Unit 7

Dr. Günter Kolousek

21. Juli 2015

Lege wiederum ein Verzeichnis an. Nennes es `07_unit7`! In diesem Verzeichnis sollen alle Dateien der jeweiligen Einheit abgelegt werden.

1 Schulübungen

1. Schreibe eine Funktion `distance1`, die die Entfernung zweier Punkte auf der x-Achse berechnet und zurückgibt.

   Füge diese Funktion zu einer Datei `geomlib.py` hinzu. Das gilt auch für alle anderen Beispiele.

2. Schreibe ein Programm `geom.py`, das den Benutzer mit der Meldung "Willkommen beim geometrischen Wunderzwerg!" (oder wie auch immer du willst) begrüßt und danach zwei Punkte auf der x-Achse abfragt und danach das Ergebnis mittels der Funktion "distance1" berechnet und auf der Konsole ausgibt.

   Das sollte so aussehen::

   ```
   Willkommen beim geometrischen Wunderzwerg!

   Bitte geben Sie den x-Wert von Punkt 1 ein: 2
   Bitte geben Sie den x-Wert von Punkt 2 ein: 5

   Die Entfernung zwischen dem Punkt 2 und dem Punkt 5 beträgt 3.
   ```

   Verwende die `format` Methode!

3. Schreibe eine Funktion `distance2`, die die Entfernung zweier Punkte des zweidimensionalen Koordinatensystems berechnet und zurückgibt. Jeder Punkt ist durch seine x- und seine y-Koordinate gegeben.

   Wieder in die Datei `geomlib.py` damit.

4. Erweitere jetzt wieder das Programm `geom.py` um eine Abfrage, ob der Benutzer die Entwernung eindimensional oder zweidimensional berechnen will. Danach rufe wieder die entsprechende Funktion auf und geben das Ergebnis schön formattiert auf der Konsole aus.

   Die Ausgabe soll folgendermaßen aussehen::

   ```
   Bitte wählen Sie:

   (1) Berechnung der Entfernung: eindimensional
   (2) Berechnung der Entfernung: zweidimensional
   ```

   Gibt der Benutzer 1 ein, dann sollen die schon vorhandenen Anweisungen eingebaut werden. Bei der Eingabe von 2 soll das Programm sich folgendermaßen verhalten::

   ```
   Bitte geben Sie den x-Wert von Punkt 1 ein: 2
   Bitte geben Sie den y-Wert von Punkt 1 ein: 2
   Bitte geben Sie den x-Wert von Punkt 2 ein: 3
   Bitte geben Sie den y-Wert von Punkt 2 ein: 3

   Die Entfernung zwischen dem Punkt (1,1) und dem Punkt (2,2) beträgt 1.41.
   ```

   Verwende für das Runden die Funktion `round`. Studiere dazu die Hilfe mittels `help(round)`.

   Gibt der Benutzer weder `1` noch `2` soll eine Fehlermeldung ausgegeben werden und das Programm beendet sich danach.

5. Schreibe eine Funktion `distance3`, die die Entfernung zweier Punkte im dreidimensionalen Raum berechnet und das Ergebnis zurückgibt.

   Baue die entsprechenden Anweisungen ebenfalls in das Programm `geomlib.py` ein!

6. Erweitere das Programm `geom.py` so, dass jetzt für ungültige Zahlen für die x-, y- bzw. z-Achse (z.B. bei `q` anstatt `1`) eine Fehlermeldung ausgegeben wird und sich das Programm danach beendet.

   Verwende dazu `try...except`!

7. Erweitere jetzt die Datei `geomlib` um eine Funktion `regular_polygon` (regelmäßiges Vieleck), die 4 Parameter mitbekommt: x-Wert und y-Wert der Startposition, die Anzahl der Ecken und die Seitenlänge.

   Das Vieleck soll gegen den Uhrzeigersinn gezeichnet werden und die erste Seite soll parallel zur x-Achse liegen.

   Weiters soll diese Funktion den Umfang des Vielecks zurückliefern.

   Es ist bekannt: am Ende soll die Turtle wieder dort sein bevor die Funktion aufgerufen wurde und auch die gleiche Ausrichtung aufweisen.

Verwende die `for` Anweisung.

Baue den Aufruf der Funktion in das Programm `geom.py` ein.

8. Schreibe nun eine Funktion `sum`, der bis zu 5 Parameter übergeben werden können und die die Summe der Zahlen zurückliefert. Verwende dazu Parameter mit Default-Werten (d.h. Standard-Werten). 0 wäre als Defaultwert nicht so schlecht, oder?

   Teste die Funktion. Baue diese Funktion aber nicht in `geomlib.py` bzw. `geom.py` ein. Das ist nicht notwendig.

9. Schreibe jetzt eine Funktion `polygon`, die mindestens zu 3 x- und y-Werte erhält und maximal 5 x- und y-Werte als Parameter erhält und diese als Polygon zeichnet. Der Umfang ist wieder zurückzuliefern. Natürlich wieder in `geomlib.py`.

   Hinweise:

   a) Nenne die Parameter: x1, y1, x2, y2,…

   b) Verwende als Default-Wert für x4, y4, x5 und y5 jeweils `None`

   c) Zeichne zuerst die Kanten von P1 zu P2 und von P2 zu P3

   d) Frage ab, ob x4 und y4 jeweils ungleich `None` sind und zeichne danach die Kante. Wenn nicht, dann Beendigung der Funktion mittels `return`.

   e) Jetzt dasselbe mit x5 und y5.

   f) Baue jetzt die Entfernung ein und verwende dazu die Funktion `distance2`.

   Baue diese Funktion wieder in das Programm `geom.py` ein.

10. Schreibe eine Funktion `stars_triangle` (in `geomlib.py`), die eine ganze Zahl als Parameter übergeben bekommt und folgendes Sternchenmuster z.B. bei 5 als Argument der auf der Konsole zeichnet:

```
*
**
***
****
*****
```

    Verwende dazu eine `for`-Schleife und den `*` Operator für Zeichenketten. Baue diese Funktion wieder in das Programm `geom.py` ein.

2 Hausübung

Kapitel 7 lesen!