INTRO TO PYTHON

PROGRAMMING

FOR-LOOPS

- Repeats a chunk of code
- Hands in different item each time

FOR-LOOPS

```
pets = [“dog”, “cat”, “fish”]
for pet in pets:
    print(pet)
```

FOR-LOOPS

```
pets = [“dog”, “cat”, “fish”]
for pet in pets:
    print(pet)
```

FOR-LOOPS

```
pets = [“dog”, “cat”, “fish”]
for pet in pets:
    print(“dog”)
```

FOR-LOOPS

```
pets = [“dog”, “cat”, “fish”]
for pet in pets:
    print(“cat”)
```

FOR-LOOPS

```
pets = [“dog”, “cat”, “fish”]
for pet in pets:
    print(“fish”)
```

FOR-LOOPS

```
pets = [“dog”, “cat”, “fish”]
for pet in pets:
    if pet == “dog”:
        print(“Man’s best friend!”)
    else:
        print(“whatever”)
```

RANGE FUNCTION

- Generates range of numbers
- Parameters match slice syntax

RANGE FUNCTION

```
range(3)
```

RANGE FUNCTION

```
range(3)
```

→ 0, 1, 2

RANGE FUNCTION

```
range(1,4)
```

RANGE FUNCTION

```
range(1,4)
```

→ 1, 2, 3

RANGE FUNCTION

```
range(3,0,-1)
```

→ 3, 2, 1

RANGE IS NOT A LIST

- Returns a “lazy” list
- Numbers are generated one-by-one to save memory

CONVERT RANGE TO LIST

```
list(range(3))
```

→ [0, 1, 2]

FOR-LOOPS

- Repeats a chunk of code
- Hands in different item each time

EXERCISES

VALUES: WHAT KINDS ARE THERE?

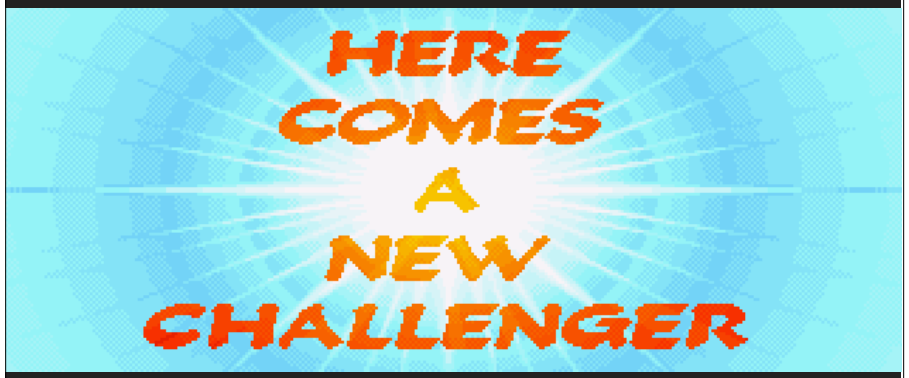
- data types: int, float, string, bool
- lists of the above

TROUBLE WITH LISTS

Hard to manage lots of data

TROUBLE WITH LISTS

Hard to manage lots of data



DICTIONARY

- Store data under a label (“key”)
- Fetch value later by key

DICTIONARY

```
my_dict = { “name”: “Rob” }
```

DICTIONARY

```
my_dict = { “name”: “Rob” }
```

dictionary literal

DICTIONARY

```
my_dict = {“name”: “Rob”,
           “hair”: “brown”}
```

DICTIONARY

```
my_dict[“hair”]
```

→ “brown”

DICTIONARY

```
my_dict[“hair”] = “blonde”
```

DICTIONARY

```
for key in my_dict:
    print(“look at this”+ key)
```

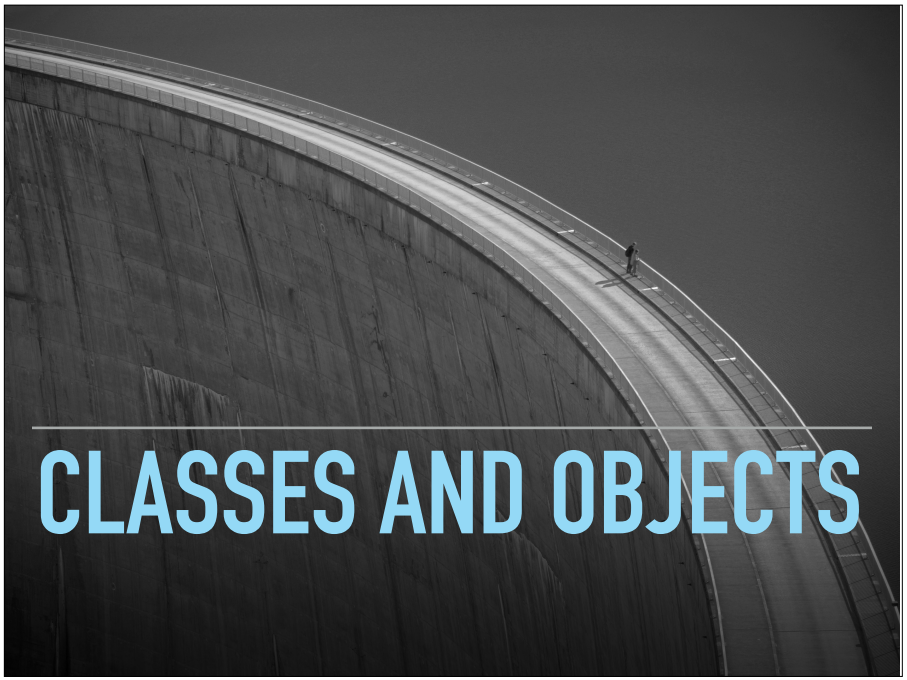
DICTIONARY

```
for value in my_dict.values():
    print(“look at this ”+value)
```

DICTIONARY

```
for key, value in my_dict.items():
    print(“look at this ”+key)
    print(“look at this ”+value)
```

EXERCISES



CLASSES AND OBJECTS

CLASS VS OBJECT

- class is a “blueprint” with empty slots
- object is one occurrence, slots filled

WHY DO WE NEED CLASSES?

- Like dictionary with guaranteed keys
- Can create methods

USING A CLASS

```
rob = Person(“Rob”)
rob.name
>>> “Rob”
```

USING A CLASS

```
rob = Person("Rob")
rob.say_name()
>>> "Hi I'm Rob"
```

WHY DO WE NEED CLASSES?

- Store data and methods together
- Can make code more intuitive

CLASS

```
class Person():
    def __init__(self, name):
        self.name = name
```

CLASS

```
class Person():
    def __init__(self, name):
        self.name = name
```

CLASS

```
class Person():
    def __init__(self, name):
        self.name = name
```

WHAT IS "SELF"??

- Class is the cookie-cutter, self is the cookie
- First argument in any method

CLASS

```
class Person():
    def __init__(self, name):
        self.name = name
```

CLASS

constructor

```
class Person():
    def __init__(self, name):
        self.name = name
```

CLASS

magic

```
class Person():
    def __init__(self, name):
        self.name = name
```

CLASS

instance

```
class Person():
    def __init__(self, name):
        self.name = name
```

CLASS

magic

```
class Person():
    def __init__(self, name):
        self.name = name
```

CLASS

```
class Person():
    def __init__(self, name):
        self.name = name
```

CLASS

```
class Person():
    def __init__(self, name):
        self.name = name
```

attribute

CLASS

```
class Person():
    def __init__(self, name):
        self.name = name
```

self[“name”] = name

CLASS

```
class Person():
    …
    def say_name(self):
        print(self.name)
```

CLASS

```
rob = Person(“Rob”)
rob.name
>>> “Rob”
```

CLASS

```
rob = Person("Rob")
rob.name
>>> "Rob"
```

attribute

CLASS

```
rob = Person("Rob")
rob.say_name()
>>> "Rob"
```

CLASS

```
rob = Person("Rob")
rob.say_name()
>>> "Rob"
```

method

EXERCISES

CONCEPTS COVERED SO FAR

- For-loops
- Dictionary
- Class

DICTIONARY

```
my_dict = {“name”: “Rob”,
           “hair”: “brown”}
```

DICTIONARY

```
my_dict[“hair”] = “blonde”
```

DICTIONARY

```
my_dict[“hair”]
```

DICTIONARY

```
my_dict[“hair”]
```

→ “brown”

DICTIONARY

```
for key in my_dict:
    print(f“look at this {key}!”)
```

DICTIONARY

```
for key, value in my_dict.items():
    print(f“look at this {key}!”)
    print(f“look at this {value}”)
```

CLASS

```
class Person():
    def __init__(self, name):
        self.name = name
```

CLASS

```
rob = Person(“Rob”)
rob.name
```

CLASS

```
rob = Person(“Rob”)
rob.name
```

“Rob”