In [9]:
```
# Q1
class Thing:
    pass

print(Thing())
example = Thing()
print(example)
```

```
<__main__.Thing object at 0x000001D504718E50>
<__main__.Thing object at 0x000001D504718E80>
```

In [10]:
```
# Q2
class Thing2:
    letters = 'abc'

print(Thing2.letters)
```

```
abc
```

In [11]:
```
# Q3
class Thing3:
    def __init__(self):
        self.letters = 'xyz'

example = Thing3()
print(example.letters)
```

```
xyz
```

In [13]:
```
# Q4
class Element:
    def __init__(self, name, symbol, number):
        self.name = name
        self.symbol = symbol
        self.number = number

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

In [14]:
```python
# Q5
class Element:
    def __init__(self, name, symbol, number):
        self.name = name
        self.symbol = symbol
        self.number = number

# Define the dictionary
hydrogen_dict = {'name': 'Hydrogen', 'symbol': 'H', 'number': 1}

# Create an instance of the Element class using the dictionary
hydrogen = Element(**hydrogen_dict)

# Print the attributes of the hydrogen object to verify it was created correctl
print(hydrogen.name)
print(hydrogen.symbol)
print(hydrogen.number)
```

```
Hydrogen
H
1
```

In [15]:
```python
# Q6
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}\nSymbol: {self.symbol}\nNumber: {self.number}

# Create an instance of the Element class
hydrogen = Element('Hydrogen', 'H', 1)

# Call the dump() method to print the attributes of the hydrogen object
hydrogen.dump()
```

```
Name: Hydrogen
Symbol: H
Number: 1
```

In [16]:
```python
# Q7
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}\nSymbol: {self.symbol}\nNumber: {self.number

# Create an instance of the Element class
hydrogen = Element('Hydrogen', 'H', 1)

# Call print() on the hydrogen object to print its string representation
print(hydrogen)

# Create a new instance of the Element class
new_hydrogen = Element('New Hydrogen', 'NH', 2)

# Call print() on the new_hydrogen object to print its string representation
print(new_hydrogen)
```

```
Name: Hydrogen
Symbol: H
Number: 1
Name: New Hydrogen
Symbol: NH
Number: 2
```

In [17]:
```
# Q8
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

    def __str__(self):
        return f"Name: {self.name}\nSymbol: {self.symbol}\nNumber: {self.number

# Create an instance of the Element class
hydrogen = Element('Hydrogen', 'H', 1)

# Call the getter properties to print the attributes of the hydrogen object
print(hydrogen.name)
print(hydrogen.symbol)
print(hydrogen.number)
```

```
Hydrogen
H
1
```

In [18]:
```
# Q9
class Bear:
    def eats(self):
        return 'berries'

class Rabbit:
    def eats(self):
        return 'clover'

class Octothorpe:
    def eats(self):
        return 'campers'

# Create an object from each class
bear = Bear()
rabbit = Rabbit()
octothorpe = Octothorpe()

# Print what each object eats
print(f"The bear eats {bear.eats()}")
print(f"The rabbit eats {rabbit.eats()}")
print(f"The octothorpe eats {octothorpe.eats()}")
```

```
The bear eats berries
The rabbit eats clover
The octothorpe eats campers
```

In [22]:
```
# Q10
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.smartphone = SmartPhone()

    def does(self):
        print(f"The robot's laser {self.laser.does()}.")
        print(f"The robot's claw {self.claw.does()}.")
        print(f"The robot's smartphone {self.smartphone.does()}.")
robot = Robot()
robot.does()
```

```
The robot's laser disintegrate.
The robot's claw crush.
The robot's smartphone ring.
```

In [ ]: