

o++o
Tabmente,
Anfragen,
Berechnungen,
Statistik
und
Visualisierung

Klaus Benecke
(Stand 22.01.2026)

Copyright © 2026 Klaus Benecke. All rights reserved.

Vorwort

Was bedeutet es, dass o++o (ottoPS) die wohl einfachste Programmiersprache ist?

Es bedeutet nicht, dass o++o aus sehr einfachen Konzepten besteht. Es bedeutet jedoch, dass ihre Anwendung relativ einfach ist. o++o ist nicht simpel, doch das Lösen von Problemen ist einfacher als mit anderen Programmiersprachen gleicher Ausdruckskraft. o++o verhält sich dabei wie eine natürliche Sprache. Englisch oder Deutsch ist auch nicht leicht zu erlernen. Natürliche Sprache kann jedoch – bis zu einem gewissen Grad – auch von Kindern im Alter von unter vier Jahren verwendet werden.

Wir sind überzeugt, dass die Grundidee hinter unserer besten Operation (Strichlistenoperation) einfacher zu verstehen ist als der Multiplikationsalgorithmus von Dezimalzahlen. Unserer Meinung nach sind die Konzepte von o++o relativ schwer zu formalisieren, aber sie lassen sich häufig durch einfache Algorithmen beschreiben, die künftig nahezu jeder Nutzer (= OttoNormalVerbraucher) verwenden kann.

Ist o++o eine Programmiersprache?

o++oPS ist als Endnutzersprache konzipiert, jedoch nicht für die Programmierung komplexer Datenbank-Systeme oder Compiler. Sie wurde entwickelt, um die Menschen bei der Lösung ihrer mathematischen alltäglichen Anforderungen zu unterstützen. Tägliche Herausforderungen sind zunächst einmal (ad hoc) Anfragen an Tabellen (Datenbanken), Dokumente oder Kollektionen von Tabellen und Dokumenten. Es gehören auch Finanzberechnungen oder in anderen täglichen Kontext: Bestimmung von Funktionswerten, Bestimmung von Nullstellen oder Extrema von Funktionen und Lösen eines Gleichungssystems (Berechnung mit Matrizen) dazu. Zudem soll o++o Bilder erzeugen und manipulieren können sowie Tabellen und Dokumente in Form von Diagrammen visualisieren. Die wichtigsten innovativen Ideen von o++o im Vergleich zu anderen Ansätzen sind mit Wiederholgruppen verbunden. Das bedeutet, ein bestimmtes Objekt darf nicht nur null oder einen Wert für ein Attribut enthalten, sondern auch mehrere Werte. Für solche Strukturen, seit über 50 Jahren in der Informatik bekannt, bietet o++o neue, leistungsstarke und einfach zu verwendende Operationen.

Das vorliegende Buch enthält eine Vielzahl von Beispielanfragen zur Veranschaulichung der Grundkonzepte.

Inhaltsverzeichnis

1	Berechnungen und Tabellenkalkulation mit o++o.....	11
2	Ein Sparkassenkonto.....	22
3	Tabellenrekursion - Exponentielles Wachstum.....	27
4	Hallo otto – Spielerei.....	36
5	o++o für den Kindergarten?.....	38
5.1	Strichlisten.....	38
5.2	Die Umrechnungsoperationen zahl und l.....	39
5.3	Die Operationen + *.....	39
5.4	o++o-Programme in den Kindergarten?.....	40
6	o++o im Schulunterricht.....	43
7	Multiplikation, Schule und Digitalisierung.....	49
7.1	Wer kann im Kopf multiplizieren?.....	49
7.2	Wer kann schriftlich multiplizieren?.....	50
7.3	Wer kann die Multiplikation programmieren?.....	50
7.4	Strichlistenmultiplikation versus Dezimalmultiplikation.....	56
7.5	Wie könnte o++o den Lehrplan der Schule bereichern?.....	57
7.6	Kann die Strichlistenoperation bereits in der dritten Klasse unterrichtet werden?.....	59
7.7	Rechnet der Schultaschenrechner von Texas-Instruments falsch?.....	60
7.8	Ist EXCEL moralisch verschlissen?.....	66
7.9	o++o Beweise.....	67
7.10	Ein Beispiel für eine tiefe Digitalisierung.....	68
8	Schemata und strukturierte Tabellen.....	70
9	Tabment-Typen (TTs) und strukturierte Dokumente.....	74
10	Eine Universitätsdatenbank.....	83
10.1	Selektion (sel sel-).....	84
10.2	Berechnungen (:=).....	89
10.3	Restrukturierung (gib).....	92
10.4	Ein einfacher Join und verschachtelte Anfragen.....	100
10.5	Ein nutzerfreundlicher Join (ext ext2).....	103
11	Anfragen an die Wikipedia (keys).....	105
12	Spezielle Umstrukturierungsoperationen (onrs).....	114
13	Textverarbeitung mit o++o (+ - -+ cut satzl).....	116
14	Formatieren mit o++o ('3 '4 norm3e norm3m mant rnd).....	118
15	Strukturierte Diagramme.....	120
16	Multiple Diagramme.....	132

17	Bilderzeugung.....	135
18	Bildbearbeitung.....	138
19	Eine Bäckeranwendung (CSS).....	146
20	Anhang A: Liste der Operationen und Schlüsselworte von o++o.....	155
21	Anhang B: Liste der o++o Farbnamen.....	172

Einführung

Die erste Computersprache ist bereits während des Zweiten Weltkriegs von Konrad Zuse entwickelt worden. Sehr frühe Sprachen wie Fortran und Lisp werden bereits seit mehr als 60 Jahren verwendet. BASIC war der erste Versuch, eine Computersprache zu entwickeln, die einfach genug ist, um von jedem benutzt zu werden. Eines der ursprünglichen Ziele von SQL war die Entwicklung einer Endnutzersprache. Mittlerweile gibt es weit über 100 Computersprachen. Python - die einfach zu verwendende Programmiersprache - und die anderen Allzwecksprachen sind für Programmierer konzipiert. Wir folgen der Überzeugung, es ist vorteilhaft, wenn jeder Mensch Computerprogramme verstehen und schreiben kann.

Richten wir unseren Blick auf Deutschland: Es ist festzustellen, dass es bis dato keine allgemein anerkannte Computersprache gibt, die in Schulen für alle gelehrt wird bzw. gelehrt werden kann. Tabellenkalkulationen kommen sogar in Mathematikbüchern aller Schüler vor, aber EXCEL ist keine Computersprache. Mit o++o kann jeder Schüler bestimmte 5-Zeilen-Programme schreiben, bei denen EXCEL 7 Arbeitsblätter benötigt.

Daher scheint es bis heute keine allgemein akzeptierte Computersprache zu geben, die es wert war, jedem vermittelt zu werden oder die zumindest benutzerfreundlich und leistungsfähig genug ist, so dass deren Sinnhaftigkeit und Nutzen bereits in der Schulausbildung hinreichend sichtbar wird. Die einzigen weit verbreiteten „Sprachen“ sind mit der Nutzung von Suchmaschinen verbunden. Hier muss der Benutzer nur ein, zwei oder drei Wörter schreiben (ein sehr einfaches Programm) und das System findet Tausende, Millionen oder noch mehr Ergebnisse. Der Nutzer ist in diesem Zusammenhang allerdings nicht in der Lage, „Programme“ mit mehr Trennschärfe zu schreiben. Er muss hoffen, dass Baidu, Google und Co. genau die Seiten finden, die seine Anforderungen erfüllen. Ausschnitte dieser Seiten werden am Anfang einer nahezu unendlichen Ergebnismenge geschrieben. Was tut der Benutzer, wenn er sich mehr für ein Dokument mit der Rangnummer 100, 1000 oder 100000 interessiert? Was kann ein Benutzer tun, wenn er eine genaue Vorstellung von 1000 gewünschten Dokumenten hat, an denen er interessiert ist, aber er möchte nur einige kleine Unterdokumente oder Untertabellen von jedem dieser Dokumente? Diese Fragen sind nicht einfach zu beantworten und zu realisieren. Die Sprache o++o (otto) zielt darauf ab, Teile solcher Probleme zu lösen. Wir fassen die wichtigsten Gestaltungsprinzipien und Anforderungen an eine Endnutzer-Computersprache oder ein Datenmodell mit entsprechenden Operationen zusammen:

1. Es sollte auf leicht anwendbaren Konzepten mit einer einfachen Syntax basieren.
2. Es sollte ausdrucksstark und leistungsfähig sein.
3. Es sollte durch neue Operationen erweiterbar sein.
4. Es sollte eine präzise Semantik haben, die auf Algorithmen basiert.
5. Es sollte Anfragen auf Tabellen (Datenbanken) und Dokumente ermöglichen.
6. Es sollte Anfragen über Dokumentkollektionen (IR-Systeme) und ganze Datenbanken ermöglichen.
7. Es sollte Berechnungen durch naive (brute force) Algorithmen ermöglichen.
8. Es soll auch für Menschen mit wenig Interesse an Mathematik und Informatik nutzbar sein (Programmierung nach Bauchgefühl).
9. Es soll neben einfachen auch anspruchsvollere Konzepte für breite Anwendungsklassen bieten, die auch für Benutzer mit großem Interesse an Mathematik und Informatik geeignet sind (Programming by Thinking).
10. Es sollte Einzeldaten- und Massendatenoperationen solide integrieren.
11. Das Ergebnis einer Massendatenoperation sollte möglichst klein sein (z. B. möglichst kein kartesisches Produkt; es müssen hochselektive Bedingungen vorliegen).
12. Es wäre schön, wenn es grafische Features auf Basis von strukturierten Tabellen nutzen könnte.

13. Es sollte effizient umsetzbar sein.

14. Zumindest der Kern der Sprache sollte optimiert werden können.

o++o wurde unter Berücksichtigung dieser Prinzipien konzipiert und entwickelt. Es begann als Datenbanksprache für Tabellen mit Wiederholgruppen. Ein Datensatz mit Wiederholungsgruppen darf an jeder Position nicht nur einen Wert enthalten, sondern auch mehrere (Sub Tupel von) Werten. Beispielsweise darf ein Studentensatz einen Namen und ein Stipendium enthalten, aber auch mehrere Hobbies oder mehrere (FACH,NOTE-) Paare. Ebenso darf ein Maschinenteil eine Nummer und die Bezeichnung einer Farbe enthalten und zu dem mehrere Unterteile oder mehrere Ebenen oder Kanten. Solche Subtupel dürfen wieder Subtupel haben. Diese Wiederholgruppen gibt es in der Informatik bereits seit mehr als 50 Jahren. Sie sind typisch für hierarchische Systeme (IMS,...), wurden aber später durch aufkommende relationale Systeme in Verruf gebracht. Auch heute sind sie in XML-, JSON- und NoSQL-Systemen weit verbreitet. Unserer Meinung nach gibt es jedoch keine allgemein akzeptierte Computersprache, die in der Lage ist, diese reichhaltigeren Strukturen angemessen zu handhaben. Mit dem Aufkommen von XML konnten wir unsere Operationen auf die neuen Möglichkeiten des beliebigen Tagging und des Alternationsoperators (or) verallgemeinern. Daher sind wir in der Lage, nicht nur Tabellen, sondern auch Dokumente zu manipulieren. Wir haben den Namen Tabment eingeführt. Ein Tabment kann als abstrakte (syntaxunabhängige) Präzisierung eines XML-Dokuments verstanden werden. Schritt für Schritt haben wir unsere Sprache o++o verbessert. Wir haben binäre Suchbäume in Tabmente eingeführt. So haben wir für mehrere Operationen große Effizienzsteigerungen erzielt.

Indizes können auch als Tabmente betrachtet werden.

Unsere Sprache o++o ist in OCaml implementiert. Einige grundlegende Schlüsselwörter von o++o sind Deutsch oder Französisch ('gib' anstelle von SELECT, 'sel' anstelle von WHERE,...), weil sie kürzer sind als die entsprechenden englischen Wörter, aber die meisten Schlüsselwörter sind Englisch. Dies scheint wichtig zu sein, da Smartphones nur einen kleinen Bildschirm haben.

Wenn wir bestimmte Anfragen betrachten, verhält sich unser o++o-Datenmodell zum relationalen Modell vielleicht wie Dezimalzahlen zum römischen System. Römische Zahlen sind vor allem bei kleinen Zahlen verständlicher, Berechnungen sind in der Regel jedoch schwieriger zu realisieren.

Das häufigste Argument gegen eine Endbenutzersprache lautet: Beahlt der Kunde für ein Produkt, für das er noch etwas lernen muss?

Wir denken 100 Jahre zurück:

Das Auto wurde erfunden, aber die meisten Menschen nahmen die Eisenbahn oder eine Pferdekutsche, wenn sie reich genug waren. Man glaubte, dass es weltweit nie mehr als 1 Million Autos geben würde, da es nicht mehr als 1 Million Fahrer geben würde. Niemand glaubte, dass der Normalverbraucher einmal einen Führerschein machen und selbst Auto fahren würde.

Heute glauben viele Menschen ebenfalls, dass niemand ein Computerprogramm kaufen würde, für das man vielleicht ein paar Stunden oder Tage lernen muss.

Wir stellen folgende Argumente dagegen:

0. Fast alle Menschen auf der Welt mussten mehrere Jahre lang lernen, um die Einzeldatenoperationen Addition, Multiplikation, Division und Differenz für jeden Zahlenbereich in der Schule zu verstehen. Sind Massendatenoperationen wie Selektionen, Berechnungen, Umstrukturierungen, Tabellen sortieren, ... nicht genauso wichtig?
1. Auch wenn Sie eine Textverarbeitung wie MS Word verwenden möchten, zahlen Sie dafür und benötigen Wochen und Monate, bis Sie alle Möglichkeiten beherrschen.

2. Eine gute Programmiersprache ist viel einfacher zu erlernen als Deutsch oder eine Fremdsprache. Die Kenntnis einer Computersprache könnte in Zukunft Teil der Allgemeinbildung werden.
3. Eine gute Programmiersprache erlaubt es, viele Probleme kürzer und präziser mit weniger Missverständnissen zu formulieren als jede natürliche Sprache.
4. Es muss nicht erklärt werden, welche Vorteile jemand hat, der einen Führerschein oder sogar ein Auto besitzt. Wenn er Problemlösungen mit dem Computer sogar selbst programmieren kann, erhöht dies die Qualität der Computernutzung, da er auch die Ergebnisse besser interpretieren kann. Er benötigt keinen Informatiker (Chauffeur). Dadurch gibt es weniger Kommunikationsprobleme und er spart die Kosten für den Informatiker und die Zeit für die Kommunikation.
5. Wenn der Einzelne präzise Anfragen stellen kann, hat er deutlich kompaktere Anfrageergebnisse und spart viel manuellen Suchaufwand. Auch das reduziert den Arbeitsaufwand und verbessert die Qualität.

