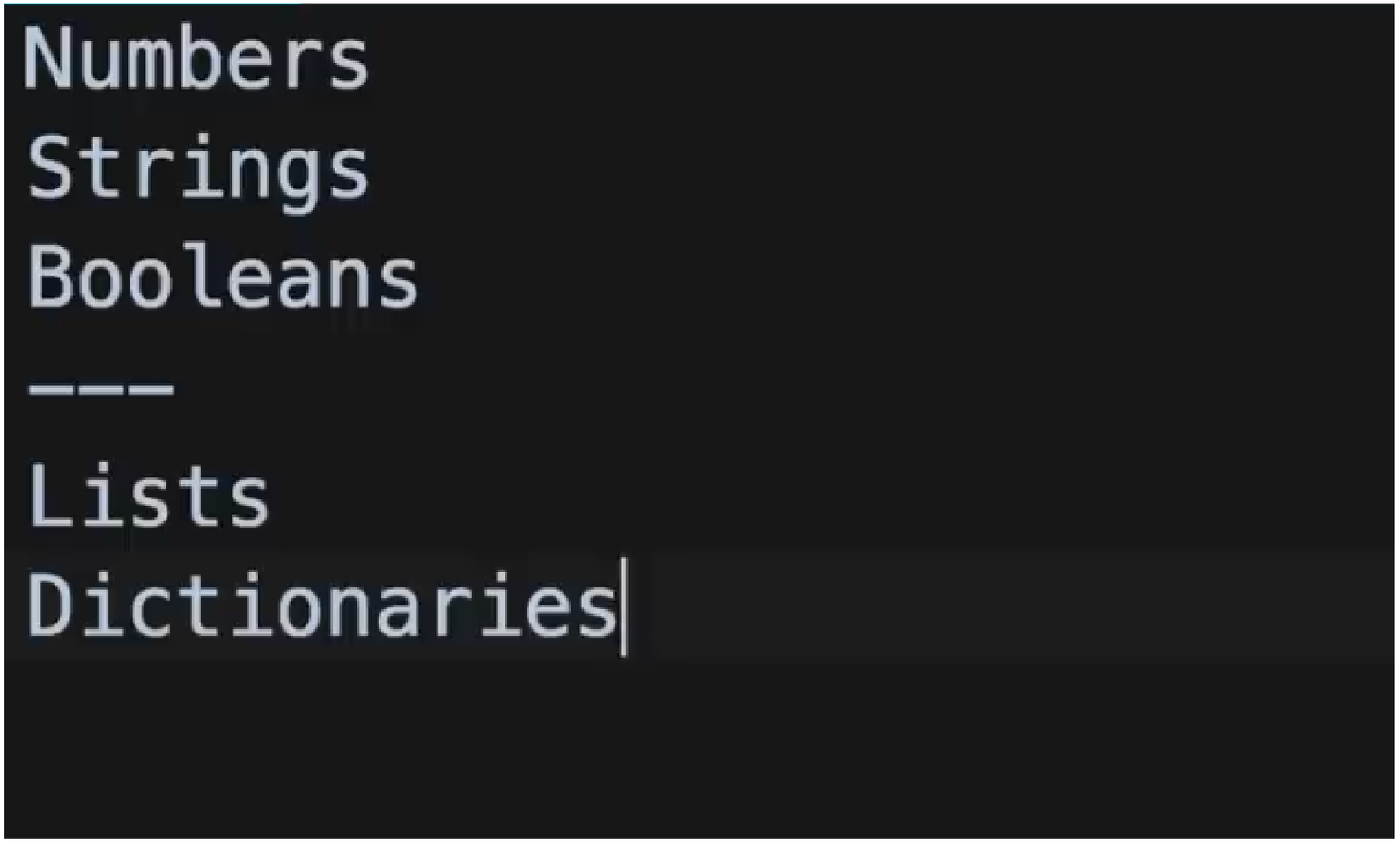
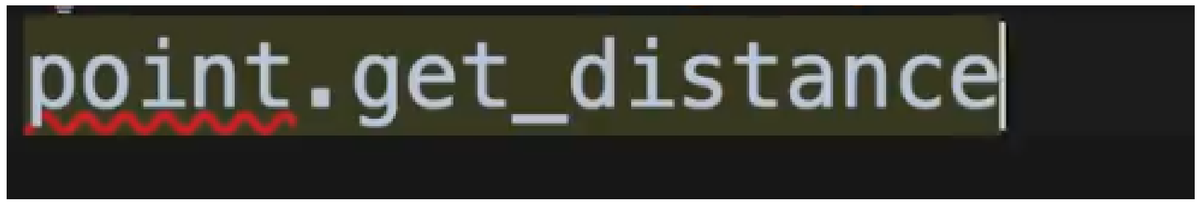
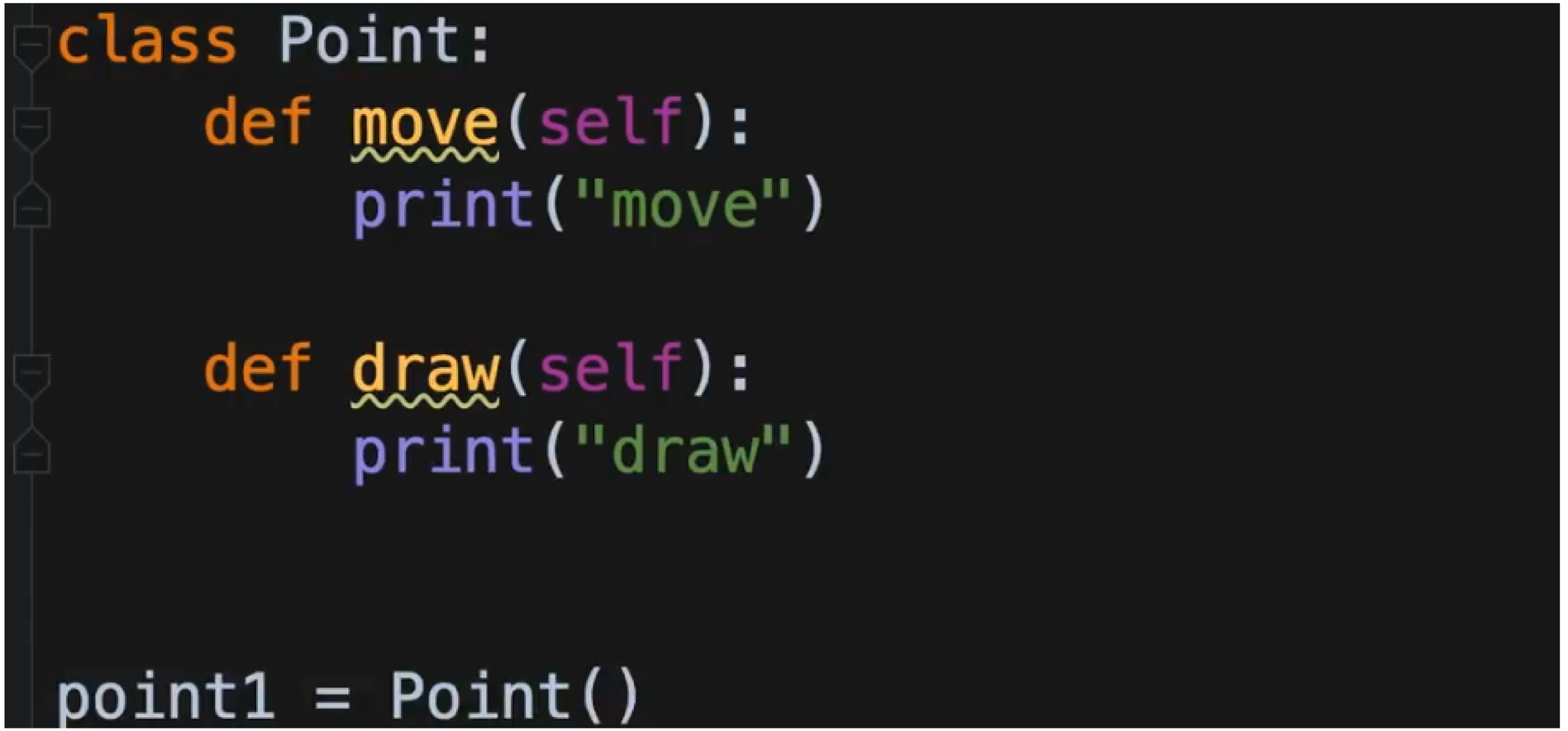
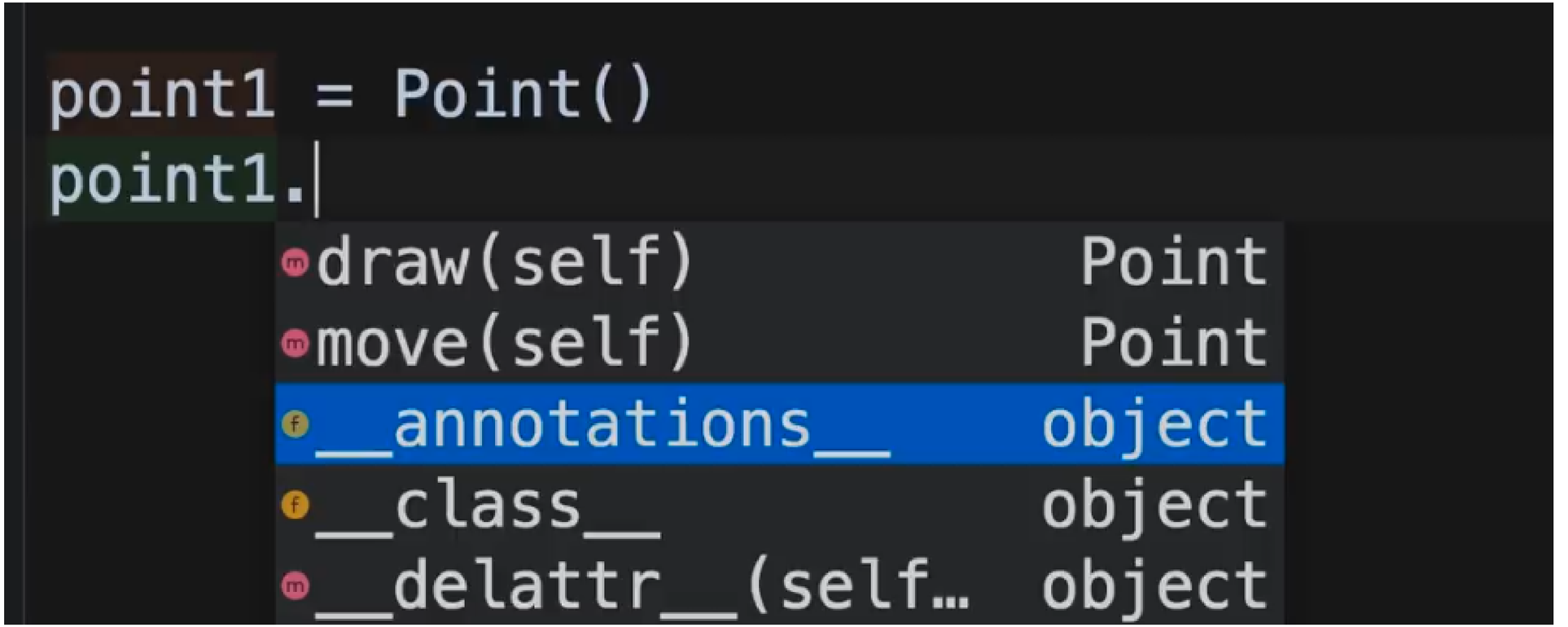
CLASSES

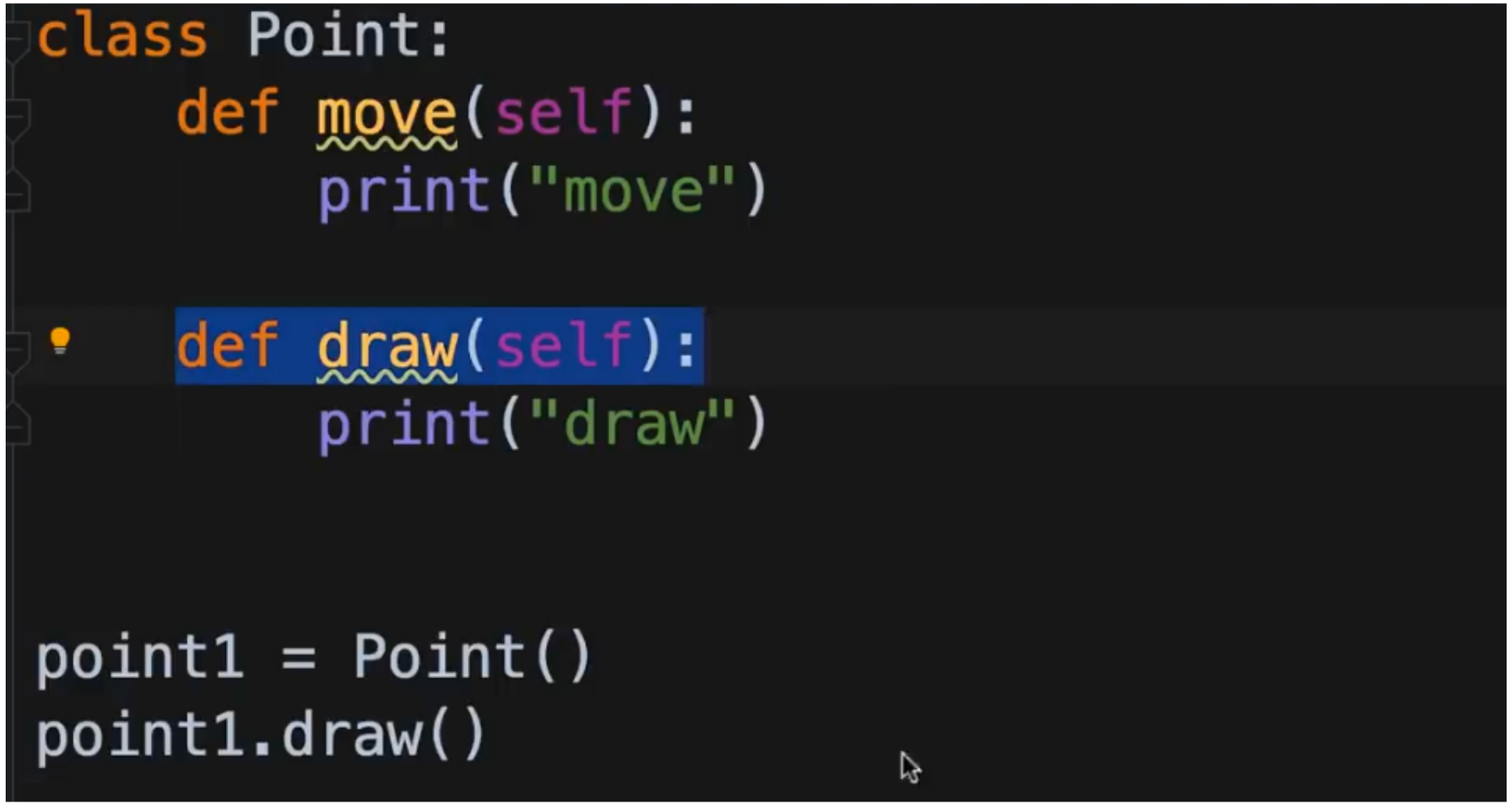

Numbers
Strings
Booleans

Lists
Dictionaries


point.get_distance


class Point:
def move(self):
print("move")
def draw(self):
print("draw")
point1 = Point()


point1 = Point()
point1.
draw(self) Point
move(self) Point
__annotations__ object
__class__ object
__delattr__(self… object

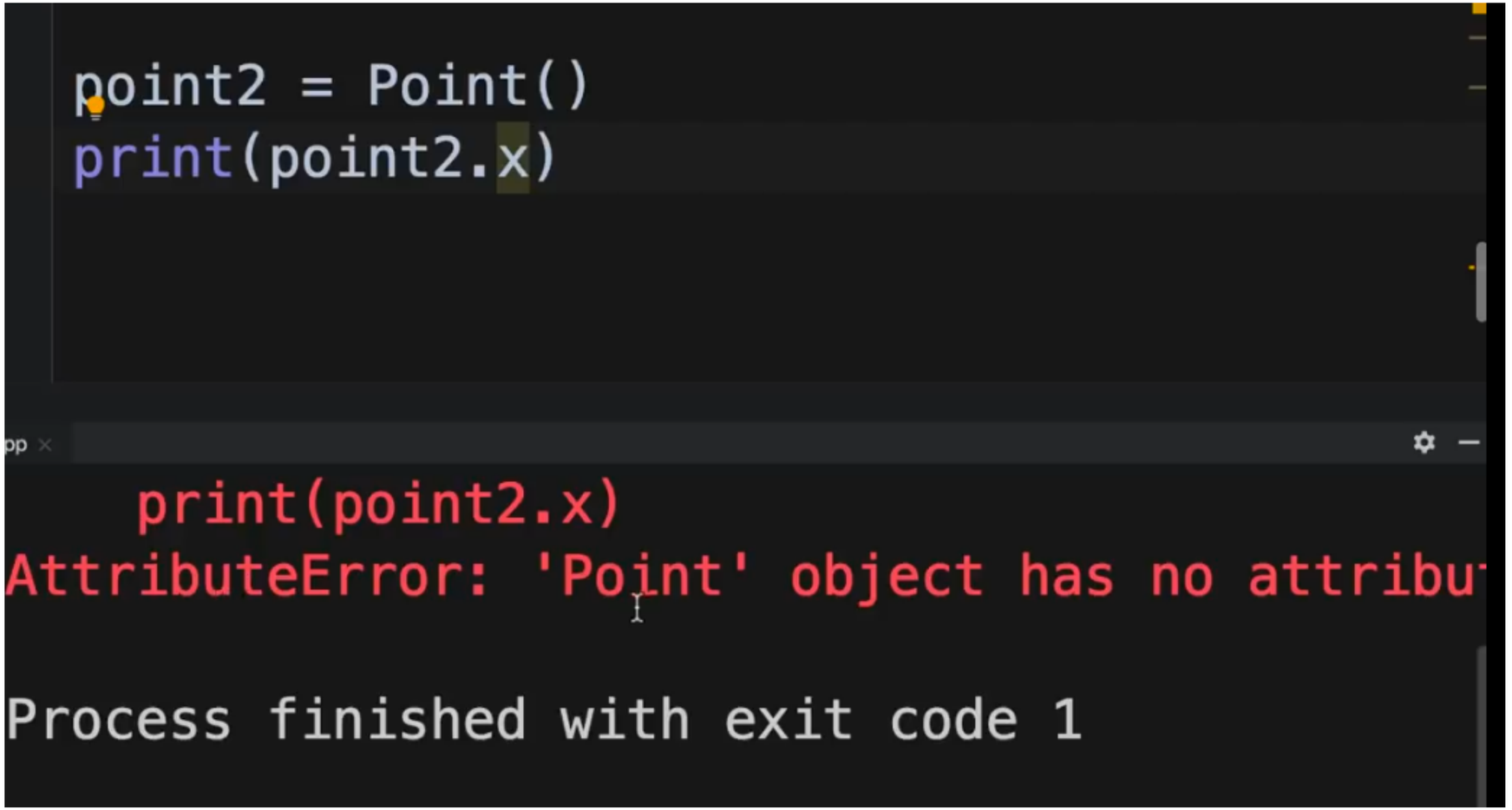

class Point:
def move(self):
print("move")
def draw(self):
print("draw")
point1 = Point()
point1.draw()

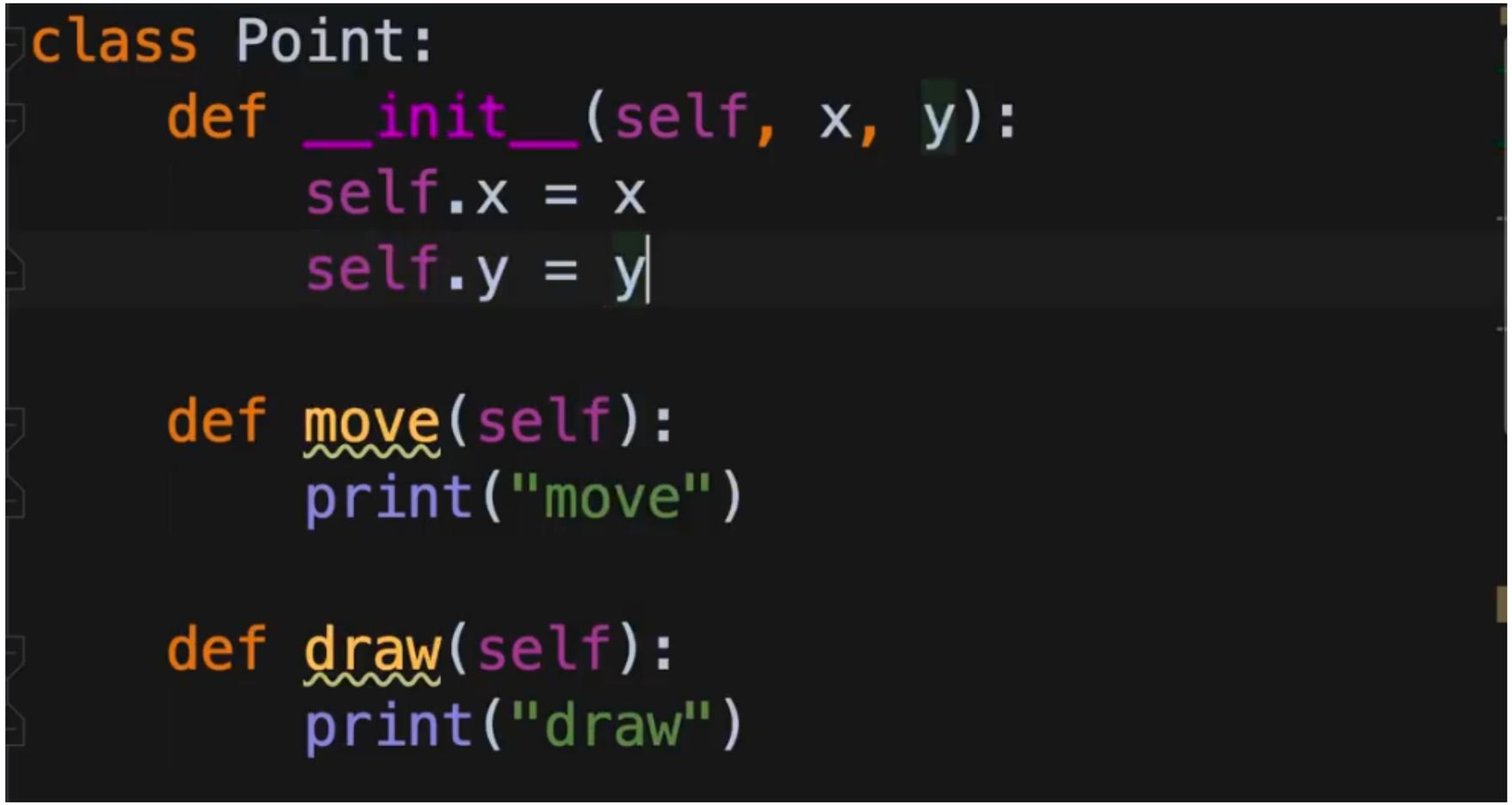
```
point1 = Point()
point1.x = 10
point1.y = 20
print(point1.x)
point1.draw()
```

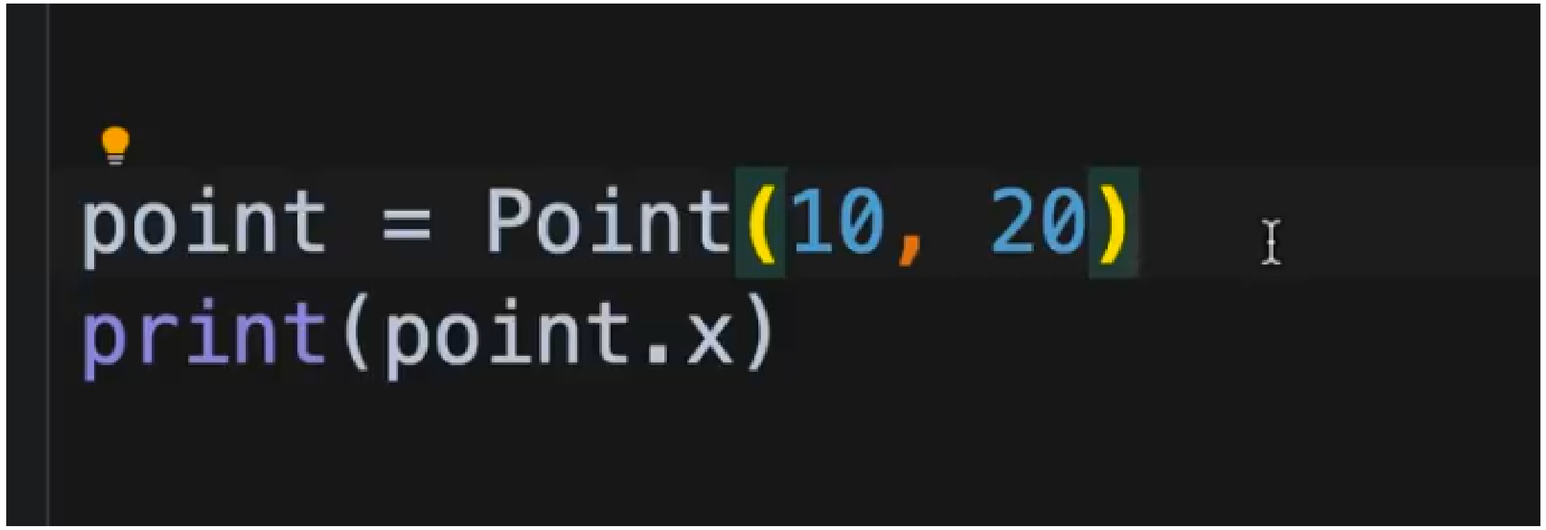
```
point2 = Point()
print(point2.x)
```

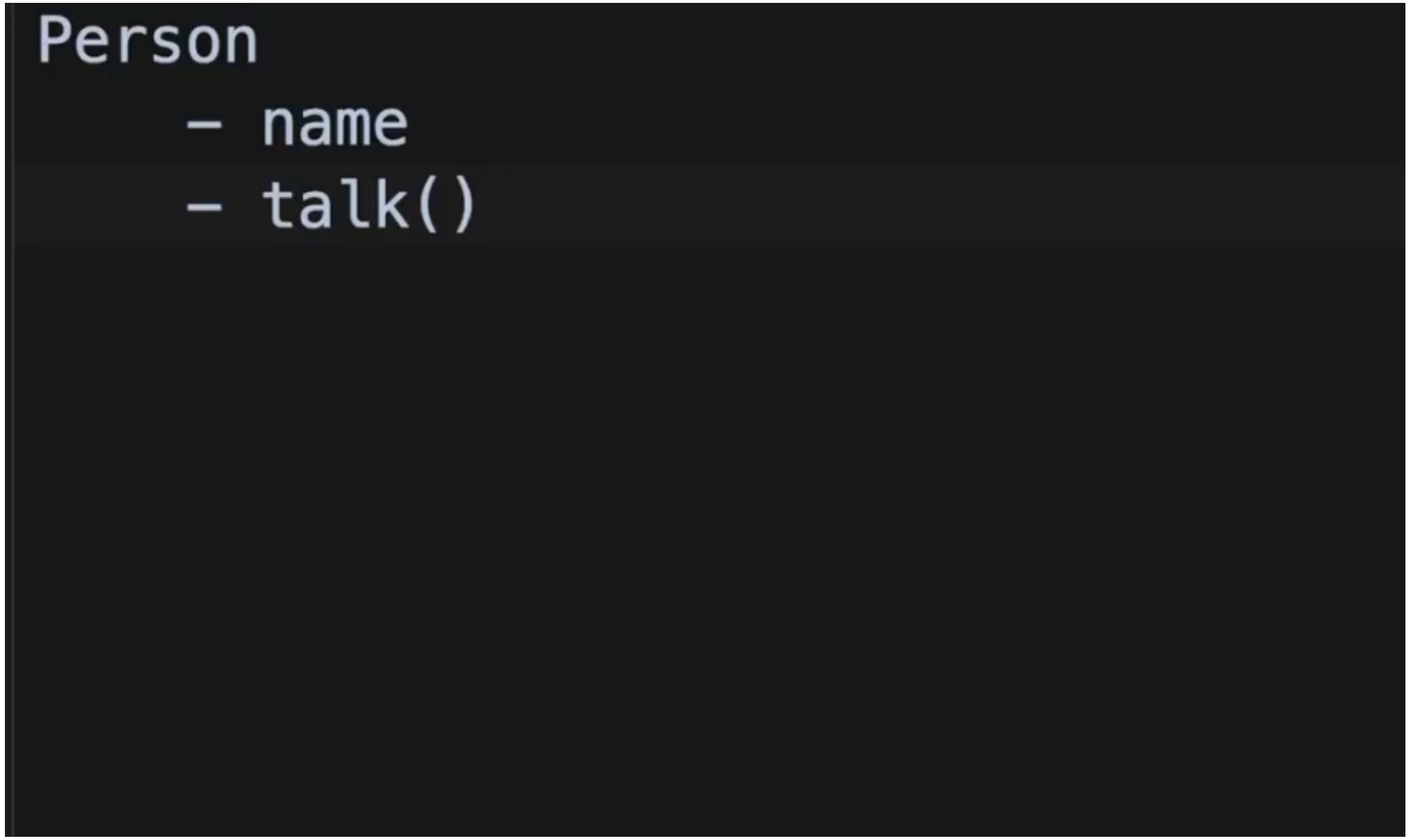
```
    print(point2.x)
AttributeError: 'Point object has no attribute 'x'

Process finished with exit code 1
```

CONSTRUCTORS


point2 = Point()
print(point2.x)
print(point2.x)
AttributeError: 'Point' object has no attribu
Process finished with exit code 1


class Point:
def __init__(self, x, y):
self.x = x
self.y = y
def move(self):
print("move")
def draw(self):
print("draw")


point = Point(10, 20)
print(point.x)


Person
- name
- talk()

Solution


class Person:
def __init__(self, name):
self.name = name
def talk(self):
print("talk")
john = Person("John Smith")
print(john.name)
john.talk()

```
bob = Person("Bob Smith")
bob.talk()
```

```
Hi, I am Bob Smith

Process finished with exit code 0
```

