Questions last lecture ?

Intelligent Control
Lecture 6 & 7 – Class

UNIVERSITY OF APPLIED SCIENCES

Hanshu Yu

16th Dec. 2022

De persoonlijke hogeschool

Contents

UNIVERSITY OF APPLIED SCIENCES

Warm-up problem - 1

type((1)) -> ?

type((‘1’)) -> ?

type((1,)) -> ?

Warm-up problem - 2

~~listA = [1, 2, 3, 4, 5, 6]~~
listB = [2, 7, 9, 1, 5, 11, 14, 2]

Get the unique elements in listB.

Warm-up problem - 3

listA = [1, 2, 3, 4, 5, 6]
listB = [2, 7, 9, 1, 5, 11, 14, 2]

Get the common elements in listA and listB.

What is class?

A blueprint for similar self-defined objects.

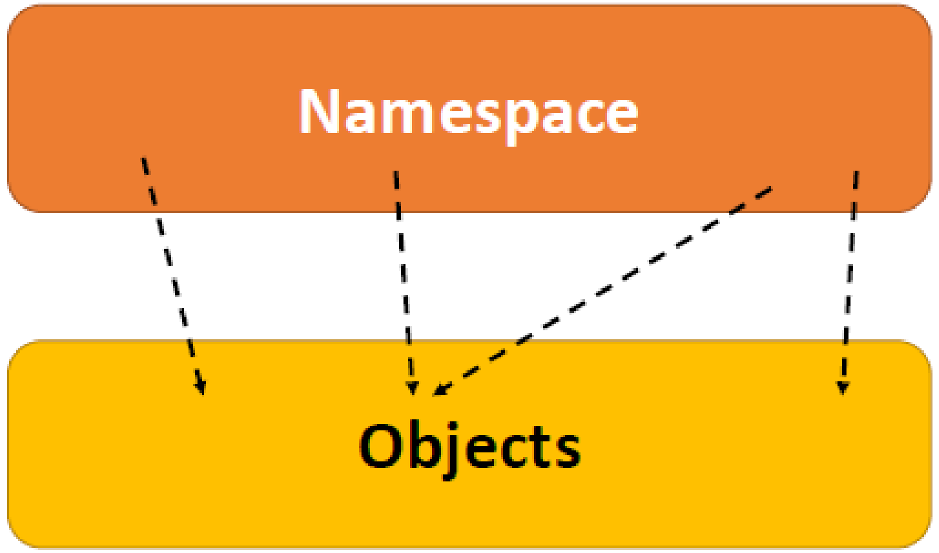
“Self-defined data types”

Python Objects

Python Object & Reference

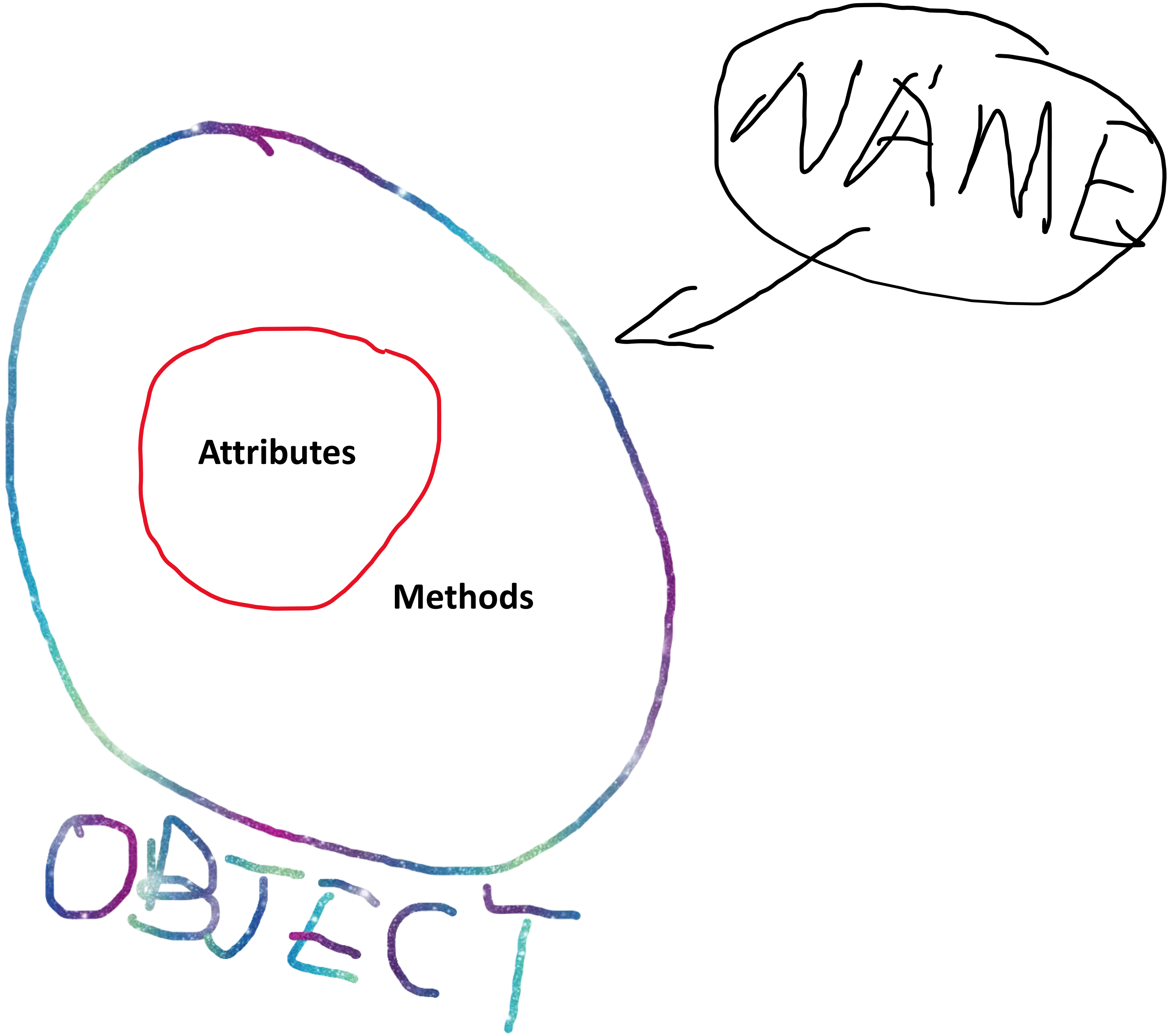
Mappings from namespace to objects,
one to one
or
one to many.

Every object have an
Unique id.
id() method
is operator



Python Objects

An object:

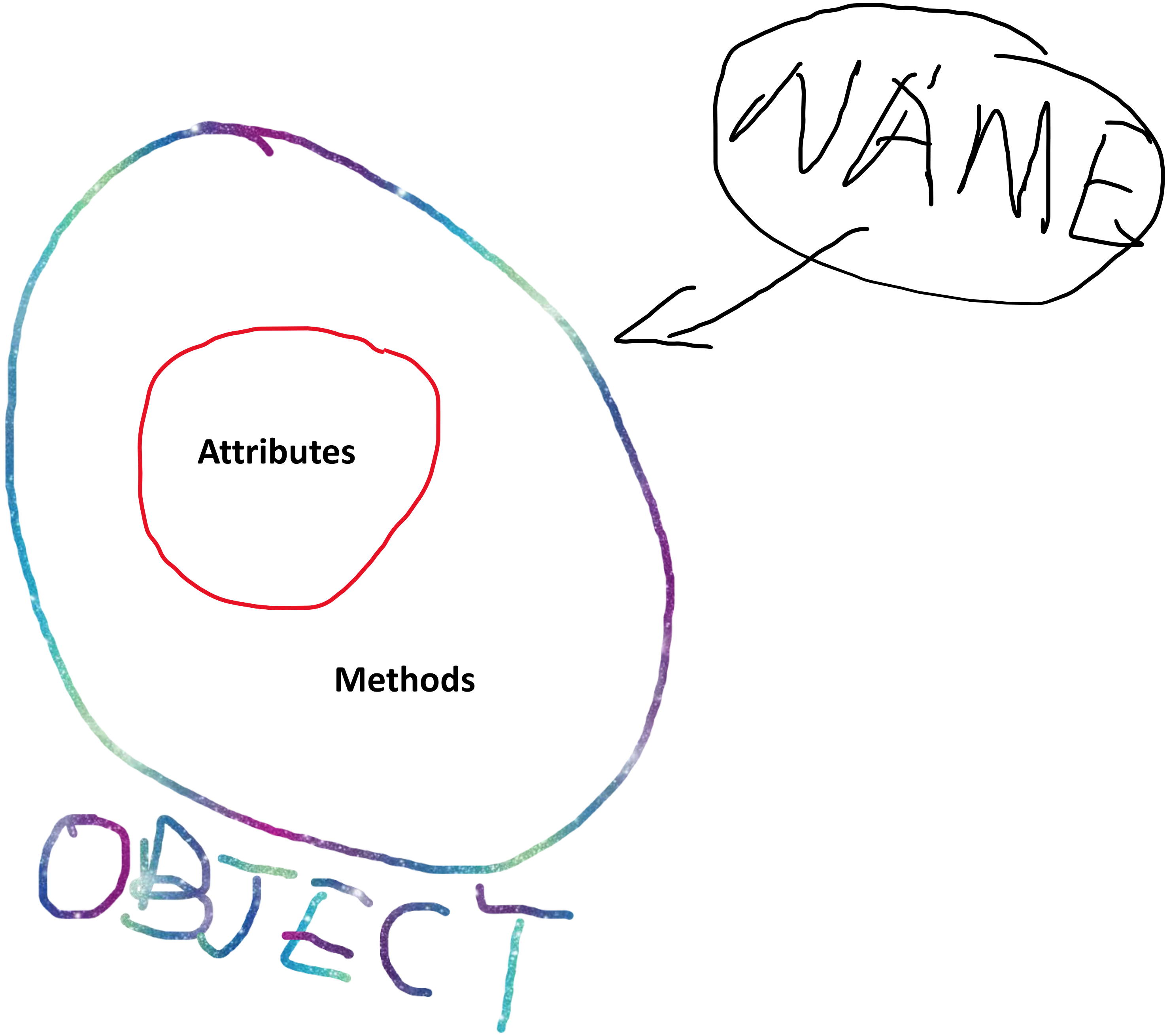


Python Objects

For example:

A point in Cartesian plane
in Euclidean geometry

(x,y)

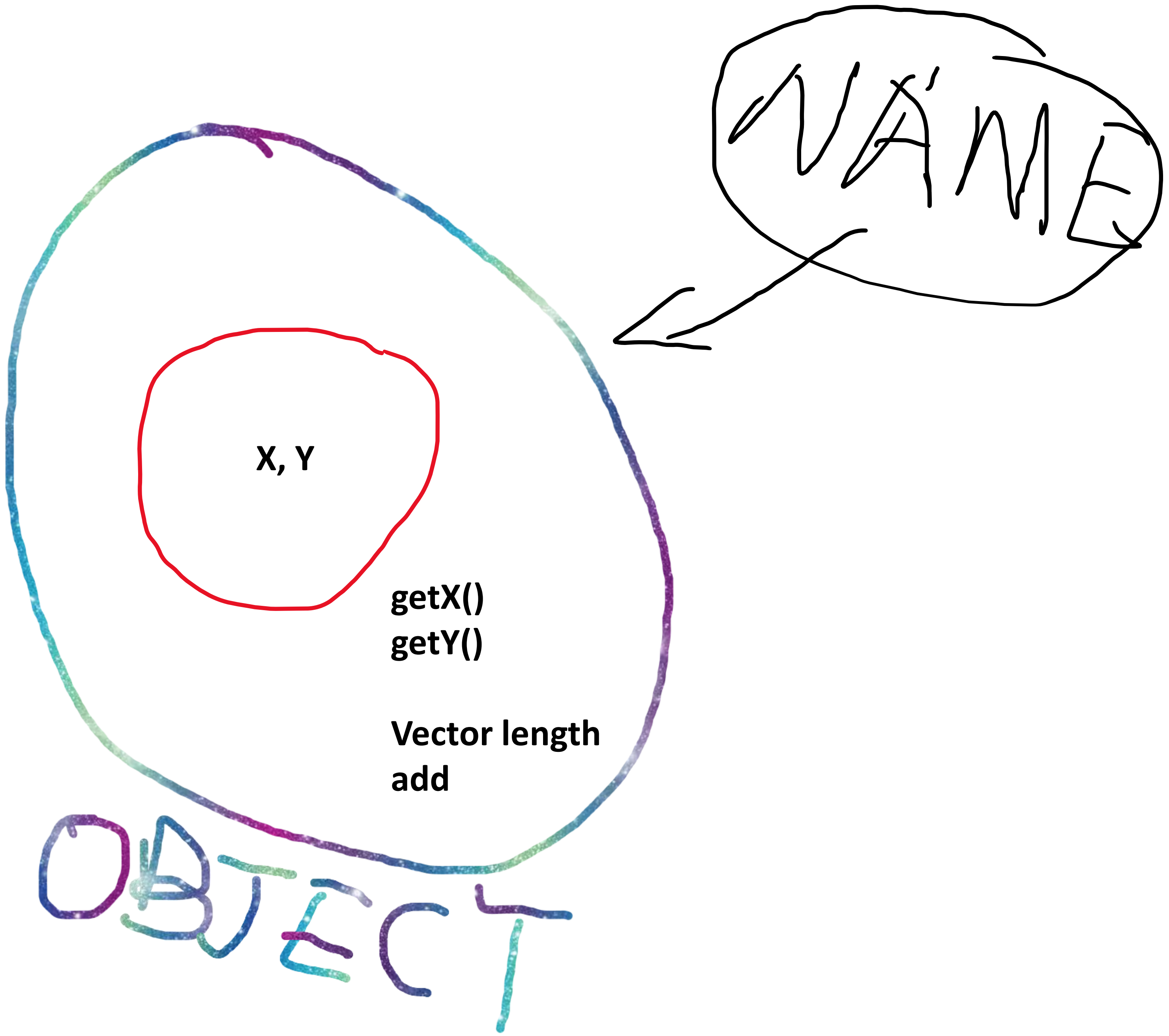


Python Objects

For example:

A point in Cartesian plane in Euclidean geometry

(x,y)



Python Class

Class definition： (~~minimum~~ component~~s~~)

- Give your class a name

class ClassIdentifier:

- Initializer method(constructor), automatically called when a new instance of your class is created.

    def __init__(self, Xini, Yini):

self: automatically set to reference the newly created object

Python Class

Continue…… class definition(some more)

- <u>Object to string:</u>
```
def __str__(self):
    return ‘Some string here’
```
- Documentation:
```
def __doc__(self):
    return ‘doc str’
```
- Your self-defined methods (these methods are locally used!)
```
def myselfdefinedfunc(self):
```

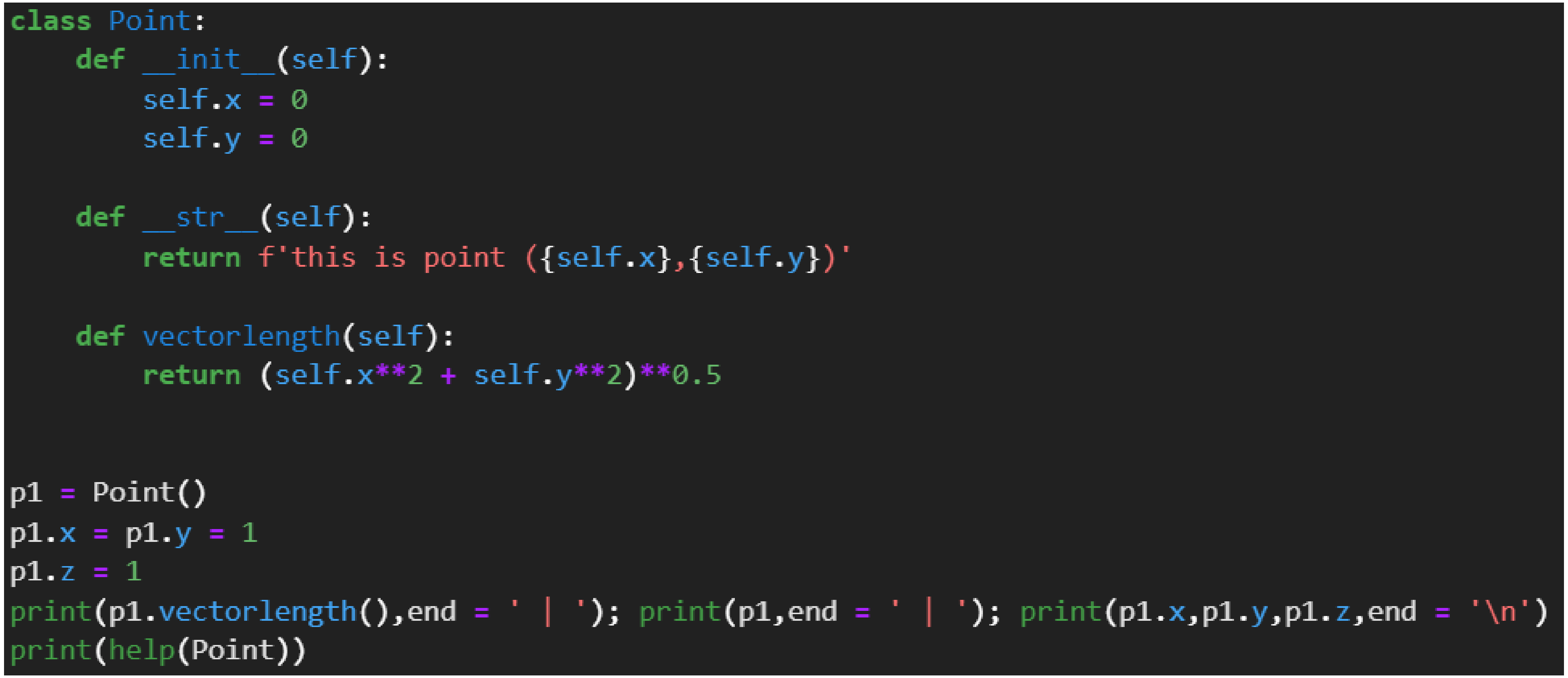
UNIVERSITY OF APPLIED SCIENCES

Create an instance of your self-defined class

example_instance = ClassIdentifier(0, 0)

Coding practice: define & use your own class object

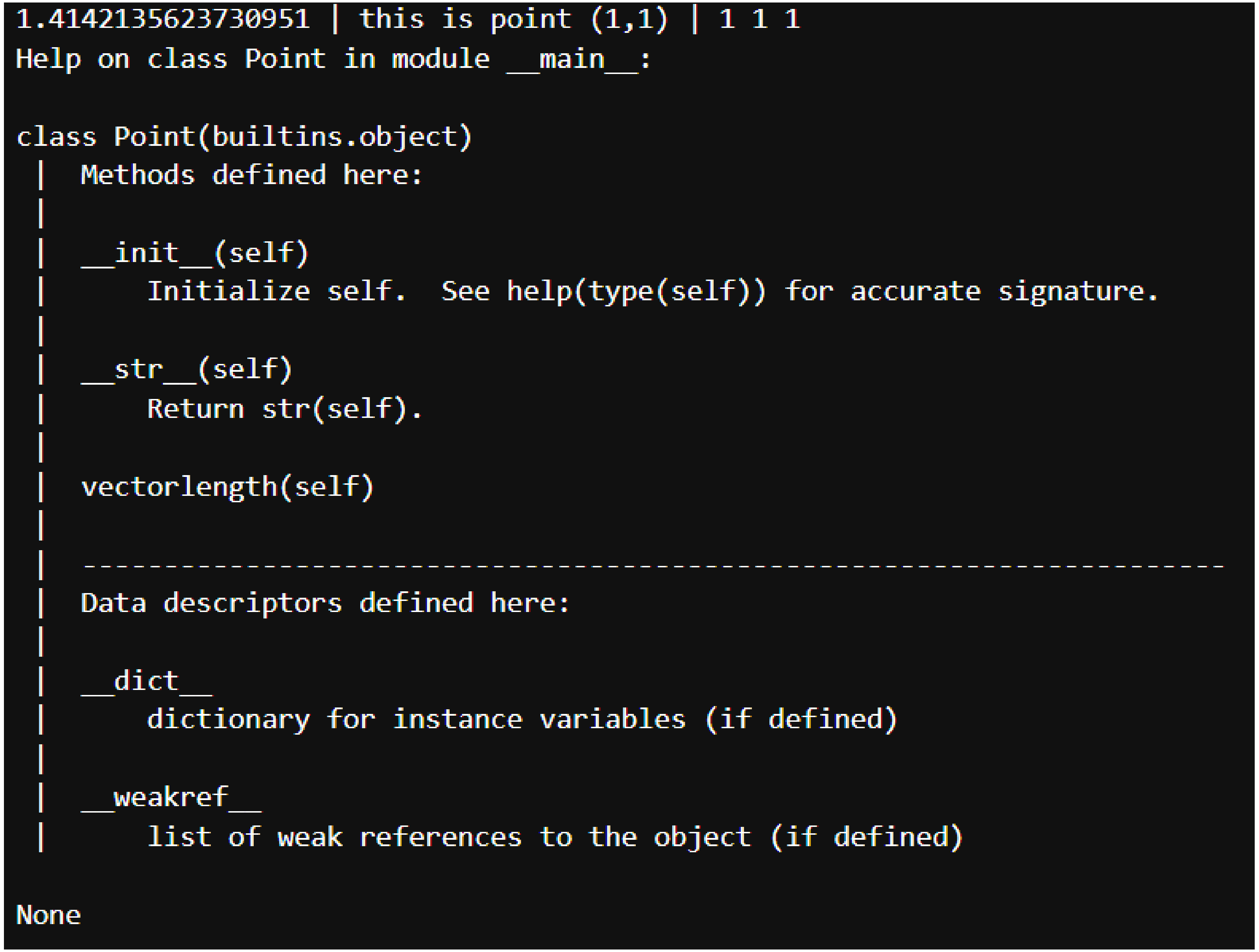
Python Class



```python
class Point:
    def __init__(self):
        self.x = 0
        self.y = 0

    def __str__(self):
        return f'this is point ({self.x},{self.y})'

    def vectorlength(self):
        return (self.x**2 + self.y**2)**0.5


p1 = Point()
p1.x = p1.y = 1
p1.z = 1
print(p1.vectorlength(),end = ' | '); print(p1,end = ' | '); print(p1.x,p1.y,p1.z,end = '\n')
print(help(Point))
```

UNIVERSITY
OF APPLIED SCIENCES

Python Class



```
1.4142135623730951 | this is point (1,1) | 1 1 1
Help on class Point in module __main__:

class Point(builtins.object)
 |  Methods defined here:
 |
 |  __init__(self)
 |      Initialize self.  See help(type(self)) for accurate signature.
 |
 |  __str__(self)
 |      Return str(self).
 |
 |  vectorlength(self)
 |
 |  ----------------------------------------------------------------------
 |  Data descriptors defined here:
 |
 |  __dict__
 |      dictionary for instance variables (if defined)
 |
 |  __weakref__
 |      list of weak references to the object (if defined)

None
```

Python Class Variables: instance variables and class variables

```
class ClassIdentifier:
        var1 = 1
        var2 = 2

        def __init__(self):
                self.var3 = 1
                self.var4 = 2
```

Python Class Variables: instance variables and class variables

```
class ClassIdentifier:
    var1 = 1
    var2 = 2

    def __init__(self):
        self.var3 = 1
        self.var4 = 2
```

$\xrightarrow{}$ shared

$\xrightarrow{}$ local

Inheritance (subclass for class)

BaseClass: books

SubClass: Novel
SubClass: Comics
SubClass: LectureNotes
……

Inheritance (subclass for class)

```
class Book:
        pass

class Novel(Book):
        pass

class Comics(Book):
        pass

class LectureNotes(Book):
        pass
```

2 useful built-in functions

isinstance(obj, classIdentifier)

issubclass(subclass, baseclass)

2 useful built-in functions

isinstance(obj, classIdentifier)

issubclass(subclass, baseclass)

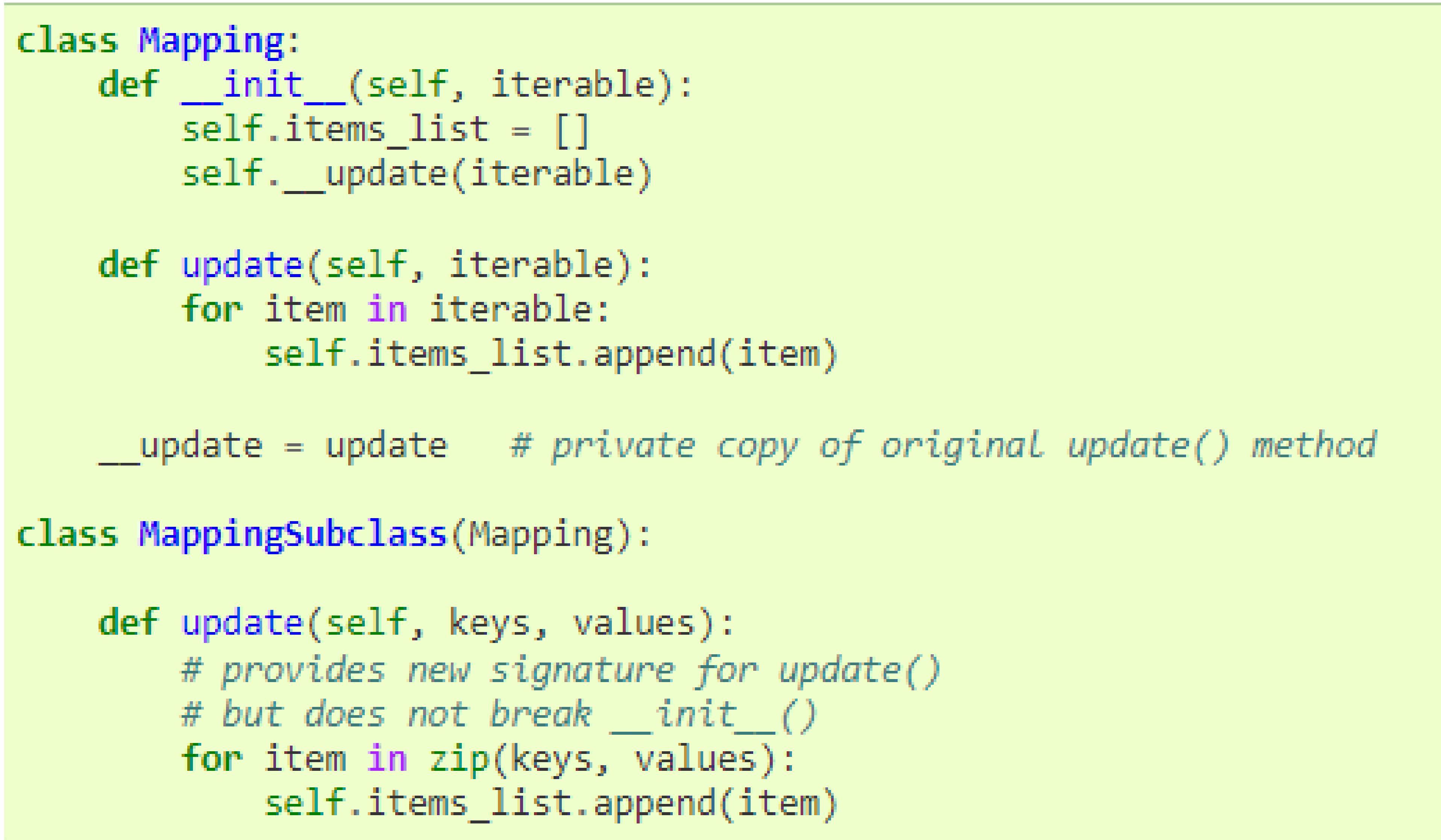
issubclass(bool, int) **-> ?**

Name-mangling

To avoid name clashes of names with those defined by subclasses.

```
class Example:

        def __init__(self):
                self.var = 0
                self._var = 0                  #Don’t use this
                self.__var = 0                 #underscores: at lease 2
                                               leading, at most one trailing
```

Name-mangling



```python
class Mapping:
    def __init__(self, iterable):
        self.items_list = []
        self.__update(iterable)

    def update(self, iterable):
        for item in iterable:
            self.items_list.append(item)

    __update = update   # private copy of original update() method

class MappingSubclass(Mapping):

    def update(self, keys, values):
        # provides new signature for update()
        # but does not break __init__()
        for item in zip(keys, values):
            self.items_list.append(item)
```

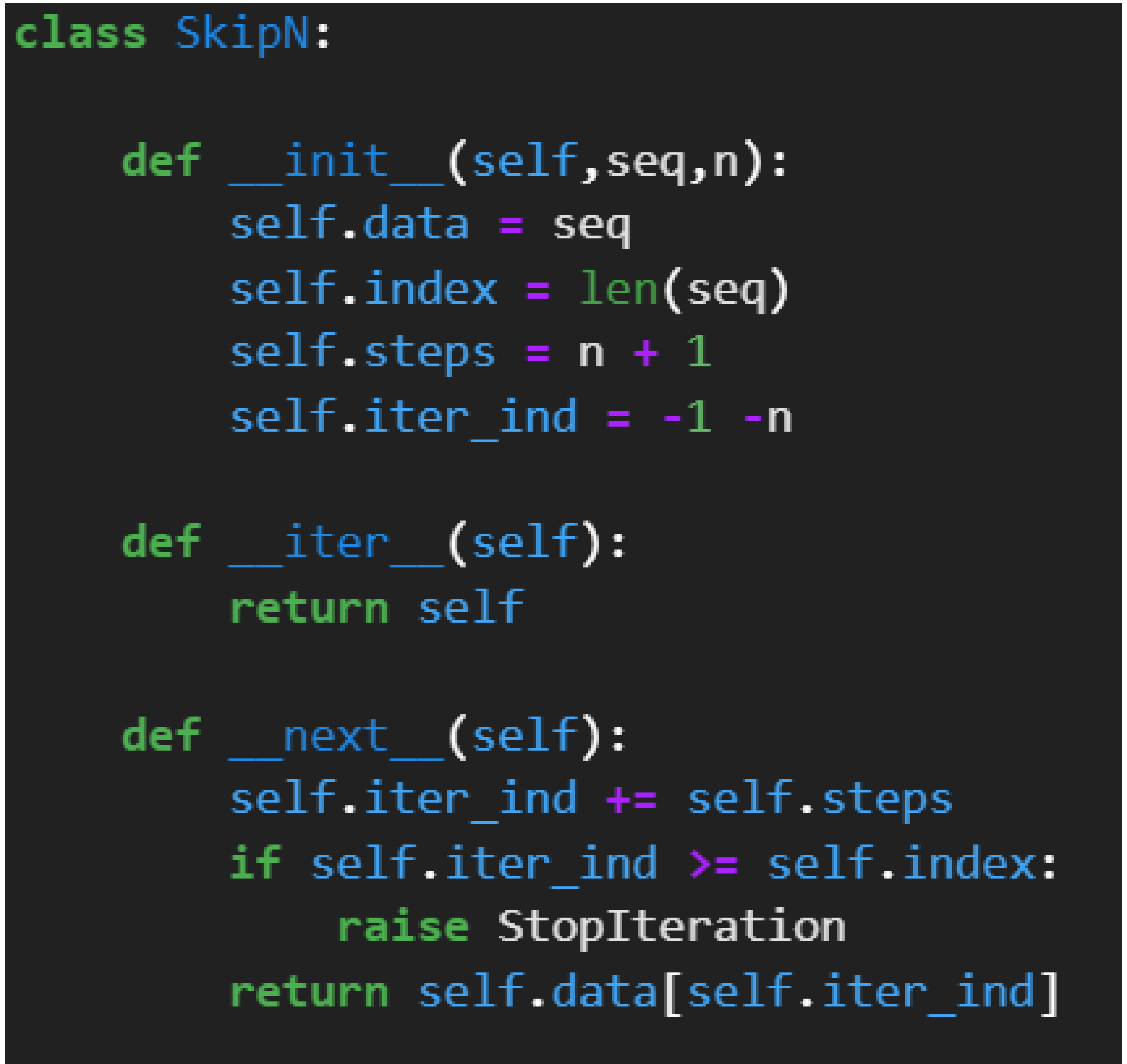
Build my own iterator with python class: iterators

iter()

For example:

```
itstr = ‘abcdefg’
example_iter = iter(itstr)
next(example_iter)                    #-> ‘a’
next(example_iter)                    #-> ‘b’
```

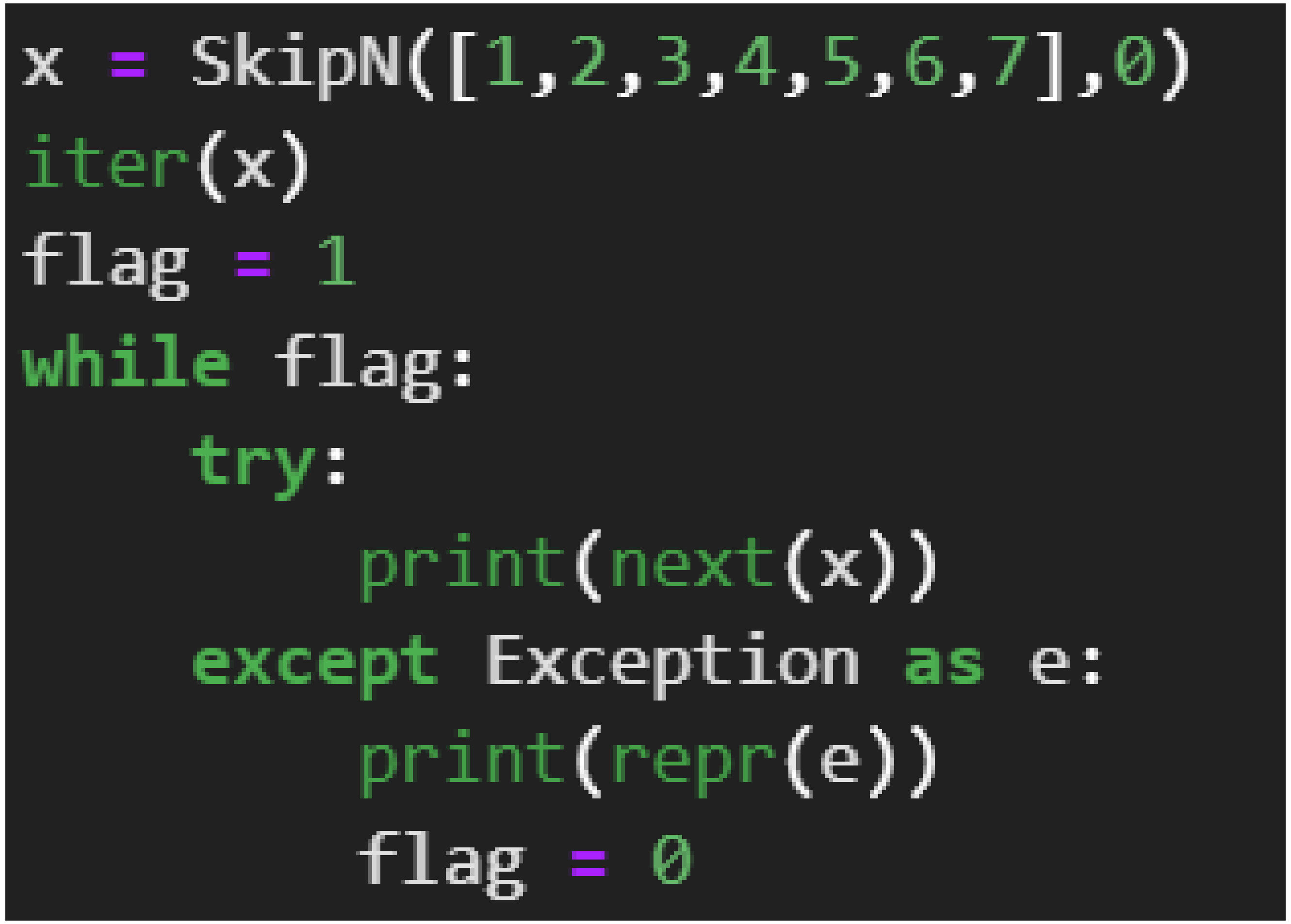
Build my own iterator with python class



```python
class SkipN:

    def __init__(self,seq,n):
        self.data = seq
        self.index = len(seq)
        self.steps = n + 1
        self.iter_ind = -1 -n

    def __iter__(self):
        return self

    def __next__(self):
        self.iter_ind += self.steps
        if self.iter_ind >= self.index:
            raise StopIteration
        return self.data[self.iter_ind]
```

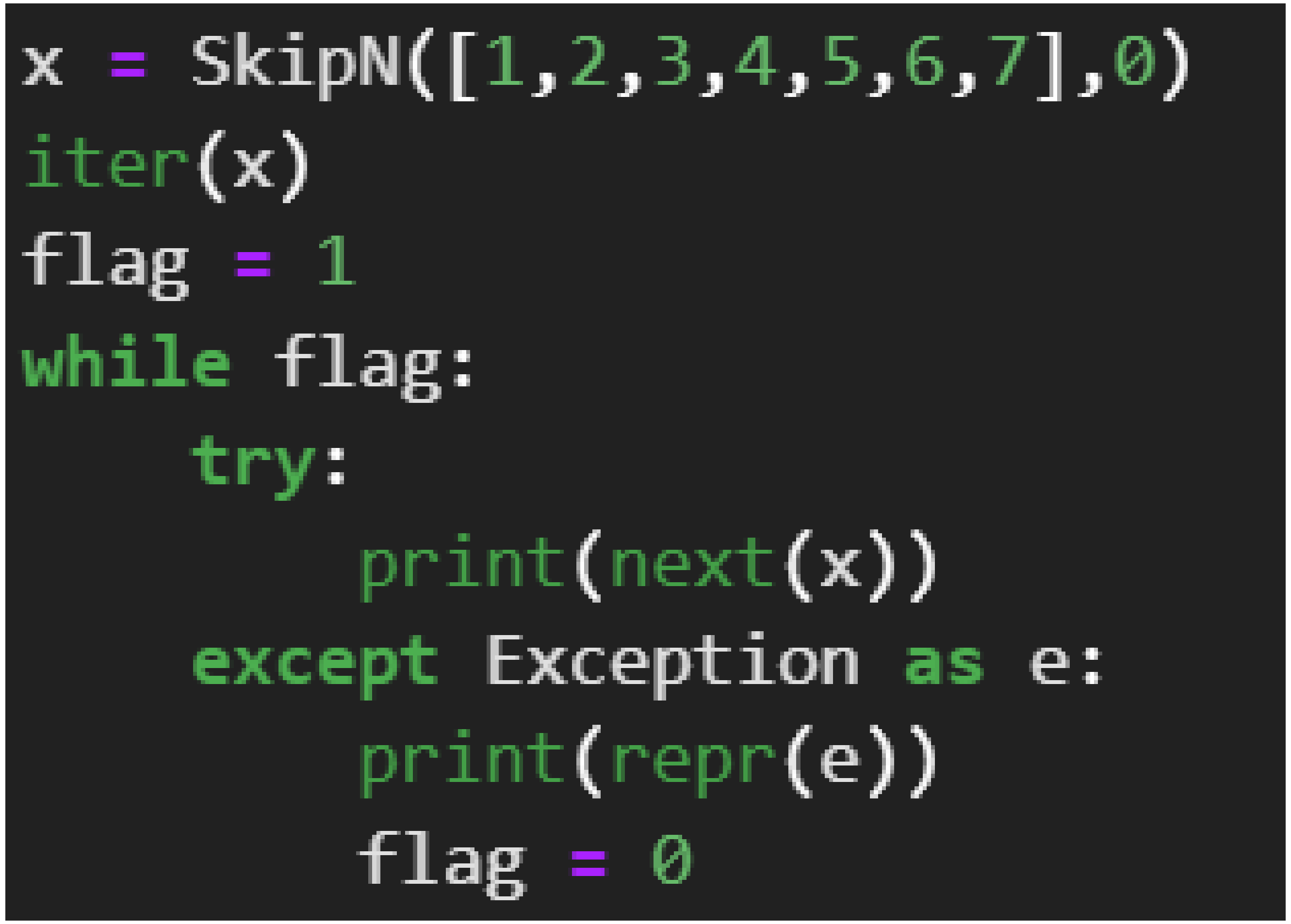
UNIVERSITY OF APPLIED SCIENCES

Build my own iterator with python class

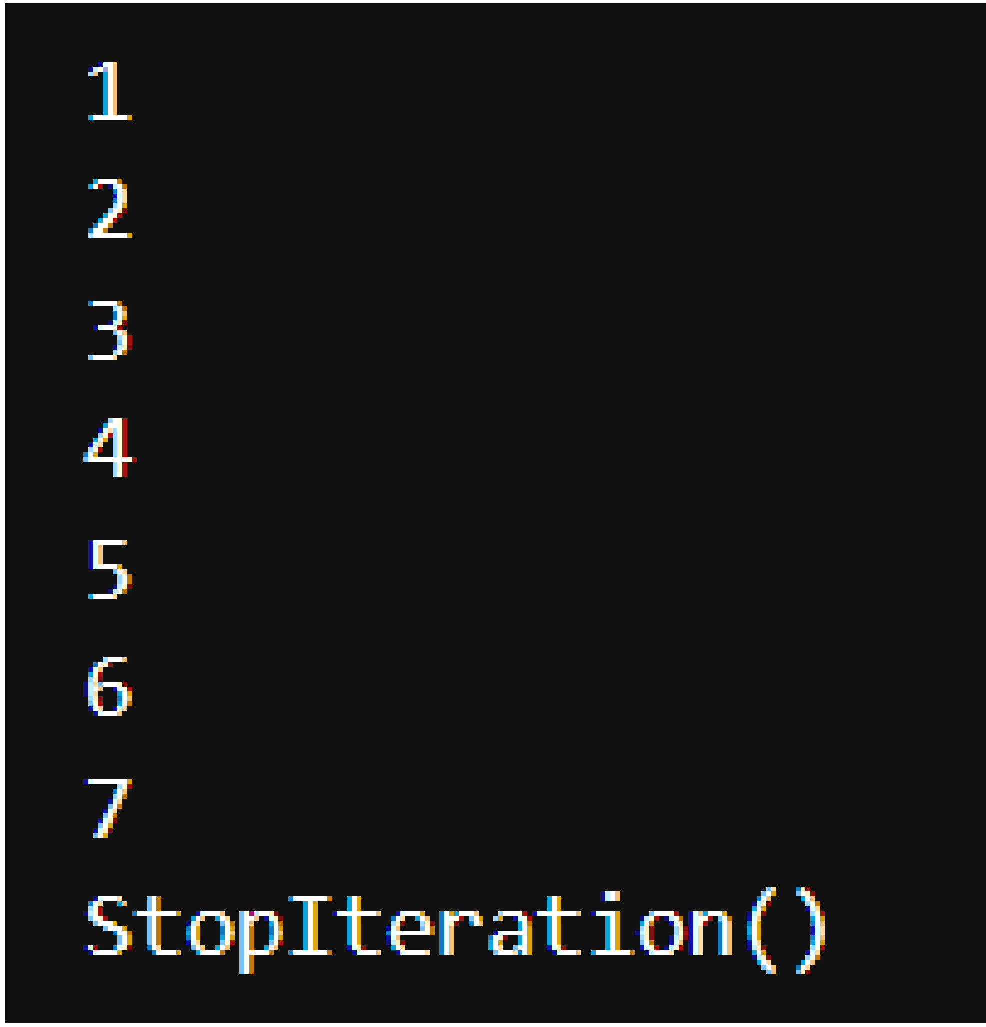


```
x = SkipN([1,2,3,4,5,6,7],0)
iter(x)
flag = 1
while flag:
    try:
        print(next(x))
    except Exception as e:
        print(repr(e))
        flag = 0
```

Build my own iterator with python class



```
x = SkipN([1,2,3,4,5,6,7],0)
iter(x)
flag = 1
while flag:
    try:
        print(next(x))
    except Exception as e:
        print(repr(e))
        flag = 0
```



```
1
2
3
4
5
6
7
StopIteration()
```

More inheritance name-mangling, iterator, generator

https://docs.python.org/3/tutorial/classes.html

One interesting practice

Create a program that generate APA reference texts.

Wrap-up

- ✓ Warm-up problems
- ✓ Introduction
- ✓ Python Objects – A more detailed view
- ✓ Create your first self-defined class
- ✓ Class Inheritance
- ✓ Name-mangling
- ✓ Create Python iterators using class objects
- ✓ Wrap-up

UNIVERSITY OF APPLIED SCIENCES

A few things to announce before the end of this lecture

- ✓ **Do as much coding practice as possible**

UNIVERSITY OF APPLIED SCIENCES