Was sind die spezifischeren Designprinzipien von o++o?

1. Das Wichtigste zuerst

1.1 Sortieren nach den ersten Attributen einer Kollektion

`gib ABTEILUNG, CHEF, (NAME, ORT m) m`

Hier ist eine strukturierte Tabelle beschrieben, die zu jeder Abteilung auch eine entsprechende Gruppe von Mitarbeitern enthält. Mengen (m) (und Multimengen) werden immer nach den ersten Spaltennamen sortiert. D.h. die äußere Menge wird nach ABTEILUNG sortiert und die innere Menge nach NAME und dann nach ORT, da der NAME nicht immer ein Schlüssel in einer Abteilung ist.

1.2 Zuerst geschrieben - zuerst berechnet:

$2+3*4$ ergibt 20

Hier hat ein Rechteck eine längere Seite, die aus zwei 2 und 3 Meter langen Teilabschnitten besteht. Die andere Seite ist 4 Meter lang. Die Fläche ergibt 20.

$3*4+2$ ergibt 14

Wenn ich zwei Rechtecke habe, eines mit den Seitenlängen 3 und 4 und eines mit der Fläche 2, kann ich zuerst $3*4$ berechnen und dann 2 addieren, um den Flächeninhalt zu erhalten.

1.3 TT-Invarianz (TT=TabmentTyp)

Bei vielen Operationen wie Addition oder Multiplikation ist der Typ des Ergebnisses derselbe wie der Typ des ersten Eingabewerts.

```
<TABH!
FACH,  NOTEI m
Mathe  1 2 4 1
Phy    2 3 5 2 4
!TABH>
*15/6
```

Hier wird eine ganze Tabelle im horizontalen Tab-Format mit einer Zahl multipliziert. Das heißt, jede Zahl der Tabelle wird mit 15/6 multipliziert und die Wörter bleiben unverändert. Somit ergibt sich wieder eine Tabelle vom Typ FACH,NOTEI m. Die Umbenennung von NOTE in PUNKTE kann vom Benutzer vorgenommen werden. D.h. auch hier ist der erste Eingabewert wichtiger als der zweite.

1.4 Exponentendarstellung von Zahlen

o++o erlaubt zusätzlich eine Darstellung, bei der der wichtigere Teil - der Exponent - der Mantisse einer Zahl vorangestellt wird. Der Exponent sagt mehr über die Größe der Zahl aus als die Mantisse:

6m12.345'678 (12 Millionen 345 Tausend ...)

9m123.456 (123 Milliarden 456 Millionen ...)

2. Pragmatik und Methodik zuerst

Wir können eine mehrzeilige Semantik auch für einen einzelnen Term zulassen. Dann könnten wir

(23+45+67) * (1111+2222+3333+4444)

durch

23+45+67

111+2222+3333+4444

ersetzen. Dies kann schneller getippt werden und ist auch übersichtlicher, indem jedem Klammerpaar eine Zeile gewidmet wird. In o++o wird diese Notation weiter verkürzt zu

23+45+67

*** 1111+2222+3333+4444**

Dies geschieht nicht aus methodischen Gründen (bessere Lesbarkeit), sondern aus Pragmatismus. Diese Notation verschwendet die zusätzliche mittlere Zeile nicht. Im Vergleich zur ersten Notation müssen Sie anstelle von 4 Klammern nur einmal eine (größere) Return-Taste benutzen.

3. Kurze einprägsame Schlüsselwörter

Kurze Programme erfordern kurze Schlüsselwörter und kurze Operationssymbole oder -namen. Wenn die Anzahl dieser Symbole jedoch zu groß wird, muss man auch vollständige Namen für Bezeichnungen zulassen, damit sie sich der Benutzer merken kann. Bei o++o gilt: Je wichtiger ein Symbol ist, d. h. je häufiger es verwendet wird, desto kürzer ist es. Diese Regel kann besser umgesetzt werden, indem auch nicht-englische Wörter zugelassen werden.

Sehr kurz sind +, *, ..., m, l Das ist sicherlich in Ordnung. Wir haben zudem viele englische Begriffe durch einprägsamere und kürzere Symbole ersetzt:

sum: ++

product: **

average: ++:

count: ++1

...

Wo wir sehr kurze einprägsame bekannte Wörter in einer anderen Sprache als Englisch gefunden haben, ersetzen wir englische Begriffe durch kürzere aus anderen Sprachen, wenn diese Wörter vielen Menschen bekannt sind:

true: si (Spanisch Italienisch)

false: no

Aus der Übersetzung von SELECT-FROM-WHERE (gib-aus-mit) sind die deutschen Wörter

gib (select) für „gib mir“

und

aus (from)

geworden.

4. Programme werden von oben nach unten und von links nach rechts ausgewertet

Programme mit Schleifen oder allgemeiner Rekursion sind ausdrucksstark und leistungsfähig, aber oft schwer zu lesen und verstehen. Von sequentiellen Programmen wird erwartet, dass sie nicht so ausdrucksstark sind. o++o wurde auch entwickelt, um das Gegenteil zu beweisen. Dies erfordert leistungsstarke und ausdrucksstarke Operationen.

```

Beispiel: Ist 37 keine Primzahl?
Zunächst werden alle Produkte bis 100 berechnet:
Xl:=2 ..50          # 49 Zahlen generieren
Yl:=2 ..10 at X     # für jeden X-Wert 9 Zahlen erzeugen
PRODUKT:=X*Y        # alle Produkte berechnen
sel PRODUKT <= 100   # auswählen der gewünschten Produkte
gib PRODUKTm         # Produkte sortieren und Duplikate
                    # eliminieren
ANTWORT:= 37 in PRODUKTm # Ist 37 ein Produkt?

```

ist das Kommentarzeichen.

Die **Lesbarkeit** von Programmen und Tabmenten ist ein wichtiges Problem.

o++o berücksichtigt das wie folgt:

1. Programme können häufig kurz geschrieben werden.
An obigem Programm zur Bestimmung aller Produkte kann man einige Konzepte von o++o gut erklären. Will man nur wissen, ob 37 eine Primzahl ist, so kann das wesentlich kürzer formuliert werden:
TEILERl:= 2 .. 19
gib ANTWORT ANTWORT:=37 rest TEILER = 0 ! ex
Das Programm ist sicher gut lesbar, wenn man verinnerlicht hat, dass ex die Existenzaggregation ist und man diese genauso anwenden kann wie die ++ Aggregation.
2. Zahlen können auch im Schweizer Stil dargestellt werden (z.B.: 12'345'678)
3. um mehr als 4 Leerzeichen eingerückte Zeilen gehören logisch zur vorhergehenden Zeile.
Z.B.:
meine_noten.tab
gib DUR, (FACH, DUR m)
 DUR:=NOTE! ++: # diese Zeile gehört vom logischen
 # Standpunkt noch zur vorangehenden
rnd 1
4. eine strukturierte Tabelle mit dem Schema
ABTEILUNG, CHEF, (NAME, GEHALT l) m
enthält jede Abteilung und jeden Chef nur einmal. Dies reduziert nicht nur die Redundanz, sondern verbessert auch die Lesbarkeit im Vergleich zu flachen Tabellen dieses Typs.

In den Kapiteln wird gezeigt, wie allgemein und einfach die Anfragemöglichkeiten von o++o sind. In Kapitel 1 werden einige Grundfunktionen unseres "Taschenrechners" vorgestellt. Alle Beispiele dort benötigen keine hinterlegten Tabellen oder Dokumente. Das bedeutet nicht, dass unsere o++o App nicht mit Dateien arbeiten kann. Tabellen und Dokumente können auch auf dem Smartphone gespeichert werden.

Als erstes muss der Benutzer verstehen, was ein Schema ist und was die Tabellen oder Dokumente sind, die zu diesem Schema gehören. Dann wird es nicht zu schwer sein, die Anfragebeispiele für Selektion, Berechnung und Umstrukturierung der ersten Kapitel zu erfassen. Alle Operationen erlauben eine kompakte und gut lesbare Formulierung von (komplexen) Anfragen. Sie gelten für verschachtelte Listen oder Mengen und sie sind neu in der Datenbankwelt. Berechnungen können oft als hierarchische "Map"-funktionen verstanden werden. Die Umstrukturierung mit der gib-Klausel ist sehr ausdrucksstark, da sie mit Sortieren (m, b), Duplikateliminierung (m), Aggregation (++ , min, max, ++1, ++:, ex, &&, **, variance) kombiniert wird.

Wir kennen keine andere Umstrukturierungsoperation in einem kommerziellen Produkt, die es erlaubt, eine gegebene Hierarchie nur durch Angabe eines Schemas oder TTs (Tabment Type) der gewünschten Struktur umzuwandeln. Obwohl die Operationen in den Beispielen nur sequentiell angewendet werden, decken sie ein weites Anwendungsgebiet ab.

In Abschnitt 10.4 werden eine "natürliche" Join-Operation und ihre ungeschachtelte und geschachtelte Verwendung vorgestellt. Es wird deutlich, dass wir das kartesische Produkt und sogar den gewöhnlichen flachen relationalen Join nicht brauchen. Ein vereinfachter Begriff der Rekursion wird in Kapitel 3 eingeführt. Mit dieser Endnutzer-Rekursion lassen sich entsprechende Anfragen mit minimalem Lernaufwand realisieren. Nachdem in Kapitel 4 gezeigt wurde, dass das Drucken von zwei Wörtern nicht nur eine syntaktische Frage ist, wird in Kapitel 6 versucht deutlich zu machen, dass o++o für alle Fächer in der Schule, insbesondere aber für Mathematik und Informatik, nützlich ist. Es soll deutlich werden, dass auch Schüler der 9. oder 10. Klasse Probleme lösen können, die als Anwendungen der Differential- und Integralrechnung gelten. Darüber hinaus wird argumentiert, dass der gewöhnliche Divisionsalgorithmus aus dem Lehrplan der Mathematik gestrichen werden könnte. Der erste „Endnutzerjoin“ (gibjoin) (Abschnitt 10.5) erweitert grob gesagt die Restrukturierungsoperation auf mehrere Eingabetabellen. Er erfordert weder kartesisches Produkt noch (versteckte) Join-Bedingungen. Dann werden in Kapitel 11 Anfragen an Wikipedia vorgestellt. Dabei werden nicht nur tabellenorientierte Anfragen mit Bezug auf Infoboxen betrachtet, sondern auch dokumentorientierte.

Kapitel 17 enthält einige Anfragen, bei denen das Ergebnis als Bild interpretiert werden kann. Grob gesagt enthält jede Ergebnistabelle die Koordinaten von Punkten eventuell kombiniert mit einem Farbwert. Außerdem wird gezeigt, dass es auf der Basis von strukturierten Tabellen einfacher ist, strukturierte Diagramme zu erstellen.

Die wichtigsten Operationen des Datenmodells werden in Kapitel 10 näher beschrieben. Abschnitt 10.3 enthält die Beschreibung der Restrukturierungsoperation, 10.2 die Zuweisungsoperation und 10.1 die Selektion.

Danksagung:

Ich möchte den folgenden Informatikern für ihre wertvollen Beiträge zu unserem System o++o und früheren Systemen danken:

Wolfgang Reichstein für die erste Implementierung der Restrukturierungsoperation in einem Schritt in C für HSQ-Dateien,

Dmitri Schamschurko für die erste Implementierung des ersten Kerns des „gib-aus-mit“-Konstrukts in einem funktionalen Stil (Caml Light),

Martin Schnabel für die Konzeption und Umsetzung von Unterprogrammen und weiteren Features,

Andreas Hauptmann für die Verbesserung vieler Konzepte in Design und Effizienz, insbesondere für Anfrageoptimierungskonzepte.

Weiterer Dank gilt Stephan Schenkl und Mirko Otto für die Unterstützung des o++o-Projekts.

Danke an Eicke Redweik für die Implementierung des Zugriffs auf ein relationales DBMS und auf die Wikipedia und an Jens Winter für die Implementierung einer ersten o++o-App für Android.

1 Berechnungen und Tabellenkalkulation mit o++o

Wir stellen zunächst einige numerische Berechnungen vor.

Programm 1.1: Addition	Ergebnis
1.23 + 4.56	5.79

Programm 1.2: Division	Ergebnis
1:7	0.142857142857

Programm 1.3: Division mit verbesserter Lesbarkeit	Ergebnis
1:7 '3	0.142'857'142'857

Programm 1.4: Division mit Rundung	Ergebnis
1:7 rnd 3	0.143

Programm 1.5: Potenzieren	Ergebnis
3 hoch 20 '3 # oder ^	3'486'784'401

ist das Kommentarzeichen. D.h., dass „oder ^“ nicht zum Programm gehört. Durch Kommentare können Programme erläutert werden.

Programm 1.6: Addition rationaler Zahlen	Ergebnis
3/4 + 1/3	13/12

Programm 1.7: Typ des ersten Eingabewerts bleibt erhalten	Ergebnis
3/4 + 0.3	21/20

Programm 1.8: Typ des ersten Eingabewerts bleibt erhalten	Ergebnis
0.3+3/4	1.05

Programm 1.9: Differenz oder Liste	Ergebnis
3 - 2 # Beachte, dass "3 -2" eine # Liste von zwei Zahlen ist	1

Programm 1.10: Sinus von pi : 2	Ergebnis
pi : 2 sin	1.

Programm 1.11: Sinus von 30 Grad	Ergebnis
30:180*pi sin	0.5

Programm 1.12: Wie viele 10-stellige Binärzahlen gibt es?	Ergebnis
2 ^ 10 # ^: hoch Basis:2 # Exponent: 10	1024

Programm 1.13: Berechne die Kantenlänge eines Würfels mit Volumen 2	Ergebnis
2 ^ 1/3 oder	1.25992104989

Program 1.14: Berechne die Kantenlänge eines Würfels mit Volumen 2 mit gewöhnlicher Divisionsoperation	Ergebnis
2 ^(1:3)	1.25992104989

Programm 1.15: Summe von 4 Zahlen	Ergebnis
3.21 4.56 6.88 9.32 ++	23.97

Programm 1.16: Summe der Zahlen von 1 bis 100	Ergebnis
1 .. 100 ++	5050

Programm 1.17: Produkt der Zahlen von 10 bis 40	Ergebnis
10 .. 40 **	2248443792019118536005322061276774400000000

Man erkennt am Ergebnis, dass man mit o++o beliebig große ganze Zahlen verarbeiten kann.

