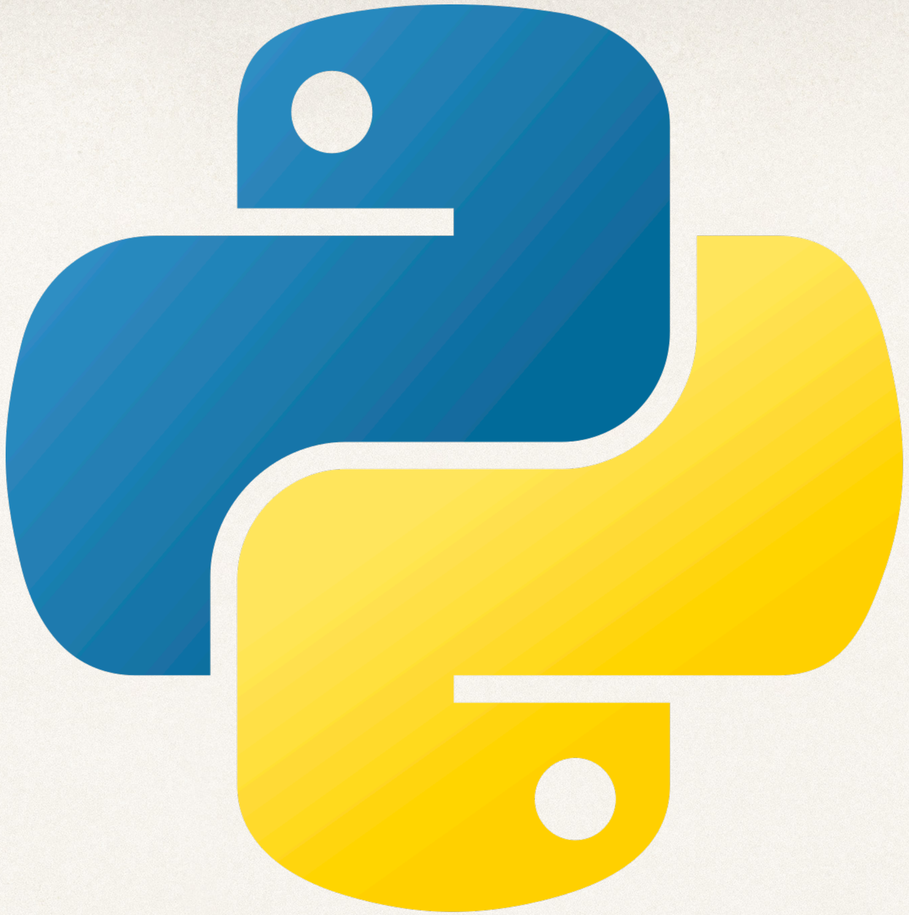

Getting Started with Python

Sandeep Soni

08/28/2023

Class logistics

Class logistics

- Refer to the class website for schedule: https://sandeepsoni.github.io/classes/qtm340.html

Class logistics

- Refer to the class website for schedule: https://sandeepsoni.github.io/classes/qtm340.html
- Join Piazza: https://piazza.com/emory/fall2023/qtm340

Class logistics

- Refer to the class website for schedule: https://sandeepsoni.github.io/classes/qtm340.html
- Join Piazza: https://piazza.com/emory/fall2023/qtm340
- Hw1 is released and is due on Wednesday

Question for the day

Question for the day

What shape is the distribution of word counts in a text collection?

Agenda

- What are constants, variables and expressions?
- What are the different data types in Python?
- What operations can be done in Python?
- What are the different data structures in standard Python?
- What are the different programming constructs?

Constants and Variables

Constants and Variables

- Constants are just values (e.g., `5`)

Constants and Variables

- Constants are just values (e.g., `5`)
- Variables are units that can store values (e.g., `x=5`)

Constants and Variables

- Constants are just values (e.g., `5`)
- Variables are units that can store values (e.g., `x=5`)
- In Python, variables need not be declared or typed

Key data types

Key data types

Type	Description	Examples

Key data types

Type	Description	Examples
int	Represents integers	`17, -3, 0`

Key data types

Type	Description	Examples
int	Represents integers	`17, -3, 0`
float	Represents real numbers*	`3.14, 0.00027, -5.8`

Key data types

Type	Description	Examples
int	Represents integers	`17, -3, 0`
float	Represents real numbers*	`3.14, 0.00027, -5.8`
bool	Represents binary values	`True , False`

Key data types

Type	Description	Examples
int	Represents integers	`17, -3, 0`
float	Represents real numbers*	`3.14, 0.00027, -5.8`
bool	Represents binary values	`True , False`
string	Represents text values	`“qtm”, ‘emory’, “mail@website.com"`

Key data types

Type	Description	Examples
int	Represents integers	`17, -3, 0`
float	Represents real numbers*	`3.14, 0.00027, -5.8`
bool	Represents binary values	`True , False`
string	Represents text values	`“qtm”, ‘emory’, “mail@website.com"`

There are other data types too.

Operators

- Arithmetic operators
- Comparison operators
- Logical operators
- Membership operators
- Identity operators
- Other operators such as assignment and bitwise operators

Arithmetic operators

Arithmetic operators

Operator	Description	Examples

Arithmetic operators

Operator	Description	Examples
+	Addition	`3+5, 3.14+5,"a"+"b"`

Arithmetic operators

Operator	Description	Examples
+	Addition	`3+5, 3.14+5,"a"+"b"`
-	Substraction	`3-5, 2-4.56`

Arithmetic operators

Operator	Description	Examples
+	Addition	`3+5, 3.14+5,"a"+"b"`
-	Substraction	`3-5, 2-4.56`
*	Multiplication	`3*5, "abc"*3`

Arithmetic operators

Operator	Description	Examples
+	Addition	3+5, 3.14+5,"a"+"b"
-	Substraction	3-5, 2-4.56
*	Multiplication	3*5, "abc"*3
/	Division	3/5

Arithmetic operators

Operator	Description	Examples
+	Addition	3+5, 3.14+5,"a"+"b"
-	Substraction	3-5, 2-4.56
*	Multiplication	3*5, "abc"*3
/	Division	3/5
%	Modulo division	3%5

Arithmetic operators

Operator	Description	Examples
+	Addition	`3+5, 3.14+5,"a"+"b"`
-	Substraction	`3-5, 2-4.56`
*	Multiplication	`3*5, "abc"*3`
/	Division	`3/5`
%	Modulo division	`3%5`
//	Floor division	`3//5`

Arithmetic operators

Operator	Description	Examples
+	Addition	3+5, 3.14+5,"a"+"b"
-	Substraction	3-5, 2-4.56
*	Multiplication	3*5, "abc"*3
/	Division	3/5
%	Modulo division	3%5
//	Floor division	3//5
**	Exponentiation	3**5

Arithmetic operators

Operator	Description	Examples
+	Addition	`3+5, 3.14+5,"a"+"b"`
-	Substraction	`3-5, 2-4.56`
*	Multiplication	`3*5, "abc"*3`
/	Division	`3/5`
%	Modulo division	`3%5`
//	Floor division	`3//5`
**	Exponentiation	`3**5`

Operators have precedence

Comparison Operators

Comparison Operators

Operator	Description	Examples

Comparison Operators

Operator	Description	Examples
	Is equal	`4==5, x==3.14`

Comparison Operators

Operator	Description	Examples
==	Is equal	`4==5, x==3.14`
!=	Not equal	`4!=5, True != False`

Comparison Operators

Operator	Description	Examples
==	Is equal	`4==5, x==3.14`
!=	Not equal	`4!=5, True != False`
>	Greater than	`5.14 > 5`

Comparison Operators

Operator	Description	Examples
==	Is equal	`4==5, x==3.14`
!=	Not equal	`4!=5, True != False`
>	Greater than	`5.14 > 5`
<	Less than	`5.14 < 6`

Comparison Operators

Operator	Description	Examples
==	Is equal	`4==5, x==3.14`
!=	Not equal	`4!=5, True != False`
>	Greater than	`5.14 > 5`
<	Less than	`5.14 < 6`
>=	Greater than or equal	`3 >= len ("the")`

Comparison Operators

Operator	Description	Examples
==	Is equal	`4==5, x==3.14`
!=	Not equal	`4!=5, True != False`
>	Greater than	`5.14 > 5`
<	Less than	`5.14 < 6`
>=	Greater than or equal	`3 >= len ("the")`
<=	Less than or equal	`3 <= (1+(2*1))`

Logical operators

Logical operators

Operator	Description	Examples

Logical operators

Operator	Description	Examples
and	Logical AND between two boolean expressions	`(3==4) and (1==1)`

Logical operators

Operator	Description	Examples
and	Logical AND between two boolean expressions	`(3==4) and (1==1)`
or	Logical OR between two boolean expression	`(3==4) or (1==1)`

Logical operators

Operator	Description	Examples
and	Logical AND between two boolean expressions	`(3==4) and (1==1)`
or	Logical OR between two boolean expression	`(3==4) or (1==1)`
not	Negation of a boolean expression	`not (3==4)`

Membership and Identity operators

Membership and Identity operators

Operator	Description	Examples

Membership and Identity operators

Operator	Description	Examples
in	True if specified value is in the sequence	`“a” in “abc”`

Membership and Identity operators

Operator	Description	Examples
in	True if specified value is in the sequence	`"a" in "abc"`
not in	True if specified value not in the sequence	`"Z" not in "abc"`

Membership and Identity operators

Operator	Description	Examples
in	True if specified value is in the sequence	`"a" in "abc"`
not in	True if specified value not in the sequence	`"Z" not in "abc"`
is	True if two variables being compared are the same object	`"a" is "A"`

Membership and Identity operators

Operator	Description	Examples
in	True if specified value is in the sequence	`"a" in "abc"`
not in	True if specified value not in the sequence	`"Z" not in "abc"`
is	True if two variables being compared are the same object	`"a" is "A"`
is not	True if the two variables being compared are not the same object	`"A" is not "A"`

Class quiz I

Class quiz I

$x=5$

Class quiz I

```
x=5

y=10
```

Class quiz I

```
x=5

y=10

print (x ** 2 > 25 and y < 25)
```

Class quiz I

```
x=5

y=10

print (x ** 2 > 25 and y < 25)
```

False

Class quiz II

Class quiz II

```
print (2 * 4 ** 2 * 4)
```

Class quiz II

```
print (2 * 4 ** 2 * 4)
```

128

Protip

Protip

If in doubt, use parenthesis to scope the operations

Data structures

Data structures

Type	Description	Examples

Data structures

Type	Description	Examples
list	Ordered but mutable sequence of items	`[1,2,"3"],[], list(), [1,[1,2]]`

Data structures

Type	Description	Examples
list	Ordered but mutable sequence of items	`[1,2,"3"],[], list(), [1,[1,2]]`
dict	Key value pairs	`{":-)":"happy",":-(":"sad"}, {}, dict ()`

Data structures

Type	Description	Examples
list	Ordered but mutable sequence of items	`[1,2,"3"],[], list(), [1,[1,2]]`
dict	Key value pairs	`{":-)":"happy",":-(":"sad"}, {}, dict ()`
set	Unordered set of items	`{"a", 1, "abcd"}, {}, set ()`

Data structures

Type	Description	Examples
list	Ordered but mutable sequence of items	`[1,2,"3"],[], list(), [1,[1,2]]`
dict	Key value pairs	`{":-)":"happy",":-(":"sad"}, {}, dict ()`
set	Unordered set of items	`{"a", 1, "abcd"}, {}, set ()`
tuple	Ordered an immutable sequence of items	`(1,2), (1,2,3,4), (), tuple()`

Explore on your own

- Use of operations to add or remove items from list, dict, or set
- Sorting or reversing a list
- Merging two container variables such as concatenating two lists

Conditional Blocks

Conditional Blocks

Syntax

Conditional Blocks

Syntax

```
if <condition>:
  do something
elif <condition>:
  do something else
else:
  do something instead
```

Conditional Blocks

Syntax

```
if <condition>:
  do something
elif <condition>:
  do something else
else:
  do something instead
```

Example

Conditional Blocks

Syntax

```
if <condition>:
  do something
elif <condition>:
  do something else
else:
  do something instead
```

Example

```
class_year='senior'

if class_year == "freshmen":
  print ("Getting started")
elif class_year == "sophomore" or class_year == "junior":
  print ("Found my safety net")
else:
  print ("Get me out of here!")
```

Loops

Loops

Syntax

Loops

Syntax

```
for <var> in <container>:

  do something
```

Loops

Syntax

```
for <var> in <container>:

  do something
```

Example

Loops

Syntax

```
for <var> in <container>:

  do something
```

Example

```
fruits=[“grape”, “apple”, “orange”]

for fruit in fruits:

  print (fruit)
```

Functions

Functions

Syntax

Functions

Syntax

```
def <function name> (<args>):
    do something
    return <some value>
```

Functions

Syntax

```
def <function name> (<args>):
    do something
    return <some value>
```

Example

```
def is_even(number):
    return number % 2 == 0
```

Functions

Syntax

```
def <function name> (<args>):
    do something
    return <some value>
```

Example

```
def is_even(number):
    return number % 2 == 0
```

Explore on your own

- The use of `break` and `continue` keywords in loops
- `while` loop
- `range()` and `enumerate()` functions

Other stuff

- Python has support for many useful libraries that can be accessed by importing them (e.g., `import csv`)
- Support for many advanced features such as decorators, generators, lambda functions, etc
- One can use Python as a scripting language or as an object-oriented language.

In class

- 02_programming.ipynb
- 02_moby_dick.ipynb

(or see README.md on class Github)