Video 19 : Magic Methods

Magic methods allow you to define how objects of the same object type can be compared. They also allow you to define what happens when mathematical operations are performed on your objects in awesome ways!

Magic methods are surrounded by double underscores. We can use magic methods to define how operators like +, -, *, /, ==, >, <, etc. will work with our custom objects.

Magic methods are used for operator overloading in Python. Here are the magic methods you can manipulate :

```
# __eq__ : Equal
# __ne__ : Not Equal
# __lt__ : Less Than
# __gt__ : Greater Than
# __le__ : Less Than or Equal
# __ge__ : Greater Than or Equal
# __add__ : Addition
# __sub__ : Subtraction
# __mul__ : Multiplication
# __truediv__ : Division
# __mod__ : Modulus
```

In this example I'll create a Time class. I'll then create custom methods that define how your Time objects are printed and added.

CODE

```
class Time:
    def __init__(self, hour=0, minute=0, second=0):
        self.hour = hour
        self.minute = minute
        self.second = second

    # Magic method that defines the string format of the object
    def __str__(self):
        # :02d adds a leading zero to have a minimum of 2 digits
        return "{}:{:02d}:{:02d}".format(self.hour,self.minute, self.second)

    def __add__(self, other_time):
        new_time = Time()

        # ---------- PROBLEM ----------
        # How would you go about adding 2 times together?

        # Add the seconds and correct if sum is >= 60
        if (self.second + other_time.second) >= 60:
            self.minute += 1
            new_time.second = (self.second + other_time.second) - 60
        else:
            new_time.second = self.second + other_time.second
```

```
        # Add the minutes and correct if sum is >= 60
        if (self.minute + other_time.minute) >= 60:
            self.hour += 1
            new_time.minute = (self.minute + other_time.minute) - 60
        else:
            new_time.minute = self.minute + other_time.minute

        # Add the minutes and correct if sum is > 60
        if (self.hour + other_time.hour) > 24:
            new_time.hour = (self.hour + other_time.hour) - 24
        else:
            new_time.hour = self.hour + other_time.hour
        return new_time

def main():
    time1 = Time(1, 20, 30)
    print(time1)
    time2 = Time(24, 41, 30)
    print(time1 + time2)

    # For homework get the Time objects to work for the other
    # mathematical and comparison operators

main()
```