sportlich ... ..

Es folgen jetzt ein paar Aufgaben, die auf höherem Schwierigkeits-grad mit den Strukturen “Schleife”, “Verzweigung” und “Array” spielen.

Das ist alles lösbar, und wurde schon von vielen Jahrgängen Technikerschule bearbeitet, allerdings in Datenverarbeitungsfächern.

Damit sind wir deutlich über dem Prüfungsniveau !

.

Iteration

Praxis

Schreiben Sie ein Programm, das die Nullstelle von y = mx + t durch Iteration sucht. m sei positiv, t negativ.

Iteration (=schrittweise Näherung an Lösung) :

Ihr Programm startet bei einem Wert x (z.b. bei 0), rechnet den y-Wert aus, macht einen Schritt dx (x also ein Stückchen größer), rechnet nochmal und prüft, ob sich das Vorzeichen des Funktionswerts geändert hat. Sie prüfen also immer zwei Werte nebeneinander. Wenn das Vorzeichen verschieden ist, wurde die Nullstelle überschritten, dann muß die Richtung geändert werden (also dx = -dx) und die Schrittweite wird kleiner (also z.b. dx = dx / 10). Das Ganze solange, bis eine gewünschte Genauigkeit in x (z.b. dx < 0.000001) erreicht wird.

Weil sie die Nullstelle von einer Geraden ja auch anders ganz einfach ermitteln können, ist das sehr gut zu testen.

Optimierung

Praxis

Sie sollen eine Blechdose für Tomatensuppe entwickeln !

Das Ding soll Zylinderform haben, und eine gegebene Menge (z.B. 1 Liter) enthalten. Sie haben Formeln für Oberfläche und Volumen. Daraus müssen sie (Mathe..) eine Formel für die Dosenoberfläche (=Blechverbrauch) machen, die von Volumen und Bodenradius abhängt (2 Gleichungen mit 2 Unbekannten - > eine Gleichung).

Schreiben Sie dann einen Iterationsalgorithmus, der für ein einzugebenes Volumen den minmalen Blechverbrauch ermittelt. Fast das Gleiche wie oben die Gerade, aber kein Nulldurchgang sondern der niedrigste y-Wert ist gesucht.

Radius und Höhe müssen rauskommen.

Sortieren

Eine sehr gute Übung für Arrays (Listen) sind Sortierverfahren.
Zum Beispiel der sogenannte “Bubblesort”.

Erst eine anschauliche Beschreibung :
Sie haben ein Röhrchen mit Wasser, das senkrecht steht.
Im Röhrchen sind Luftblasen, verschieden groß, alle in einer Reihe vertikal (keine nebeneinander).
Wenn sie das schütteln, sind (jedenfalls im Bild ;-) die Blasen nach Größe sortiert, die dickste ganz oben.

Das Verfahren wird so programmiert, daß man unten anfängt, und prüft, ob die unterste “Blase” größer ist als die zweite. Wenn ja, tauschen die beiden Platz. Jetzt prüfen sie die zweite und die dritte : wenn nötig, Platz tauschen.

Die Blase wird so lange hochsteigen, bis die drübere größer ist. Dann macht das Verfahren einfach mit der Größeren drüber weiter, die steigt auf, falls nötig.

Wenn ich oben angekommen bin, beginnt alles von vorne.
Wieder beginne ich unten, und führe das beschriebene Verfahren aus, allerdings höre ich schon bei der vorletzten oben auf, ich weiß ja, daß die oberste schon die Größte ist.

Das jetzt alles dauernd immer wieder, bis alle Zahlen bis zur zweiten von unten feststehen. Immer eine weiter unten aufhören. Das wars.

(Ist ein Knaller, ich weiß ;-)

Ich habe zum Testen ein Array (Liste) mit 6 Elementen definiert.

```
zahlen = [0,1,2,3,4,5]
```

Die Werte in der Klammer sind übrigens völlig egal, Hauptsache es sind Zahlen, 6 Nuller sind genauso gut.

Werte für diese 6 Zahlen gebe ich am Programmanfang ein :

```
i=0
while i < 6 :
    zahlen[i]=int(input("zahl : "))
    i=i+1
```

Test : Ich gebe ein : 9, 2, 3, 10, 4, 1

Ergibt :

```
bubble : 9 - 2 - 3 - 10 - 4 – 1   : die 9 steigt hoch
bubble : 2 - 9 - 3 - 10 - 4 - 1
bubble : 2 - 3 - 9 - 10 - 4 – 1
bubble : 2 - 3 - 9 - 10 - 4 – 1   : bis hier, 10 ist größer
bubble : 2 - 3 - 9 - 4 - 10 – 1   : jetzt steigt die 10
bubble : 2 - 3 - 9 - 4 - 1 – 10   : 2-3 ?
bubble : 2 - 3 - 9 - 4 - 1 – 10   : 3-9 ?
bubble : 2 - 3 - 9 - 4 - 1 – 10   : 9-4 !
bubble : 2 - 3 - 4 - 9 - 1 – 10   : die 9 steigt hoch
bubble : 2 - 3 - 4 - 1 - 9 – 10   : 2-3 ?
bubble : 2 - 3 - 4 - 1 - 9 – 10   : 3-4 ?
bubble : 2 - 3 - 4 - 1 - 9 - 10   : 4-1 !
bubble : 2 - 3 - 1 - 4 - 9 - 10   : die 4 steigt hoch
bubble : 2 - 3 - 1 - 4 - 9 - 10
bubble : 2 - 1 - 3 - 4 - 9 - 10

Fertig : 1 - 2 - 3 - 4 - 9 - 10
```