In [1]:
```
class Person:
    def __init__(self, name):
        self.name = name
    def __str__(self):
        return f"Person: {self.name}"
    def __repr__(self):
        return f"Person('{self.name}')"

p = Person("Alice")
print(p)      # Person: Aliceprint(repr(p))# Person('Alice')
```

```
Person: Alice
```

In [3]:
```
class Demo:
    def __init__(self, name):
        self.name = name
    def __del__(self):
        print(f"{self.name} is deleted!")

obj = Demo("Test")
del obj   # Output: Test is deleted!
```

```
Test is deleted!
```

In [11]:
```
class Point:
    def __init__(self, x):
        self.x = x
    def __sub__(self, other):
        return Point(self.x - other.x)
    def __str__(self):
        return f"Point({self.x})"
print(Point(3) - Point(7))   # Point(10)
```

```
Point(-4)
```

In [15]:
```
class Student:
    def __init__(self, marks):
        self.marks = marks
    def __gt__(self, other):
        return self.marks > other.marks
print(Student(80) > Student(90))  # True
```

```
False
```

In [17]:
```
class MyList:
    def __init__(self, data):
        self.data = data
    def __getitem__(self, index):
        return self.data[index]
    def __len__(self):
        return len(self.data)

nums = MyList([10, 20, 30])
print(len(nums))   # 3print(nums[1])     # 20
```

```
3
```

In [ ]:
```
class Greet:
    def __call__(self, name):
        return f"Hello {name}!"

say = Greet()print(say("Alice"))   # Hello Alice!
```

In [21]:
```
class MyContext:
    def __enter__(self):
        print("Entering...")
    def __exit__(self, exc_type, exc_val, exc_tb):
        print("Exiting...")
with MyContext():
    print("Inside block")
```

```
Entering...
Inside block
Exiting...
```

In [19]:
```
class Greet:
    def __call__(self, name):
        return f"Hello {name}!"

say = Greet()
print(say("Alice"))
```

```
Hello Alice!
```

In [ ]: