Machine Learning and Programming in Python

Lecture for Master and PhD students

Chair of Data Science in Economics

Ruhr University Bochum

Summer semester 2024

Lecture 3

Object-oriented programming in Python

- Computer programs manipulate/ change data in the form of objects
- objects: data, variables, files, ...
- objects have types
  - scalar - indivisible
  - non-scalar — with internal structure, can be ordered/unordered and mutable/immutable
- We can do things with objects:
  - use variables to associate them with names
  - combine objects and operators to evaluate expressions
  - pass objects to functions
  - call methods on objects

Data types

Type		Scalar	Mutability	Order
integer	int	scalar	immutable	
float	float	scalar	immutable	
boolean	bool	scalar	immutable	
None	NoneType	scalar	immutable	
string	str	non-scalar	immutable	ordered
tuple	tuple	non-scalar	immutable	ordered
list	list	non-scalar	mutable	ordered
set	set	non-scalar	mutable	unordered
dictionary	dict	non-scalar	mutable	unordered

- String – sequence of characters/values (immutable, ordered)
- Tuple – sequence of characters/values (immutable, ordered)
- List – sequence of characters/values (mutable, ordered)
- Set – collection of unique characters/values (mutable, unordered)
- Dictionary – a set of key and character/value pairs (mutable, unordered)

Operators

- Arithmetic: +, -, *, /, ** exponent, % modulus, // floor division
- Boolean: and, or, not
- Comparison: ==, ! = does not equal, >, <=
- Assignment: = , +=, -=
- Membership: in

Indexing and Slicing (for ordered data types/ sequences: string, tuple, list):

- Strings, tuples and lists are indexed by numbers. Indexing in Python starts from 0!
- Use elem[index] to extract individual sub-elements
- Use elem[start:end] to get sub-sequence starting from index start and ending at index end-1
- Use elem[start:end:step] to get sub-sequence starting from index start, in steps of step, ending at index end-1

Functions

- function(object)
- examples for functions: 1. use the name of a data type to convert values/ characters to that data type, 2. the len() function returns the length of the element

Methods

- object.method()
- Use the dot . to link the method to the object

- S = 'Hello world'
  (This is a string (an object))
- S + '!'
  (+ is an operator. Operators combine objects)
- len(S)
  (This is a function. Objects are passed to functions.)
- S.upper()
  (This is a method. Methods are called on objects.)

Some Methods for Strings

S is a string, e.g. S = "hello world"

- S.upper() – change to upper case
- S.lower() – change to lower case
- S.capitalize() – capitalize the first word
- S.find("S1") – return the index of the first instance of S1
- S.replace("S1", "S2") – find all instances of S1 and change to S2
- S.strip(" ") – remove whitespace characters from the beginning and end of a string (useful when reading in from a file)
- S.split("S1") – split the string into a list, S1 as separator
- "S1".join("S2") – combine each element of the input sequence S2 (except for the last one) with S1, give out a single string

https://docs.python.org/3/library/stdtypes.html#string-methods

Mutability of objects (lists, sets, dicts)

Some Methods for Lists

L is a list, e.g. L = [1, 2, 3, 4, 5], e represents an element, i the index

- L.append(e) – extends by e at the end of the sequence
- L.insert(i, e) – inputs e at position i
- L.remove(e) – removes the element from L where L is e
- L.extend(L1) – extends L with L1
- L.pop(i) – removes element at position L[i]
- L.sort() – sorts elements in in L
- L.reverse() – reverses sorting of elements in L

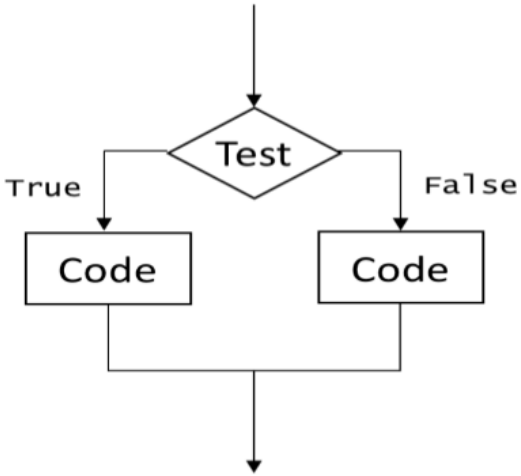
https : *//docs.python.org/3/library/stdtypes.html#mutable − sequence − types*

Control Flow

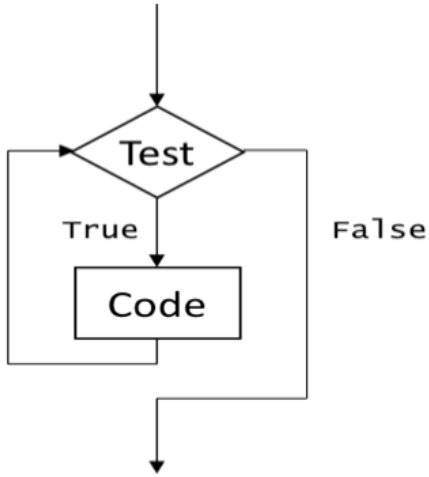
- Control flow is the order in which statements are executed or evaluated

- In Python, there are three main categories of control flow:
  - Branches (conditional statements) – execute only if some condition is met
  - Loops (iteration) – execute repeatedly
  - Function calls – execute a set of distant statements and return back to the control flow

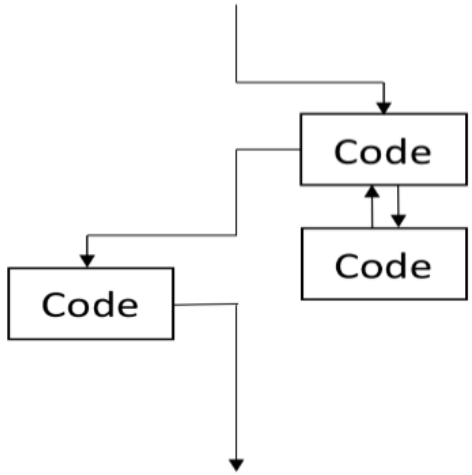
Conditional statement



Loop



Function call



Conditional Statements:

- if *Boolean expression*:
  block of code

- if *Boolean expression*:
  block of code
  else:
  block of code

Conditional Statements:

- if *Boolean expression*:
  block of code
  elif *Boolean expression*:
  block of code
  else:
  *block of code

Conditional

In [1]:
```
x = 5

if x > 0:
    print('Positive')
elif x < 0:
    print('Negative')
else:
    print('Zero')
```

```
Positive
```

Loops:

- while *Boolean expression*:
  block of code

- for *element* in *sequence*:
  block of code

While Loop

In [75]:
```
x = 0

while x < 5:
    x += 1
    print(x)
```

```
1
2
3
4
5
```

For Loop

In [78]:
```
for i in [1, 2, 3, 4, 5]:
    print(i)
```

```
1
2
3
4
5
```

In [5]:
```
for i in range(1, 6):
    print(i)
```

```
1
2
3
4
5
```

Functions

- Either built-in, for example len(), max(), range(), open(), etc.
- or user-defined by you, collaborators, or the open-source community

- Defining a function:

  def *functionname*(*list of parameters*):
    body of function

- Calling a function:

  functionname(*arguments*)

Function

In [73]:
```
def get_larger(x, y):

    if x > y:
        return x
    else:
        return y

m = get_larger(7, 9)
print(m)
```

```
9
```

Classes

- Object-oriented programming in Python
- A programming paradigm based on the concept of objects
- An object is a data abstraction that captures:
  - Internal representation (data attributes)
  - Interface for interacting with object (methods)

- Differences in using classes or using functions; remember: objects are passed on to functions
- Drawback of using programming code that has functions: might be dependent on the data type of the object. What if we change the type? Function might not work any more
- Advantage when using programming code that has classes: Methods, however, are tied to the object. Methods could be operated still, when change of type of object

- Objects have types (belong to classes)
- Objects also have a set of procedures for interacting with them (methods)

In [2]:
```
s = 'This is a string'
print(type(s))
print(s.upper())
```

```
<class 'str'>
THIS IS A STRING
```

Defining Classes in Python

- Data attributes — name, age
- Methods
  - bark()
  - compute_ageinmonths()
  - _init_() — called when a class is instantiated
  - _str_() — called by print() and str()
- Operations
  - Instantiation: cat1 = Cat('Garfield', 10) calls method _init_()
  - Attribute/method reference: cat1.compute_ageinmonths()

Classes vs. Objects

- Cat is a class
- cat1 is an instance of the class Cat; it is an object of type Cat

We have worked with classes and objects all the time before. Consider the data type "string". String is a class. In our example 'This is a string' is an object of type str ! We can attach methods to the string with a dot !

What you do, when coding classes, is to do more advanced programming in Python. While there are built-in classes in Python like str, list, tuple, etc., you have now learned how to programme your own classes in Python!

In [95]:
```
class Cat(object):

    def __init__(self, name, age):
        self.name = name
        self.age = age

    def bark(self):
        return "Mmmm!"

    def compute_ageinmonths(self):
        return 12*self.age

    def __str__(self):
        var = str(self.age)
        return "Cat " + self.name + ' is ' + var + " years old! Mmmm!"
```

In [96]:
```
cat1 = Cat("Garfield", 10)

print(cat1.name)

print(cat1.age)

print(cat1.bark())

print(cat1.compute_ageinmonths())

print(cat1)


cat2 = Cat("Tom", 5)

print(cat2.bark())
print(cat2)
```

```
Garfield
10
Mmmm!
120
Cat Garfield is 10 years old! Mmmm!
Mmmm!
Cat Tom is 5 years old! Mmmm!
```

Data Project

- Central bank communication
- Analysis of a speech by Christine Lagarde, President of the European Central Bank (ECB)
- Data Source of ECB Speeches:
  https : *//www.ecb.europa.eu/press/key/html/downloads.en.html*
- See Jupyter Notebook for analyses

- Clean the text data
- Extract a list of all the words that are used in the speech
- Find out the length of the speech (number of words)
- Find out the number of unique words

- Rewrite the programming code to use classes for the cleaning of the text data, the extraction of words, the determination of the length of the speech