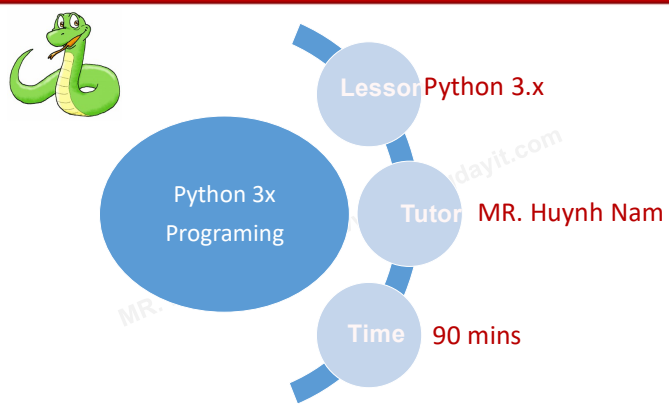


1

Contents

- Python language & History
- Comparison among programing languages
- Variable in Python
- Numeric in Python
- Conversion
- Fraction in Python
- Input / Output in Python
- Comment in Python
- How to code and run Python program

2

Example 1

- Print out a screen 100 times "I love you"

C language

```
#include <iostream.h>
void main()
{
    for (int I = 0; I < 100; i++)
        cout << "I love you \n";
}
```

Python 3.x

```
>print ("I love you" * 100)
```


3

Example 2

- Multiplying 2 big numbers

- 1234567891234567891234567891234567891234567891234567891234567891234567891234567891234567 9 * 1000000000

C++
- Using dynamic list
- Many codes

Python

```
>print
(1234567891234567891234567891234567891234567891234567891234567891234567891234567
89 * 1000000000)
```


4

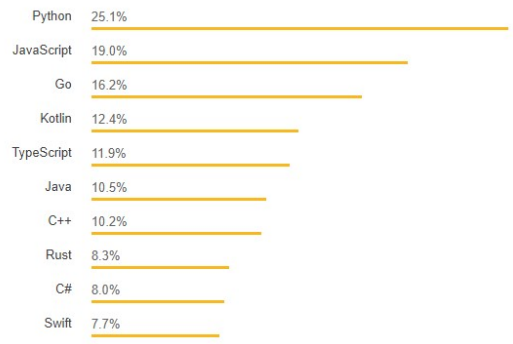
Introduction to Python

- Born 1991, author: Guido Van Rossum
- Features:
  - Be a script and interpeter language
  - Clearly, friendly and easy to learn
    - Increasing to use English keyword rather than symbols
    - Appoarching simple way
      - Using "while" and remove "do while"
      - Using "if - else" and remove "switch – case"
  - Versatile
    - Web prgraming
    - Desktop, graphic, game application
    - Mobile programing
    - High effictive in sciencetific and computing

Introduction to Python

- Features:
  - Multi-variants
    - Allow many ways in programing
    - Dynamic, Static and Strong type
  - Python is powerful and fast
    - Battery included: having a rich and versatile standard library which is immediately available, without making the user download separate packages
    - Code quickly
  - Well adapted with many language
    - Java → Jython
    - .NET → IronPython, Python for .NET
    - Extension module in C/C++
    - …
  - Python can run every platform
    - Unix
    - Windows
    - Mac
    - Nokia S60

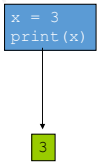
Comparison among programing languages - Wanted Language



Variable in Python

- Python is dynamically typed, which means that you don't have to declare what type each variable is. variables are a storage placeholder for texts and numbers.
- Python variables work more like tags. When you do an assignment in Python, it tags the value with the variable name.

