

02. Python Data Types & Operators

3ikakke

Outline

- Learning Objectives
- Learning a new programming language
- What is Python
- Learning Python
- Data Types
 - Strings
 - Integers
 - Floats
 - Booleans
 - None Type
 - Collections
 - * Lists
 - * Tuples
 - * Sets
 - * Dictionaries
- Operators
 - The assignment operator
 - Mathematical operators
 - Comparison operators
 - The concatenation operator
 - Logical operators
 - Object oriented operator

Learning Objectives

- Understand the core components of programming languages
- Understand the core construct of python
- Understand data types in python and be able to name and describe them
- Understanding how data types in python map to data types in data science
- Understanding operators and operator types
- Introduce two functions
 - print
 - type

Learning a new programming language

- Simple rule(cheat)
 - Data types
 - Operators
 - Control structure logic
 - Control structure loops
 - Functions (in built and user defined)
 - Object oriented programming (OOP)

What is Python

- Object Oriented Language
- In python (almost) everything is an object
- Data types have ‘methods’ that are unique to their type
- Python has a limited set of functions and instead of functions uses methods mostly
- Version 2 vs **Version 3**

Learning Python

- Follow the same rules for learning a new language just described!

Data Types

- Strings
- Integers
- Floats

- None
- Boolean
- Collections
 - Tuples
 - Lists
 - Dictionaries
 - Sets

Strings

- Characters of any type and any length wrapped in quotes that may be single or double

```
print("Hello world")
print('Hello world')
```

- Some string methods

```
"Hello world".lower()
"Hello world".upper()
"hello world".capitalize()
```

Integers

- Integers are simply whole numbers
- Examples 1, 2, 10, 55 ...

Floats

- Floats are decimal numbers and may include whole numbers with decimal place
- Example - 1.2, 4.73, 5.0, 7.00

Booleans

- Fancy name for True or False
- Notice they are not wrapped in quotes. That would make them strings!
- Examples - True, False

None Type

- Simply nothing!
- Same as NULL in some other languages
- Written as

None

Collections

- As the name implies a mix of different things put together
- 0 indexed (you count them from 0 not 1)
 - Tuples - once set cannot be changed formed with brackets
 - Lists - Like tuples but can be changed (modified) formed with square brackets
 - Sets - Like tuples but with no duplicates formed with curly braces

```
days = ('monday', 'tuesday', 'wednesday', 'thursday', 'friday', 'saturday', 'sunday') #tuple
basket = ['mango', 'banana', 'guava', 'orange', 'guava', 'pineapple', 'guava'] #list
fruits = {'mango', 'banana', 'guava', 'orange', 'pineapple'} #set

print(days[0])
print(basket[1])
```

Collections (contd.)

- Dictionaries - Key value pairs

```
details = {'name': 'Adamu Okechuckwu Adewumi', 'age': 35, 'sex': 'male'}

print(details['name'])
print(details['age'])
print(details['sex'])
```

Collections (contd.)

- Special collections
 - Series
 - Numpy Arrays (2 dimensional arrays)
 - Pandas Dataframes (collections that are like spreadsheets)

Data primitives in Data Science and how they map to Python Data Types

- Qualitative AKA Categorical AKA Character (Strings, Booleans)
- Quantitative AKA Numeric (Floats, Integers)
- Data science deals broadly with these 2 types only
- If data is missing that would map to None

Operators

- These are symbols that carry out some assigned function
- The assignment operator (=)

- Mathematical operators (+, -, /, *, **, %)
- The concatenation operator (+)
- Comparison operators (==, >, <, >=, <=, !=, is, is not)
- Logical operators (and, or)
- Object oriented programming operator (.)

The assignment operator =

```
a = 1
firstname = 'Danladi'
fruits = ['guava', 'orange', 'banana']
```

Mathematical operators + - / * ** %

```
#assign data to variables
a = 2
b = 3
c = 'Hello'

#Mathematical operators
a + b #addition
a - b #subtraction
a / b #division
a * d #multiplication
a ** d #exponent
a % b #modulus(remainder)
```

- There is operator precedence (PEDMAS)
 - Parenthesis (Bracket)
 - Exponent (Power)
 - Division
 - Multiplication
 - Addition
 - Subtraction

The concatenation operator +

- Plus(+) when used between string joins them

```
c + c #concatenation
```

Comparison operators (==, >, <, >=, <=, !=, is, is not, in, not in)

- Comparison operators return a boolean as result

- for equality we use the double equality symbol since the single one is reserved for assignment

```
#assigning variables
x = 4
y = 5
i = 1
j = 1
z = x
fruits = ['mango', 'guava', 'orange']

#testing comparison operators
x == y
i == j
x > y
x < y
x >= y
x <= y
x != y
x is y
x is not y
i is j
i is not j
x is z

'mango' in fruits
'mango' not in fruits
'pineapple' in fruits
'pineapple' not in fruits
```

Logical operators (and, or)

- This lets you evaluate multiple booleans and get a single boolean
- Seeing an example will make it easier to understand

```
#simple boolean comparison
True and True
True and False
False and False
True or True
True or False
False or False

(True and True) or (True or False)

((x == y) and ('mango' in fruits)) or (z > y)
```

Useful to know

- White space in python
- Writing comments in python
- We have seen variables, some other languages have constants but python doesnt have that in its current

construct

- f-strings

```
name = 'Stephen'  
print(f"Hello {name}")
```

Gist and training docs

- Data Types and Operators
- PDF version of module
- The Jupyter Notebook will be shared over **Slack**

Homework

Conclusion

- Lets go over our learning objectives again
 - Understand the core components of programming languages
 - Understand the core construct of python
 - Understand data types in python and be able to name and describe them
 - Understanding how data types in python map to data types in data science
 - Understanding operators and operator types

Q&A

Thank you for listening and contributing!