# [320] Object Oriented Programming

Yiyin Shen

# Creating New Types

# Classes and Other Types


# Objects

https://www.macys.com/shop/product/martha-stewart-collection-set-of-6-cookie-cutters-created-for-macys?ID=5467270

# Classes

# **dicts** can represent many kinds of things

https://earthquake.usgs.gov/fdsnws/event/1/query?format=geojson&starttime=2021-09-21&endtime=2021-09-22

**classes** (today) are often a better option when all your keys are the same

```
m1 = {...}
m2 = {...}

p1 = {}
p2 = {}
p3 = dict()

p1["Fname"] = "Joseph"
p2["fname"] = "Peyman"
p3["fname"] = "Shri Shruthi"

print(type(m1))
print(type(p1))
```

create some objects
of type **dict** for **movies**

create some objects
of type **dict** for **people**

set some keys/values

```
class Person:
    pass

p1 = Person()
p2 = Person()
p3 = Person()

p1.Fname = "Joseph"
p2.fname = "Peyman"
p3.fname = "Shri Shruthi"

print(type(p3))
```

create a Person type/class

create some objects of type Person

set some attributes

Objects created from classes are mutable.
Attribute names are not fixed at creation.

# PythonTutor: Compare dicts to class types

Python 3.6
known limitations

```
1  p1 = {"x": 4, "y": 5}
2
3  class Coord:
4      pass
5
6  p2 = Coord()
7  p2.x = 4
8  p2.y = 5
```

Edit this code

line that just executed
next line to execute

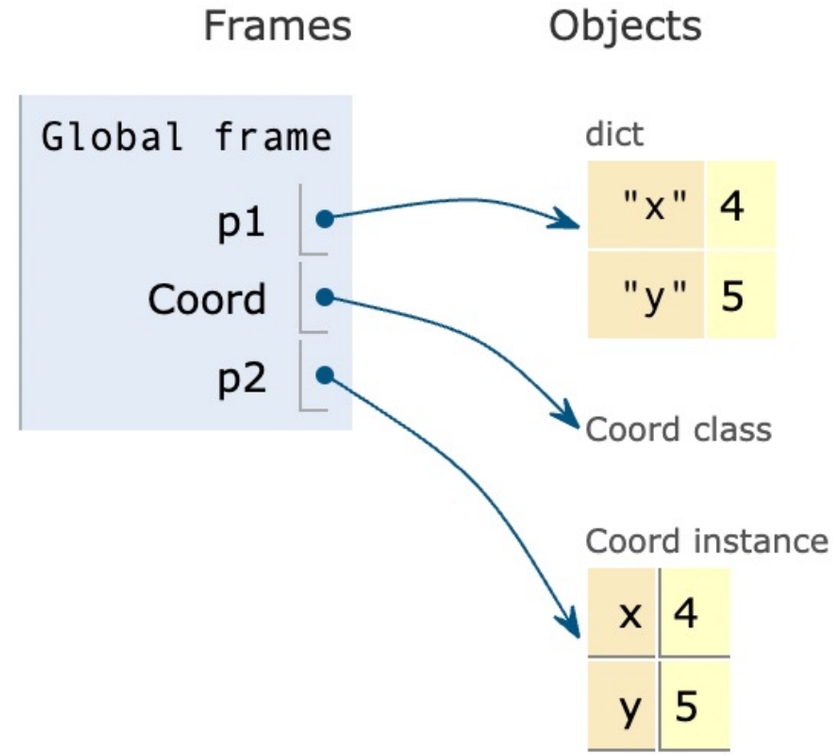
<< First | < Prev | Next > | Last >>

Done running (5 steps)

Visualized with pythontutor.com

NEW: subscribe to our YouTube

Move and hide objects



# Coding Examples: Animal Classes

**Principals**

- objects and functions
- methods
- checking object type
- type-based dispatch
- receiver (self parameter)
- constructors