

## Objects Continued

### Textbook

## Objects Continued



Object Oriented Programming is a powerful tool for programming. Let's explore more things that can be done with the objects and classes.

As a review, here is a basic class with some objects created using the constructor.

```
1 class Dog:
2
3     def __init__(self, name, age, gender):
4         self.name = name
5         self.age = age
6         self.gender = gender
7
8 pet = Dog("Jasmine", "15", "Female")
9
10 print(pet)
```

Try it!

## Adding an Attribute to the Class

We can also add attributes to the class.

```
1 class Dog:
2
3     def __init__(self, name, age, gender):
4         self.name = name
5         self.age = age
6         self.gender = gender
7
8 pet = Dog("Jasmine", "15", "Female")
9
10 pet.height = 12
11
12 print(pet.height)
```

Try it!

## Removing an Attribute to the Class

We can also remove an attribute to the class by using `delattr()`

```
1 class Dog:
2
3     def __init__(self, name, age, gender):
4         self.name = name
5         self.age = age
6         self.gender = gender
7
8 pet = Dog("Jasmine", "15", "Female")
9
10 delattr(name)
11
12 print(pet.name)
```

Try it!

## Modify Object Properties

We can also change the values in our objects.

```
1 class Dog:
2
3     def __init__(self, name, age, gender):
4         self.name = name
5         self.age = age
6         self.gender = gender
```

```

7
8 pet = Dog("Jasmine", "15", "Female")
9
10 print(pet.name)
11
12 pet.name = "Franz"
13
14 print(pet.name)

```

Try it!

## Delete Object Properties

We can also remove the values in our objects. This is done with the `del` keyword.

```

1 class Dog:
2
3     def __init__(self, name, age, gender):
4         self.name = name
5         self.age = age
6         self.gender = gender
7
8 pet = Dog("Jasmine", "15", "Female")
9
10 print(pet.name)
11
12 del pet.name
13
14 print(pet.name)

```

Try it!

This will throw an error because there is no longer a value for `pet.name`.

## Object Methods

We can also add functions to our classes. These functions are called methods.

Let's add a function named `greeting` and put it in the class named `Dog`. This function will print out `Good Morning`. Then we will call the function down at the bottom of our code.

```

1 class Dog:
2     def __init__(self, name, age, gender):
3         self.name = name
4         self.age = age
5         self.gender = gender
6
7     def greeting(self):
8         print("Good Morning")
9
10
11 pet = Dog("Jasmine", "15", "Female")
12
13 pet.greeting()

```

Try it!

This will print out `Good Morning` .

Notice how we passed the parameter `self` into the function declaration? This is so that the function is used in this particular instance of this class.

## Abstraction

Abstraction is like creating a simplified view of something complex, showing only what's necessary and hiding the complicated details. When you define a **class**, you're essentially creating an **abstraction**. For example, the `Dog` class abstracts the idea of a dog by focusing on key characteristics like `name` , `age` , and `gender` , without needing to worry about every single biological detail of a real dog. This allows you to work with `Dog` objects at a higher, more manageable level, without getting bogged down by the internal complexities of how those characteristics are stored or how the dog behaves internally.

## Checkpoint

---

### Objects Continued

Practice making modifications to your objects!

1. Create a class named `Vacation`
2. Inside the class named `Vacation` , create the `__init__()` function with four parameters: `self` , `place` , `distance` , and `weather` . Make sure to assign these values to `self`.
3. Inside the class named `Vacation` , create a method named `tuesday()` . Inside the method named `tuesday()` , create a print statement that prints `We will be hiking on Tuesday` . Remember to include the `self` parameter in the method.
4. After the class named `Vacation` , create a variable named `summer` . Place an instance of the class named `Vacation` inside the variable named `summer` . This instance of the class will have the following parameters: `"Hawaii"` , `2000` , `"Sunny"`
5. Add an attribute named `rating` to the object named `summer` . Assign the attribute to `10` .
6. Update the `weather` attribute in the object named `summer` to `"rainy"`
7. In a separate print statement, print the object named `summer` .
8. In a separate print statement, print the attribute named `rating` in the object named `summer` .
9. In a separate print statement, print the updated attribute named `weather` in the object named `summer` .

### Requirements:

- Create a class named `Vacation`
- Inside the class named `Vacation`, create the `__init__()` function with four parameters: `self`, `place`, `distance`, and `weather`. Make sure to assign these values to `self`.
- Inside the class named `Vacation`, create a method named `tuesday()`. Inside the method named `tuesday()`, create a print statement that prints `We will be hiking on Tuesday`. Remember to include the `self` parameter in the method.
- After the class named `Vacation`, create a variable named `summer`. Place an instance of the class named `vacation` inside the variable named `summer`. This instance of the class will have the following parameters: `"Hawaii"`, `2000`, `"Sunny"`
- Add an attribute named `rating` to the object named `summer`. Assign the attribute to `10`.
- Update the `weather` attribute in the object named `summer` to `"rainy"`
- In a separate print statement, print the object named `summer`.
- In a separate print statement, print the attribute named `rating` in the object named `summer`.
- In a separate print statement, print the updated attribute named `weather` in the object named `summer`.

## Questions (10)

MULTIPLE CHOICE

### 1. What is the object in the following example?

```
class Dog:
    def __init__(self, name, age, gender):
        self.name = name
        self.age = age
        self.gender = gender
    def greeting(self):
        print("Good Morning")
pet = Dog("Jasmine", "15", "Female")
pet.greeting()
```

Choose the correct answer:

- A. pet
- B. greeting
- C. Dog
- D. \_\_init\_\_()

MULTIPLE CHOICE

### 2. What will the following code print out?

```
class House:
    def __init__(self, build, size, location):
        self.build = build
        self.size = size
        self.location = location
myhouse = House("A frame", 2000, "Chicago")
myhouse.location = "Portland"
print(myhouse.build)
```

Choose the correct answer:

- A. Chicago
- B. Portland
- C. A frame
- D. 2000
- E. myhouse
- F. House
- G. <myhouse object at 0x9fbb08>

MULTIPLE CHOICE

### 3. What will the following code print out?

```
class House:
    def __init__(self, build, size, location):
        self.build = build
        self.size = size
        self.location = location
myhouse = House("A frame", 2000, "Chicago")
myhouse.location = "Portland"
print(myhouse.location)
```

Choose the correct answer:

- A. Portland
- B. Chicago
- C. 2000
- D. A frame
- E. <myhouse object at 0x9fbb08>
- F. House

**4. What will the following code print out?**

```
class House: def __init__(self, build, size, location): self.build = build self.size = size self.location = location myhouse = House("A frame", 2000, "Chicago") del House.size print(myhouse.size)
```

**Choose the correct answer:**

- A. A frame
- B. Chicago
- C. Portland
- D. 2000
- E. House
- F. It will throw an error

**5. What is the object method in the following example?**

```
class Dog: def __init__(self, name, age, gender): self.name = name self.age = age self.gender = gender def greeting(self): print("Good Morning") pet = Dog("Jasmine", "15", "Female") pet.greeting()
```

**Choose the correct answer:**

- A. greeting
- B. pet
- C. Dog
- D. There is no object method

**6. In the given code, what are "name," "age," and "gender" examples of?**

```
class Dog: def __init__(self, name, age, gender): self.name = name self.age = age self.gender = gender pet = Dog("Jasmine", "15", "Female") print(pet)
```

**Choose the correct answer:**

- A. Methods
- B. Attributes
- C. Functions
- D. Classes

**7. What will be printed when the following code is executed?**

```
class Dog: def __init__(self, name, age, gender): self.name = name self.age = age self.gender = gender pet = Dog("Jasmine", "15", "Female") print(pet.name)
```

**Choose the correct answer:**

- A. Jasmine
- B. 15
- C. Female
- D. Dog

**8. How is a new attribute added to an object?**

**Choose the correct answer:**

- A. Using the add\_attr function
- B. With the create\_attr keyword
- C. By using the setattr method
- D. Just by assigning a new attribute to a value.

**9. What does the following code snippet delattr(name) do in the following code?**

```
class Dog: def __init__(self, name, age, gender): self.name = name self.age = age self.gender = gender pet = Dog("Jasmine", "15", "Female") delattr(name) print(pet.name)
```

**Choose the correct answer:**

- A. Deletes the entire object
- B. Removes the 'name' attribute from the object
- C. Adds a new attribute named 'age'
- D. Prints the 'age' attribute

**10. True or False: Objects can also have their own methods.**

**Choose the correct answer:**

- A. True
- B. False

## Challenges (2)

### 1. Weekend

Create a class that describes things you do on the weekend!

1. Create a class named `Friday`
2. Inside the class named `Friday`, create the `__init__()` function with three parameters: `self`, `activity` and `friend`. Make sure to assign these values to `self`.
3. Inside the class named `Friday`, create a method named `pictures()`. Inside the method named `pictures()`, create a print statement that prints `We took so many pictures!`. Remember to include the `self` parameter in the method.
4. After the class named `Friday`, create a variable named `thisWeekend`. Place an instance of the class named `Friday` inside the variable named `thisWeekend`. This instance of the class will have the following parameters: `"Movie"`, `"Charlotte"`
5. Add an attribute named `money` to the object named `thisWeekend`. Assign the attribute to `20`.
6. Update the `friend` attribute in the object named `thisWeekend` to `"Shane"`
7. In a separate print statement, print the object named `thisWeekend`.
8. In a separate print statement, print the attribute named `money` in the object named `thisWeekend`.
9. In a separate print statement, print the updated attribute named `friend` in the object named `thisWeekend`.

#### Requirements:

- Create a class named `Friday`
- Inside the class named `Friday`, create the `__init__()` function with three parameters: `self`, `activity` and `friend`. Make sure to assign these values to `self`.
- Inside the class named `Friday`, create a method named `pictures()`. Inside the method named `pictures()`, create a print statement that prints `We took so many pictures!`. Remember to include the `self` parameter in the method.
- After the class named `Friday`, create a variable named `thisWeekend`. Place an instance of the class named `Friday` inside the variable named `thisWeekend`. This instance of the class will have the following parameters: `"Movie"`, `"Charlotte"`
- Add an attribute named `money` to the object named `thisWeekend`. Assign the attribute to `20`.
- Update the `friend` attribute in the object named `thisWeekend` to `"Shane"`
- In a separate print statement, print the object named `thisWeekend`.
- In a separate print statement, print the attribute named `money` in the object named `thisWeekend`.
- In a separate print statement, print the updated attribute named `friend` in the object named `thisWeekend`.



## 2. Shopping

Create a class for a Shopping trip. This class will have a method that adds items to an empty list.

1. Create a class named `Shopping` .
2. Inside the class named `Shopping` , create the `def __init__()` function. Inside the function, include the following parameters: `self` , `item` , `quality` . Make sure to assign these values to `self` .
3. Inside the `def __init__()` function, create an attribute `self.total` and assign it to an empty list.
4. Inside the class named `Shopping` , create a method named `spending` . Include the following parameters: `(self, cost)`
5. Inside the method named `spending` , append the value of `cost` to the `self.total` attribute.
6. Create an instance of the class named Shopping. Name this instance sportStore and include the following arguments: `("Kayak", "High Quality")`
7. Call the method named `spending` in 3 different function calls. Use different integer values for each call.
8. In a separate print statement, print the variable named `sportStore.total` .

### Requirements:

- Create a class named Shopping
- Inside the class named Shopping, create the `def __init__()` function. Inside the function, include the following parameters: `self`, `item`, `quality`. Make sure to assign these values to `self`.
- Inside the `def __init__()` function, create an attribute `self.total` and assign it to an empty list.
- Inside the class named Shopping, create a method named `spending`. Include the following parameters: `(self, cost)`
- Inside the method named `spending`, append the value of `cost` to the `self.total` attribute.
- Create an instance of the class named Shopping. Name this instance sportStore and include the following arguments: `("Kayak", "High Quality")`
- Call the method named `spending` in 3 different function calls. Use different integer values for each call.
- In a separate print statement, print the variable named `sportStore.total`.

## Answer Keys & Solutions

### Checkpoint Solutions

#### Objects Continued

```
1 class Vacation:
2     def __init__(self, place, distance, weather):
3         self.place = place
4         self.distance = distance
5         self.weather = weather
6
7     def tuesday(self):
8         print("We will be hiking on Tuesday.")
9
10
11 summer = Vacation("Hawaii", 2000, "Sunny")
12
13 summer.rating = 10
14
15 summer.weather = "rainy"
16
17 print(summer)
18 print(summer.rating)
19 print(summer.weather)
```

### Questions

#### 1. What is the object in the following example?

MULTIPLE CHOICE

Correct Answer:

- |               |             |
|---------------|-------------|
| A. pet        | ✓ Correct   |
| B. greeting   | ✗ Incorrect |
| C. Dog        | ✗ Incorrect |
| D. __init__() | ✗ Incorrect |

#### Explanation:

Dog is the name of the class, or the object constructor. greeting is the name of the method inside the class named Dog.