	Other language (variable)	Python (name)
a = 1		
a = 2		
b = a		



Data type – Python Numberic

- Python supports integers, floating point numbers and complex numbers. They are defined as int, long, float and complex class in Python.
- Operator : `+ - / * %`

```
x = 14
y = 3
print ("Sum: ", x + y)
print ("Product: ", x * y)
print ("Remainder: ", x % y)
```

```
Sum: 17
Product: 42
Remainder: 2
```

```
x = 5
print ("x started as", x)
x = x * 2
print ("Then x was", x)
x = x + 1
print ("Finally x was" ,x)
```

```
x started as 5
Then x was 10
Finally x was 11
```


Conversion in Python

- Python supports integers, floating point numbers and complex numbers. They are defined as int, float and complex class in Python.
- Integers and floating points are separated by the presence or absence of a decimal point.
  - Example: 5 is integer whereas 5.0 is a floating point number.
- Complex numbers are written in the form, ***x + yj***, where x is the real part and y is the imaginary part.
- Using the ***type()*** function to know which class a variable or a value belongs to and ***isinstance()*** function to check if it belongs to a particular class.

```
a = 5

# Output: <class 'int'>
print(type(a))

# Output: <class 'float'>
print(type(5.0))

# Output: (8+3j)
c = 5 + 3j
print(c + 3)

# Output: True
print(isinstance(c, complex))
```


Conversion in Python

- Convert one type of number into another. This is also known as coercion.
- Operations like addition, subtraction coerce integer to float implicitly (automatically), if one of the operand is float.
  - Example 1: **1 + 1** → output: **2**
  - Example 2: **1 + 4.0** → output: **5.0**
- Use built-in functions like ***int()***, ***float()*** and ***complex()*** to convert between types explicitly. These function can even convert from ***strings***.

```
>>> int(2.3)
2
>>> int(-2.8)
-2
>>> float(5)
5.0
>>> complex('3+5j')
(3+5j)
```


Fractions in Python

- Python provides operations involving fractional numbers through its fractions module.
- A fraction has a numerator and a denominator, both of which are integers. This module has support for rational number arithmetic.

```
import fractions

# As float
# Output: 2476979795053773/2251799813685248
print(fractions.Fraction(1.1))

# As string
# Output: 11/10
print(fractions.Fraction('1.1'))
```

```
import fractions

# Output: 3/2
print(fractions.Fraction(1.5))

# Output: 5
print(fractions.Fraction(5))

# Output: 1/3
print(fractions.Fraction(1,3))
```

```
from fractions import Fraction as F

# Output: 2/3
print(F(1,3) + F(1,3))

# Output: 6/5
print(1 / F(5,6))

# Output: False
print(F(-3,10) > 0)

# Output: True
print(F(-3,10) < 0)
```

Input / Output in Python (Output)

- Use the print() function to output data to the standard output device (screen).

```
print('This sentence is output to the screen')
# Output: This sentence is output to the screen

a = 5

print('The value of a is', a)
# Output: The value of a is 5
```

- The actual syntax of the print() function is :

```
print(*objects, sep=' ', end='\n', file=sys.stdout, flush=False)
```

```
print(1,2,3,4)
# Output: 1 2 3 4

print(1,2,3,4,sep='*')
# Output: 1*2*3*4

print(1,2,3,4,sep='#',end='&')
# Output: 1#2#3#4&
```


Input / Output in Python (Input)

- To allow flexibility we might want to take the input from the user. In Python, we have the input() function to allow this. The syntax for input() is:

```
input([prompt])
```

```
>>> num = input('Enter a number: ')
Enter a number: 10
>>> num
'10'
```


Comment in Python

- Comments are lines that exist in computer programs that are ignored by compilers and interpreters. Including comments in programs makes code more readable for humans as it provides some information or explanation about what each part of a program is doing.
- Depending on the purpose of your program, comments can serve as notes to yourself or reminders, or they can be written with the intention of other programmers being able to understand what your code is doing.
- In general, it is a good idea to write comments while you are writing or updating a program as it is easy to forget your thought process later on, and comments written later may be less useful in the long term.

```
# Print "Hello, World!" to console
print("Hello, World!")
```

```
x = 8  # Initialize x with an arbitrary number
```

```
def parse_token(token):
    """
    This function parses a token.
    TODO: write a decent docstring :-)
    """

    if token == '\\and':
        do_something()
```


Review

- Variables declaration
- Output / input


Data Visualization
EDU-BOOST
THANK YOU
Q & A
WRITTEN BY MR. HUYNH NAM
www.giangdayit.com