Exercise_2_solutions

January 5, 2019

```
In [0]: import math
```

1.

```
In [0]: class Line:

          def __init__(self, coor1, coor2):
            self.x1, self.y1 = coor1
            self.x2, self.y2 = coor2

          def distance(self):
            return ((self.x2 - self.x1)**2 + (self.y2 - self.y1)**2)**0.5

          def slope(self):
            return (self.y2 - self.y1) / (self.x2 - self.x1)

        li = Line((3,2), (8,10))
        print(li.distance())
        print(li.slope())
```

```
9.433981132056603
1.6
```

2.

```
In [0]: class Cylinder:

          def __init__(self, height=1, radius=1):
            self.height = height
            self.radius = radius

          def volume(self):
            return math.pi*self.radius**2*self.height

          def surface_area(self):
            return 2*math.pi*self.radius*(self.height + self.radius)
```

```
c = Cylinder(2,3)
print(c.volume())
print(c.surface_area())
```

```
56.548667764616276
94.24777960769379
```

3.

```
In [19]: # metody min, max, avg zostay zastpione jedn globaln metod statistics
         # zaoyem ponadto, e wszystkie pola numeryczne w pliku s typu int

         def avg(seq):
             return sum(seq)/len(seq)

         class DataFile(object):

           def __init__(self, filename):
             with open(filename, 'r') as file:
               self.__names_init(file)
               self.__values_init(file)

           def __names_init(self, file):
             self.names = file.readline().strip().split(';')

           def __values_init(self, file):
             self.values = []
             for line in file:
               values = []
               for v in line.strip().split(';'):
                 try:
                   values.append(int(v))
                 except ValueError:
                   values.append(v)
               self.values.append(values)

           def statistic(self, colnum=0, colname='', func=min):
             if(colname):
               colnum = self.names.index(colname)

             col_values = [row_values[colnum] for row_values in self.values]
             if all([isinstance(v, int) for v in col_values]):
               return func(col_values)
             else:
               None
```

```
  def info(self):
    print("{:>20}{:>10}{:>10}".format('Min', 'Max', 'Avg'))
    for name in self.names:
      col_min, col_max, col_avg = self.statistic(colname=name), self.statistic(colname
      if(col_min and col_max and col_avg):
        print("{:10}{:>10}{:>10}{:>10.2f}".format(name, col_min, col_max, col_avg))
      else:
        print("{:10}{:>10}{:>10}{:>10}".format(name, '-', '-', '-'))

df = DataFile('myfile.csv')
df.info()
```

```
                 Min       Max       Avg
Name               -         -         -
Age                4         8      6.00
Weight            18        32     25.00
Height            98       138    119.67
```

In [0]: