Lab. Python's class basic II

1. 사용 tool
   -Jupyter Notebook
   -Microsoft Visual Studio Code

2. Code

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

       def print_pt(self):
      print('({}, {})'.format(self.x, self.y))

       def add(self, pt):
      new_x = self.x + pt.x
      new_y = self.y + pt.y
      return Point(new_x, new_y)

       def multiply(self, factor):
      return Point(self.x * factor, self.y * factor)

       def length(self):
      return self.x ** 2 + self.y ** 2

       def get_x(self):
      return self.x

       def get_y(self):
      return self.y

       #Base Overloading Methods
       def __str__(self):
      return '({}, {})'.format(self.x, self.y)

       def __add__(self, pt):
      new_x = self.x + pt.x
      new_y = self.y + pt.y
      return Point(new_x, new_y)

       def __sub__(self, pt):
      new_x = self.x - pt.x
      new_y = self.y - pt.y
      return Point(new_x, new_y)

       def __mul__(self, factor):
      return Point(self.x * factor, self.y * factor)

       def __len__(self):
```

```
    return self.x ** 2 + self.y ** 2

     def __getitem__(self, index):
    if index == 0 : return self.x
    elif index == 1 : return self.y


p1 = Point(100, 200)
p2 = Point(300, 450)

p1.print_pt()   #(100, 200)
p2.print_pt()   #(300, 450)

print(p1)   #(100, 200)   call __str__()
print(p2)   #(300, 450)

p3 = p1.add(p2)
print(p3)     #(400, 650)

p4 = p1 + p2   #call __add__()
print(p4)     #(400, 650)

p5 = p2 - p1  #call __sub__()
print(p5)       #(200, 250)

p6 = p1.multiply(7)
print(p6)      #(700, 1400)

p7 = p1 * 7
print(p7)       #(700, 1400)


print('p1\'s length =', p1.length())   #p1's length = 50000
print('p1\'s length =', len(p1))      #call __len__()  p1's length = 50000

print('p1(x, y) = ({}, {})'.format(p1.get_x(), p1.get_y()))   #p1(x, y) = (100, 200)
print('p1(x, y) = ({}, {})'.format(p1[0], p1[1]))   #p1(x, y) = (100, 200)
```

Refer to 3.3.8. Emulating numeric types(https://docs.python.org/3/reference/datamodel.html?object.__add__#emulating-numeric-types)