

### TOPIC 01: INTRODUCTION

- Python Introduction
- Getting Started
- Keywords & Identifiers
- Statements & Comments
- Python Variables
- Python Datatypes
- Python Type Conversion
- Python I/O and Import
- Python Operators
- Python Namespace

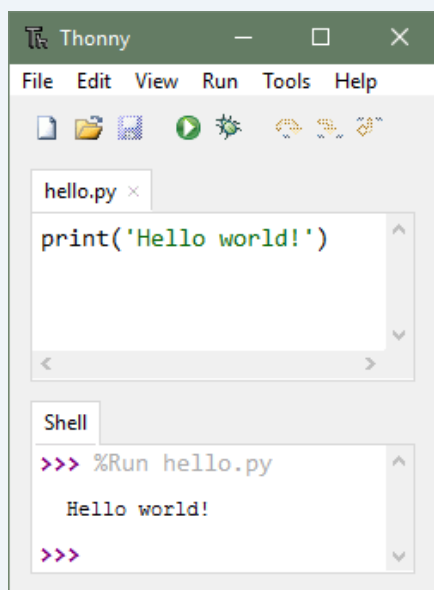
#### 1.1. Install Thonny Python IDE

Link to download Thonny Python IDE for beginners

<https://thonny.org/>

Thonny is a popular Python Integrated Development Environment (IDE) that is designed for beginners and students learning to program in Python.

#### 1.2. Run Thonny Python IDE



Thonny comes with Python 3.10 built in, so just one simple installer is needed and you're ready to learn programming. (You can also use a separate Python installation, if necessary.) The initial user interface is stripped of all features that may distract beginners.

#### 1.3. Python Programming Keywords

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

In Python, All keywords are case sensitive. All the keywords except True, False and None are in lowercase and they must be written as it is.

#### 1.4. Create a program to test Keywords

```
>>> import keyword
>>> keyword.d. i s k e y w o r d ( " t u t s m a k e . c o m " )
False
>>> keyword.d. i s k e y w o r d ( " t r y " )
True
>>>
```

#### 1.5. Python Programming identifiers

Identifier rules of Python

##### Rule 1:

Use either a lowercase (A to Z) or an uppercase (A to Z) sequence of letters. However, you can also add digits (0 to 9) or an underscore ( \_ ) while writing identity in python.

For example:- Names like myClass, my\_1, and upload\_image\_to\_db are all valid identifiers.

##### Rule 2:

You cannot write digit with the start of an identifier. This will assume invalid. But you can write digit with the end of the identifier.

For Example:- If you write identifier like 1variable, it is invalid, but variable1 is perfectly fine.

##### Rule 3:

Python Programming Reserved Keywords cannot be used as identifiers. Like del, global, not, with, as, if, etc.

### 1.5. Python Programming identifiers (cont)

#### Rule 4:

Also, you can not use special character as an identifier in python programming like '.', '!', '@', '#', '\$', '%', etc.

#### Rule 5:

The identifier can be of any length.

In the Python programming language, identifiers are user-defined names. Which represents functions, classes, variables, and any other objects in the code. It helps to distinguish one identity from another.

### 1.6. Create a program to test identifier

```
>>> 'john'.isidentifier()
True
>>> 'lhell o'.isidentifier()
False
>>> 'tutsmake.com'.isidentifier()
False
>>> 'tutsmake_com'.isidentifier()
True
```

### 1.7. Python Statement

#### Simple Assignment Statement

##### Syntax:

```
variable = expression
```

##### Example:

```
lang = " Learn Python "
```

#### Multi-line statement

Use a new line character to break the statement line

```
x = 5 + 4 + 2 + \
    1 + 6 + 4 + \
    9 + 7 + 8
```

Use parentheses ( ), brackets [ ] and braces { }.

```
x = (5 + 4 + 2 +
    1 + 6 + 4 +
    9 + 7 + 8)
```

```
colors = ['red',
          'blue',
          'green']
```

Multiple line statements in single line.

### 1.7. Python Statement (cont)

Use semicolons

```
a = 1; b = 2; c = 3
```

In python programming, A statement is a logical instruction. Which can read and execute by Python interpreter. In Python, it could be an expression or an assignment statement.

### 1.8. Python Indentation

#### Python uses indentation

A code block starts with indentation and ends with the first unindented line. And a code block that represents the body of a function, loop, etc. For example:

```
for i in range(1,9):
    print(i)
    if i == 5:
        break
```

#### Python Comments

##### Single line comment

```
#This is hello world by tutsmake.com
print( 'Hello world')
```

##### Multi line comment

```
#This is hello world program
#by tutsmake.com{\n #hello{\n print( 'Hello world
'
```

```
" " "This is also a
multi line
comments""
print( 'Hello world')
```

### 1.9. Docstring in Python

#### Docstring in Python

```
def myFunc(no):
    " " " Fun ction to 10x the value""
    return 10*no
```

#### access the code above

```
>>> print( myF unc.__ doc__)
```

### 1.10. Python Variable and Type

#### Rules for creating a variable in python

Rule 1:

A variable name must begin with a letter or the underscore character

Rule 2:

A variable name cannot begin with a number

Rule 3:

The python variable name can only contain alpha-numeric characters and underscores (A-z, 0-9, and \_)

Rule 4:

A Variable names are case-sensitive Like name, Name and NAME are three different variables

#### Python declare/create variable

Here first we will create/declare a variable. And also assign value to it.

```
x = 10
print("My fav no is " + x)
```

#### Change variable value in python

We will create a variable and change the value of this variable at runtime.

```
x = 10
print("My fav no is " + x)
x = 15
print("My fav no is " + x)
```

#### Assigning a single value to multiple variables

Python allows to assign a single value to several variables simultaneously.

```
x = y = z = 10
print(x)
print(y)
print(z)
```

#### Assigning a different values to multiple variables

```
x, y, z = 5, " str ing ", 99
print(x)
print(y)
print(z)
```

#### Types of variables

##### Local Variable

Which variables are made inside the function. They are called local variables. Local variables can be used only inside the function. For example:

### 1.10. Python Variable and Type (cont)

```
def valfun():
    x = "great"
    print(" Python is " + x)
valfun()
```

##### Global Variable

Which variables are made outside the function. They are called global variables. Global variables can be used both inside and outside the function. For example:

```
x = "great"
def valfun():
    print(" Python is " + x)
valfun()
```

### 1.11. Data Types in Python

## Numbers Data Type

Integer, floating-point numbers, and complex numbers under the category numbers. Those you define as int, float and complex classes. For exam

```
a = 10
print(a, "is of type", type(a))

a = 50.0
print(a, "is of type", type(a))

a = 11+2j
print(a, "is complex number ?", isinstance(a, complex))
```

Output of the above program is following:

```
10 is of type <class 'int'>
50.0 is of type <class 'float'>
(11+2j) is complex number? True
```

## List Data Type

list data types hold different types of data. it does not need to be of the same type. The items stored in the list are separated with a comma (,) and enclosed in square brackets [].

```
a = [10, 5.5, 'python']
print(a)
a = [5, 10, 15, 20, 25, 30, 35, 40]
# a[2] = 15
print("a[2] = ", a[2])
# a[0:3] = [5, 10, 15]
print(" a[0:3] = ", a[0:3])
# a[5:] = [30, 35, 40]
print(" a[5:] = ", a[5:])
```

The output of the above program is following:

```
[10, 5.5, 'python']
a[2] = 15
a[0:3] = [5, 10, 15]
a[5:] = [30, 35, 40]
```

## Tuple Data Type



### 1.11. Data Types in Python (cont)

A tuples data type is alike to list data type. But it is only one difference, once tuples created can not be changed/modify.

Tuples contain the collection of the items of different data types alike list data type. The items of the tuple are separated with a comma (,) and enclosed in parentheses ().

```
a = (8, 'python', 1+2j)
# a[1] = 'program'
print( "a[1] = ", a[1])
# a[0:3] = (5, 'program', (1+3j))
print( " a[0:3] = ", a[0:3])
```

Output of the above code is following:

```
a[1] = python
a[0:3] = (8, 'python', (1+2j))
```

#### Strings Data Type

A string is a sequence of characters in Python. In python, Strings are either enclosed with single quotes, double, and triple quotes.

```
string = 'Hello world!'
# string[1] = 'e'
print( " str ing[1] = ", string[1])
mString = " " " Hello world!
           this is multi
           line string exampl e""
print( mSt ring)
```

Output of the above program is the following:

```
string[1] = e
Hello world!
           this is multi
           line string example
```

#### Set Data Type

Set data types hold unordered collection of unique items. The items stored in the set data types are separated with a comma (,) and enclosed within square brackets {}.

```
abc = {5,2,3 ,1,4}
# printing set variable
print( "abc = ", abc)
# data type of variable a
print( typ e(abc))
```

Output of the above program is following:

```
abc = {1, 2, 3, 4, 5}
<class 'set'>
```

#### Dictionary Data Type

### 1.11. Data Types in Python (cont)

Dictionary data types is held data in key and value pairs. Note:- A dictionary data type does not allow duplicate keys in collection. But the values can be duplicate. For example:

```
dict = {1:" joh n", "l ast nam e": "s mit h", " age
":25}
# prints the value where key value is 1
print( dic t[1])
# prints the value where key value is " las tna me
"
print( dic t["l ast nam e"])
# prints the value where key value is " age "
print( dic t["a ge"])
```

Output of the above program is following:

```
john smith 25
```

#### TOPIC: FLOW CONTROL

#### TOPIC: FUNCTIONS

#### TOPIC: DATATYPES

#### TOPIC: FILE HANDLING

- File Handling Operation
- Python open and close
- Python Read and Write
- Python File Delete

#### TOPIC: Python built-in Methods

- Math Functions
- Math Module Functions
- String Methods
- List Methods
- Dictionary Methods
- Set Methods