**ANSWERS**

**Q1. 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?**

**Ans.** Please find below the code:

class Thing:

pass

print(Thing)

example = Thing()

print(example)

The printed values will be different. The first print statement will display information about the class **Thing**, and the second print statement will display information about the instance example of the class **Thing**.

**Q2. Create a new class called Thing2 and add the value 'abc' to the letters class attribute. Letters should be printed.**

**Ans.** Please find below the code:

class Thing2:

letters = 'abc'

print(Thing2.letters)

This will print **'abc'**.

**Q3. Make yet another class called, of course, Thing3. This time, assign the value 'xyz' to an instance (object) attribute called letters. Print letters. Do you need to make an object from the class to do this?**

**Ans.** Please find below the code:

class Thing3:

def \_\_init\_\_(self):

self.letters = 'xyz'

obj = Thing3()

print(obj.letters)

You need to create an object from the class to access and print the instance attribute **letters.**

**Q4. Create an Element class with the instance attributes name, symbol, and number. Create a class object with the values 'Hydrogen,' 'H,' and 1.**

**Ans.** Please find below the code:

class Element:

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

self.name = name

self.symbol = symbol

self.number = number

element\_obj = Element('Hydrogen', 'H', 1)

**Q5. Make a dictionary with these keys and values: 'name': 'Hydrogen', 'symbol': 'H', 'number': 1. Then, create an object called hydrogen from class Element using this dictionary.**

**Ans.** Please find below the code:

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

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

**Q6. 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.**

**Ans.** Please find below the code:

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()

**Q7. 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.**

**Ans.** Please find below the code:

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)

**Q8. Modify Element to make the attributes name, symbol, and number private. Define a getter property for each to return its value.**

**Ans.** Please find below the code:

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

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

print(hydrogen.name)

print(hydrogen.symbol)

print(hydrogen.number)

**Q9. Define three classes: Bear, Rabbit, and Octothorpe. For each, define only one method: eats(). This should return 'berries' (Bear), 'clover' (Rabbit), or 'campers' (Octothorpe). Create one object from each and print what it eats.**

**Ans.** Please find below the code:

class Bear:

def eats(self):

return 'berries'

class Rabbit:

def eats(self):

return 'clover'

class Octothorpe:

def eats(self):

return 'campers'

bear = Bear()

rabbit = Rabbit()

octothorpe = Octothorpe()

print(bear.eats())

print(rabbit.eats())

print(octothorpe.eats())

**Q10. Define these classes: Laser, Claw, and SmartPhone. Each has only one method: does(). This returns 'disintegrate' (Laser), 'crush' (Claw), or 'ring' (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.**

**Ans.** Please find below the code:

class Laser:

def does(self):

return 'disintegrate'

class Claw:

def does(self):

return 'crush'

class SmartPhone:

def does(self):

return 'ring'

class Robot:

def \_\_init\_\_(self):

self.laser = Laser()

self.claw = Claw()

self.phone = SmartPhone()

def does(self):

return f"Laser does: {self.laser.does()}, Claw does: {self.claw.does()}, Phone does: {self.phone.does()}"

robot = Robot()

print(robot.does())