Programm 1.18: Maximum von Zahlen	Ergebnis
1/3 2/7 max	1/3

Programm 1.19: Durchschnitt von mehreren Noten	Ergebnis
1 3 2 1 3 4 ++:	2.33333333333

Programm 1.20: Einführung von zwei Spaltennamen (Werte zweier Terme gleichzeitig ausgeben)	Ergebnis
X:=2 hoch 10 # := : Zuweisung Y:=X : 10	X, Y 1024 102.4

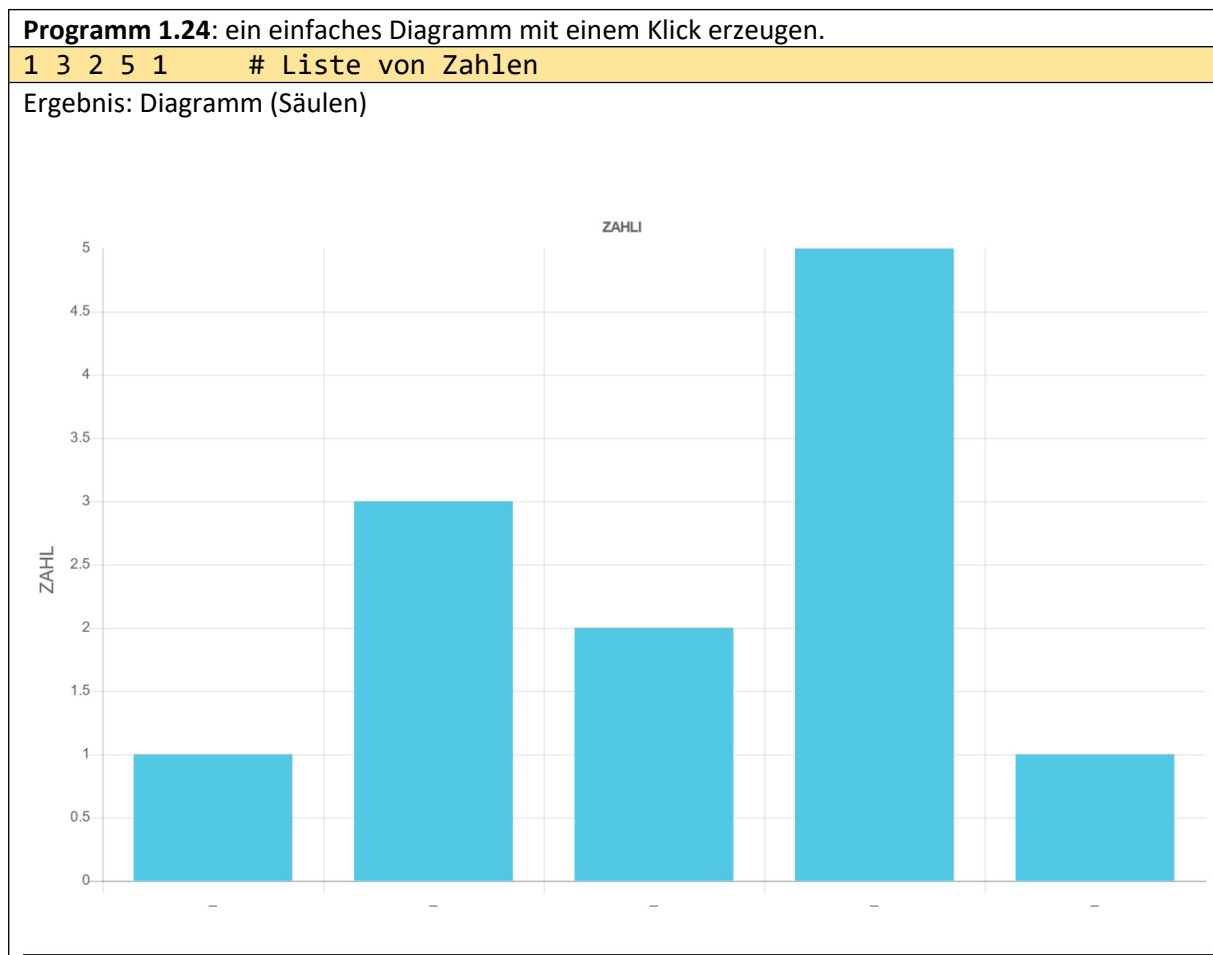
Programm 1.21: ein Paar von zwei unabhängigen Termen	Ergebnis
2 sqrt; 3 sqrt # ; trennt # stärker als ,	PZAHl, PZAHl 1.41421356237 1.73205080757

In Tabellen sind kaum Kommata enthalten. Dies würde die Lesbarkeit zerstören. Daher finden wir beispielsweise auch in .tab -Dateien kaum Kommata, auch wenn Paare oder Tupel repräsentiert

werden. Die Paarbildung wird jedoch in den Metadaten (Tabellenköpfen) der Tabellen abgebildet, um Missverständnissen vorzubeugen. PZAHL ist eine Zahl mit Punkt.

Programm 1.22: "," ist eine gewöhnliche Operation: Berechnung von links nach rechts	Ergebnis
2 sqrt,3 sqrt # das letzte sqrt # wirkt sowohl über # "2 sqrt" als auch # über 3	PZAHL, PZAHL 1.189207115 1.73205080757

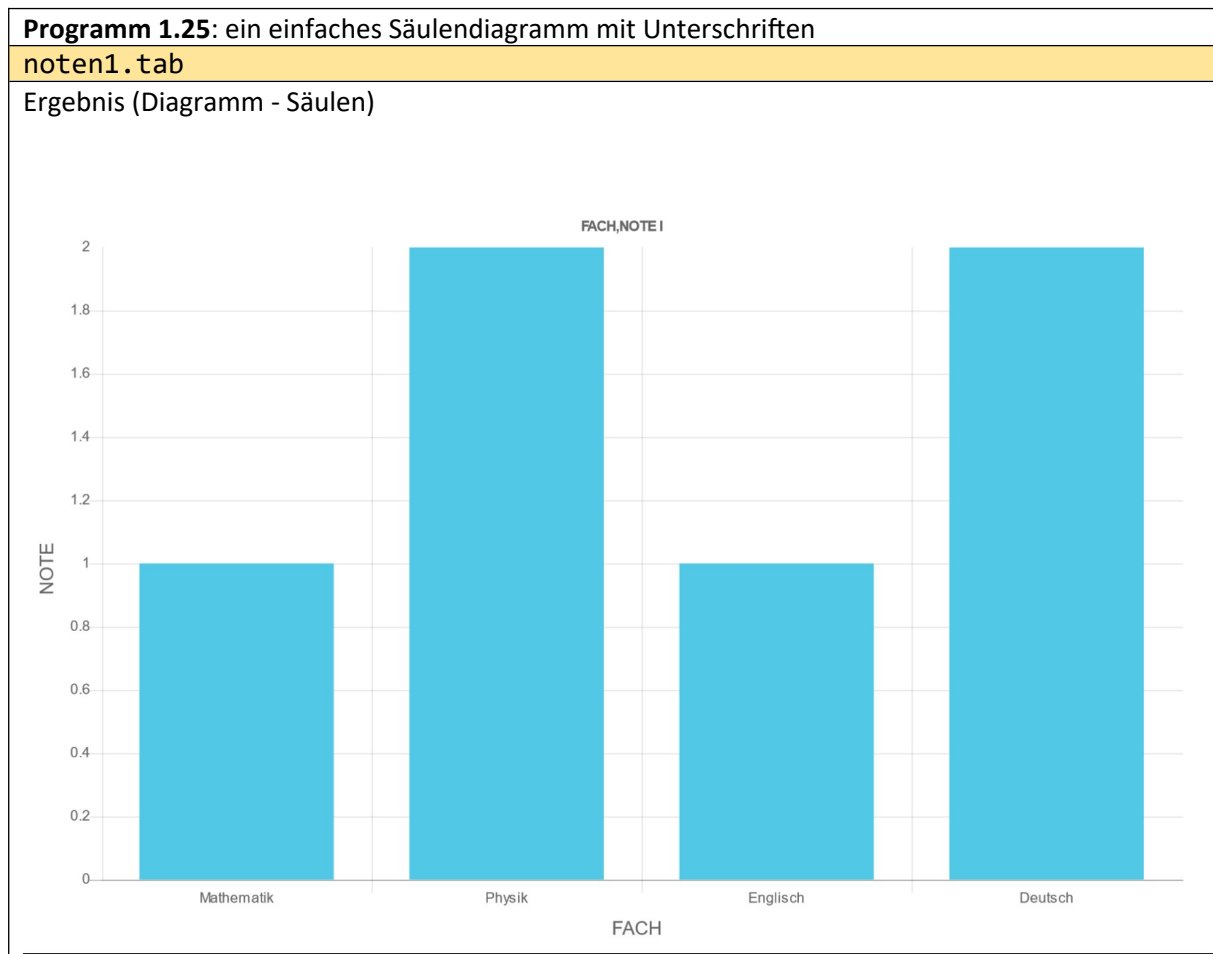
Programm 1.23: ein Paar von Zahlen durch einen Term ermitteln	Ergebnis
DIV,REST:=13 divrest 5	DIV, REST 2 3



FACH,	NOTE 1
Mathematik	1
Physik	2
Englisch	1
Deutsch	2

noten1.tab

Die obige Tabelle repräsentiert eine Liste von (FACH,NOTE)-Paaren. Sie kann mit einem beliebigen Texteditor erstellt oder in das Ausgabefeld der o++o-Oberfläche eingetippt werden. I steht für Liste.



Es ist auch möglich, die folgende Zeile in das Programmfeld der Otto-Oberfläche einzugeben.

```
FACH,NOTE I:=Mathematik Physik Englisch Deutsch posjoin 1 2 1 2
```

Dabei werden die beiden Listen elementweise durch Komma verbunden. Die entstandene Liste besteht aus 4 (WORT,ZAHL)-Paaren. Die erste Spalte erhält den Namen FACH und die zweite den Namen NOTE.

Die Basisdaten der folgenden Anfrage können durch die folgende kleine strukturierte Tabelle generiert werden. Hier steht I für Liste. Es benötigt die Endung tabh, da die Noten horizontal angeordnet sind. Listen wurden in Venedig erfunden (Lista). Die einzelnen Einträge (=Elemente = Zeilen) der Liste wurden untereinander angeordnet. Die Fächer werden in noten2.tabh ebenfalls vertikal angeordnet. Einfache Listen wurde bereits vor tausenden Jahren horizontal angeordnet. Ein Satz ist eine Liste von Worten, die im Wesentlichen horizontal angeordnet wurden. Da dies viel Platz auf dem Bildschirm und Papier einspart, können einfache (einspaltige) Listen in o++o auch horizontal angeordnet werden. Das ist möglich, da die Liste abstrakt verstanden wird. Dadurch kann o++o beispielsweise auch JSON-Listen verstehen, obwohl die Listenelemente nicht einfach durch Leerzeichen getrennt werden. In Fragen der Darstellung der Elemente sind Mengen und Multimengen den Listen gleichgestellt. Es werden aber andere Klammern benutzt.

FACH,	NOTE1 m
Mathematik	2 1 3
Physik	2 2 3
Englisch	1 4

noten2.tabh

Diese Tabelle kann auch durch folgende Programmzeile mit Mengenklammern {} erzeugt werden:

FACH,NOTE1 m:={Mathematik,[2 1 3] Physik,[2 2 3] Englisch,[1 4]}

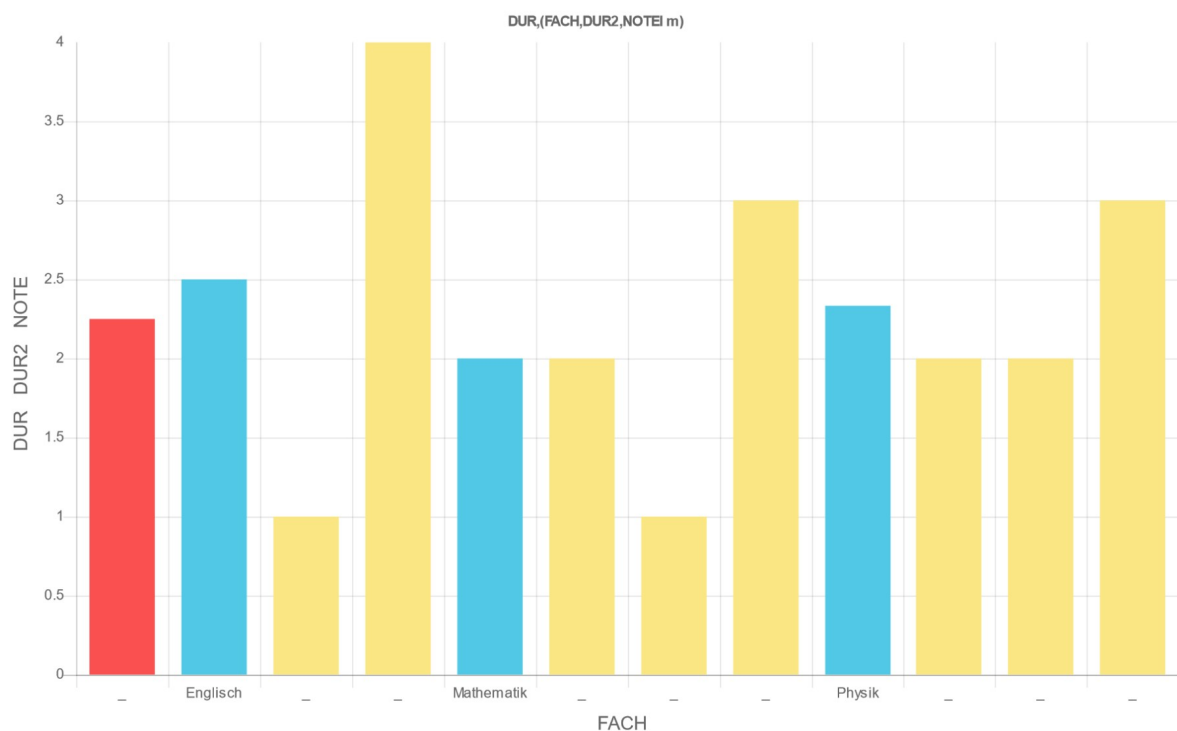
Programm 1.26: ein strukturiertes Diagramm

noten2.tabh

gib DUR,(FACH,DUR,NOTE1 m)

DUR:=NOTE! ++:

Ergebnis (Diagramm-Säulen)



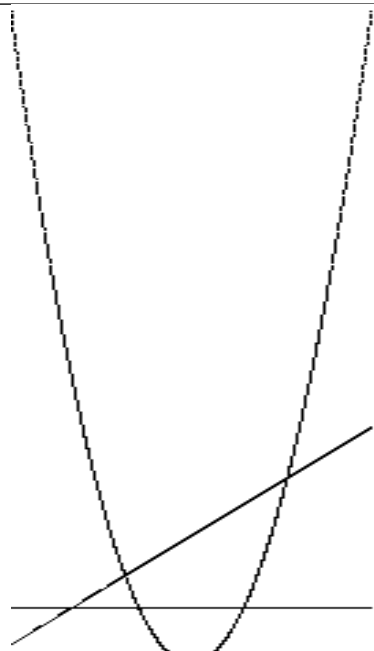
Ergebnis (tabh-Ausgabe)

DUR, (FACH,	DUR2,	NOTE1 m)
2.25 English	2.5	1 4
Mathematik 2.	2.0	2 1 3
Physik 2.33333333333333	2.33333333333333	2 2 3

Es folgen Beispiele zu einer Kurvendiskussion am Beispiel einer Parabel.

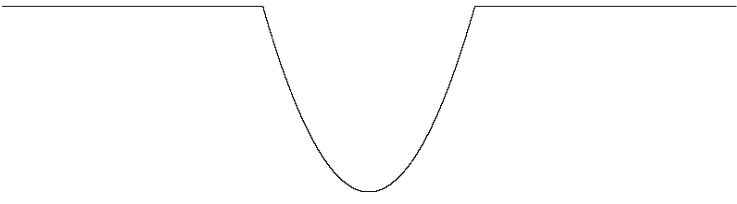
Programm 1.27: Berechnung einer kleinen Wertetabelle der quadratischen Funktion mit den Koeffizienten 1 -8 13 ($x^2 - 8x + 13$)	Ergebnis (tab)
X1:= -2 .. 10	X, Y 1
Y := X poly 1 -8 13	-2 33

	-1	22
	0	13
	1	6
	2	1
	3	-2
	4	-3
	5	-2
	6	1
	7	6
	8	13
	9	22
	10	33

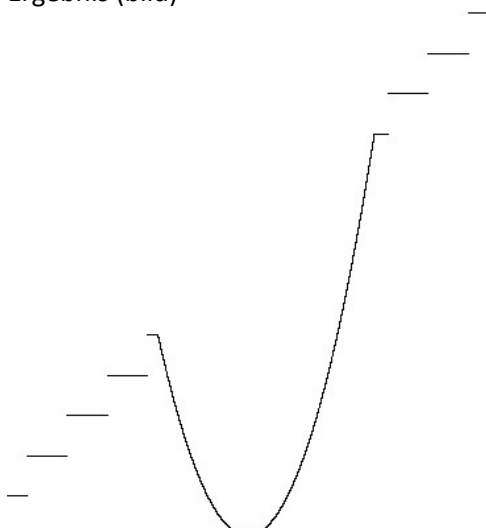
Programm 1.28: Erweitern der Wertetabelle, so dass ein Funktionsgraph zu sehen ist. Zeichne den Graphen der Parabel (quadratische Funktion) mit der x-Achse und der Funktion $y=x$ im Intervall $[-2\ 10]$.	Ergebnis (bild)
<pre> X1:= -2 ... 10! 0.01 Y := X poly 1 -8 13 GERADE:= X Y0:= 0*X </pre>	

Programm 1.29: Näherungsweise Bestimmung des (lokalen) Minimums der Parabel	Ergebnis
<pre>-2 ... 10! 0.0001 poly [1 -8 13] min</pre>	-3

Programm 1.30: Näherungsweise Bestimmung der beiden Nullstellen	Ergebnis (tab)						
<pre> X1:= -2 ... 10!0.0001 Y := X poly [1 -8 13] sel Y succ * Y <= 0 # succ: successor rnd 7 </pre>	<table> <tr> <th>X,</th><th>Y 1</th></tr> <tr> <td>2.2679000</td><td>0.0001704</td></tr> <tr> <td>5.7320000</td><td>-0.0001760</td></tr> </table>	X,	Y 1	2.2679000	0.0001704	5.7320000	-0.0001760
X,	Y 1						
2.2679000	0.0001704						
5.7320000	-0.0001760						

Programm 1.31: Bestimmung der Fläche unter einer (zusammengesetzten) Funktion	Ergebnis (bild) (ohne 2 letzten Programmzeilen)
	
<pre> X1:= -2 ... 10! 0.0001 Y := (X poly [1 -8 13],0) min RECHTECK:= Y*0.0001 gib RECHTECK1 ++ </pre>	Ergebnis (tab) -6.92820323316

Wenn wir die letzten beiden Programmzeilen im folgenden Programm weglassen, kann die Funktion durch Klicken auf bild visualisiert werden:

Programm 1.32: Bestimmung der Fläche unter einer nicht stetigen Funktion	Ergebnis (bild)
	
<pre> X1:= -2 ... 10! 0.0001 Y := X poly (1 -8 13),(X rnd 0) min RECHTECK:=Y*0.0001 gib RECHTECK1 ++ </pre>	Ergebnis (tab) 21.970089131

Programm 1.33: Anwendung der Funktion divrest zur Ausgabe von Zahlenpaaren	Ergebnis (tab)																											
X1:=1 ..10 DIV,REST:=X divrest 3	<table><tr><th colspan="3">X, DIV, REST 1</th></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>2</td><td>0</td><td>2</td></tr><tr><td>3</td><td>1</td><td>0</td></tr><tr><td>4</td><td>1</td><td>1</td></tr><tr><td>5</td><td>1</td><td>2</td></tr><tr><td>6</td><td>2</td><td>0</td></tr><tr><td>7</td><td>2</td><td>1</td></tr><tr><td>8</td><td>2</td><td>2</td></tr></table>	X, DIV, REST 1			1	0	1	2	0	2	3	1	0	4	1	1	5	1	2	6	2	0	7	2	1	8	2	2
X, DIV, REST 1																												
1	0	1																										
2	0	2																										
3	1	0																										
4	1	1																										
5	1	2																										
6	2	0																										
7	2	1																										
8	2	2																										

	9	3	0
	10	3	1

Programm 1.34: Bestimmung aller Primzahlen bis 70

```
X1:= 2 .. 35
Y1:= 2 .. 9 at X
PRODUKT:= X*Y
sel PRODUKT <= 70 # sel: Seelektion
gib PRODUKTm
PRIM1:= 2 ..70
sel- PRIM in PRODUKTm # sel-: die spezifizierten Elemente entfernen
gib PRIM1
```

Ergebnis (tabh-Ausgabe):

2 3 5 7 11 13 17 19 23 29 31 37 41 43 47 53 59 61 67

Programm 1.35: Berechnen Sie den Umfang mehrerer Kreise, deren Radien gegeben sind. Die Ergebnisse sind auf 2 Stellen nach dem Punkt zu runden.

```
4 5 6 2 3.7 9.77 *pi*2 rnd 2
```

Ergebnis (tabh)

25.13 31.42 37.7 12.57 23.25 61.39

Man erkennt, dass dieses Programm in einer Zeile geschrieben werden kann.

Programm 1.36: Berechnung von Umfang und Fläche mehrerer Kreise, deren Radien gegeben sind

```
R1:= 4 5 6 2 3.7 9.77
UMFANG :=R*pi*2
FLAECHE:=R*R*pi
rnd 1
```

Ergebnis (tab)

	R,	UMFANG,	FLAECHE	1
4.	25.1	50.3		
5.	31.4	78.5		
6.	37.7	113.1		
2.	12.6	12.6		
3.7	23.2	43.0		
9.8	61.4	299.9		

Durch R1:= wird jedem Element der gegebenen Liste der Name R („tag“ genannt) zugeordnet.

Bei einer Zuweisung („:=“) wird die angegebene Tabelle um eine neue Spalte erweitert. Im obigen Programm werden nacheinander die Spalten UMFANG und FLAECHE hinzugefügt, wodurch eine Tabelle vom Typ R,UMFANG,FLAECHE | entsteht. | steht für Liste. Leider kann dies leicht mit der Eins verwechselt werden.

Programm 1.37: Berechnung von Umfang und Fläche mehrerer Rechtecke

```
<TAB!
A, B 1
1.23 5.67
7.65 4.32
9.87 6.54
!TAB>
UMFANG :=A+B*2
FLAECHE:=A*B
```

Ergebnis (tab)

A,	B,	UMFANG,	FLAECHE	1
1.23	5.67	13.8	6.9741	
7.65	4.32	23.94	33.048	
9.87	6.54	32.82	64.5498	

Die TAB-Klammern („<TAB!“, „!TAB>“) werden nur im Programmteil des Systems benötigt. In einer Datei erkennt das System den Typ an der Endung „.tab“. In der TAB-Darstellung müssen die Werte an der linken Seite der zugehörigen Spaltennamen ausgerichtet werden.

Programm 1.38: Gesamtpreis einer einfachen Rechnung	Ergebnis
<pre><TAB! ARTIKEL, PREIS 1 Bier 0.61 Limonade 0.23 Steak 2.40 !TAB> ++</pre>	3.24

Hier wird einfach die Summe über die Zahlen in der gegebenen Tabelle (einer Liste von Paaren) gebildet. Die Artikelwerte sind Wörter und haben daher keinen Einfluss auf das Ergebnis. Jetzt ersetzen wir ++ durch * 1.1. Das erzeugt eine Tabelle mit 2 Spalten und drei Zeilen (Datensätze, Tupel). Jede Zahl beinhaltet jetzt noch 10% Trinkgeld:

Programm 1.39: „Multiplikation“ einer Tabelle mit einer Zahl	Ergebnis
<pre><TAB! ARTIKEL, PREIS 1 Bier 0.61 Limonade 0.23 Steak 2.40 !TAB> +% 10</pre>	<pre>ARTIKEL, PREIS 1 Bier 0.671 Limonade 0.253 Steak 2.64</pre>

Dann können Sie erneut mit ++ addieren, um die Gesamtsumme (3.564) zu erhalten.

Programm 1.40: Finden Sie den Gesamtpreis einer komplizierteren Berechnung anhand einer einfachen Tabelle	Ergebnis
<pre><TAB! ARTIKEL, PREIS, ANZAHL 1 Bier 0.61 7 Limonade 0.23 3 Steak 2.40 4 !TAB> POSPREIS:= PREIS*ANZAHL gib POSPREIS1 ++</pre>	14.56

Als Ergebnis der Zuweisung wird die angegebene Tabelle um eine neue Spalte mit dem Spaltennamen POSPREIS erweitert, wobei jeder der drei PREIS-Werte mit dem zugehörigen ANZAHL-Wert multipliziert wird.

Der erste Eingabewert einer Operation, die am Anfang einer Programmzeile steht, ist immer das Ergebnis der vorherigen Programmzeile.

Das Zeichen „:=“ der Zuweisung ist vom Gleichheitszeichen = zu unterscheiden. Zur Formulierung von Bedingungen wird das Gleichheitszeichen, sowie <, >, <=, "in" etc. benötigt. Zur Selektion werden Bedingungen verwendet (Filterung von (komplexen) Zeilen strukturierter Tabellen).

Fügen Sie beispielsweise eine Bedingung

sel ARTIKEL = Bier

oder nur

sel Bier

hinzu, dann ist das Endergebnis der Gesamtpreis für die sieben Biere. Wenn Sie stattdessen nur den Preis für die anderen Artikel berechnen möchten, verwenden Sie

sel- ARTIKEL = Bier

oder einfach

sel- Bier.

Spaltennamen (Metadaten) müssen immer in Großbuchstaben geschrieben werden. Die Schlüsselwörter (gib, sel-, sel, ...) müssen immer klein geschrieben werden. Wenn Sie ein Wort der Primärdaten immer mit Groß- und Kleinbuchstaben schreiben, wird das Programm leichter lesbar. Der Verweis auf die Aggregation (hier ++) ergibt sich aus der Kopfzeile der gewünschten Tabelle. TOTAL ist eine Aggregation pro NAME. Mengen (m, m-) werden immer nach den zuerst angegebenen Spaltennamen sortiert.

Programm 1.41: Finden Sie einen gewichteten Durchschnitt für 3 Schüler und den Gesamtdurchschnitt	Ergebnis (tabh)
<pre> <TABH! NAME, EXAM1,NOTE1 1 Ernst 1 2 1 2 3 1 3 1 1 Clara 1 1 3 Sophia 1 3 1 !TABH> TOT:=EXAM1 ++: *0.6 +(NOTE1 ++: *0.4) TOTAL:=TOT1 ++: rnd 2 </pre>	<pre> TOTAL,(NAME, TOT, EXAM1,NOTE1 1) 1.66 Ernst 1.59 1 2 1 2 3 1 3 1 1 Clara 1.8 1 1 3 Sophia 1.6 1 3 1 </pre>

Programm 1.42: Eine Frau wiegt 40 kg plus die Hälfte ihres Gewichts. Wie viel wiegt sie?	Ergebnis
<pre> GEWICHT1:=40 .. 100 sel GEWICHT:2+40=GEWICHT </pre>	80

Programm 1.43: Eine Flasche mit Korken kostet einen Euro und zehn Cent. Die Flasche ist ein Euro teurer als der Korken. Wie viel kostet die Flasche?	Ergebnis
<pre> FLASCHE1:= 0 .. 110 sel 110-FLASCHE=FLASCHE- 100 # = KORK </pre>	<pre> FLASCHE 105 </pre>

Programm 1.44: Eine Flasche mit Korken kostet einen Euro und zehn Cent. Die Flasche ist ein Euro teurer als der Korken. Wie viel kostet der Korken?	Ergebnis
<pre> FLASCHE1:= 0 .. 110 KORKEN:= FLASCHE - 100 sel KORKEN+FLASCHE = 110 </pre>	<pre> FLASCHE, KORKEN 1 105 5 </pre>

Die erste Zuordnung gibt jeder der Zahlen von 0 bis 110 den Tag FLASCHE. Das sieht man am besten, wenn man sich die ment-Darstellung ansieht:	Wenn wir die Zuweisung FLASCHE:= 0 ..110 geschrieben hätten, würde der FLASCHE-Tag nur einmal erscheinen:
---	---

<TABM> <FLASCHE>0</FLASCHE> <FLASCHE>1</FLASCHE> <FLASCHE>2</FLASCHE> <FLASCHE>3</FLASCHE> ... <FLASCHE>108</FLASCHE> <FLASCHE>109</FLASCHE> <FLASCHE>110</FLASCHE> </TABM>	<FLASCHE> 0 1 2 3 ... 108 109 110 </FLASCHE>
--	---

Programm 1.45: Eine Flasche mit Korken kostet einen Euro und zehn Cent. Die Flasche ist ein Euro teurer als der Korken. Wie viel kostet die Flasche?	Ergebnis
FLASCHE1:= 0 ..110 KORKEN1 := FLASCHE - 100 KORKEN2 := 110 - FLASCHE sel KORKEN1=KORKEN2	FLASCHE, KORKEN1, KORKEN2 1 105 5 5

Diese Lösung ist aus methodischer Sicht vorteilhaft, da die ersten 3 Programmzeilen durch Klicken auf den bild-Button dargestellt werden können. Man kann erkennen, dass es 2 Geraden gibt, deren Schnittpunkt durch die Bedingungen bestimmt wird.

