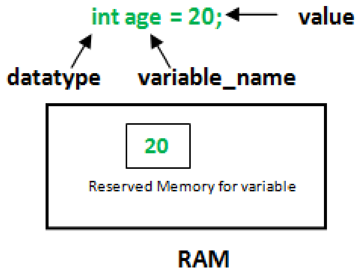


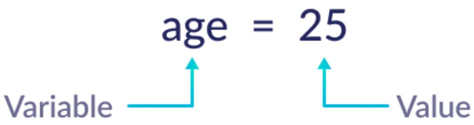
comments.py

```
# This is the workshop for UPNM Students
# This program prints important messages.
print("Hello, world!")
print()                    # blank line
```

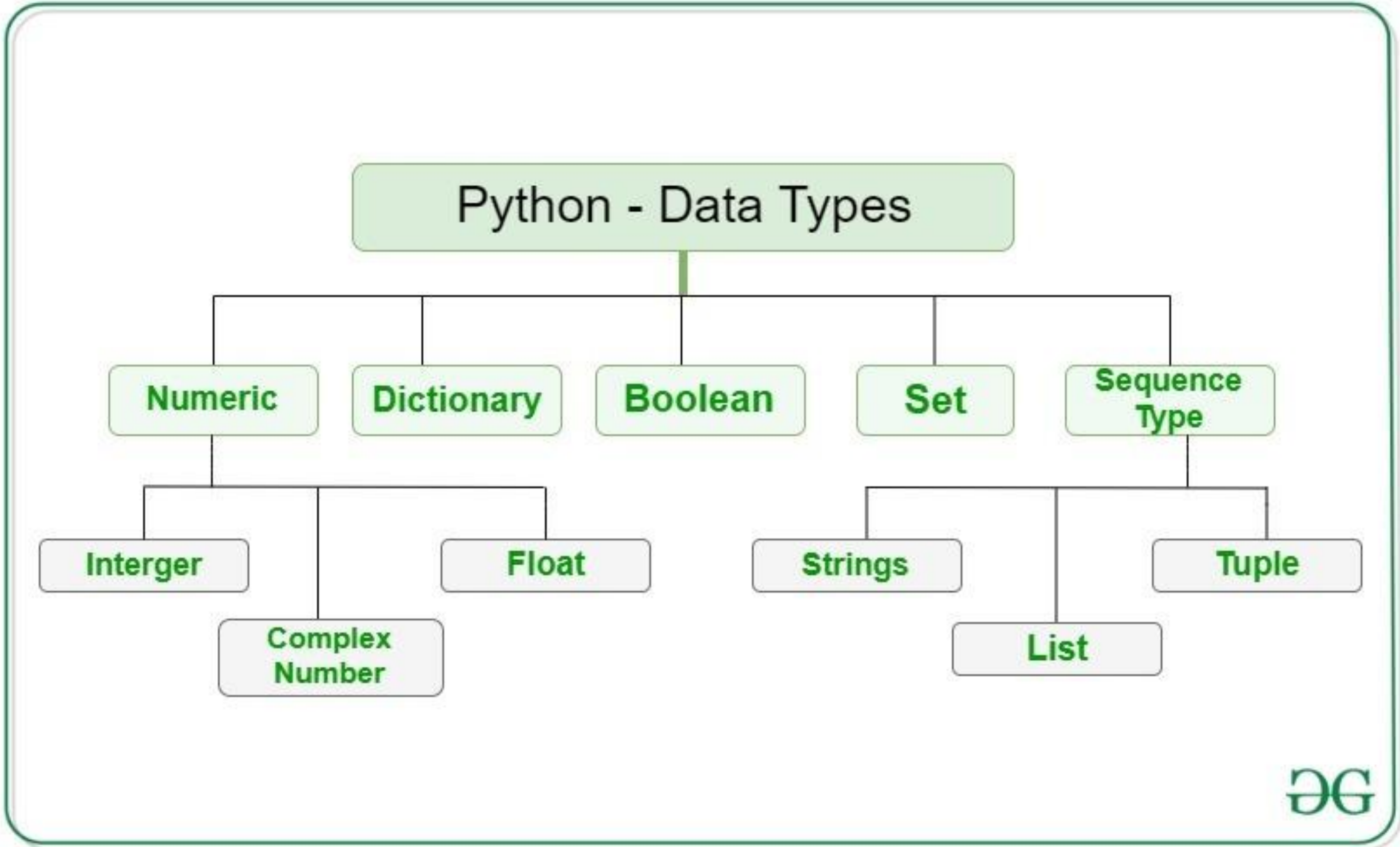
Variables

- Used to **store the data** in computer’s memory
- Each variable is assigned to an unique name.
- It allows programmers to store and update values **dynamically**.





Data Types


Python - Data Types
Numeric
Dictionary
Boolean
Set
Sequence Type
Interger
Complex Number
Float
Strings
List
Tuple

Numeric

- Represents numbers and supports arithmetic operations.
- **int** is for whole numbers, **float** for decimal numbers, and **complex** for numbers with real and imaginary parts.

numeric.py

```
# These are the types of numeric operators.
X = 10                   #int
Y = 3.14                 #float
Z = 2+3j                 #complex
```

String

- A sequence of characters enclosed in quotes (' ', " ").
- Supports indexing, slicing, and various string operations.

string.py

```
# These are the types of string operators.
S1 = ‘Hello’                     #single quotes
S2 = “World”                     #double quotes
```

List

- An ordered, mutable (changeable) collection of elements.
- Supports heterogeneous data types and allows indexing/slicing.

numeric.py

```
# These are the types of list operators.
lst1 = [1,7,9]                #collection of elements
lst2 = [8,’Hello’,3.5]        #Supports multiple dtypes
```

Tuples

- An ordered, immutable collection of elements.
- Faster than lists and used when data should remain unchanged.

tuple.py

```
# These are the types of tuple operators.
X = (1,2,3)                         #Similar to list
Y = (7,'UPNM',6.2)                  #but immutable
```

Set

- An unordered collection of unique element.
- They are mutable and cannot contain duplicate elements.

set.py

```
# These are the types of set operators.
S1 = {2,4,1}                    #included with curly braces
```

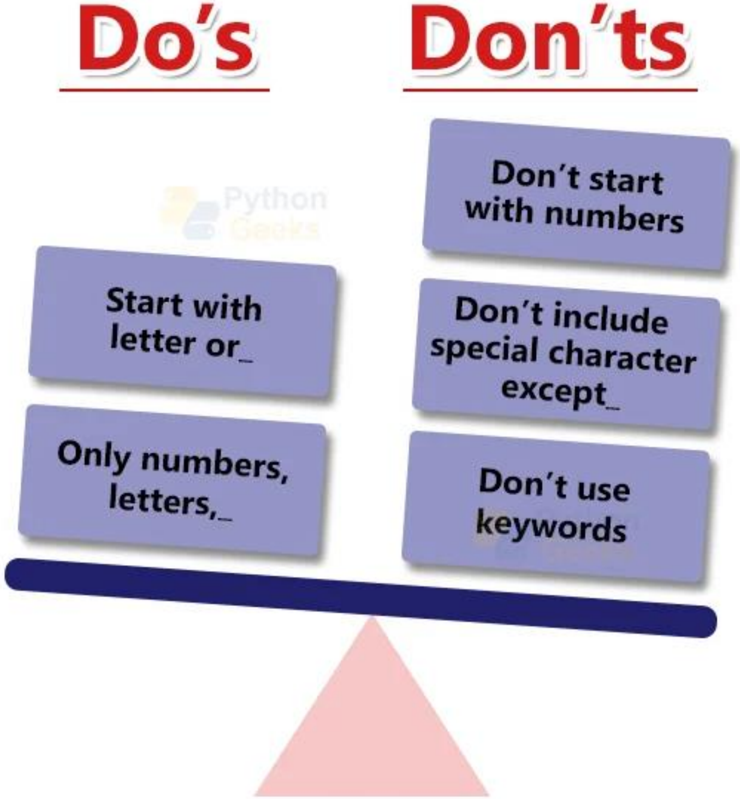
Dictionary

- Stores key-value pairs in an unordered format.
- Keys must be unique and immutable, while values can be any type.

dict.py

```
# These are the types of dictionary dtype.
d = {”name” : “Suriya”,
     ”Age”  : 22,
     “Gender” : “Male”}
```

Rules for Naming Variables



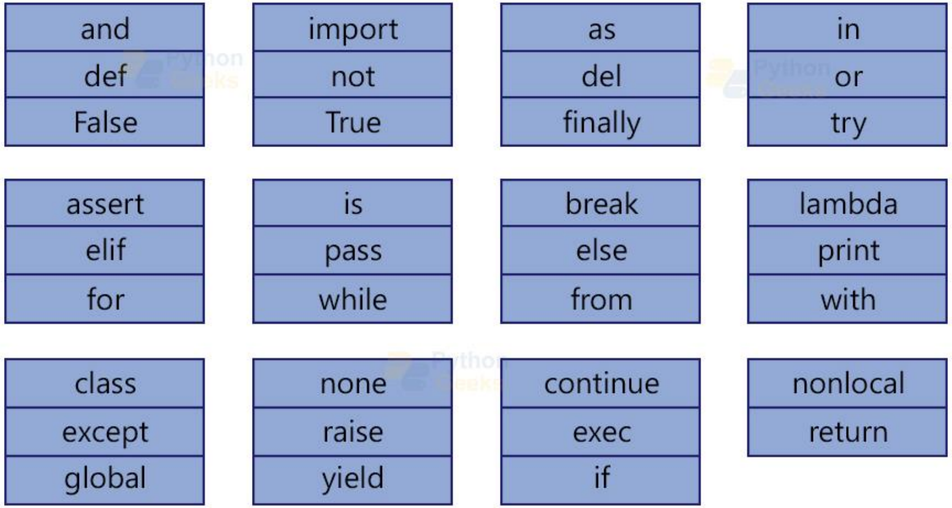
Do's

- Start with letter or_
- Only numbers, letters,_

Don'ts

- Don't start with numbers
- Don't include special character except_
- Don't use keywords

List of Reserved Keywords



and	import	as	in
def	not	del	or
False	True	finally	try
assert	is	break	lambda
elif	pass	else	print
for	while	from	with
class	none	continue	nonlocal
except	raise	exec	return
global	yield	if	

(These keywords cannot be used as a variable name)

Types of Operators

ARITHMETIC

Basic mathematical operations
(+,-,*,/,**,//,%)

RELATIONAL

Compares two values
(==, >=,<=,<,>,!=)

LOGICAL

Logical operations
(AND, OR, NOT)

BITWISE

Deals with binary digits

ASSIGNMENT

Assigns and modify values
(+=,-=,/=,*=)

SPECIAL

Membership operators
(is, is not)

Arithmetic

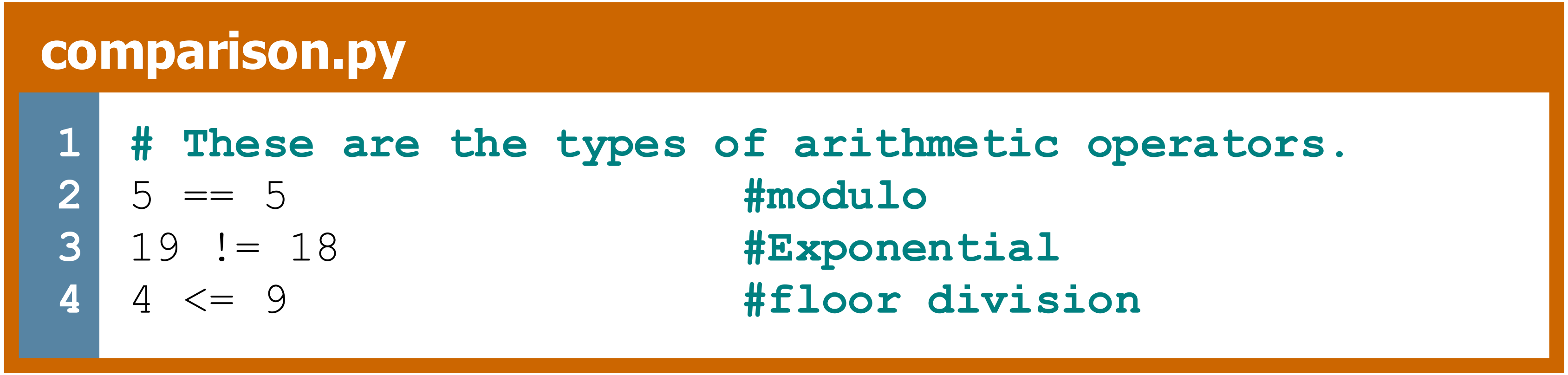
- Performs mathematical calculations like **addition, subtraction, multiplication, and division**.
- Supports both integer (//, %) and floating-point (/, **) operations.

arithmetic.py

```
# These are the types of list operators.
x = 3 + 17                  #addition
y = 17%2                    #modulo
Z = 2**3                    #Exponential
a = 3//2                    #floor division
```

Comparison

- Compares two values and returns a boolean **(True or False).**
- Used for decision-making in conditions **(if, while).**



comparison.py

```
# These are the types of arithmetic operators.
5 == 5                 #modulo
19 != 18               #Exponential
4 <= 9                 #floor division
```

Logical

- Evaluates boolean expressions using logical operations.
- Used to combine multiple conditions in control structures.

arithmetic.py

```
# These are the types of list operators.
True and True          #and operator
False or True          #or operator
```

Assignment

- Assigns values to variables using **=** and modifies them using shorthand operators **(+=, -=, etc.).**
- Helps in reducing redundancy when updating variables.

assignment.py

```
# These are the types of arithmetic operators.
x = 3                  #variable defining
x += 6                 #addition assignment
x -= 2                 #subtraction assignment
```