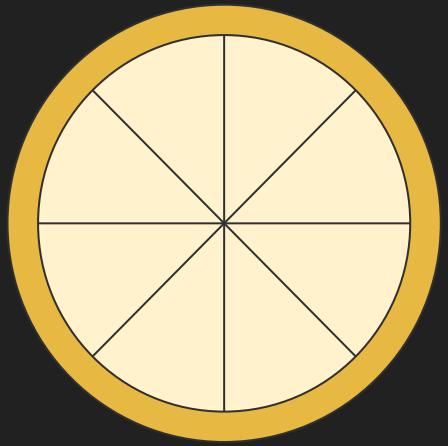
Object Oriented Programming

Example: Pizza

size: small

toppings: 0

gluten free: no

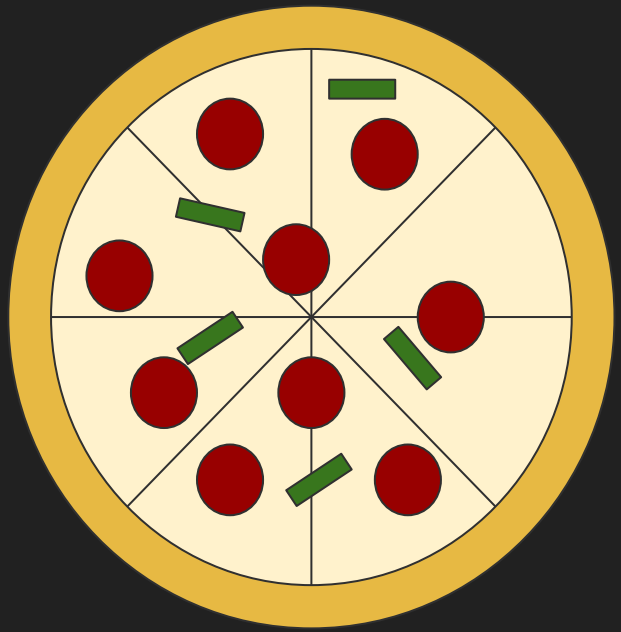


Example: Pizza

size: large

toppings: 2

gluten free: yes



Object Oriented Programming

Lets you create new objects in your program.

“Type” ~> “Class”

“Data/Variables” ~> “Attributes”

“Functions” ~> “Methods”

Creating a class

```
class <class name>:

    <class body>
```

Let's try it!
Create a file called pizza_orders.py.
Create a class called Pizza with an empty body.

Attributes

- first part of class body
- variables that belong to each instantiation of the object
- Syntax:

```
<attribute name> : <type>

gluten_free : bool
```

Let's try it!

Give the Pizza class the following attributes:

- **gluten_free:** boolean of whether or not pizza is GF
- **size:** string storing the size of the pizza
- **num_toppings:** number of toppings on the pizza

Constructor

- Method that defines what happens when new object is created
- Signature Syntax:

```
def __init__(self, <other parameters>):
```

*Essentially returns self

Let's try it!

Write a constructor that takes the following inputs and uses it to initialize the corresponding parameters

- **gf_input: bool**
- **size_input: str**
- **num_toppings_input: int**

Constructor

- Method that defines what happens when new object is created
- Signature Syntax:

```
def __init__(self, <other parameters>):
```

- Instantiation:

```
<class name>(<arguments>)
```

*Essentially returns self

Let's try it!

Create a Pizza object with the following arguments:

- **gf_input=False**
- **size_input="large"**
- **num_toppings_input:2**

Methods

- Functions that belong to an object
- Defining a method:

```
def <method_name>(self, <other params>) -> <ret type>:
```

```
def price(self) -> float:
```

- Calling a method:

```
my_pizza.price()
```

Methods

- Functions that belong to an object
- Defining a method:

```
def <method_name>(self, <other params>) -> <ret type>:
```

```
def price(self) -> float:
```

- Calling a method:

```
my_pizza.price()
```
 * as opposed to `price(my_pizza)`

Let's try it!

Write a method called price with the following behavior:

- Size "small" costs $5.25, other sizes cost $7.50
- Each topping is $.25
- If gluten free, add $1

And call it!

Functions that use class objects

- You can also define a function outside the class that takes a class objects as input!

Let's try it!

Write a function called num_orders that takes as input a list[Pizza] and returns the number of of Pizza objects in the list.

Memory Diagram

```python
class Dog:

    age: int
    breed: str

    def __init__(self, inp_age: int, inp_breed: str):
        self.age = inp_age
        self.breed = inp_breed

    def greet(self) -> None:
        if self.age < 2:
            print("Hi puppy!")
        else:
            print(f"Hi pupper!")

Bear: Dog = Dog(4, "Poodle")
Bear.greet()
print(Bear.age)
```

Quiz Questions + Review?