University of Tripoli
Faculty of Information Technology
Department of Software Engineering

ITSE305 مواضيع مختارة
Python Programming
S2025

Lecture (1): Python Basics

Python Installing

- Download Python from python.org
  - Check first if python is installed on your OS

```
python --version
```

- Download VSC (IDE)

Executing Python Syntax

- Execute Python syntax in the command Line
  - Run Python code from a file

```
C:\Users\Your Name>python helloworld.py
```

  - Run Python code directly in cmd

```
C:\Users\Your Name>python
Python 3.6.4 (v3.6.4:d48eceb, Dec 19 2017, 06:04:45) [MSC v.1900 32 bit
(Intel)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> print("Hello, World!")
Hello, World!
```

Don't forget exit()

Python Indentation

- indentation in Python is very important
  - Python uses indentation to indicate a block of code.
  - Python will give you an error if you skip the indentation
    - At least one space (commonly 4 spaces)
    - But use the same number of spaces in the same block of code

```
if 5 > 2:
print("Five is greater than two!")
```

```
if 5 > 2:
 print("Five is greater than two!")
if 5 > 2:
        print("Five is greater than two!")
```

```
if 5 > 2:
 print("Five is greater than two!")
        print("Five is greater than two!")
```

Python Variables

- Variables do not need to be declared with any particular *type*
- Variables are created when you assign a value to it
- Using casting to specify the data type of a variable
- Variable names are case-sensitive.

```
a = 4
A = "Sally"
A = 6
y = int(3)
z = float(3)

print(a)
print(A)
print(y)
print(z)
```

```
4
6
3
3.0
```


Python-Variable Names

- A variable name must start with a letter or the underscore character
- A variable name cannot start with a number
- A variable name can only contain alpha-numeric characters and underscores
- Variable names are case-sensitive
- A variable name cannot be any of the Python keywords such as class, break and import.
- Use Camel case, Pascal case or Snake case techniques for multi-words variables names (Why?)

Python – Output Variables

- To output variables use the Python print() function

```python
x = "Python is awesome"
y = "Python is perfect"
z = 5

print(x)
print(y)
print(z)

print(x,y,z)
print(x+y)
print(x+""+y)
print(z+z)
```

```
Python is awesome
Python is perfect
5
Python is awesome Python is perfect 5
Python is awesomePython is perfect
Python is awesomePython is perfect
10
```

- What x+y is called??

Python Data Types

- Python has the following data types built-in:

Text Type:	str	x = "Hello World"
Numeric Types:	int Float Complex	x = 20 x = 20.5 x = 1j
Sequence Types:	List Tuple Range	x = ["apple", "banana", "cherry"] x = ("apple", "banana", "cherry") x = range(6)
Mapping Type:	dict	x = {"name" : "John", "age" : 36}
Set Types:	set, frozenset	x = {"apple", "banana", "cherry"} x = frozenset({"apple", "banana", "cherry"})
Boolean Type:	bool	x = True
Binary Types:	bytes	x = b"Hello"
None Type:	NoneType	x = None

Python Data Types

- Use the type() function to get the data type of any object
- In Python, the data type is set when you assign a value to a variable:
- To specify the data type, you can use the constructor functions such as: x = str("Hello")
- To convert (casting) from one type to another with Python numbers use the int(), float(),complex() and str() methods
  - But you cannot convert complex numbers into another number type.
- Python does not have a character data type, a single character is simply a string with a length of 1.
- To get the length of a string, use the len() function.
- The first character in a string has index 0.

Python-Modify Strings

- Python has a set of built-in methods that can use on strings:

Upper Case	upper()	a = "Hello, World!" print(a.upper())
Lower Case	lower()	a = "Hello, World!" print(a.lower())
Remove Whitespace	strip()	a = " Hello, World! " print(a.strip()) # returns "Hello, World!"
Replace String	replace()	a = "Hello, World!" print(a.replace("H", "J"))
Split String	split()	a = "Hello, World!" print(a.split(",")) # returns ['Hello', ' World!']

- All string methods return new values, they do not change the original string.

F-Strings

- Introduced in Python 3.6 to combine strings and numbers
- Simply put an f in front of the string literal, and add curly brackets {} as placeholders for variables and other operations.

Escape Characters

\’	Single Quote
\\	Backslash
\n	New Line
\t	Tab
b	Backspace

Comments

- Why ?
  - Comments can be used to explain Python code.
  - Comments can be used to make the code more readable.
  - Comments can be used to prevent execution when testing code.
- Comments start with a #, and Python will render the rest of the line as a comment
- Python does not really have a syntax for multiline comments
  - multiline string !!!
  - triple-quoted strings !!!

Python Operators

- Python divides the operators in the following groups:
  - Arithmetic operators
  - Assignment operators
  - Comparison operators
  - Logical operators
  - Identity operators
  - Membership operators

Python Arithmetic Operators

- They are used with numeric values to perform common mathematical operations:

+	Addition	x + y
-	Subtraction	x - y
*	Multiplication	x * y
/	Division	x / y
%	Modulus	x % y
**	Exponentiation	x ** y
//	Floor division	x // y

Python Assignment Operators

- They are used to assign values to variables:

Operator	Example	Same As
=	x = 5	x = 5
+=	x += 3	x = x + 3
-=	x -= 3	x = x - 3
*=	x *= 3	x = x * 3
/=	x /= 3	x = x / 3
%=	x %= 3	x = x % 3
//=	x //= 3	x = x // 3

Operator	Example	Same As
**=	x **= 3	x = x ** 3
&=	x &= 3	x = x & 3
|=	x |= 3	x = x | 3
^=	x ^= 3	x = x ^ 3
>>=	x >>= 3	x = x >> 3
<<=	x <<= 3	x = x << 3
:=	print(x := 3)	x = 3 print(x)

Python Comparison Operators

- They are used to compare two values:

==	Equal	x == y
!=	Not equal	x != y
>	Greater than	x > y
<	Less than	x < y
>=	Greater than or equal to	x >= y
<=	Less than or equal to	x <= y

Python Logical Operators

- They are used to combine conditional statements:

Operator	Description	Example
and	Returns True if both statements are true	x < 5 and x < 10
or	Returns True if one of the statements is true	x < 5 or x < 4
not	Reverse the result, returns False if the result is true	not(x < 5 and x < 10)

Python Identity Operators

- They are used to compare the objects, not if they are equal, but if they are actually the same object, with the same memory location:

Operator	Description	Example
is	Returns True if both variables are the same object	x is y
is not	Returns True if both variables are not the same object	x is not y

Python Membership Operators

- They are used to compare the objects, not if they are equal, but if they are actually the same object, with the same memory location:

Operator	Description	Example
in	Returns True if a sequence with the specified value is present in the object	x in y
not in	Returns True if a sequence with the specified value is not present in the object	x not in y

The END

by: Fatima Ben Lashihar