**1. Make a class called Thing with no contents and print it. Then, create an object called example**

**from this class and also print it. Are the printed values the same or different?**

**2. Create a new class called Thing2 and add the value &#39;abc&#39; to the letters class attribute. Letters**

**should be printed.**

**3. Make yet another class called, of course, Thing3. This time, assign the value &#39;xyz&#39; to an instance**

**(object) attribute called letters. Print letters. Do you need to make an object from the class to do**

**this?**

**4. Create an Element class with the instance attributes name, symbol, and number. Create a class**

**object with the values &#39;Hydrogen,&#39; &#39;H,&#39; and 1.**

**5. Make a dictionary with these keys and values: &#39;name&#39;: &#39;Hydrogen&#39;, &#39;symbol&#39;: &#39;H&#39;, &#39;number&#39;: 1. Then,**

**create an object called hydrogen from class Element using this dictionary.**

**6. For the Element class, define a method called dump() that prints the values of the object’s**

**attributes (name, symbol, and number). Create the hydrogen object from this new definition and**

**use dump() to print its attributes.**

**7. Call print(hydrogen). In the definition of Element, change the name of method dump to \_\_str\_\_,**

**create a new hydrogen object, and call print(hydrogen) again.**

**8. Modify Element to make the attributes name, symbol, and number private. Define a getter**

**property for each to return its value.**

**9. Define three classes: Bear, Rabbit, and Octothorpe. For each, define only one method: eats(). This**

**should return &#39;berries&#39; (Bear), &#39;clover&#39; (Rabbit), or &#39;campers&#39; (Octothorpe). Create one object from**

**each and print what it eats.**

**10. Define these classes: Laser, Claw, and SmartPhone. Each has only one method: does(). This**

**returns &#39;disintegrate&#39; (Laser), &#39;crush&#39; (Claw), or &#39;ring&#39; (SmartPhone). Then, define the class Robot that**

**has one instance (object) of each of these. Define a does() method for the Robot that prints what its**

**component objects do.**

*SOLUTIONS*

*1.class Thing:*

*pass*

*print(Thing)*

*example = Thing()*

*print(example)*

The printed values are different. The first one is the class definition **Thing** and the second one is an instance of the class **Thing** created using **example = Thing()**.

*2.* *class Thing2:*

*letters = 'abc'*

*print(Thing2.letters)*

*3.* *class Thing3:*

*pass*

*example = Thing3()*

*example.letters = 'xyz'*

*print(example.letters)*

*4.* *class Element:*

*def \_\_init\_\_(self, name, symbol, number):*

*self.name = name*

*self.symbol = symbol*

*self.number = number*

*hydrogen = Element('Hydrogen', 'H', 1)*

*5.* *element\_dict = {'name': 'Hydrogen', 'symbol': 'H', 'number': 1}*

*hydrogen = Element(\*\*element\_dict)*

*6.* *class Element:*

*def \_\_init\_\_(self, name, symbol, number):*

*self.name = name*

*self.symbol = symbol*

*self.number = number*

*def dump(self):*

*print(f"name: {self.name}, symbol: {self.symbol}, number: {self.number}")*

*hydrogen = Element('Hydrogen', 'H', 1)*

*hydrogen.dump()*

*7.* *class Element:*

*def \_\_init\_\_(self, name, symbol, number):*

*self.name = name*

*self.symbol = symbol*

*self.number = number*

*def \_\_str\_\_(self):*

*return f"name: {self.name}, symbol: {self.symbol}, number: {self.number}"*

*hydrogen = Element('Hydrogen', 'H', 1)*

*print(hydrogen)*

*8.* *class Element:*

*def \_\_init\_\_(self, name, symbol, number):*

*self.\_name = name*

*self.\_symbol = symbol*

*self.\_number = number*

*@property*

*def name(self):*

*return self.\_name*

*@property*

*def symbol(self):*

*return self.\_symbol*

*@property*

*def number(self):*

*return self.\_number*

*9.* *class Bear:*

*def eats(self):*

*return 'berries'*

*class Rabbit:*

*def eats(self):*

*return 'clover'*

*class Octothorpe:*

*def eats(self):*

*return 'campers'*

*bear = Bear()*

*print(bear.eats())*

*rabbit = Rabbit()*

*print(rabbit.eats())*

*octothorpe = Octothorpe()*

*print(octothorpe.eats())*

*10.* *class Laser:*

*def does(self):*

*return 'disintegrate'*