Object-Oriented Programming (OOP)

What are objects *in the real world*?

Things that can be perceived, used, or interacted with

They can be *physical*:   **or *abstract*:**

- Chair is a type of furniture
- Human is a type of mammal
- Fork is a type of utensil

- Happiness is a type of emotion
- Friendship is a type of relationship
- Learning is a type of experience

And they all serve distinct purposes!

What are objects *in Python*?

Many types of data in Python:

```
23        "hello world!"          3.14159          [24, 26, 25, 27]

{110.001: "Lytle and Jordan", 110.003: "Hinks"}        True
```

Every object has:
- A type
- An internal data representation
- A set of procedures for interaction with the object

An object is an instance of a type
- `23` is an instance of an `int`
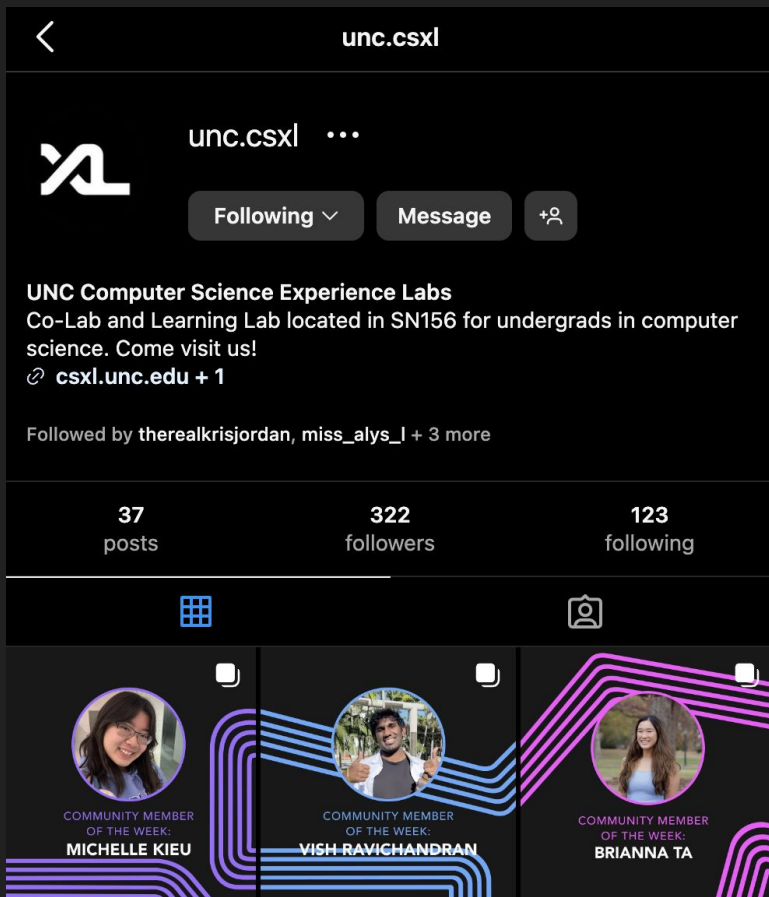- `"hello world!"` is an instance of a `str`

Modeling an Instagram profile with code

What data should we keep track of?

```
username: str = "unc.csxl"
bio: str = "UNC CS Experience Labs"
posts: int = 37
followers: int = 322
following: int = 123
private: bool = False
```

What functions would be useful?

- View # followers or following
- Write or update a bio
- (Un)follow an account
- Make an account private/public

Modeling an Instagram profile with code

What data should we keep track of?

```python
username: str = "unc.csxl"
bio: str = "UNC CS Experience Labs"
posts: int = 37
followers: int = 322
following: int = 123
private: bool = False
```

What functions would be useful?
- View # followers or following
- Write or update a bio
- (Un)follow an account
- Make an account private/public

Instagram has over **2 billion** user profiles…

What challenges could we encounter?

It'd be nice to be able to bundle these attributes and functions into one object per profile…

5

Modeling an Instagram profile with a `class`

declaring a new data type!

```
class Profile:
```

Modeling an Instagram profile with a `class`

declaring a new data type!

```
class Profile:
    username: str
    bio: str
    followers: int
    following: int
    private: bool
```

declaring attributes
(every Instagram profile has these!)

Modeling an Instagram profile with a `class`

declaring a new data type!

```python
class Profile:
    username: str
    bio: str
    followers: int
    following: int
    private: bool
```

declaring attributes
(every Instagram profile has these!)

```python
    def __init__(self):
        self.username = "usr9"
        self.bio = ""
        self.followers = 0
        self.following = 0
        self.private = False
```

initializing attributes
(what are the default values?)

8

Modeling an Instagram profile with a `class`

declaring a new data type!

```python
class Profile:
    username: str
    bio: str
    followers: int
    following: int
    private: bool
```

declaring attributes
(every Instagram profile has these!)

```python
    def __init__(self):
        self.username = "usr9"
        self.bio = ""
        self.followers = 0
        self.following = 0
        self.private = False
```

initializing attributes
(what are the default values?)

```python
my_prof: Profile = Profile()
```

Construct a new profile!

9

Modeling an Instagram profile with a `class`

declaring a new data type!

```python
class Profile:
    username: str
    bio: str
    followers: int
    following: int
    private: bool
```

declaring attributes
(every Instagram profile has these!)

```python
    def __init__(self):
        self.username = "usr9"
        self.bio = ""
        self.followers = 0
        self.following = 0
        self.private = False
```

initializing attributes
(what are the default values?)

```python
my_prof: Profile = Profile()
my_prof.username = "comp110fan"
print(my_prof.private)
```

Memory diagram

```python
 1 class Profile:
 2     username: str
 3     bio: str
 4     followers: int
 5     following: int
 6     private: bool
 7
 8     def __init__(self):
 9         self.username = "usr9"
10         self.bio = ""
11         self.followers = 0
12         self.following = 0
13         self.private = False
14
15
16 my_prof: Profile = Profile()
17 my_prof.username = "comp110fan"
18 print(my_prof.private)
```

Memory diagram

```
1  class Profile:
2      username: str
3      bio: str
4      followers: int
5      following: int
6      private: bool
7
8      def __init__(self):
9          self.username = ""
10         self.bio = ""
11         self.followers = 0
12         self.following = 0
13         self.private = False
14
15
16 my_prof: Profile = Profile()
17 your_prof: Profile = Profile()
18 your_prof.username = "unccompsci"
19 my_prof.username = "unc.csxl"
20
21 print(my_prof.username)
```

What if we wanted to keep track of usernames of followers/accounts we're following?

```
 1  class Profile:
 2      username: str
 3      bio: str
 4      followers: int
 5      following: int
 6      private: bool
 7
 8      def __init__(self):
 9          self.username = ""
10          self.bio = ""
11          self.followers = 0
12          self.following = 0
13          self.private = False
14
15
16  my_prof: Profile = Profile()
17  your_prof: Profile = Profile()
18  your_prof.username = "unccompsci"
19  my_prof.username = "unc.csxl"
20
21  print(my_prof.username)
```

How could we change our attributes
to do this?

What if we wanted to keep track of usernames of followers/accounts we're following?

```
1 class Profile:
2     username: str
3     bio: str
4     followers: set[str]
5     following: set[str]
6     private: bool
7
8     def __init__(self):
9         self.username = "usr9"
10        self.bio = ""
11        self.followers = set()
12        self.following = set()
13        self.private = False
14
15 my_prof: Profile = Profile()
16 my_prof.username = "comp110fan"
17 my_prof.following.add("unc.latinosintech")
18 print(my_prof.following)
```

Use a set[str]!

Initially empty…

Until we follow an account!