LandkartenErstellung

API Documentation

May 17, 2019

Contents

1 Module Berechnung

1.1 Class Berechnung

1.1.1 Methods

distance(*self*, *landA*, *landB*)

Berechnet den Kreisabstand zwischen landA und landB anhand der Radien und des direkten Abstandes

berechne(*self*, *karte*)

Abstrakte Methode der Berechnung Muss in der abgeleiteten Klasse implementiert werden

normVektor(*self*, *landA*, *landB*)

Berechnet den normierten Vektor zwischen den Mittelpunkten von landA und landB

1.2 Class StandardBerechnung

Berechnung.Berechnung
└ **Berechnung.StandardBerechnung**

1.2.1 Methods

mittelwertRadien(*self*, *karte*)

Berechnet den Mittelwert aller Radien

voriteration(*self*, *karte*, *faktor*=0)

Berechnung der Voriteration durch Skalieren der Mittelpunkte und Verkleinern der Radien

berechne(*self*, *karte*)

Berechnet einen Iterationsschritt fuer alle Laender in karte

Overrides: Berechnung.Berechnung.berechne

Inherited from Berechnung.Berechnung(Section 1.1)

distance(), normVektor()

2 Module InputReader

2.1 Functions

zusammenhaengend(*laenderliste*)

prueft ob alle Laender in der Liste zusammenhaengen; gibt true zurueck wenn alle zusammenhaengen, sonst false

zusammenhaengendRek(*startland*, *besucht*)

Rekursionsfunktion zum pruefen ob alle laender zusammenhaengen void

addNachbarnToLand(*self*, *nachbarn*, *heimatland*)

fuegt einem Land ein anderes als Nachbarn hinzu

2.2 Variables

Name	Description
__package__	**Value:** `None`

2.3 Class InputReader

2.3.1 Methods

__init__(*self*, *filename*)

Konstruktor erhaelt den Dateinamen der Eingabedatei und speichert ihn ab

read(*self*)

liest die Eingabedatei ein und gibt ein Objekt von Karte mit allen eingelesenen Informationen zurueck

2.3.2 Class Variables

Name	Description
mFilename	dateiname fuer eingabedatei **Value:** ''

3 Module Karte

3.1 Variables

Name	Description
__package__	**Value:** `None`

3.2 Class Karte

3.2.1 Methods

__init__(*self*, *l*, *kennzBz*)

Konstruktor der Klasse Karte bekommt eine Liste an Laendern sowie die Kennzahlbezeichnung

scale(*self*)

Passt den Wertebereich von x und y so an, dass xmax-xmin = ymax-ymin

berechneMinimum(*self*)

Berechnet die minimalen und maximalen x- und y-Werte der Karte

3.2.2 Class Variables

Name	Description
laenderliste	**Value:** `[]`
kennzahlBz	**Value:** `''`
xmin	minimaler x-Wert der Karte **Value:** `0.0`
xmax	maximaler x-Wert der Karte **Value:** `0.0`
ymin	minimaler y-Wert der Karte **Value:** `0.0`
ymax	maximaler y-Wert der Karte **Value:** `0.0`

4 Module Land

4.1 Variables

Name	Description
__package__	**Value:** `None`

4.2 Class Land

4.2.1 Methods

__init__(*self*, *x*, *y*, *kennZe*, *kennZa*)

Konstruktor der Klasse Land bekommt Mittelpunkt (x,y) das Laenderkennzeichen kennZe und die Kennzahl kennZa

__str__(*self*)

Gibt String-Repraesentation fuer Ausgabe zurueck

__repr__(*self*)

Gibt String-Repraesentation fuer Ausgabe zurueck Wird zu Debugging-Zwecken benoetigt

addNachbar(*self*, *nachbar*)

fuegt Nachbarn der Liste hinzu

containsNachbar(*self*, *nachbar*)

True wenn nachbar ein Nachbar des self-Objektes

4.2.2 Class Variables

Name	Description
xPos	x-Position des Mittelpunktes **Value:** `0.0`
yPos	y-Position des Mittelpunktes **Value:** `0.0`
kennzeichen	Laenderkennzeichen **Value:** `''`

continued on next page

Name	Description
kennzahl	Kennzahl **Value:** `0`
kraftX	aktuelle Kraft in x-Richtung (nur waehrend einer Iteration benoetigt) **Value:** `0.0`
kraftY	aktuelle Kraft in y-Richtung (nur waehrend einer Iteration benoetigt) **Value:** `0.0`
radius	Radius des Kreises **Value:** `0.0`
nachbarlaender	Nachbarlaender **Value:** `[]`

5 Module Main

5.1 Functions

main(*arguments*)

Main-Methode zum Steuern von Input, Berechnung und Output

5.2 Variables

Name	Description
ap	**Value:** `argparse.ArgumentParser("Hello there!")`
results	**Value:** `ap.parse_args()`

5.3 Class Main

5.3.1 Methods

printUsage(*self*)

5.3.2 Class Variables

Name	Description
mFilename	string dateiname fuer ausgabe **Value:** `""`
mInput	InputReader fuer eingabe **Value:** `None`
mOutput	OutputWriter fuer ausgabe **Value:** `None`
mBerechnung	(Standard-)Berechnung fuer die Iterationsberechnung **Value:** `None`
karte	eingelesene und ggf. schon berechnete Daten **Value:** `None`
iterationen	iterationsschritte **Value:** `100`

6 Module OutputWrite

6.1 Variables

Name	Description
__package__	**Value:** `None`

6.2 Class OutputWriter

6.2.1 Methods

__init__(*self*, *iFilename*, *iTemplate*, *iKarte*, *iIterationen*)

Konstruktor fuer die Ausgabe; iFilename - Dateipfad fuer die Ausgabedatei; iTemplate - Template mit allgemeiner Form; iKarte - berechnete Karte zur Ausgabe; iIterationen - Anzahl der durchgefuehrten Iterationen

write(*self*)

Schreibt die Datei

6.2.2 Class Variables

Name	Description
mFilename	Dateipfad der Ausgabedatei **Value:** `''`
mTemplate	Dateipfad des Templates **Value:** `''`
mKarte	Karte die ausgegeben werden soll **Value:** `Karte([], "")`
mIterationen	Anzahl der berechneten Iterationen **Value:** `0`

Index