COMP1531

🔨 Python
3.1 - Objects

In this lecture

Why?

- Python might be one of the first languages you've used that has a widespread notion of objects

What?

- Object model
- Examples

What are objects?

Various Types in C

```
# C
int a;
```

4 bytes
no functions

```
# C
struct point {
  int x;
  int y;
}
```

8 bytes
no functions

- Simple types in C were basic types that occupied limited memory
- Structs were collections of primitive types wrapped into an "object"
  - We would create instances of these "objects" and then access **properties** of them
  - We can expand this concept into Python

Python Objects

```
# python
lst = [1,2,3]
```

many bytes
some functions

- In python, basically every data type acts like an "object"
- An "object" being a data type that:
  - Can be created via a constructor
  - Contains 0 or more properties (/attributes)
  - Contains 0 or more functions (/methods)
- To oversimplify: It's **structs with functions**

A simple example

obj.py

```
1  from datetime import date
2
3  # CONSTRUCTIONS
4  today = date(2019, 9, 26)
5
6  # PROPERTIES / ATTRIBUTES
7  print(today.year)
8  print(today.month)
9  print(today.day)
10
11 # FUNCTIONS / METHODS
12 print(today.weekday())
13 print(today.ctime())
14
15 # 'date' is its own type
16 print(type(today))
```

Everything* is an object

- Almost all values in python are objects
- We refer to object types as "classes"
- For example:
  - lists have an append() method

```
1 animals = ["dog", "cat", "chicken"]
2 animals.append("sheep") # Modifies the list 'animals'
```

- - strings have a capitalize() method

```
1 greeting = "hi there!"
2 print(greeting.capitalize()) # Returns a new string
```

Can I define my own objects?

- Is python an objct-oriented language?
  - Sort of!
    - C is a purely procedural language
    - Java is a purely object-oriented language
    - Python is a proedural language with OO capabilities
- In python you can create your own classes (object types), though in this course you won't need to

Feedback