## 2. What will the following code print out?

Correct Answer:

- |                                 |             |
|---------------------------------|-------------|
| A. Chicago                      | ✗ Incorrect |
| B. Portland                     | ✗ Incorrect |
| C. A frame                      | ✓ Correct   |
| D. 2000                         | ✗ Incorrect |
| E. myhouse                      | ✗ Incorrect |
| F. House                        | ✗ Incorrect |
| G. <myhouse object at 0x9fbb08> | ✗ Incorrect |

### Explanation:

The build attribute is being printed. The location attribute is what was updated.

## 3. What will the following code print out?

Correct Answer:

- |                                 |             |
|---------------------------------|-------------|
| A. Portland                     | ✓ Correct   |
| B. Chicago                      | ✗ Incorrect |
| C. 2000                         | ✗ Incorrect |
| D. A frame                      | ✗ Incorrect |
| E. <myhouse object at 0x9fbb08> | ✗ Incorrect |
| F. House                        | ✗ Incorrect |

### Explanation:

The location attribute of the object named myhouse was updated to Portland.

## 4. What will the following code print out?

Correct Answer:

- |                           |             |
|---------------------------|-------------|
| A. A frame                | ✗ Incorrect |
| B. Chicago                | ✗ Incorrect |
| C. Portland               | ✗ Incorrect |
| D. 2000                   | ✗ Incorrect |
| E. House                  | ✗ Incorrect |
| F. It will throw an error | ✓ Correct   |

**Explanation:**

We deleted the attribute named size, so it cannot be printed.

**5. What is the object method in the following example?**

MULTIPLE CHOICE

**Correct Answer:**

- |                              |             |
|------------------------------|-------------|
| A. greeting                  | ✓ Correct   |
| B. pet                       | ✗ Incorrect |
| C. Dog                       | ✗ Incorrect |
| D. There is no object method | ✗ Incorrect |

**Explanation:**

The object method is a function inside of the class

**6. In the given code, what are "name," "age," and "gender" examples of?**

MULTIPLE CHOICE

**Correct Answer:**

- |               |             |
|---------------|-------------|
| A. Methods    | ✗ Incorrect |
| B. Attributes | ✓ Correct   |
| C. Functions  | ✗ Incorrect |
| D. Classes    | ✗ Incorrect |

**Explanation:**

These are all attributes of the class

## 7. What will be printed when the following code is executed?

MULTIPLE CHOICE

**Correct Answer:**

- A. Jasmine ✓ Correct
- B. 15 ✗ Incorrect
- C. Female ✗ Incorrect
- D. Dog ✗ Incorrect

**Explanation:**

The name attribute is being called

## 8. How is a new attribute added to an object?

MULTIPLE CHOICE

**Correct Answer:**

- A. Using the add\_attr function ✗ Incorrect
- B. With the create\_attr keyword ✗ Incorrect
- C. By using the setattr method ✗ Incorrect
- D. Just by assigning a new attribute to a value. ✓ Correct

**Explanation:**

To

## 9. What does the following code snippet `delattr(name)` do in the following code?

MULTIPLE CHOICE

**Correct Answer:**

- A. Deletes the entire object ✗ Incorrect
- B. Removes the 'name' attribute from the object ✓ Correct

C. Adds a new attribute named 'age'

✗ Incorrect

D. Prints the 'age' attribute

✗ Incorrect

## 10. True or False: Objects can also have their own methods.

MULTIPLE CHOICE

Correct Answer:

A. True

✓ Correct

B. False

✗ Incorrect

### Explanation:

Objects can have methods too

## Challenges

### 1. Weekend

Solution:

```
1 class Friday:
2
3     def __init__(self, activity, friend):
4         self.activity = activity
5         self.friend = friend
6
7     def pictures(self):
8         print("We took so many fun pics this weekend!")
9
10 thisWeekend = Friday("Movie", "Charlotte")
11
12 thisWeekend.money = 20
13 thisWeekend.friend = "Shane"
14
15 print(thisWeekend)
16 print(thisWeekend.money)
17 print(thisWeekend.friend)
```

### 2. Shopping

Solution:

```
1 class Shopping:
2
3     def __init__(self, item, quality):
```

```
4     self.item = item
5     self.quality = quality
6     self.total = []
7
8     def spending(self, cost):
9         self.total.append(cost)
10
11 sportStore = Shopping("Kayak", "High Quality")
12
13 sportStore.spending(20)
14 sportStore.spending(5)
15 sportStore.spending(10)
16
17 print(sportStore.total)
```