Oefening 1

Collapse context

In deze reeks maken we oefeningen op klassen. Wanneer in de opgave enkel gevraagd wordt om een bepaalde methode te implementeren, dan is het voldoende om enkel deze methode te geven in het code block. De grader gaat alle nodige stukken code uit vorige opdrachten halen en bij elkaar zetten. Het is daarom belangrijk om de eerste oefeningen in volgorde te maken en ervoor te zorgen dat je oplossingen correct zijn. Merk op dat nadat je een klasse gemaakt hebt, de volgende oefeningen telkens functies zijn met als eerste parameter `self`.

Voorbeeld

Oefening 1:

```
class User:
    def __init__(self,naam):
        self.naam = naam
```

Oefening 2:

```
def toon_naam(self):
    print(self.naam)
```

Your answer passed the tests! Your score is 100.0%. [Submission #6719232b3baf09455ab93234 (**2024-10-23 16:24:11**)]

Question 1: Maak een nieuwe klasse aan

✓ Perfect

Maak een klasse `Point` aan die twee coördinaten als members heeft: `x` en `y`. Zorg ervoor dat deze members geïnitialiseerd worden in de constructor.

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

Submit

Question 2: Print

✓ Perfect

Maak een methode print voor de klasse `Point` die het punt als een tuple afdrukt op het scherm.

```
>>> Point(3, 5).print()
(3,5)
```

Schrijf enkel deze methode, de rest van de klasse definitie wordt automatisch overgenomen uit je antwoorden van vorige vragen.

```
def print(self):
    print(f"({self.x},{self.y})")
```

Submit

Question 3: Reflect x

✓ Perfect

Voeg een methode `reflect_x` toe aan `Point` die een nieuw `Point` teruggeeft, dat de spiegeling van het punt is ten opzichte van de x-as. Voorbeeld: de reflectie van punt (3,5) is (3,-5).

```
def reflect_x(self):
    return Point(self.x, -self.y)
```

Submit

Question 4: Distance

✓ Perfect

Voeg een methode `distance` toe die een extra `Point` als argument meekrijgt en de afstand tussen de twee punten teruggeeft.

Hint: we hebben `sqrt` uit de module `math` voor jou reeds geïmporteerd.

```
from math import sqrt
def distance(self, p):
    return sqrt((p.x-self.x)**2 + (p.y-self.y)**2)
```

Submit

Question 5: Compute line to

✓ Perfect

De vergelijking voor een rechte is $y = ax + b$. De coëfficiënten `a` en `b` beschrijven de rechte volledig. Voeg een methode `compute_line_to` toe aan `Point` zodat, als een ander punt gegeven is, het de vergelijking voor de rechte die de twee punten verbindt teruggeeft. Je methode moet enkel de twee coëfficiënten `a` en `b` als een tuple teruggeven.

```
>>> print(Point(4,11).compute_line_to(Point(6,15)))
(2.0, 3.0)
```

```
1  def compute_line_to(self, p):
2      slope = (p.y - self.y) / (p.x - self.x)
3      return (slope, p.y - slope * p.x)
```

Submit