In [1]:

```python
# Create a Time class with hours and minutes.

# Overload the + operator to add two Time objects correctly.

class Time:
    def __init__(self, hours, minutes):
        self.hours = hours
        self.minutes = minutes

    def __str__(self):
        # Return a string representation of the time in "hh:mm" format
        return f"{self.hours:02d}:{self.minutes:02d}"

    def __add__(self, other):
        # Add the hours and minutes of two Time objects
        total_minutes = self.minutes + other.minutes
        total_hours = self.hours + other.hours

        # If total minutes exceed 60, adjust the hours and minutes
        total_hours += total_minutes // 60
        total_minutes = total_minutes % 60

        # Return a new Time object with the calculated hours and minutes
        return Time(total_hours, total_minutes)

# Create two Time objects
time1 = Time(2, 45)  # 2 hours 45 minutes
time2 = Time(3, 30)  # 3 hours 30 minutes

# Add the two Time objects using the overloaded + operator
result = time1 + time2

# Print the result
print(f"Time 1: {time1}")
print(f"Time 2: {time2}")
print(f"Result of addition: {result}")
```

```
Time 1: 02:45
Time 2: 03:30
Result of addition: 06:15
```

In [2]:

```python
# Create a Distance class with attributes feet and inches.

# Overload the * operator to multiply the distance by a scalar value.(any numeric value)

class Distance:
    def __init__(self, feet, inches):
        self.feet = feet
        self.inches = inches

    def __str__(self):
        # Return the string representation of the distance in "feet' inches" format
        return f"{self.feet} feet {self.inches} inches"

    def __mul__(self, scalar):
        # Multiply the distance by a scalar (numeric value)
        total_inches = (self.feet * 12 + self.inches) * scalar
        new_feet = total_inches // 12
        new_inches = total_inches % 12

        # Return a new Distance object with the calculated feet and inches
        return Distance(new_feet, new_inches)

# Create a Distance object
distance1 = Distance(5, 9)  # 5 feet 9 inches

# Multiply the Distance object by a scalar value (e.g., 3)
result = distance1 * 3

# Print the result
print(f"Original Distance: {distance1}")
print(f"Result of multiplying by 3: {result}")
```

```
Original Distance: 5 feet 9 inches
Result of multiplying by 3: 17 feet 3 inches
```

In [3]:

```python
# Create a Rectangle class with length and width.

# Overload the == operator to compare the area of two rectangles.

class Rectangle:
    def __init__(self, length, width):
        self.length = length
        self.width = width

    def __str__(self):
        # Return a string representation of the rectangle with length and width
        return f"Rectangle({self.length} x {self.width})"

    def area(self):
        # Calculate the area of the rectangle
        return self.length * self.width

    def __eq__(self, other):
        # Overload the == operator to compare the area of two rectangles
        if isinstance(other, Rectangle):
            return self.area() == other.area()
        return False

# Create two Rectangle objects
rectangle1 = Rectangle(4, 5)  # Area = 20
rectangle2 = Rectangle(2, 10)  # Area = 20
rectangle3 = Rectangle(3, 7)   # Area = 21

# Compare the areas of the rectangles using the overloaded == operator
print(f"Rectangle1 == Rectangle2: {rectangle1 == rectangle2}")  # Should be True
print(f"Rectangle1 == Rectangle3: {rectangle1 == rectangle3}")  # Should be False
```

```
Rectangle1 == Rectangle2: True
Rectangle1 == Rectangle3: False
```

In [ ]: