Chap17_TimePoint_Theory&Exercises_Part1

May 15, 2020

In [ ]:
```
# Chapter 17. Classes and Methods
# First part: Object-oriented features, Printing objects, The special methods,
#             Operator overloading
```

In [1]:
```
#Printing objects
class Time(object):
    """ Represents the time of day"""
def  print_time(time):
    print '%.2d:%.2d:%.2d'%(time.hour, time.minutes, time.second)
start=Time()
start.hour=9
start.minutes=45
start.second=00
print_time(start)
```

```
09:45:00
```

In [2]:
```
# Exercise 17.1
# rewrite time_to_int as a METHOD

class Time(object):
    def time_to_int(self):
        minutes = time.hour * 60 + time.minute
        seconds = minutes * 60 + time.second
        return seconds
    # It is not appropriate to rewrite <int-to-time()> as a method
    # because there is no object to invoke it on :)
    # <int_to_time()> remains a function in a complete program;

time = Time()
time.hour = 9
time.minute = 55
time.second = 50

print time.time_to_int()
```

```
35750
```

```
In [5]: # Example from Section 17.3
        # increment as a METHOD (it rewrites the function from Section 16.3)
        class Time(object):
            def print_time(self):
                print '%.2d:%.2d:%.2d'%(self.hour,self.minute,self.second)
            def time_to_int(self):
                return self.hour*3600+self.minute*60+self.second
            def increment(self,seconds):
                seconds+=self.time_to_int()
                return int_to_time(seconds)
        def int_to_time(seconds):
            t = Time()
            minutes,t.second = divmod(seconds,60)
            t.hour,t.minute = divmod(minutes,60)
            return t

        start = Time()
        start.hour=9
        start.minute=45
        start.second=00
        start.print_time()
        end = start.increment(1337)
        end.print_time()
```

```
09:45:00
10:07:17
```

```
In [8]: # Example from Section 17.4
        # is_after as a METHOD (it rewrites the function from Section 16.2)
        class Time(object):
            def print_time(self):
                print '%.2d:%.2d:%.2d'%(self.hour,self.minute,self.second)
            def time_to_int(self):
                return self.hour*3600+self.minute*60+self.second
            def increment(self,seconds):
                seconds+=self.time_to_int()
                return int_to_time(seconds)
            def is_after(self,other):
                return self.time_to_int() > other.time_to_int()
        def int_to_time(seconds):
            t = Time()
            minutes,t.second = divmod(seconds,60)
            t.hour,t.minute = divmod(minutes,60)
            return t

        start = Time()
        start.hour=9
```

```
        start.minute=45
        start.second=00
        start.print_time()
        end = start.increment(1337)
        end.print_time()

        print end.is_after(start)
        #!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
        #end = becomes subject = the first parameter of is_after
        #start = remains object = the second parameter of is_after
```

```
09:45:00
10:07:17
True
```

```
In [19]: #Metode speciale
         class Time(object):
             def print_time(self):
                 print '%.2d:%.2d:%.2d'%(self.hour,self.minute,self.second)
             def __init__(self,hour=0,minute=0,second=0):
                     self.hour = hour
                     self.minute = minute
                     self.second =second
             def __str__(self):
                 return '%.2d:%.2d:%.2d' % (self.hour, self.minute, self.second)
         time=Time()
         time.print_time()
         time=Time(9,50)
         time.print_time()
         time=Time(9,45)
         print time
```

```
00:00:00
09:50:00
09:45:00
```

```
In [1]: # Exercise 17.2
        # add the special method __init__ to the class POINT from Chapter 15

        class Point(object):
            def __init__(self, x=0, y=0):
                self.x = x
                self.y = y

            def print_point(self):
                print "x =", self.x, ",",
                print "y =", self.y
```

```
        point = Point()
        point.print_point()

        point = Point(30)
        point.print_point()

        point = Point(40, 50)
        point.print_point()
```

```
x = 0 , y = 0
x = 30 , y = 0
x = 40 , y = 50
```

```
In [4]: # Exercise 17.3
        # add the special method __str__ to the class POINT from Chapter 15

        class Point(object):
            def __init__(self, x=0, y=0):
                self.x = x
                self.y = y

            def __str__(self):
                return '(%d, %d)' % (self.x, self.y)

        point = Point()
        print point

        point = Point(20)
        print point

        point = Point(20, 25)
        print point
```

```
(0, 0)
(20, 0)
(20, 25)
```

```
In [5]: # Exercise 17.4
        # add the special method __add__ to the class POINT from Chapter 15
        class Point(object):
            def __init__(self, x=0, y=0):
                self.x = x
                self.y = y

            def __str__(self):
                return '(%d, %d)' % (self.x, self.y)
```

```
            def __add__(self, other):
                x = self.x + other.x
                y = self.y + other.y
                return Point(x, y)

        point1 = Point(4, 5)
        point2 = Point(7, 9)

        print point1 + point2
```

```
(11, 14)
```