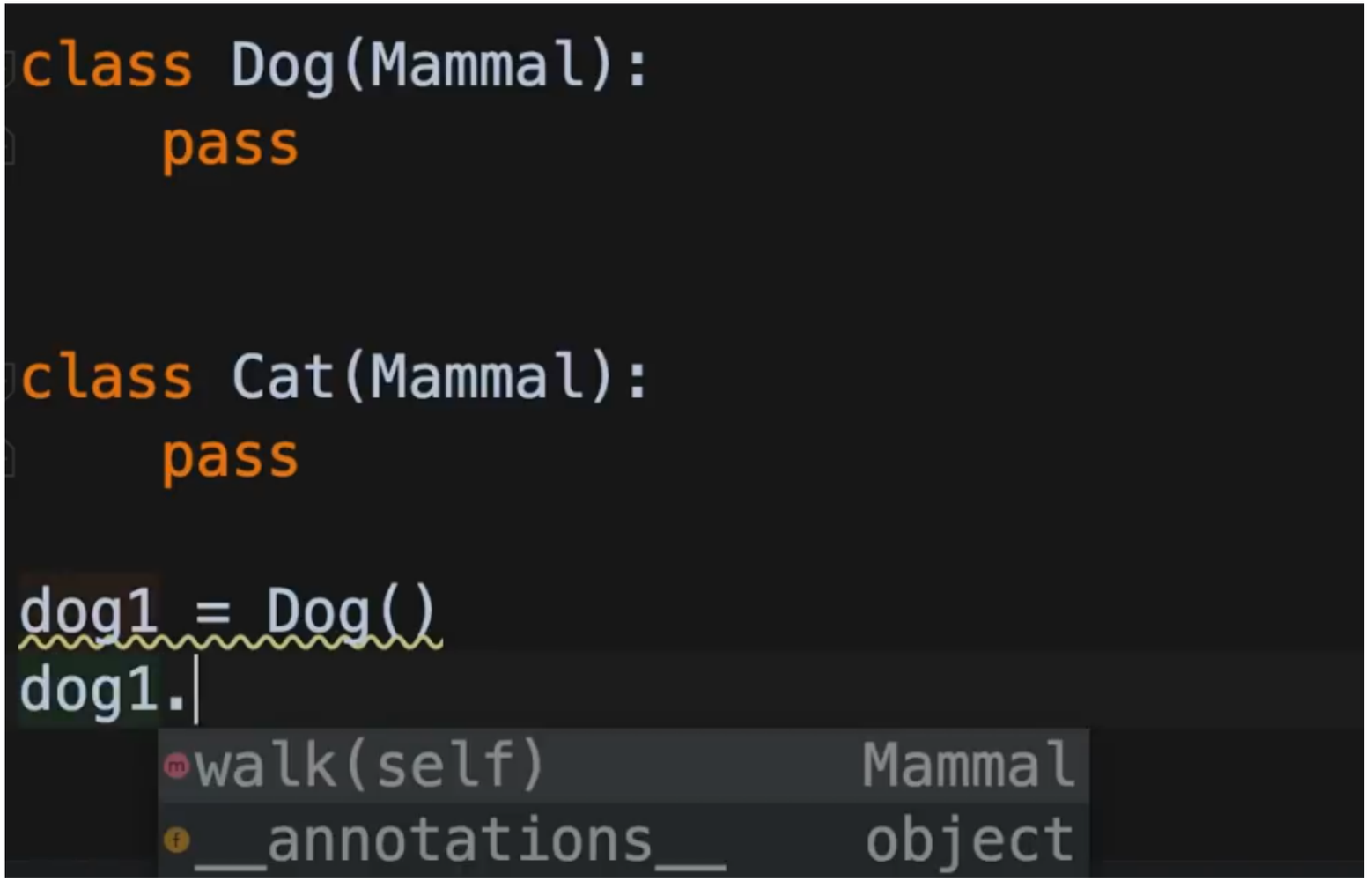
INHERITANCE

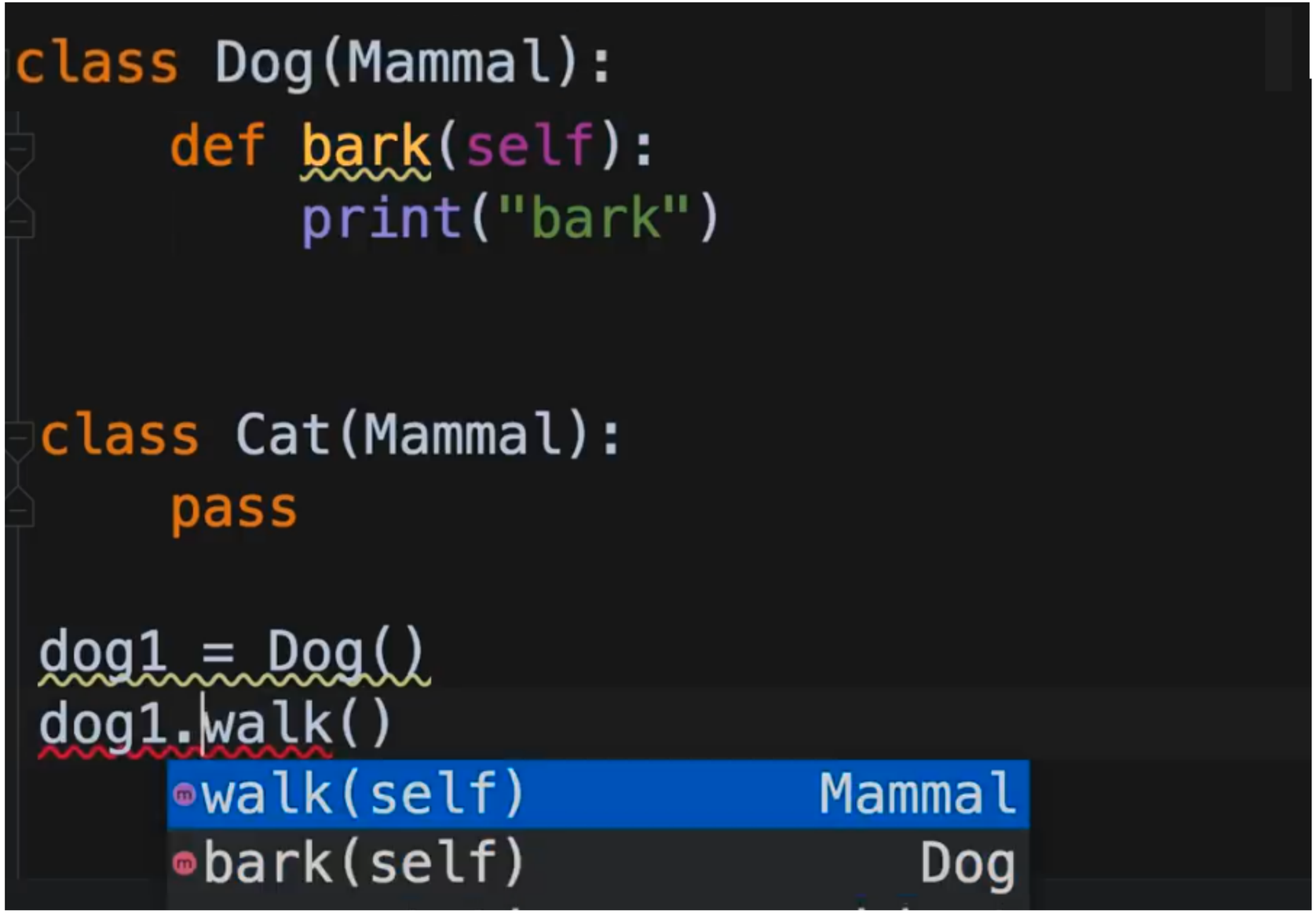

class Dog:
def walk(self):
print("walk")

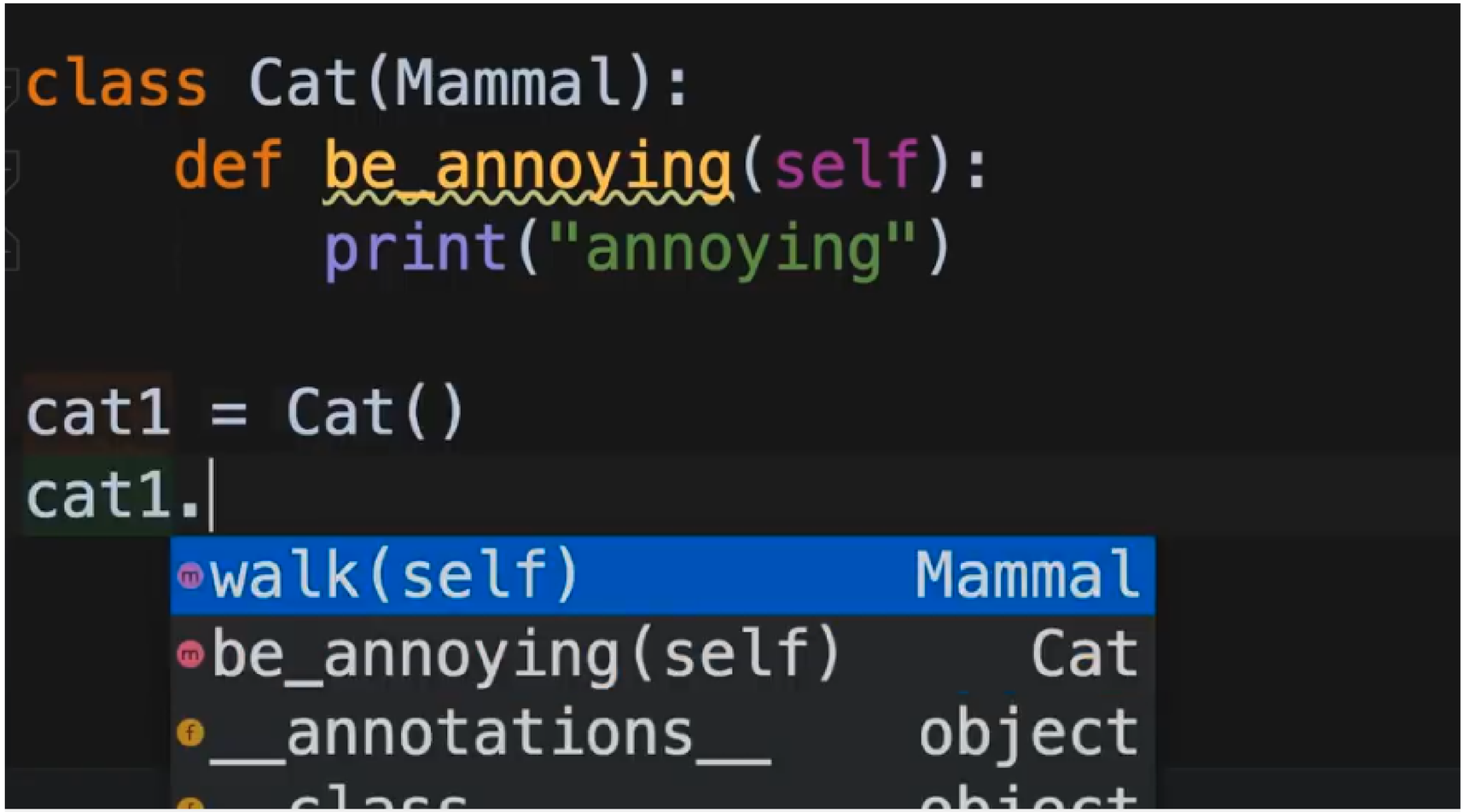

class Dog:
def walk(self):
print("walk")
class Cat:
def walk(self):
print("walk")


DRY


class Mammal:
def walk(self):
print("walk")
class Dog(Mammal):
class Cat:
def walk(self):
print("walk")


class Dog(Mammal):
pass
class Cat(Mammal):
pass
dog1 = Dog()
dog1.
walk(self) Mammal
__annotations__ object


class Dog(Mammal):
def bark(self):
print("bark")
class Cat(Mammal):
pass
dog1 = Dog()
dog1.walk()
walk(self) Mammal
bark(self) Dog

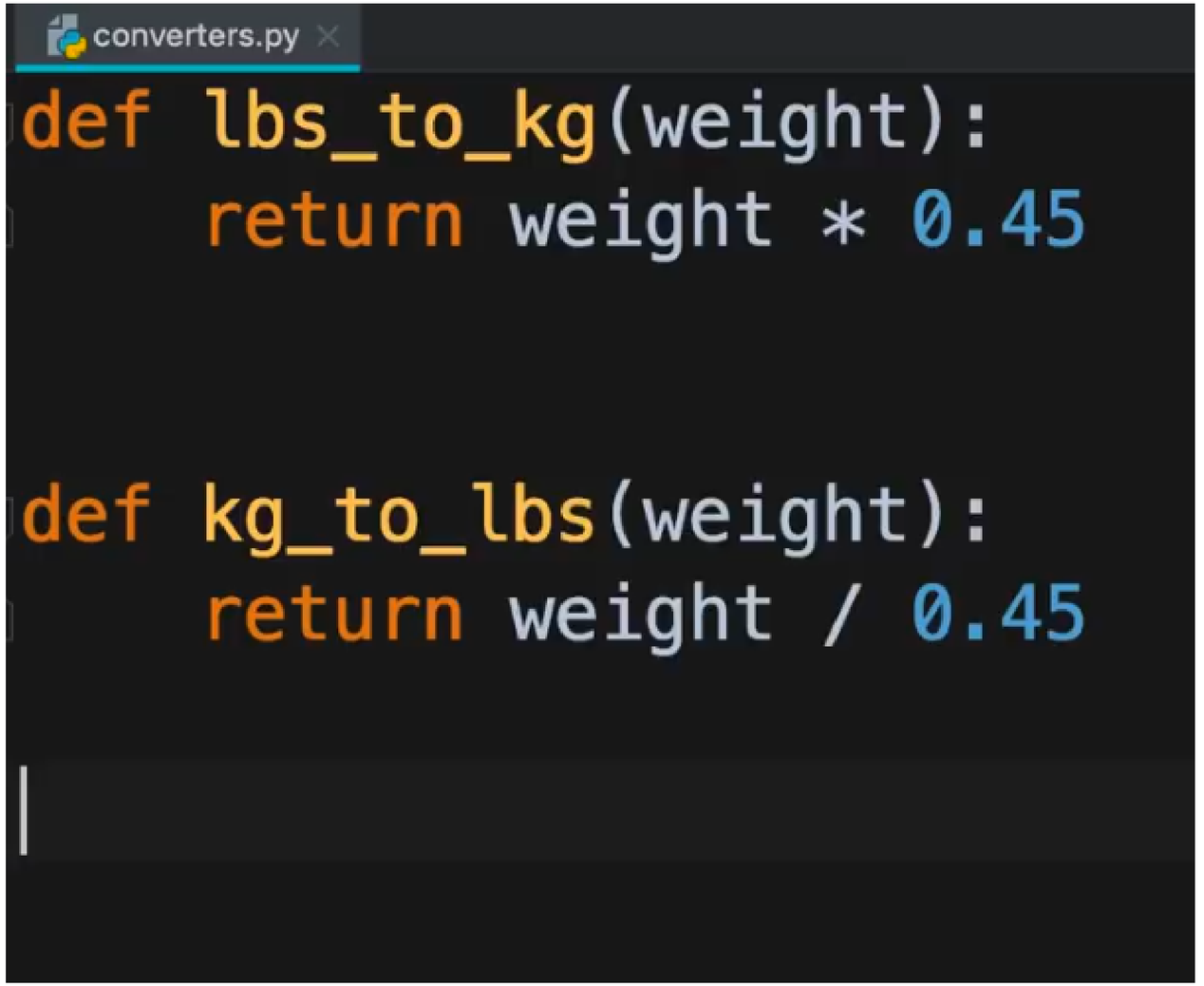

class Cat(Mammal):
def be_annoying(self):
print("annoying")
cat1 = Cat()
cat1.
walk(self) Mammal
be_annoying(self) Cat
__annotations__ object

