Location filter

Preparation

Python functions are first-class objects, so that they can be used as any other object. In particular, functions can be assigned to variables or passed as arguments to other functions. As an example, consider the following three functions.

```
def repeat(value, func, count):
    for i in range(count):
        value = func(value)
    return value

def increase(value):
    return value + 1

def decrease(value):
    return value - 1
```

The following interactive Python sessions illustrates how these functions can be used. Start an interactive Python session of your own, define the above functions, and then inspect how Python responds if you call the functions as given below. Before you start with the actual assignment, you should make sure that you fully understand why a certain value is returned and what exactly happens in the interactive session.

```
>>> increase(5)
>>> decrease(101)
>>> repeat(5, increase, 3)
>>> repeat(5, decrease, 3)
>>> repeat(5, decrease, increase(3))
```

Introduction

If you think of a circle, you probably have in mind a round drawing that can be made by a compass. However, circles are only round if we use the *Euclidean distance*. A circle is defined as the set of points in a plane that are at a given distance from a given point (the centre of the circle). If the distance concept is redefined, then the geometry of circles also changes.

But why should we redefine the distance concept? The Euclidean distance corresponds to the distance from a bird's eye view. Such a distance is not always useful in practical applications. *Graph distance*, for example, defines distance based on a given road map. This way of computing distances is slightly more complicated and requires knowledge of the road map at hand. In this exercises we will — apart from the Euclidean distance — consider two alternative ways to compute distances, which are slightly less complicated than the graph distance.

The *Manhattan distance* owes its name to the typical pattern of streets in Manhattan where all street either are perpendicular or parallel to each other. In this case, the distance is given as the sum of the difference of the x-co-ordinates and the difference of the y-co-ordinates. This corresponds to the path you would take to go from one point to another point if you could only walk parallel to the x-axis and the y-axis.

The *Chebyshev distance* or *chessboard distance* owes its name to the fact that it corresponds to

the number of moves a king needs to go on a chessboard to go from one point to another. In this case, the distance is given as the maximum of the difference of the x-co-ordinates and the difference of the y-co-ordinates.

The following table gives an overview of the different formulas that compute the distance between two points (x_1, y_1) and (x_2, y_2) as defined by the different distance measures. Based on these formulas, can you predict how a circle will look like for each of the different distance measures?

name	formula
Euclidean distance	$d=\sqrt{(x_1-x_2)^2 + (y_1-y_2)^2}$
Manhattan distance	$d=\|x_1-x_2\| + \|y_1-y_2\|$
chessboard distance	$d=\max(\|x_1-x_2\|,\|y_1-y_2\|)$

As a result, if we say that a given location is within a given radius around a point P, we actually should also specify what distance measure we are using. This is what we will do in this exercise.

Assignment

1. Write three functions `euclideanDistance`, `manhattanDistance` and `chessboardDistance` corresponding to the same distance concepts as defined in the above table. Each of these functions takes exactly two arguments. Each argument is a tuple of two floating point numbers that represent the x- and the y-coordinate of a point in the plane. The function must return the distance between both points as a floating point number, in accordance with the corresponding distance measure.
2. Write a function `locationFilter` that has two mandatory and two optional parameters. It is mandatory to pass a centre (as a tuple of co-ordinates) and a list of points (as a list of tuples of co-ordinates) to the function. In addition, it should also be possible to pass a radius and a distance function. By default, the latter two arguments are set to 1 and the Euclidean distance. The function must return a new list that only contains the points that are in or on the circle that is defined by the given radius and distance function.

Check the example below for more details on how the functions can be used and how they should behave. After submitting your solution, you will get to see several maps give a graphical representation of the circles and the points that are in or on the circle according to your implementation of the function `locationFilter`.

Example

```
>>> euclideanDistance((0, 0), (3, 4))
5.0
>>> manhattanDistance((0, 0), (3, 4))
7.0
>>> chessboardDistance((0, 0), (3, 4))
4.0
>>> points = [(0, 0), (1, 0), (0, 1), (1, 1), (0, -1), (-1, 1)]
```

```
>>> locationFilter((0, 0), points)
[(0, 0), (1, 0), (0, 1), (0, -1)]
>>> locationFilter((0, 0), points, 2)
[(0, 0), (1, 0), (0, 1), (1, 1), (0, -1), (-1, 1)]
>>> locationFilter((0, 0), points, 1, chessboardDistance)
[(0, 0), (1, 0), (0, 1), (1, 1), (0, -1), (-1, 1)]
>>> locationFilter((1, 0), points, 2, manhattanDistance)
[(0, 0), (1, 0), (0, 1), (1, 1), (0, -1)]
```

Voorbereiding

In Python zijn functies zelf ook objecten, waardoor je ze kunt gebruiken zoals alle andere objecten. In het bijzonder kun je functies toekennen aan variabelen of doorgeven als argument aan andere functies. Bekijk bijvoorbeeld onderstaande drie functies.

```
def herhalen(waarde, functie, aantal):
    for i in range(aantal):
        waarde = functie(waarde)
    return waarde

def verhogen(waarde):
    return waarde + 1

def verlagen(waarde):
    return waarde - 1
```

Hieronder staat een voorbeeld van hoe deze functies gebruikt kunnen worden. Ga na hoe Python reageert als je achtereenvolgens de volgende instructies uitvoert binnen een interactieve Python sessie waarin je eerst zorgt dat bovenstaande functies gedefinieerd werden. Verzeker jezelf ervan dat je goed begrijpt waarom je een bepaalde waarde krijgt en wat er juist gebeurt in de interactieve sessie vooraleer je verder gaat met de eigenlijke opgave.

```
>>> verhogen(5)
>>> verlagen(101)
>>> herhalen(5, verhogen, 3)
>>> herhalen(5, verlagen, 3)
>>> herhalen(5, verlagen, verhogen(3))
```

Inleiding

Als je aan een cirkel denkt, dan denk je meteen aan de ronde tekening die je met een passer kan maken. Dat is echter alleen maar zo omdat we zo gewoon zijn om te werken met de *Euclidische afstand*. Een cirkel wordt gedefinieerd als een verzameling van punten die allemaal op een bepaalde afstand liggen van een gegeven punt (het centrum of middelpunt van de cirkel). Als je het concept afstand zou herdefiniëren dan verandert ook het uitzicht van een cirkel.

Maar waarom zou je het concept afstand willen herdefiniëren? De Euclidische afstand komt overeen met de afstand in vogelvlucht. Deze is niet altijd even bruikbaar bij praktische problemen. Daarom heb je bijvoorbeeld de *graafafstand* die de afstand definieert aan de hand van de afstand via een wegennet. Deze manier om een afstand te berekenen is echter iets ingewikkelder en vraagt de kennis van het weggennet *in casu*. We gaan in deze opgave — naast de Euclidische afstand — twee andere manieren beschouwen om de afstand te berekenen, die iets gemakkelijker zijn dan deze graafafstand.

De *Manhattan-afstand* dankt zijn naam aan het typische stratenpatroon van Manhattan waar alle straten ofwel loodrecht op elkaar staan ofwel evenwijdig met elkaar lopen. In dit geval wordt de afstand gegeven als de som van het verschil van de x-coördinaten en het verschil van de y-coördinaten. Dit komt overeen met de weg die je zou afleggen om van het ene punt naar het andere punt te gaan als je enkel evenwijdig met de x-as en de y-as zou mogen wandelen.

De *Chebyshev-afstand* of de *schaakbord-afstand* dankt zijn naam aan het feit dat op een discreet rooster deze afstand overeenkomt met het aantal zetten dat het schaakstuk koning nodig heeft om van het ene punt naar het andere punt te gaan. De afstand wordt in dit geval gegeven door het maximum van het verschil van de x-coördinaten en het verschil van de y-coördinaten.

In onderstaande tabel vind je een overzicht van de verschillende formules voor de afstand tussen twee punten (x_1, y_1) en (x_2, y_2) zoals die wordt gedefinieerd door deze verschillende afstandsmaten. Kan je op basis van deze formules voorspellen hoe een cirkel er zal uitzien voor elk van deze vormen van afstand?

naam	**formule**
Euclidische afstand	$d=\sqrt{(x_1-x_2)^2 + (y_1-y_2)^2}$
Manhattan-afstand	$d=\|x_1-x_2\| + \|y_1-y_2\|$
schaakbord-afstand	$d=\max(\|x_1-x_2\|,\|y_1-y_2\|)$

Als je dus zegt dat een bepaalde plaats binnen een gegeven straal rond het punt P ligt, dan moet je er eigenlijk ook bijzeggen welke definitie van afstand je gebruikt. Dit is dan ook wat we in deze opgave gaan doen.

Opgave

1. Schrijf drie functies euclidischeAfstand, manhattanAfstand en schaakbordAfstand die corresponderen met de gelijknamige concepten voor het begrip afstand zoals die in bovenstaande tabel gedefinieerd werden. Aan elk van deze functies moeten verplicht twee argumenten doorgegeven worden. Elk argument is een tuple van 2 reële getallen die de x- en de y-coördinaat van een punt in het vlak voorstellen. Als resultaat geven deze functies telkens de afstand tussen de twee gegeven punten terug als een reëel getal, overeenkomstig de corresponderende definitie van afstand.
2. Schrijf een functie locatieFilter die twee verplichte parameters en twee optionele parameters heeft. Verplicht moeten eerst een centrum als een tuple van coördinaten en een lijst van punten (dit is dus een lijst van tuples van coördinaten) doorgegeven worden. Vervolgens kan je ook een straal en een afstandsfunctie doorgeven. Standaard worden deze laatste argumenten ingesteld op respectievelijk 1 en de Euclidische afstand. Deze functie geeft dan een nieuwe lijst terug die enkel die punten bevat die op of binnen de cirkel liggen die beschreven wordt door de straal en de afstandsfunctie.

Bekijk zeker ook onderstaand voorbeeld voor meer details over hoe de functies kunnen gebruikt worden en hoe ze zich moeten gedragen. Na het indienen van een oplossing krijg je enkele kaartjes te zien die de verschillende cirkels tonen en de locaties die volgens jouw functie binnen

deze cirkels gelegen zijn.

Voorbeeld

```
>>> euclidischeAfstand((0, 0), (3, 4))
5.0
>>> manhattanAfstand((0, 0), (3, 4))
7.0
>>> schaakbordAfstand((0, 0), (3, 4))
4.0
>>> punten = [(0, 0), (1, 0), (0, 1), (1, 1), (0, -1), (-1, 1)]
>>> locatieFilter((0, 0), punten)
[(0, 0), (1, 0), (0, 1), (0, -1)]
>>> locatieFilter((0, 0), punten, 2)
[(0, 0), (1, 0), (0, 1), (1, 1), (0, -1), (-1, 1)]
>>> locatieFilter((0, 0), punten, 1, schaakbordAfstand)
[(0, 0), (1, 0), (0, 1), (1, 1), (0, -1), (-1, 1)]
>>> locatieFilter((1, 0), punten, 2, manhattanAfstand)
[(0, 0), (1, 0), (0, 1), (1, 1), (0, -1)]
```