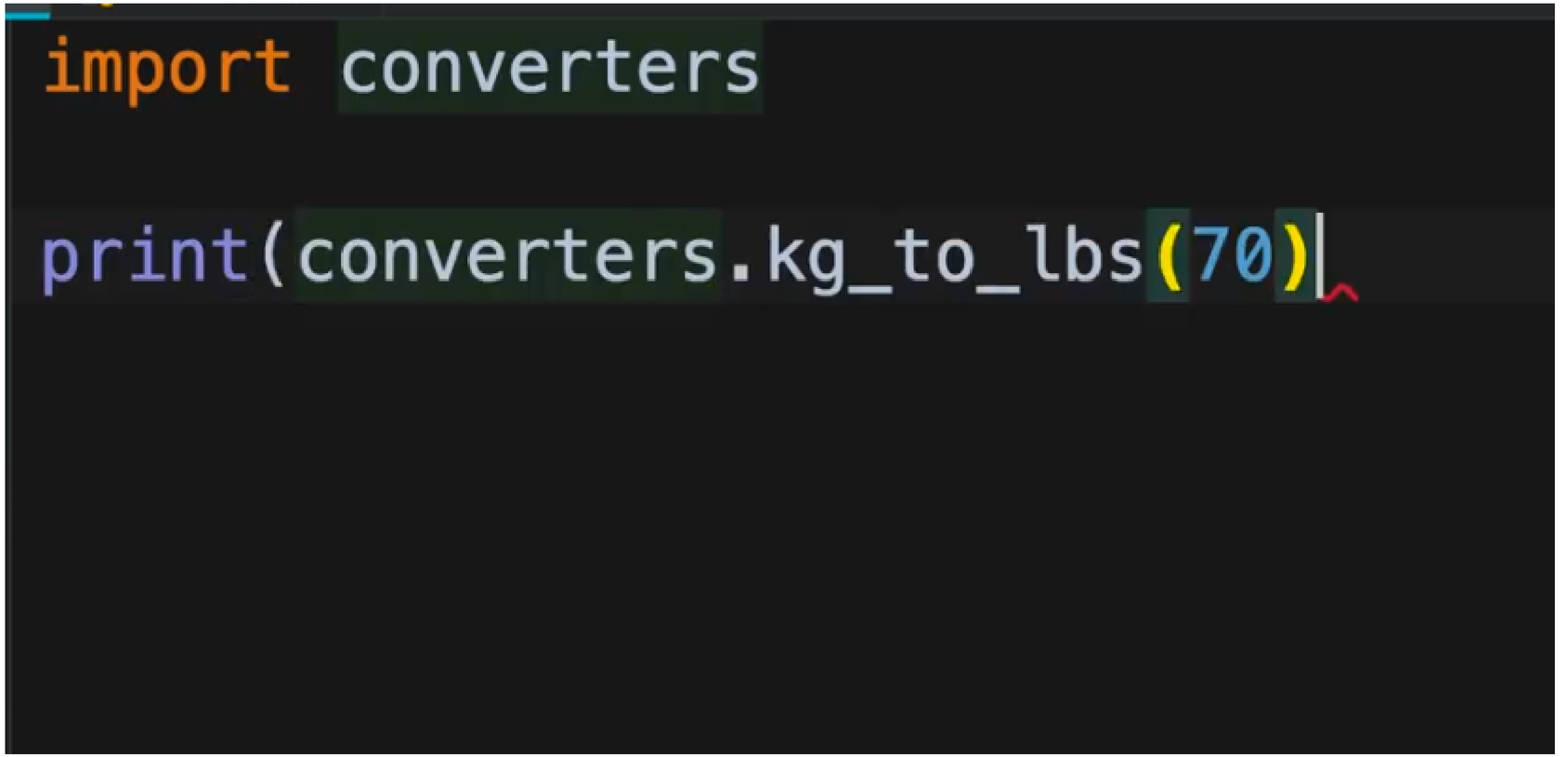
MODULES

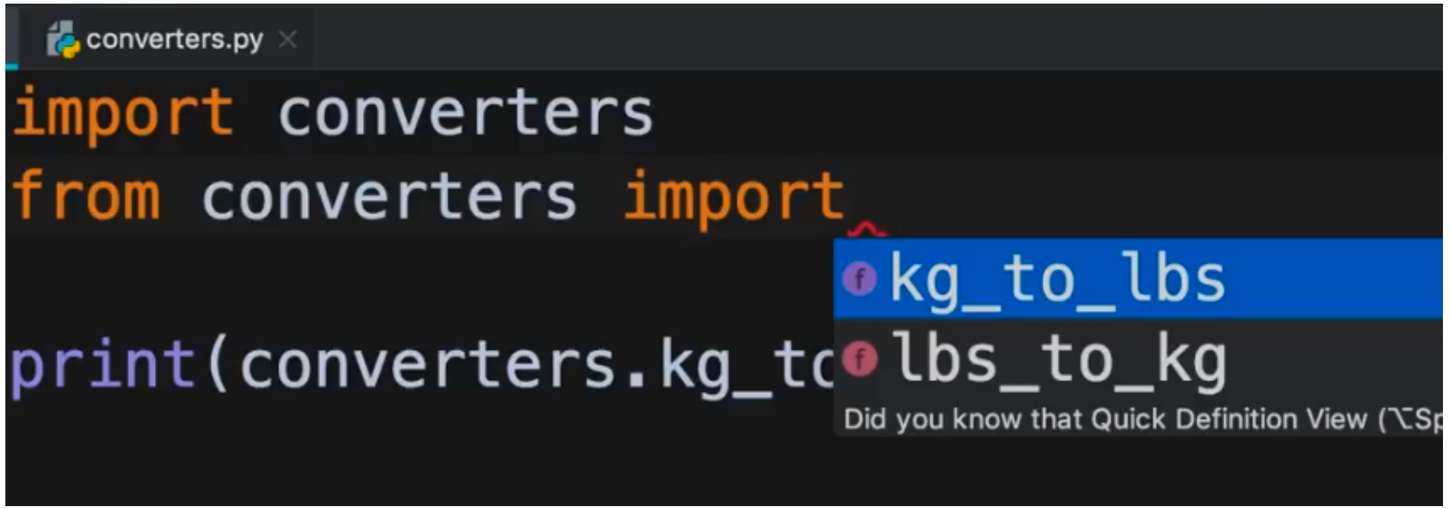
app.py

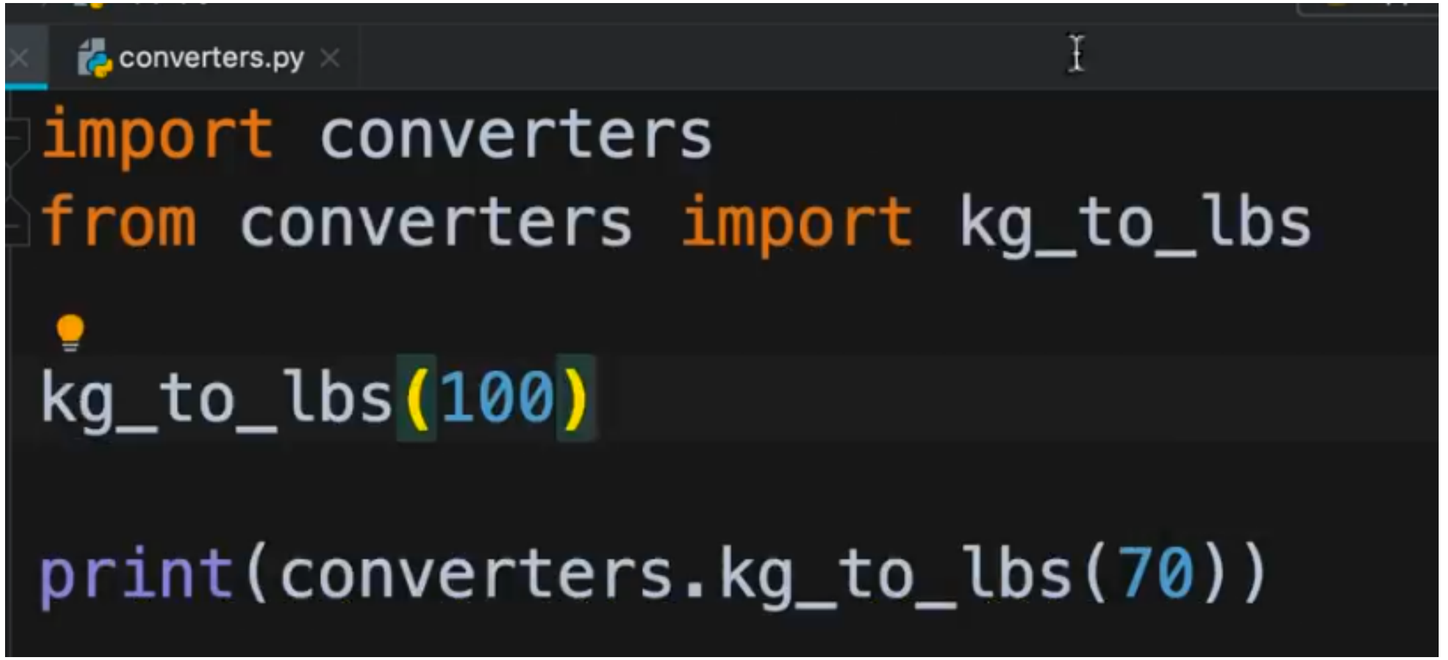
converter.py

Both app.py and converter.py are separate modules


converters.py
def lbs_to_kg(weight):
return weight * 0.45
def kg_to_lbs(weight):
return weight / 0.45


import converters
print(converters.kg_to_lbs(70)


converters.py
import converters
from converters import
print(converters.kg_tc
kg_to_lbs
lbs_to_kg
Did you know that Quick Definition View (⌥Sp


converters.py
import converters
from converters import kg_to_lbs
kg_to_lbs(100)
print(converters.kg_to_lbs(70))

Modules

A module is a file with some Python code. We use modules to break up our program into multiple files. This way, our code will be better organized. We won't have one gigantic file with a million lines of code in it!

There are 2 ways to import modules: we can import the entire module, or specific objects in a module.

Modules

A module is a file with some Python code. We use modules to break up our program into multiple files. This way, our code will be better organized. We won't have one gigantic file with a million lines of code in it!

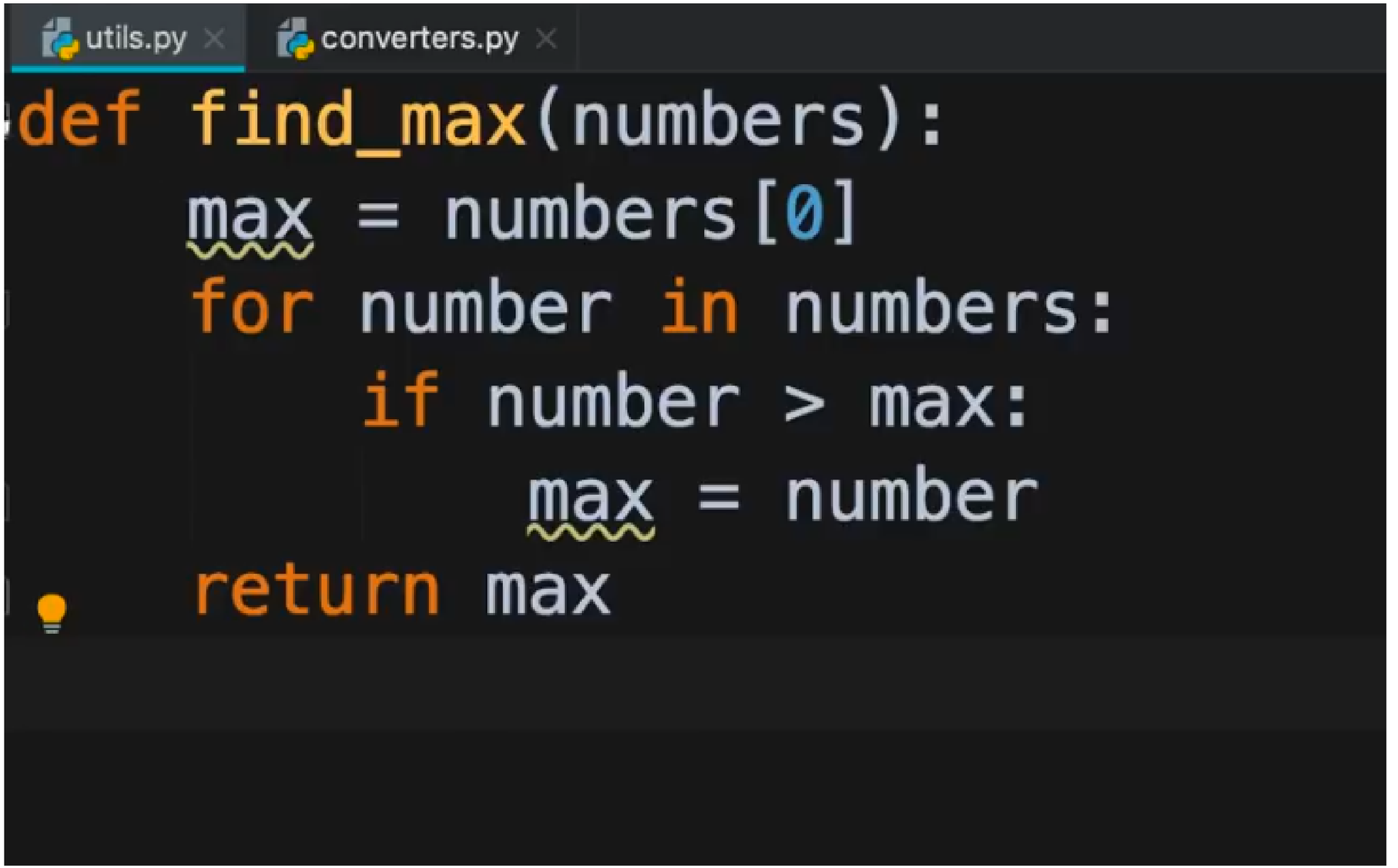
There are 2 ways to import modules: we can import the entire module, or specific objects in a module.

EXERCISE

```
numbers = [10, 3, 6, 2]
max = numbers[0]
for number in numbers:
    if number > max:
        max = number
print(max)
```

```
find_max()
utils
```

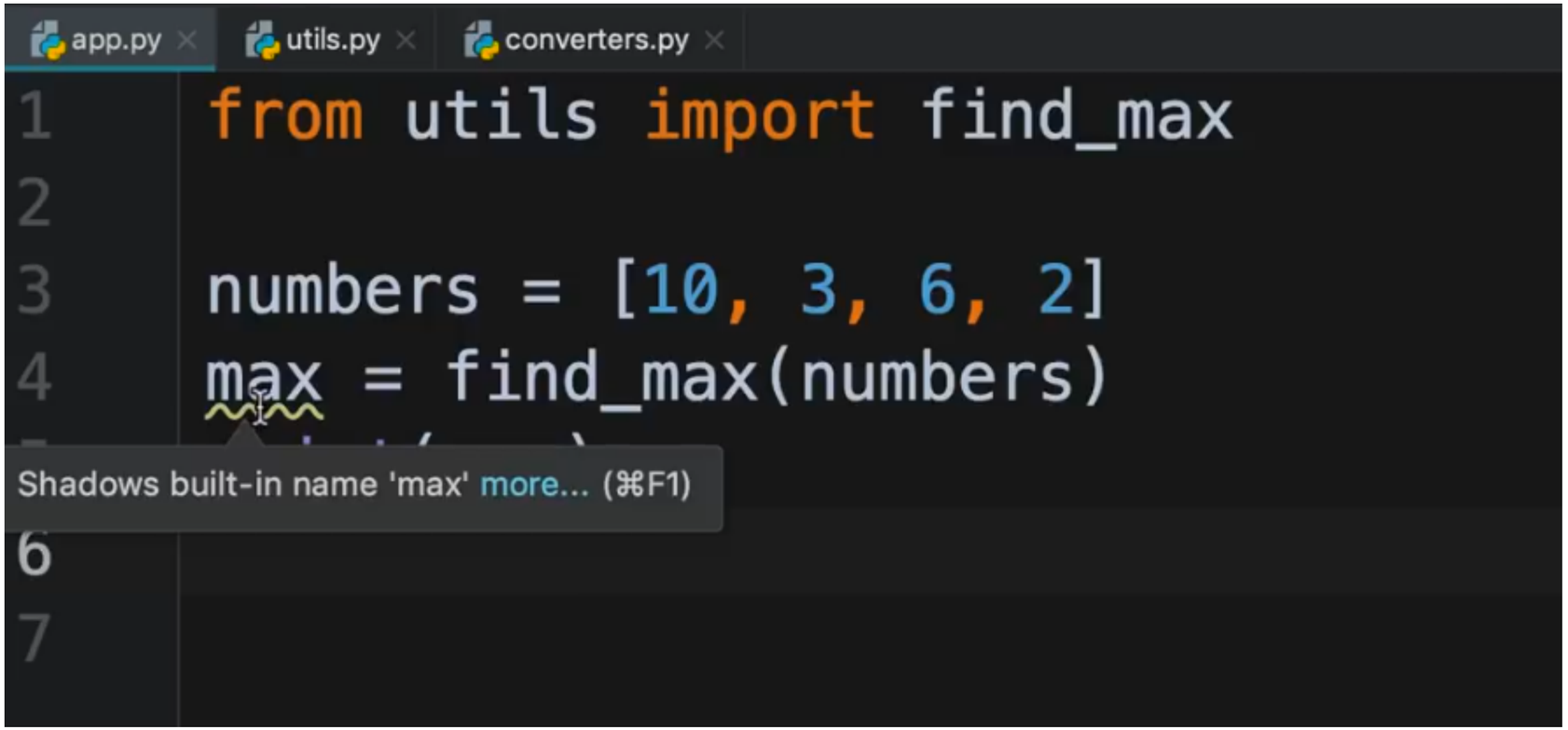
Solution


utils.py
converters.py
def find_max(numbers):
max = numbers[0]
for number in numbers:
if number > max:
max = number
return max

app.py | utils.py | converters.py

```python
import utils

utils.find_max()

numbers = [10, 3, 6, 2]
print(max)
```

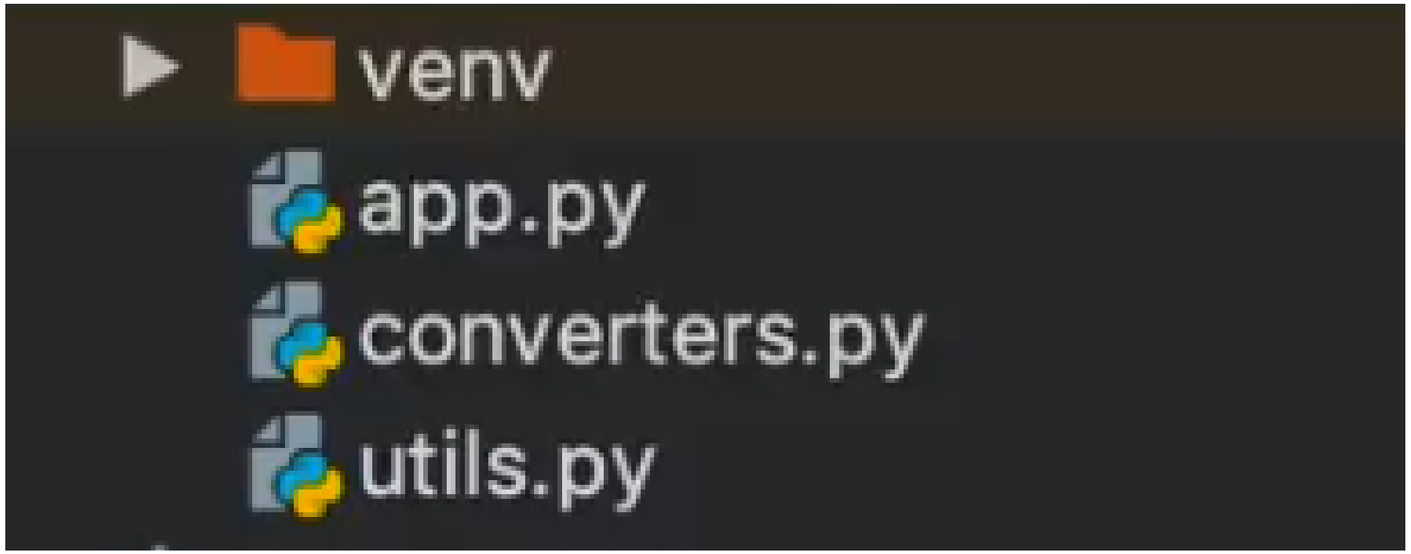
```python
from utils import find_max

numbers = [10, 3, 6, 2]
max = find_max(numbers)

print(max)
```

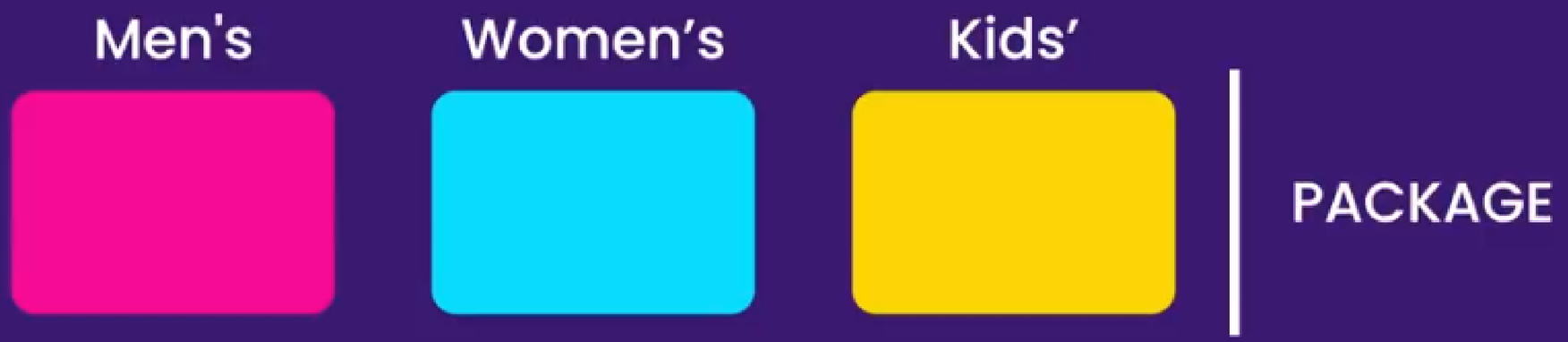

app.py
utils.py
converters.py
1 from utils import find_max
2
3 numbers = [10, 3, 6, 2]
4 max = find_max(numbers)
Shadows built-in name 'max' more... (⌘F1)
6
7

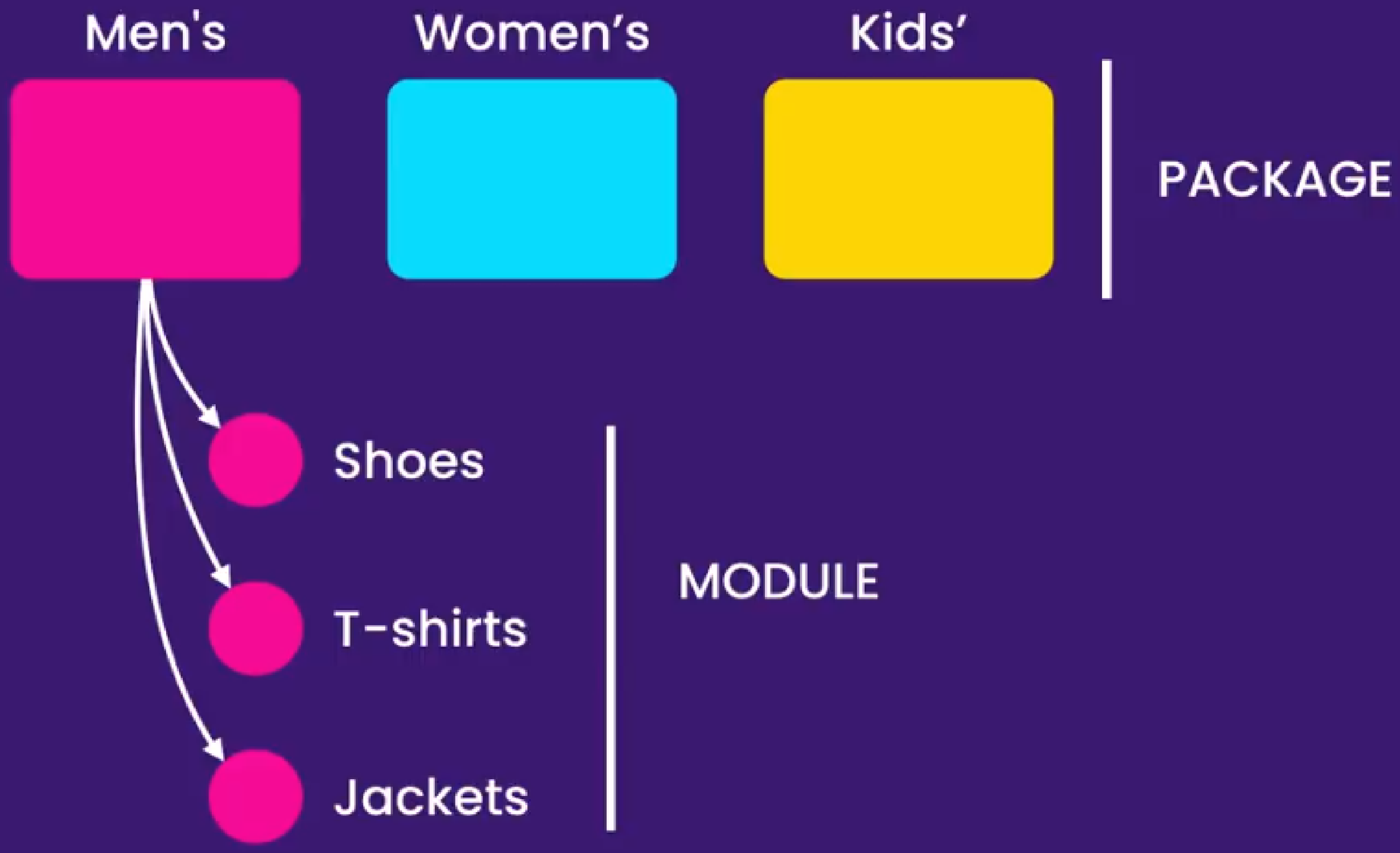
```python
from utils import find_max

numbers = [10, 3, 6, 2]
# max = find_max(numbers)
print(max)
```

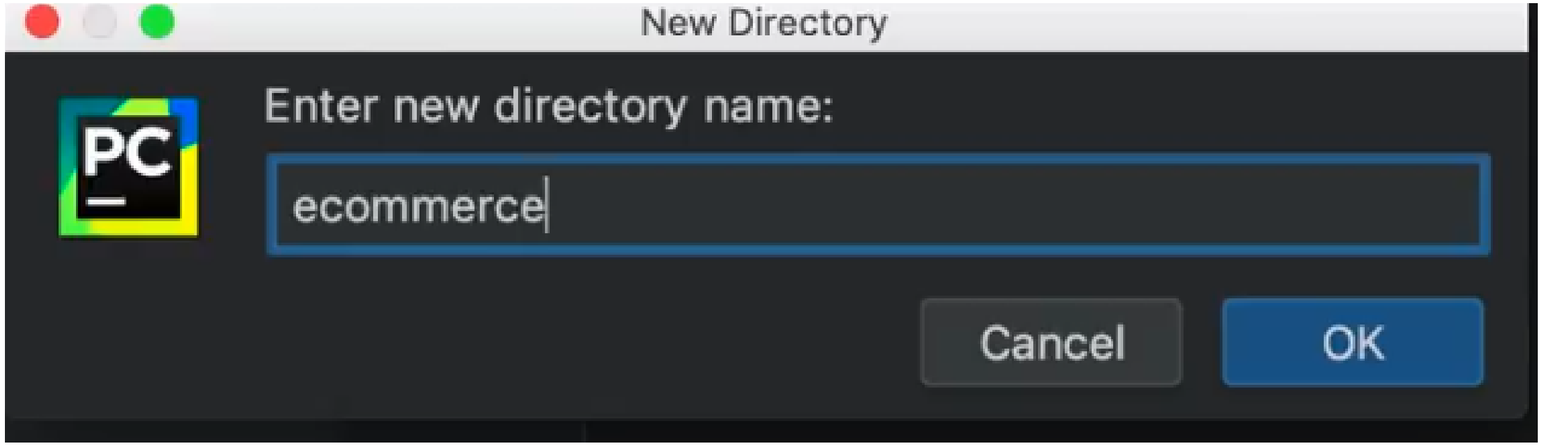
```python
from utils import find_max

numbers = [10, 3, 6, 2]
# max = find_max(numbers)
print(max(numbers))
```

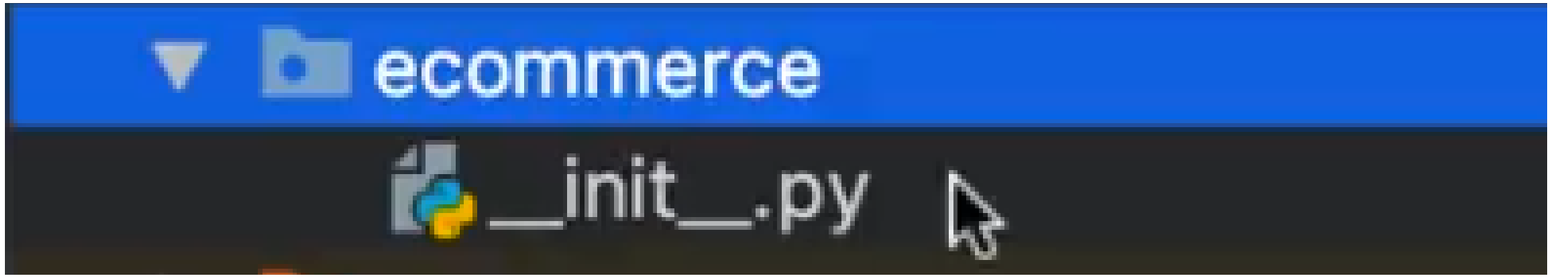
PACKAGES

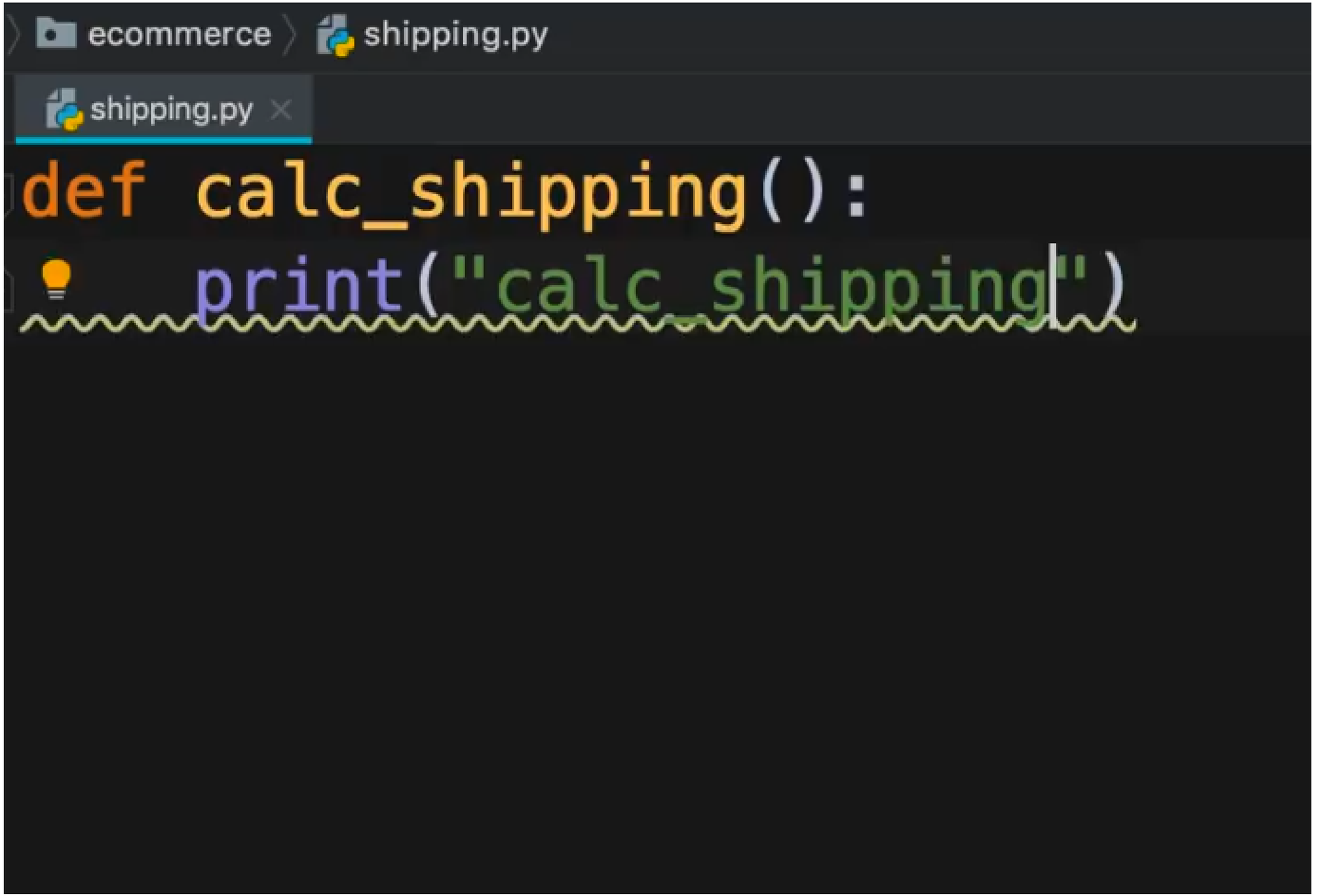

venv
app.py
converters.py
utils.py

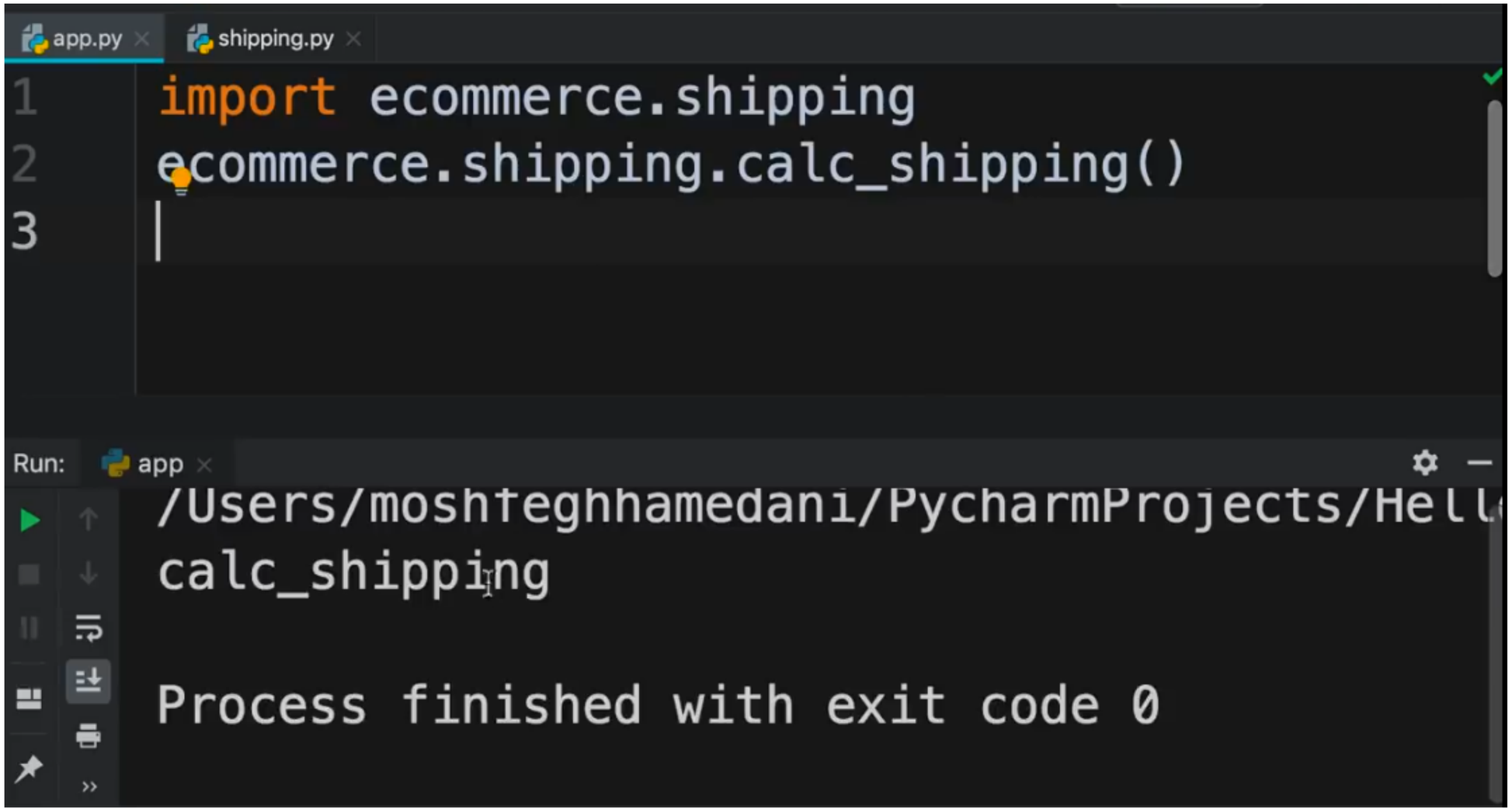

Men's
Women's
Kids'
PACKAGE


Men's
Women's
Kids'
PACKAGE
Shoes
T-shirts
Jackets
MODULE


New Directory
Enter new directory name:
ecommerce
Cancel
OK


ecommerce
__init__.py


ecommerce
shipping.py
shipping.py
def calc_shipping():
print("calc_shipping")

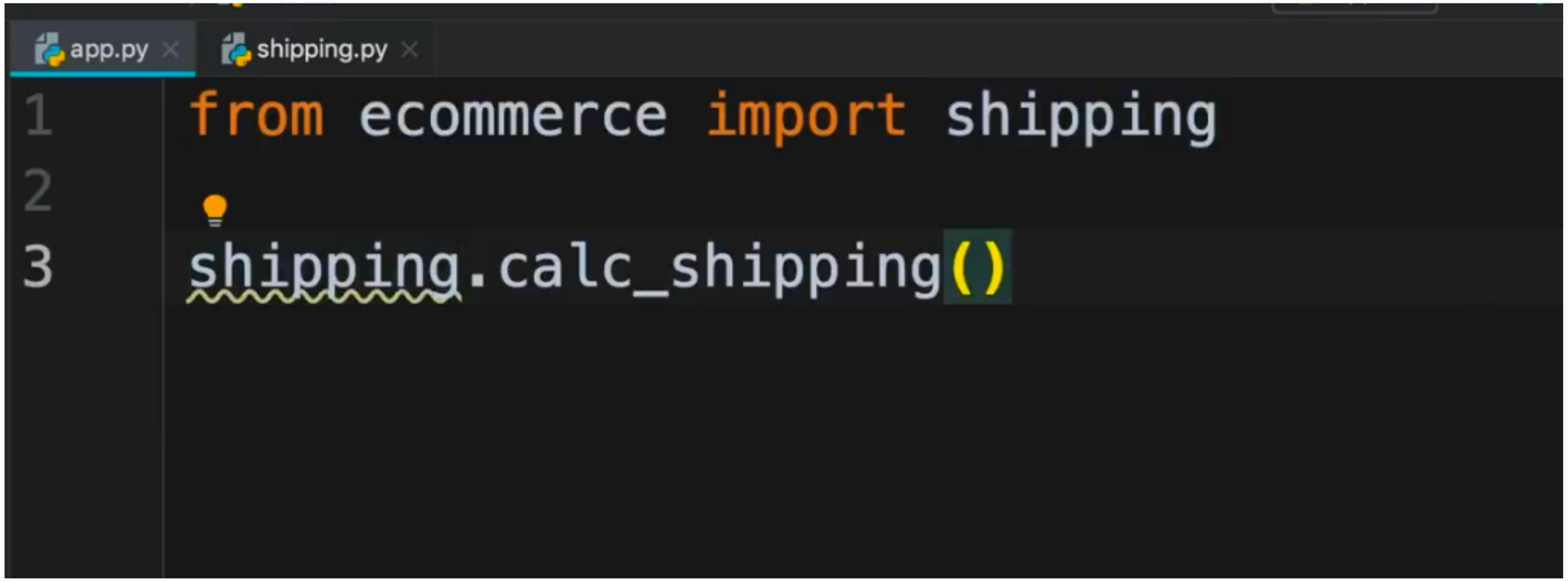

app.py
shipping.py
1
2
3
import ecommerce.shipping
ecommerce.shipping.calc_shipping()
Run:
app
/Users/moshteghhamedani/PycharmProjects/Hell
calc_shipping
Process finished with exit code 0

app.py shipping.py

```
1  from ecommerce.shipping import calc_shipping
2
3  calc_shipping()
```

app.py shipping.py

```
from ecommerce.shipping import calc_shipping, calc_tax

calc_shipping()
```


app.py
shipping.py
1 from ecommerce import shipping
2
3 shipping.calc_shipping()

Packages

A package is a directory with **__init__.py** in it. It can contain one or more modules.

```
# importing the entire sales module
from ecommerce import sales
sales.calc_shipping()

# importing one function in the sales module
from ecommerce.sales import calc_shipping
calc_shipping()
```

GENERATING RANDOM VARIABLES

Python Standard Library

Python comes with a huge library of modules for performing common tasks such as sending emails, working with date/time, generating random values, etc.

Random Module

```
import random

random.random()        # returns a float between 0 to 1
random.randint(1, 6)   # returns an int between 1 to 6

members = ['John', 'Bob', 'Mary']
leader = random.choice(members) # randomly picks an item
```

Challenge

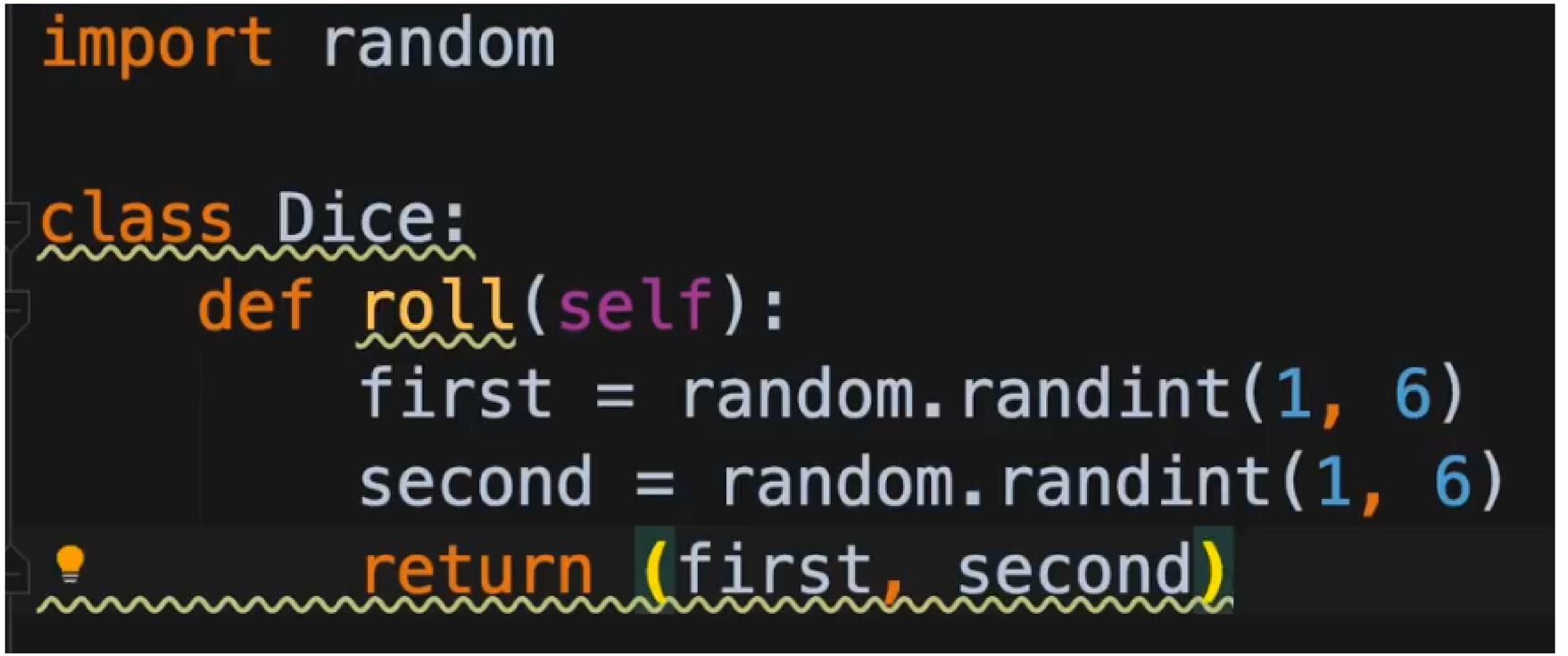
```
(3, 1)
```

```
(3, 6)
```

```
(5, 6)
```

```
Dice
    - roll()
```

Solution


import random
class Dice:
def roll(self):
first = random.randint(1, 6)
second = random.randint(1, 6)
return (first, second)

```python
import random


class Dice:
    def roll(self):
        first = random.randint(1, 6)
        second = random.randint(1, 6)
        return first, second


dice = Dice()
print(dice.roll())
```

WORKING WITH DIRECTORIES

PYPI AND PIP

THANK YOU!