2 Ein Sparkassenkonto

Die folgenden Anfragen beziehen sich auf Datensätze der Sparkasse. Hier kann der Kunde seine Daten als csv-Datei herunterladen. csv-Dateien sind sehr einfach strukturiert. Da sie sehr viele Anführungszeichen enthalten sind sie relativ schwer lesbar. Der otto-Nutzer muss sich mit dieser Syntax nicht vertraut machen. Er kann sie bzw. Teile der Datei in gewohnter Weise als tab-, hsq, ment, web oder json-Datei ansehen. Wir betrachten eine Datei `umsatz.csv`, die Transaktionen aus 3 Jahren enthält.

Programm 2.1: Wie viele Umsätze gibt es?	Ergebnis
<code>umsatz.csv</code> <code>++1</code>	162

Programm 2.2: Gib die ersten Spalten der ersten und letzten Transaktion!
<code>umsatz.csv</code> <code>sel BETRAG pos=1 or BETRAG pos- =1</code>
Ergebnis (tab-Ausgabe):
<code>AUFTRAGSKONTO, BUCHUNGSTAG, VALUTADATUM, BUCHUNGSTEXT, VERWENDUNG...</code>
<code>DE598105327206411 20.07.22 20.07.22 ONLINE-UEBERWEISUNG ReNr2</code>
<code>DE598105327206411 13.05.20 13.05.20 ONLINE-UEBERWEISUNG Lohn</code>
pos ermittelt die Position eines Tupels. pos beginnt vom Anfang mit 1 zu zählen und pos- von hinten. „or“ ist das logische oder-Zeichen.

Programm 2.3: Wie viel Geld wurde auf das Konto überwiesen?	Ergebnis
<code>umsatz.csv</code> <code>sel BETRAG > 0</code> <code>++ BETRAG</code> <code>'3</code>	110'729.17

Programm 2.4: Wie viel Geld wurde vom Konto überwiesen?	Ergebnis
<code>umsatz.csv</code> <code>sel BETRAG < 0</code> <code>gib BETRAG1</code> <code>++</code> <code>'3</code>	-94'713.65

Programm 2.5: Stelle die Gesamteinnahmen und Ausgaben gegenüber?	Ergebnis (tab)
<code>umsatz.csv</code> <code>gib EINNAHMEN,AUSGABEN</code> <code>EINNAHMEN:=BETRAG if BETRAG > 0!0!++</code> <code>AUSGABEN :=BETRAG if BETRAG < 0!0!++</code> <code>'3</code>	<code>EINNAHMEN , AUSGABEN</code> <code>110'729.17 -94'713.65</code>

Programm 2.6: Wie viel wurde insgesamt an Frau Heyer überwiesen?	Ergebnis
---	----------

umsatz.csv	
sel Heyer	-54'538.28
gib BETRAG1	
++	
'3	

Hier muss der Nutzer seine Daten kennen. Wenn es zwei Heyer's gibt, ist das obige Ergebnis sicher nicht das gewünschte. Man könnte dann den Vornamen hinzunehmen:

sel Heyer Erika

oder

sel Heyer & Erika

Man kann auch die Kontonummer benutzen, dann ist das Programm aber nicht so gut lesbar, da die meisten Menschen sich keine Kontonummer oder IBAN merken können.

Programm 2.7: Wie viel wurde Frau Heyer in jedem Jahr und Monat überwiesen?	Ergebnis (tab)
umsatz.csv	Jahr, SUM, (Monat, SUM2 m) m
sel Heyer	20 -16'807.22 05 -1'967.66
MONAT:=VALUTADATUM subtext 4!2	06 -1'777.31
JAHR:=VALUTADATUM subtext 7!2	07 -1'943.89
gib JAHR,SUM,(MONAT,SUM m) m	08 -2'110.27
SUM:=BETRAG ! ++	09 -1'833.69
'3	10 -3'189.39
	11 -1'973.48
	12 -2'011.53
rnd 2	21 -20'727.13 01 -2'011.53
	02 -1'911.90
	03 -2'006.17
	04 -2'443.65
	05 -2'438.02
	06 -583.32
	08 -812.47
	09 -2'636.73
	10 -993.13
	11 -2'635.68
	12 -2'254.53
	22 -17'003.93 01 -1'630.30
	02 -2'202.03
	03 -1'324.38
	04 -1'438.72
	05 -4'460.36
	06 -2'999.59
	07 -2'948.55

Programm 2.8: Gib eine Gegenüberstellung der Einnahmen und Ausgaben für jedes Jahr!
umsatz.csv
JAHR:=20 wort + (VALUTADATUM subtext 7!2)
gib JAHR,PLUS,MINU,SUM m
PLUS:=BETRAG if BETRAG>0!0!++
MINU:=BETRAG if BETRAG<0!0!++

SUM:=BETRAG!++ '3			
Ergebnis (tab-Ausgabe):			
JAHR,PLUS	,MINU	,SUM	m
2020	24'921.	-23'257.11	1'663.89
2021	50'468.04	-34'274.22	16'193.82
2022	35'340.13	-37'182.32	-1'842.19

Subtext benötigt 3 Inputwerte. Hier ist VALUTADATUM der erste, die Anfangszeichennummer 7 der zweite und die Länge der gewünschten Zeichenkette 2 der dritte. VALUTADATUM ist hier ein in deutscher Schreibweise aufgebautes Wort, z.B.: 18.06.12. Im obigen Beispiel soll das Jahr aber mit 4 Stellen ausgegeben werden. Dazu muss das Wort „20“ mit den beiden Ziffern, die subtext ermittelt, verschmolzen werden.

Programm 2.9: Gib mir für jeden Monat des Jahres 2021 die Einnahmen und Ausgaben mit den größeren Überweisungen!
<pre>umsatz.csv sel VALUTADATUM subtext 7!2 = 21 MONAT:=VALUTADATUM nthzahl 2 VERWENDUNG:=VERWENDUNGSZWECK subtext 3!6 EMPFÄNGER:=BEGUENSTIGTER_ZAHLUNGSPFLICHTIGER subtext 3!6 gib MONAT,PLUS,MINU,(BETRAG,VERWENDUNG,EMPFÄNGER b-) m PLUS:=BETRAG if BETRAG>0 ! 0 ! ++ MINU:=BETRAG if BETRAG<0 ! 0 ! ++ '3 sel BETRAG! BETRAG abs>2'000 rnd 2</pre>
Ergebnis (web-Ausgabe):

MONAT	PLUS	MINU	(BETRAG, VERWENDUNG, EMPFÄNGER b-)		
1	0	-3'536.75	BETRAG	VERWENDUNG	EMPFÄNGER
2	18'391.40	-1'922.80	BETRAG	VERWENDUNG	EMPFÄNGER
			11'119.40	SZAHLU	vestit
			7'272.00	SZAHLU	vestit
3	0	-2'969.82	BETRAG	VERWENDUNG	EMPFÄNGER
4	0	-3'238.44	BETRAG	VERWENDUNG	EMPFÄNGER
5	0	-3'572.81	BETRAG	VERWENDUNG	EMPFÄNGER
6	3'513.33	-2'417.52	BETRAG	VERWENDUNG	EMPFÄNGER
			3'513.33	/2018/	vestit
7	279.37	-2'001.98	BETRAG	VERWENDUNG	EMPFÄNGER
8	410.68	-2'258.82	BETRAG	VERWENDUNG	EMPFÄNGER
9	0	-3'932.65	BETRAG	VERWENDUNG	EMPFÄNGER
10	27'527.11	-1'152.42	BETRAG	VERWENDUNG	EMPFÄNGER
			17'413.36	971240	vestit
			10'113.75	971240	vestit
11	0	-2'764.63	BETRAG	VERWENDUNG	EMPFÄNGER
12	346.15	-4'505.58	BETRAG	VERWENDUNG	EMPFÄNGER

Aus Gründen der Darstellbarkeit wurden entsprechende Daten des Empfängers und des Verwendungszwecks gekürzt.

Würde man die letzte Zeile durch

`sel BETRAG abs > 2'000`

ersetzen, so würden nur die Daten der Monate 2, 6 und 10 verbleiben, da die anderen keine größeren Transaktionen (über 2'000 Euro) enthalten.

Programm 2.10: Gib die Kontostände für 2021 als Säulendiagramm aus!

umsatz.csv

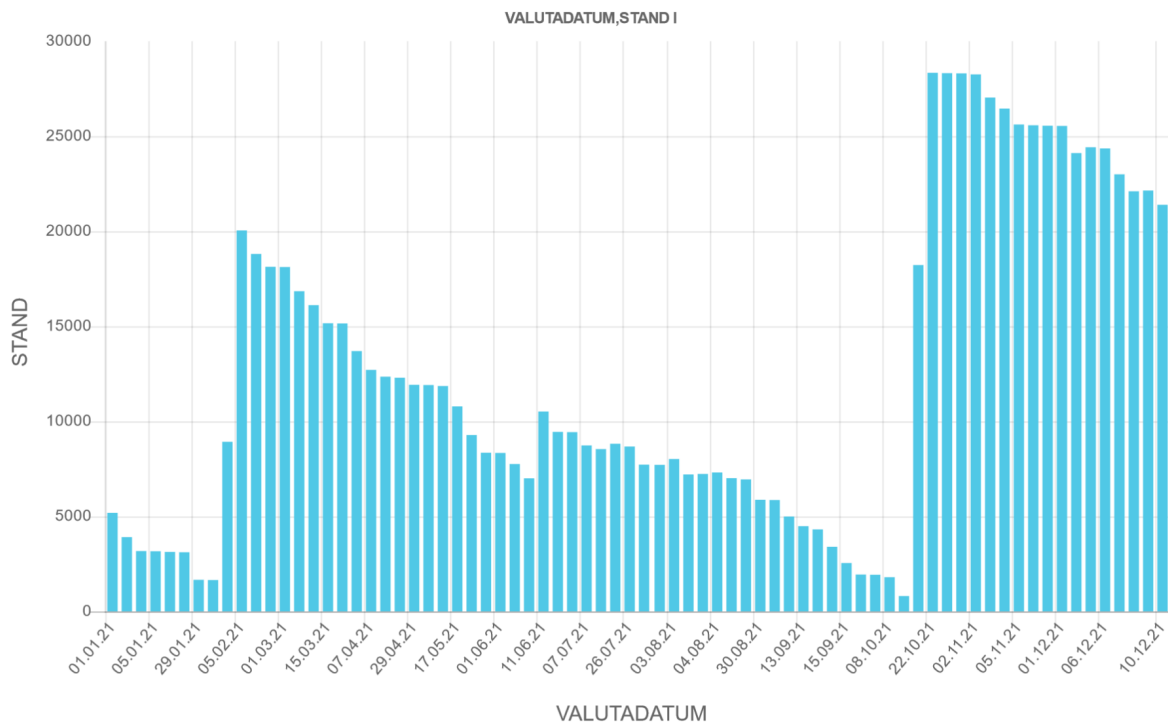
```
sel VALUTADATUM subtext 7!2=21
```

```
gib VALUTADATUM,BETRAG 1-
```

```
STAND:= 5'200 +BETRAG next STAND pred +BETRAG at BETRAG
```

```
gib VALUTADATUM,STAND 1
```

Ergebnis (Säulendiagramm):



Hier wird angenommen, dass der Anfangskontostand 5'200 ist. Diese Zahl wird im obigen Beispiel lediglich zu dem ersten BETRAG-Wert hinzugefügt.

Die Auswertungsmöglichkeiten einer Kontodatei und anderer Dateien hängen von den vorhandenen Daten ab. Wenn im Verwendungszweck der Datensätzen beispielsweise kein Name für den Empfänger angegeben wird, kann man auch keine gut lesbaren Anfragen schreiben. Je besser das Datenmaterial, desto einfacher werden die o++o-Programme und desto mehr Auswertungen werden ermöglicht. Damit wird aber auch klar, dass derjenige, der die Inputdaten kennt, die besten o++o-Programme schreiben kann. Ein Informatiker, der allgemeine Auswertungen solcher Daten programmieren will, wird nie die Vielfalt ermöglichen können, die ein Endnutzer erreicht, der die Inhalte der Sätze genau kennt. Der Verwendungszweck allein bietet viele Möglichkeiten, die Auswertungen zu verbessern, die von vielen sicher noch nicht ausgeschöpft werden.

Die Bedeutung einer einfachen Anfragesprache wird noch vergrößert, wenn auch in Deutschland der Geldverkehr vollständig bargeldlos erfolgt. Wenn jeder Mensch Zugriff auf die Daten seiner Käufe in einem Supermarkt oder Tankstelle hat, kann er genau ermitteln wann und wofür er sein Geld ausgegeben hat.

3 Tabellenrekursion - Exponentielles Wachstum

Rekursivität ist ein mächtiges Werkzeug, um Funktionen oder Datenstrukturen in kurzer Form zu beschreiben. Sie wird besonders in funktionalen Sprachen wie OCaml oder HASKEL genutzt. Wir stellen eine Art von „Vorwärtsrekursivität“ vor, die leicht zu handhaben ist. Es wird stets ein Anfangswert durch einen Wert oder einem Term beschrieben und die nachfolgenden Werte ergeben sich aus jeweils aus dem direkten Vorgänger mittels eines zweiten Terms. Alle erzeugten Werte sind in der Ergebnistabelle sichtbar.

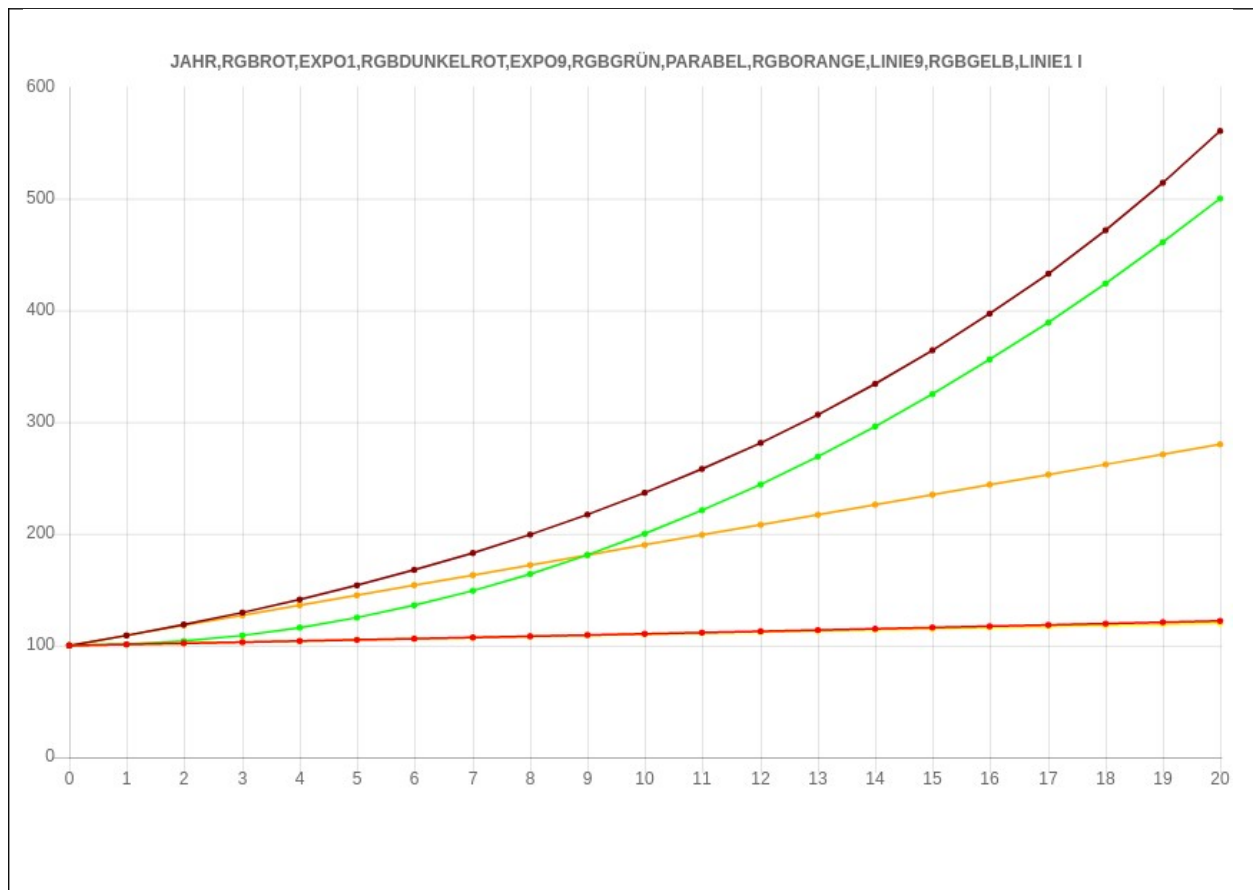
Auf diese Art und Weise kann man exponentielles Wachstum beschreiben. EXPO9 und EXPO1 sind die Programmzeilen hierfür. Gleichzeitig enthalten diese beiden Spalten in der tab-Repräsentation die Wachstumswerte. Ein Prozent Wachstum ist exponentielles Wachstum, wenn man den Zinseszins berücksichtigt. Das ist in beiden Formeln gegeben. Fügt man jeweils lediglich 9 % bzw. 1% von 100 zu dem Vorgänger, so wird der Zins vom Zins nicht berücksichtigt. Das wäre also das Wachstum, wenn man den Zins vom Zins jedes Jahr entnimmt. Dem entsprechen unsere Formeln für LINIE9 und LINIE1. Diese Geradenformeln und die Exponentialfunktionskurven unterscheiden sich hier also lediglich an einer Stelle. Wird die Operation + durch +% ersetzt, wird aus linearem Wachstum ein exponentielles. Bekanntlich ist exponentielles Wachstum jedem anderen Wachstum und somit insbesondere dem linearen Wachstum weit überlegen. Dass neun Prozent Zinsen nach 20 Jahren einen weit besseren Gesamtbetrag ergeben als ein Prozent, zeigt die letzte Zeile der Tabelle (560 € gegen 122 €). Ohne Zinseszins ergeben sich 280 € bzw. 120 €. Um besser mit polynomialen Wachstum zu vergleichen, haben wir eine Parabel eingeschlossen.

Die grüne Parabel zeigt in diesem Bereich von 20 Jahren offensichtlich ein ähnliches Verhalten wie das exponentielle Wachstum von 9 Prozent (dunkelrot). Im nächsten Beispiel werden wir sehen, dass sich das völlig ändert, wenn man anstelle von 20 200 Jahre betrachtet. Die gelbe (1 % ohne Zinseszins) und die rote Kurve (exponentielles Wachstum 1 %) unterscheiden sich praktisch fast nicht. Es soll bereits hier erwähnt werden, dass die Kurven „verzerrt“ sind, damit sie schöner aussehen. In der Schule müsste LINIE1 mit einem Winkel von 45° gezeichnet werden. LINIE9 würde mit einem Winkel von mehr als 83° fast senkrecht stehen. Wenn man das machen würde, würden die Werte der schnell wachsenden Funktionen keinen Platz auf dem Papier oder Bildschirm haben, oder man müsste die X-Achse (hier JAHR) entsprechend verkürzen. Dann würde es aber so aussehen, als ob alle Punkte und Kurven senkrecht verlaufen. Diese unverzerrte reale Darstellung der Punkte wird durch die Ausgabe ‚bild‘ realisiert. Das sieht zwar nicht schön aus, aber man soll die Menschen auch ab und zu mit der Realität konfrontieren. Dann können sie auch die Visualisierungen unten besser einordnen.

Programm 3.1: Wie entwickelt sich ein Betrag von 100 Euro bei einem „einfachen“ und normalen Zins von 1 % und 9 % und bei quadratischem Wachstum innerhalb von 20 Jahren.

```
JAHR1 := 0 .. 20
EXPO9 := 100. next EXP09 pred +% 9 at JAHR
EXPO1 := 100. next EXP01 pred +% 1 at EXPO9
PARABEL:= JAHR * JAHR + 100
LINIE9 := 100. next LINIE9 pred + 9 at PARABEL
LINIE1 := 100. next LINIE1 pred + 1 at LINIE9
rnd 0
JAHR:=JAHR wort
RGBDUNKELROT :=darkred leftat EXP09
RGBROT :=red leftat EXP01
RGBGRÜN :=green leftat PARABEL
RGBORANGE :=orange leftat LINIE9
RGBGELB :=yellow leftat LINIE1
```

Ergebnis (Liniendiagramm):



Ergebnis (tab-Ausgabe):

JAHR	EXPO9	EXPO1	PARABEL	LINIE9	LINIE1
0	100.	100.	100	100.	100.
1	109.	101.	101	109.	101.
2	119.	102.	104	118.	102.
3	130.	103.	109	127.	103.
4	141.	104.	116	136.	104.
5	154.	105.	125	145.	105.
6	168.	106.	136	154.	106.
7	183.	107.	149	163.	107.
8	199.	108.	164	172.	108.
9	217.	109.	181	181.	109.
10	237.	110.	200	190.	110.
11	258.	112.	221	199.	111.
12	281.	113.	244	208.	112.
13	307.	114.	269	217.	113.
14	334.	115.	296	226.	114.
15	364.	116.	325	235.	115.
16	397.	117.	356	244.	116.
17	433.	118.	389	253.	117.
18	472.	120.	424	262.	118.
19	514.	121.	461	271.	119.
20	560.	122.	500	280.	120.

Die neue Spalte EXPO9 wird durch zwei Formeln definiert. Dem ersten Element der Liste der Jahre wird der Wert der ersten Formel zugewiesen. Der zweite Wert wird durch die zweite Formel berechnet, wobei „EXPO9 pred“ der Wert des Vorgängers ist. Daher erhalten wir 100 + % 9=109 für den zweiten Wert. Der dritte Wert wird wieder durch die zweite Formel berechnet,

aber jetzt müssen wir $109 + \% 9 = 118.81$ (gerundet 119) berechnen. Auf die gleiche Weise werden alle folgenden Werte Wert für Wert mit der zweiten Formel errechnet. Die Rundung bereitet keine Ungenauigkeiten, da sie erst nach allen Rechnungen erfolgt.

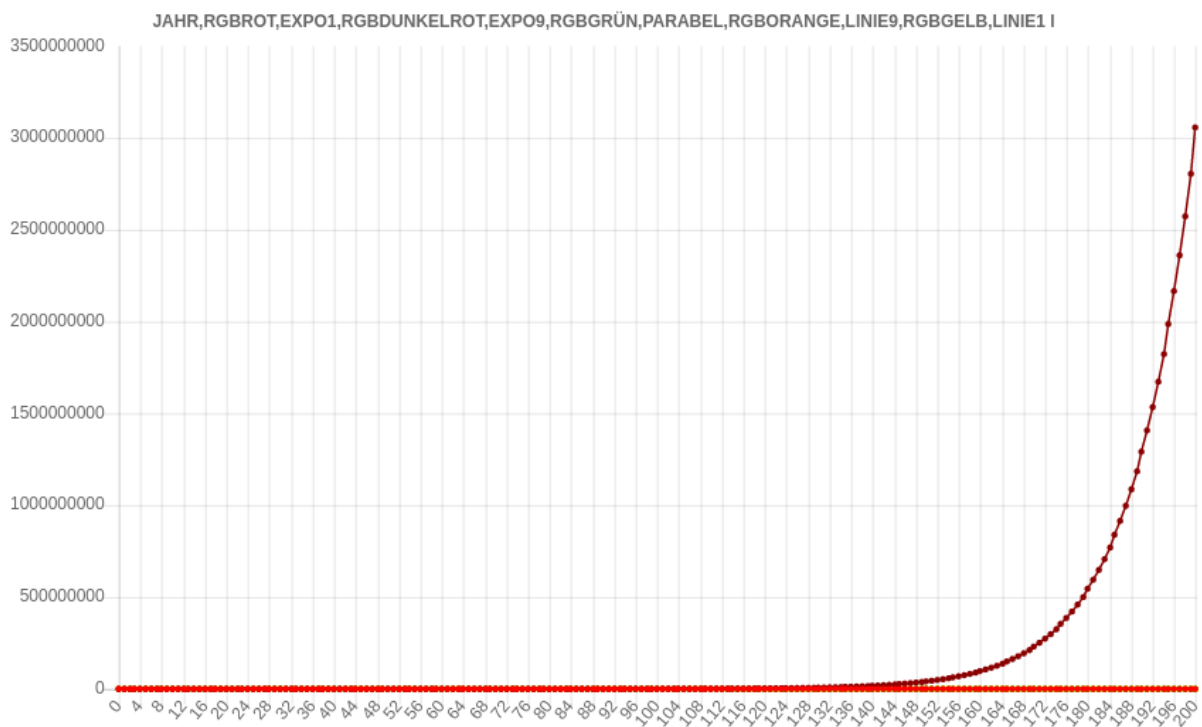
Programm 3.2: Wie entwickelt sich ein Betrag von 100 Euro bei einem „einfachen“ und normalen Zins von 1 % und 9 % und bei quadratischem Wachstum innerhalb von 200 Jahren.

```

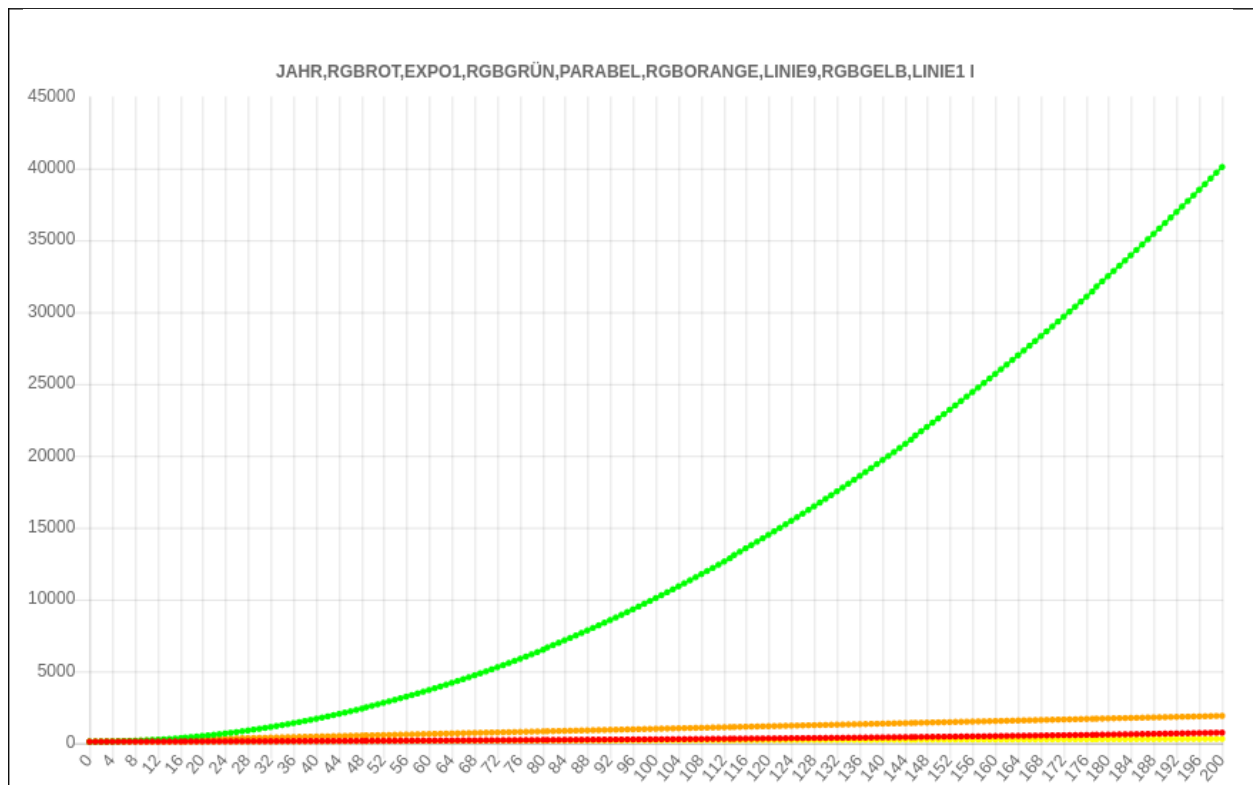
JAHR1 := 0 .. 200
EXP09 := 100. next EXP09 pred +% 9 at JAHR
EXP01 := 100. next EXP01 pred +% 1 at JAHR #EXP09
PARABEL:= JAHR * JAHR + 100
LINIE9 := 100. next LINIE9 pred + 9 at PARABEL
LINIE1 := 100. next LINIE1 pred + 1 at LINIE9
rnd 0
#sel JAHR rest 10 = 0 diese Bedingung wurde angewandt, um das Volumen der
#tab-Ausgabe zu verringern.
JAHR::=JAHR wort
'3
RGBDUNKELROT :=darkred leftat EXP09
RGBROT        :=red      leftat EXP01
RGBGRÜN       :=green    leftat PARABEL
RGBORANGE     :=orange   leftat LINIE9
RGBGELB       :=yellow   leftat LINIE1

```

Ergebnis (Liniendiagramm)



Ergebnis (Liniendiagramm ohne EXP09)



Ergebnis (tab-Ausgabe):

JAHR, EXPO9	, EXPO1, PARABEL, LINIE9, LINIE1 1				
0	100.	100.	100	100.	100.
10	237.	110.	200	190.	110.
20	560.	122.	500	280.	120.
30	1'327.	135.	1'000	370.	130.
40	3'141.	149.	1'700	460.	140.
50	7'436.	164.	2'600	550.	150.
60	17'603.	182.	3'700	640.	160.
70	41'673.	201.	5'000	730.	170.
80	98'655.	222.	6'500	820.	180.
90	233'553.	245.	8'200	910.	190.
100	552'904.	270.	10'100	1'000.	200.
110	1'308'925.	299.	12'200	1'090.	210.
120	3'098'702.	330.	14'500	1'180.	220.
130	7'335'754.	365.	17'000	1'270.	230.
140	17'366'396.	403.	19'700	1'360.	240.
150	41'112'576.	445.	22'600	1'450.	250.
160	97'328'419.	491.	25'700	1'540.	260.
170	230'411'765.	543.	29'000	1'630.	270.
180	545'468'442.	600.	32'500	1'720.	280.
190	1'291'322'174.	662.	36'200	1'810.	290.
200	3'057'029'208.	732.	40'100	1'900.	300.

Die grüne Parabel ist im ersten Bild nicht sichtbar. Die Punkte befinden sich hinter den anderen nicht-dunkelroten Punkten. Daher sieht die Parabel hier aus wie eine Gerade. Die Gerade verwandelt sich in eine schnell wachsende Kurve, wenn die noch schneller wachsende dunkelrote Exponentialkurve

aus dem Bild genommen wird. Das kann man nur verstehen, wenn man die Skalierungen der Ordinaten (Y-Achsen) vergleicht.

Programm 3.3: Das Schachbrettproblem: Lege auf das erste Feld ein Weizenkorn, auf das zweite zwei, das dritte 4, dann acht u.s.w. . Diese Exponentialentwicklung wird mit dem Polynom X^8 verglichen.

```
X1 := 1 .. 64
FELD := 1. next FELD pred +% 100 at X
HOCH8:= X hoch 8
norm3m
```

Ergebnis (tab-Ausgabe):

X	,FELD	,HOCH8	1
1.	1.		1.
2.	2.		256.
3.	4.		6561.
4.	8.		65536.
5.	16.		390625.
6.	32.	6m1.679616	
7.	64.	6m5.764801	
8.	128.	6m16.777216	
9.	256.	6m43.046721	
10.	512.	6m100.	
11.	1024.	6m214.358881	
12.	2048.	6m429.981696	
13.	4096.	6m815.730721	
14.	8192.	9m1.475789056	
15.	16384.	9m2.562890625	
16.	32768.	9m4.294967296	
17.	65536.	9m6.975757441	
18.	131072.	9m11.019960576	
19.	262144.	9m16.983563041	
20.	524288.	9m25.6	
21.	6m1.048576	9m37.822859361	
22.	6m2.097152	9m54.875873536	
23.	6m4.194304	9m78.310985281	
24.	6m8.388608	9m110.075314176	
25.	6m16.777216	9m152.587890625	
26.	6m33.554432	9m208.827064576	
27.	6m67.108864	9m282.429536481	
28.	6m134.217728	9m377.801998336	
29.	6m268.435456	9m500.246412961	
30.	6m536.870912	9m656.1	
31.	9m1.073741824	9m852.891037441	
32.	9m2.147483648	12m1.099511627776	
33.	9m4.294967296	12m1.406408618241	
34.	9m8.589934592	12m1.785793904896	
35.	9m17.179869184	12m2.251875390625	
36.	9m34.359738368	12m2.821109907456	
37.	9m68.719476736	12m3.512479453921	
38.	9m137.438953472	12m4.347792138496	
39.	9m274.877906944	12m5.352009260481	

40.	9m549.755813888	12m6.5536
41.	12m1.09951162778	12m7.984925229121
42.	12m2.19902325555	12m9.682651996416
43.	12m4.3980465111	12m11.688200277601
44.	12m8.79609302221	12m14.048223625216
45.	12m17.5921860444	12m16.815125390625
46.	12m35.1843720888	12m20.047612231936
47.	12m70.3687441777	12m23.811286661761
48.	12m140.737488355	12m28.179280429056
49.	12m281.474976711	12m33.232930569601
50.	12m562.949953421	12m39.0625
51.	15m1.12589990684	12m45.767944570401
52.	15m2.25179981369	12m53.459728531456
53.	15m4.50359962737	12m62.259690411361
54.	15m9.00719925474	12m72.301961339136
55.	15m18.0143985095	12m83.733937890625
56.	15m36.028797019	12m96.717311574016
57.	15m72.0575940379	12m111.429157112001
58.	15m144.115188076	12m128.063081718016
59.	15m288.230376152	12m146.830437604321
60.	15m576.460752303	12m167.9616
61.	18m1.15292150461	12m191.707312997281
62.	18m2.30584300921	12m218.340105584896
63.	18m4.61168601843	12m248.155780267521
64.	18m9.22337203685	12m281.474976710656

Man erkennt, dass das Polynom auf dem sechsten Feld bereits die Million überschritten hat, die Exponentialfunktion aber erst bei 32 ist. In der letzten Zeile wird dagegen deutlich, dass die Exponentialfunktion etwa um den Faktor 10'000 größer ist als der Polynomwert.

Programm 3.4: Berechne das Gesamtwachstum des Bruttoinlandsprodukts in Ostdeutschland, Westdeutschland und China in den Jahren von 1992 bis 2014 mit dem angegebenen Wachstumsdaten.

```
<TAB!
JAHR, DDRWA, BRDWA, CHINAWA 1
1988    0.    0.    0.
1989    1.85  3.9   4.2
1991   -47.8 11.09 13.56
1992     6.2  1.7   14.3
1993     8.7 -2.6   13.9
1994     8.1  1.4   13.1
1995     3.5  1.4   11.
1996     1.6  0.6    9.9
1997     0.5  1.5    9.2
1998     0.2  2.3    7.8
1999     1.8  2.1    7.6
2000     1.2  3.1    8.4
2001    -0.6  1.1    8.3
2002     0.2  0.1    9.1
2003    -0.3 -0.1   10.
2004     1.3  1.6   10.1
2005    -0.2  0.8   11.3
2006     3.4  3.8   12.7
```

```

2007    2.9    3.3    14.2
2008    0.6    1.     9.6
2009   -3.9   -6.1    9.2
2010    3.2    4.3   10.6
2011    1.9    3.8    9.5
2012    0.6    0.4    7.7
2013   -0.1    0.1    7.7
2014    1.4    1.6    7.4

```

```
!TAB>
```

```
sel JAHR>1991
```

```
DDR,BRD,CHINA:=100.,100.,100. next preds +% (DDRWA,BRDWA,CHINAWA)
      at CHINAWA
```

```
rnd 1
```

```
JAHR:= JAHR wort #subtext 3!2
```

```
TITEL:="Rot:DDR  Schwarz:BRD  Gelb:China"
```

```
gib TITEL,(JAHR,DDR,BRD,CHINA 1)
```

```
RGB:=red    leftat DDR
```

```
RGB:=black  leftat BRD
```

```
RGB:=yellow leftat CHINA
```

Ergebnis (tab-Ausgabe)(ohne TITEL und RGB):

```
JAHR ,DDR ,BRD ,CHINA 1
```

```

1992  100.0 100.0 100.0
1993  108.7  97.4 113.9
1994  117.5  98.8 128.8
1995  121.6 100.1 143.0
1996  123.6 100.7 157.1
1997  124.2 102.3 171.6
1998  124.4 104.6 185.0
1999  126.7 106.8 199.0
2000  128.2 110.1 215.8
2001  127.4 111.3 233.7
2002  127.7 111.4 254.9
2003  127.3 111.3 280.4
2004  128.9 113.1 308.8
2005  128.7 114.0 343.7
2006  133.1 118.3 387.3
2007  136.9 122.3 442.3
2008  137.7 123.5 484.8
2009  132.4 115.9 529.3
2010  136.6 120.9 585.5
2011  139.2 125.5 641.1
2012  140.0 126.0 690.4
2013  139.9 126.2 743.6
2014  141.9 128.2 798.6

```

Mit einem Gesamtwachstum von 100 auf 142 ist Ostdeutschland in diesem Zeitintervall deutlich besser als Westdeutschland mit einem Wachstum von 100 auf 128. Jetzt soll die Bedingung JAHR>1991 entfallen. Ferner nehmen wir an, dass die obigen in TAB-Klammern eingeschlossenen Daten sich in der Datei wachstum.tab befinden.

Programm 3.4: Berechne das Wachstum des Bruttoinlandsprodukts in Ostdeutschland, Westdeutschland und China in den Jahren 1988 bis 2014 mit dem angegebenen Wachstum.

wachstum.tab

```
DDR,BRD,CHINA:=100.,100.,100. next preds +% (DDRWA,BRDWA,CHINAWA)
                                     at CHINAWA
```

```
rnd 1
```

```
JAHR:= JAHR wort #subtext 3!2
```

```
TITEL:="Rot:DDR Schwarz:BRD Gelb:China"
```

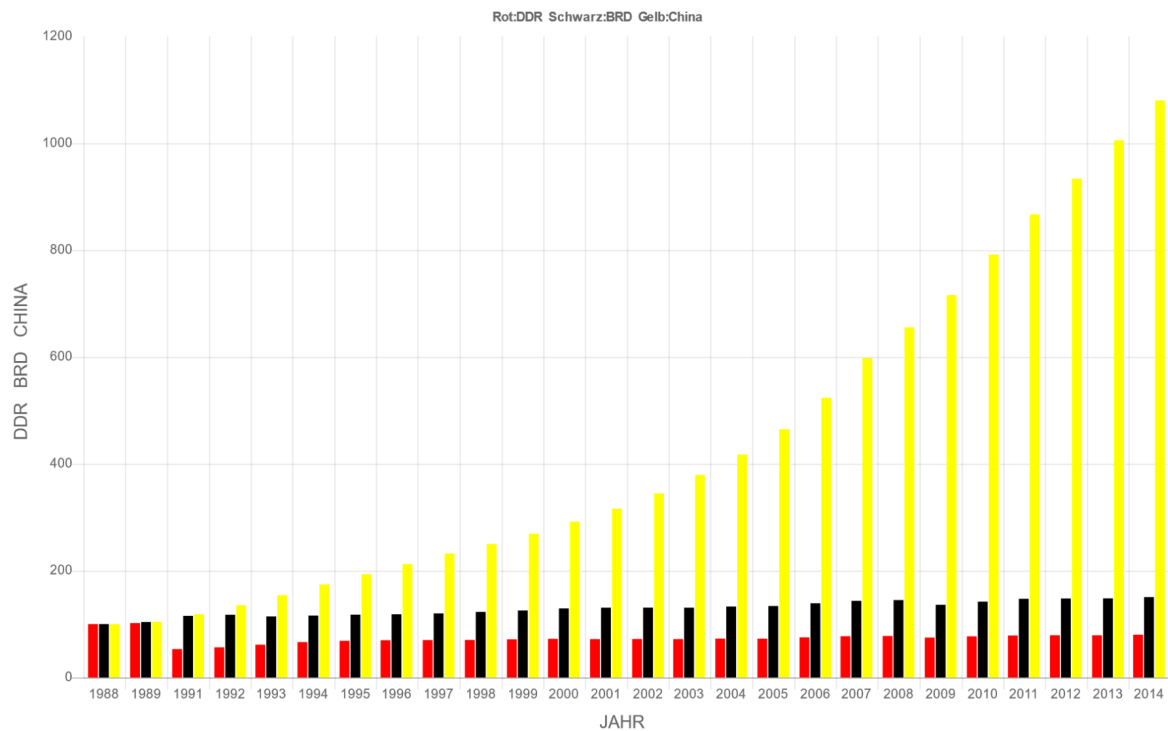
```
gib TITEL,(JAHR,DDR,BRD,CHINA 1)
```

```
RGB:=red leftat DDR
```

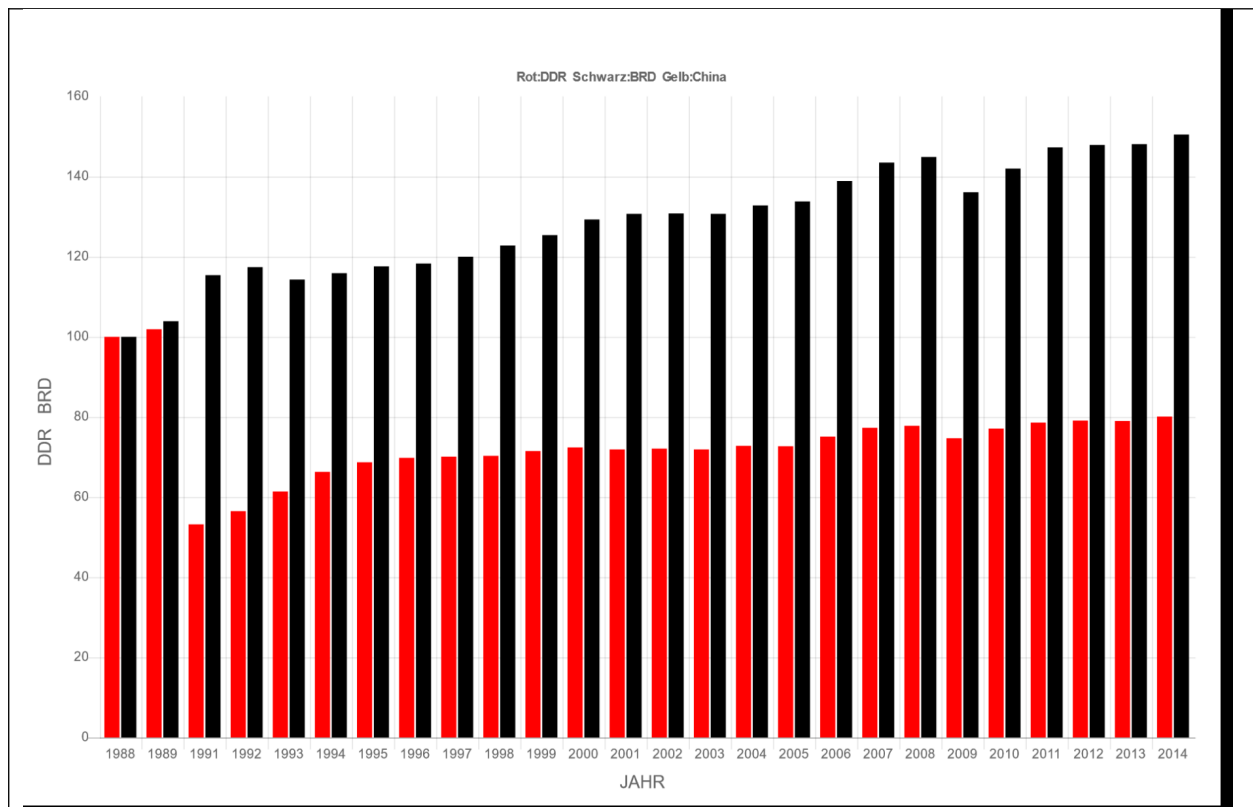
```
RGB:=black leftat BRD
```

```
RGB:=yellow leftat CHINA
```

Ergebnis (Säulendiagramm):



Ergebnis ohne China (Säulendiagramm):



Ergebnis (tab-Ausgabe):

TITEL			,(JAHR	,RGB	,DDR	,RGB	,BRD	1)
Rot:DDR	Schwarz:BRD	Gelb:China	1988	1.,0.,0.	100.0	0.,0.,0.	100.0	
			1989	1.,0.,0.	101.9	0.,0.,0.	103.9	
			1991	1.,0.,0.	53.2	0.,0.,0.	115.4	
			1992	1.,0.,0.	56.5	0.,0.,0.	117.4	
			1993	1.,0.,0.	61.4	0.,0.,0.	114.3	
			1994	1.,0.,0.	66.3	0.,0.,0.	115.9	
			1995	1.,0.,0.	68.7	0.,0.,0.	117.6	
			1996	1.,0.,0.	69.8	0.,0.,0.	118.3	
			1997	1.,0.,0.	70.1	0.,0.,0.	120.0	
			1998	1.,0.,0.	70.3	0.,0.,0.	122.8	
			1999	1.,0.,0.	71.5	0.,0.,0.	125.4	
			2000	1.,0.,0.	72.4	0.,0.,0.	129.3	
			2001	1.,0.,0.	71.9	0.,0.,0.	130.7	
			2002	1.,0.,0.	72.1	0.,0.,0.	130.8	
			2003	1.,0.,0.	71.9	0.,0.,0.	130.7	
			2004	1.,0.,0.	72.8	0.,0.,0.	132.8	
			2005	1.,0.,0.	72.7	0.,0.,0.	133.8	
			2006	1.,0.,0.	75.1	0.,0.,0.	138.9	
			2007	1.,0.,0.	77.3	0.,0.,0.	143.5	
			2008	1.,0.,0.	77.8	0.,0.,0.	144.9	
			2009	1.,0.,0.	74.7	0.,0.,0.	136.1	
			2010	1.,0.,0.	77.1	0.,0.,0.	142.0	
			2011	1.,0.,0.	78.6	0.,0.,0.	147.3	
			2012	1.,0.,0.	79.1	0.,0.,0.	147.9	
			2013	1.,0.,0.	79.0	0.,0.,0.	148.1	
			2014	1.,0.,0.	80.1	0.,0.,0.	150.5	

Wir haben China im zweiten Diagramm ausgeblendet, damit man die beiden Wachstumsdaten Deutschlands besser erkennt. Beispielsweise hat Ostdeutschland eine geringere Wertschöpfung als zu DDR-Zeiten. Die Bankenkrise hatte großen negativen Einfluss auf die ostdeutsche Wirtschaft, obwohl Ostdeutschland keine Bank besitzt, Ein Zuviel an Informationen kann scheinbar Wesentliches verdecken.

4 Hallo otto – Spielerei

Programm 4.1: Gib zwei Worte aus.

Hallo otto

Ergebnis (tabh)

WORT1

Hallo otto

Programm 4.2: Gib ein Paar von zwei Worten aus.

Hallo, otto

Ergebnis (tab)

WORT, WORT

Hallo otto

Programm 4.3: Gib einen Text mit Leerzeichen aus.

"Hallo Otto"

Ergebnis (tab)

TEXT

Hallo otto

Programm 4.4: Verkette zwei Worte mit Leerzeichen.

Hallo + " " + otto

Ergebnis (tab)

TEXT

Hallo otto

Programm 4.5: Gib einen Gruß mit einer Liste von zwei Worten aus.

GRUSS := Hallo otto

Ergebnis (ment)

TABMENT! GRUSS

GRUSS! WORT1

<GRUSS>
Hallo
otto
</GRUSS>

Programm 4.6: Gib zwei Worte mit je einem eigenen Spaltennamen aus.

LIEBER:=Hallo

GRUSS:=otto

Ergebnis (tab)

LIEBER, GRUSS

Hallo otto

Programm 4.7: Gib zwei Worte mit einem Spaltennamen aus.
GRUSS:= "Hallo otto"
Ergebnis (tabh)
GRUSS
Hallo otto

Programm 4.8: Sortiere eine Menge von Worten.
GRUSS:= {otto Hallo}
Ergebnis (tabh)
TABMENT! GRUSS
GRUSS! WORT1
GRUSS
Hallo otto

Programm 4.9: Repräsentiere ein Wort durch Metadaten und das andere durch Primärdaten.
HALLO := otto
Ergebnis (tab)
HALLO
otto

5 o++o für den Kindergarten?

Die Strichliste ist historisch gesehen die erste Darstellung einer Zahl. Sie könnte bereits eine Million Jahre alt sein. Gekerbte Hölzer sind nachweislich 150 Tausend Jahre alt. Konzepte, die in der Geschichte zuerst entwickelt wurden, sind in der Regel einfacher als spätere Konzepte. Deshalb sollten Strichlisten auch im Kindergarten einen breiteren Raum einnehmen.

Mit dem Einsatz von o++o im Kindergarten könnten folgende Ziele verfolgt werden:

1. Durch die gleichzeitige Darstellung von Dezimalzahlen und Strichlisten kann ein Kind die Größenordnung von Zahlen besser einschätzen. Zum Beispiel unterscheidet sich die Zahl Hundert von der Zahl Zehn nur durch eine Ziffer Null. Die entsprechenden Strichlisten unterscheiden sich jedoch erheblich.
2. Die Operationssymbole + * - : könnten gelehrt werden. Sie lassen sich wahrscheinlich leichter auf Strichlisten erklären. Die Strichlisten könnten in Dezimalzahlen umgewandelt werden und umgekehrt.
3. Der Algorithmus, der hinter der Strichlistenoperation (gib-Anweisung) steht, könnte anhand geeigneter Beispiele vermittelt werden.
4. Können Kinder im Vorschulalter entsprechende o++o-Programme formulieren? Hier sollte man sich vorstellen, dass eine App entwickelt wird, die Buchstaben oder Wörter durch Bilder ersetzt.

5.1 Strichlisten

Die Zählung von Tieren verschiedener Arten könnte die folgende kleine Zwischentabelle ergeben:

Elefant				
Hirsch				
Schwein				

Wenn ein weiterer Hirsch kommt, wird in der zweiten Zeile ein Strich hinzugefügt. Kommt dagegen ein Truthahn, muss eine neue Zeile mit dem Namen Truthahn und einem Strich am Ende angefügt werden.

Dies birgt bereits viele Probleme, obwohl Vorschulkinder eine solche Tabelle bereits erstellen können, wenn die Wörter durch Bilder oder einzelne Buchstaben ersetzt wurden. Es ist nicht klar, wie viele Spalten diese Tabelle hat, wenn nur "normale" Tabellen betrachtet werden. Wenn wir strukturierte Tabellen zulassen, können wir sagen, dass diese Tabelle eine Spalte TIER und eine Spalte STRICH enthält, aber die Werte der Spalte STRICH können für jedes Tier "wiederholt" werden. Ein zugehöriges Schema `TIER,STRICH l m` oder `TIER,STRICH l l` würde dies ausdrücken. Dabei ist `l` eine Abkürzung für Liste und `m` steht für Menge. Diese Symbole werden wieder postfix verwendet, d.h. sie werden nachgestellt. Das `m` ist in einem gib-Teil notwendig, damit jedes Tier nur einmal in der Zieltabelle erscheint.

Da sich viele Kinder für Autos interessieren, könnte man analog zum Zählen von Tieren auch Autos zählen. Daraus könnte sich die folgende Tabelle ergeben:

Golf				
A6				
Polo				
Wartburg				
A8				

Ist es im Kindergarten möglich, beim Zählen die Strukturtiefe der Tabelle zu erhöhen? Dann könnte die folgende (hsqh-) Tabelle erstellt worden sein:

VW									
Golf									
Polo									
Audi									
A6									
A8									
IFA									
Wartburg									

5.2 Die Umrechnungsoperationen zahl und |

Als Strich verwendet o++o das | (oder) Zeichen. Mehrere solcher Zeichen müssen bei der Eingabe in eckige Klammern eingeschlossen werden, damit sie von o++o als eine Liste von Strichen interpretiert werden. Wir werden die Operationen im folgenden Text mit selbsterklärenden Beispielen illustrieren.

Programm 5.2.1: Strichliste zu Zahl	Ergebnis
zahl	3

Umwandlung einer Zahl in eine Liste von Strichen:

Programm 5.2.2: Zahl zu Strichliste	Ergebnis
4 1	

5.3 Die Operationen + *

Verschiedene Darstellungen einer Additionsaufgabe. Der erste Inputtyp bestimmt wieder den Outputtyp.

Programme 5.3.1: Vier plus vier	Ergebnisse
4 + 4	8
4 +	8
+	
+ 4	
4 + 4 1	

Verschiedene Darstellungen einer Multiplikationsaufgabe:

Programme 5.3.2: Zehn mal zehn	Ergebnisse
10 * 10	100
10 * 10 1	

Programme 5.3.3: Zwei Darstellungen einer Subtraktionsaufgabe	Ergebnis
10 - 5	5
- 5	

Programm 5.3.4: Setze neben jede Zahl kleiner als 16 die entsprechende Strichliste	Ergebnis
X1:=0 ..15 Y :=X 1	X, Y 1 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15

5.4 o++o-Programme in den Kindergarten?

Welche der folgenden Programme sind für das Verständnis nützlich und welche sind lehrbar? Wann ist eine Syntax zu unverständlich? Diese Fragen werden im Folgenden umrissen.

Die Multiplikation ist das Zählen der Striche in einem Rechteck?

Programm 5.4.1: Jedes von vier Kindern bekommt 3 Äpfel. Wie viele Äpfel sind es insgesamt ?	Ergebnis (tabh)
NAME1 := Ernst Clara Sophia Claudia APFEL1:= at NAME ++	Zwischenergebnis nach den ersten 2 Zeilen NAME, APFEL1 1 Ernst Clara Sophia Claudia Endergebnis (++ steht für viele Additionen) 12

Die letzte Zeile kann auch ersetzt werden und man erhält das gleiche Ergebnis.

Programm 5.4.2: Jedes von vier Kindern bekommt 3 Äpfel. Wie viele Äpfel sind es insgesamt?	Ergebnis (tabh)
NAME1 := Ernst Clara Sophia Claudia APFEL1:= at NAME gib APFEL1 ++1	Zwischenergebnis nach den ersten 2 Zeilen NAME, APFEL1 1 Ernst Clara Sophia Claudia Endergebnis (++1 zählt) 12

Programm 5.4.3: Zählen von Tieren				
TIER1:=Elefant Hirsch Elefant Schwein Elefant Hirsch Schwein Schwein Elefant gib TIER,ANZAHL m ANZAHL:=TIER! ++				
Ergebnis (tabh-Ausgabe):				
TIER,	ANZAHL	1		
Elefant				
Hirsch				
Schwein				

Programm 5.4.4: Zählen von Autos				
<TAB! MARKE ,FARBE, TYP, GEWICHT 1 VW Blau Polo 1250 IFA Papyrus 500 580 VW Blau Golf 1450 Audi Gelb Quatro 2070 VW Blau Polo 1380 IFA Beige 601 620 VW Rot Golf 1400 Audi Rot Quatro 2100 IFA Beige 601 620 VW Beige Polo 1300 !TAB> gib MARKE,ANZAHL,(FARBE,ANZAHL m) m ANZAHL:=TYP! ++				
Ergebnis (tabh)				
MARKE,	ANZAHL,	(FARBE,	ANZAHL2 1)	1
Audi			Gelb	
			Rot	
IFA			Beige	
			Papyrus	
VW			Beige	
			Blau	
			Rot	

Programme 5.4.5: 3 Divisionsoperationen	Ergebnisse
13 div 4	3
13 : 4	3.25
13 divrest 4	3,1

All diese Operationen scheinen für den Kindergarten zu kompliziert zu sein.
Wenn man nicht nur mit Zahlen, sondern auch mit Tabellen rechnet, könnte man neue Divisionsoperationen einführen. Dies kann jedoch an dieser Stelle nicht zu Ende diskutiert werden.

Programm 5.4.6: Problem: Verteile 15 Äpfel unter 4 Kindern. Wer entwirft das o++o-Programm?	Ergebnis (tabh)
	Ernst
	Clara
	Sophia

	Claudia			
--	---------	--	--	--

Eine weitere sehr wichtige Operation der Digitalisierung ist die Selektion. Wäre eine Datenbankoperation wie die Selektion bis zu einem gewissen Grad lehrbar?

gegeben:

NAME,	ALTER	1
Ernst	8	
Clara	6	
Sophia	6	
Claudia	4	
Ulrike	5	
Käthe	4	

meinefamilie.tab

Programm 5.4.7: Wie alt ist Claudia?	Ergebnis
aus meinefamilie.tab	NAME, ALTER 1
sel Claudia	Claudia 4

sel kürzt Selektion ab

Programm 5.4.8: Gesucht sind alle 6jährigen Kinder.	Ergebnis
aus meinefamilie.tab	NAME, ALTER 1
sel ALTER = 6	Clara 6
	Sophia 6

Programm 5.4.9: Gesucht sind alle Kinder, die jünger als 6 sind.	Ergebnis
meinefamilie.tab	NAME, ALTER 1
sel ALTER < 6	Claudia 4
	Ulrike 5
	Käthe 4

6 o++o im Schulunterricht

Es gibt viele mögliche Anwendungen für o++o in der Schule. Vor allem in den Fächern Mathematik und Informatik. Aber auch in allen anderen Fächern kann o++o verwendet werden, um Daten aus vorgegebenen Tabellen, Dokumenten oder aus Wikipedia zu extrahieren (siehe: Kapitel 11). Wir wollen hier nicht alle möglichen, typischen Anfragebeispiele vorstellen. Wir wollen uns auf die so genannten "Brute-Force-Algorithmen" für die Mathematik beschränken. Das sind die einfachsten, d.h. die methodisch besten. Da alle diese Algorithmen im Hauptspeicher realisiert werden, brauchen wir uns um die Effizienz keine Sorgen zu machen. Jetzt beginnen wir mit einem einfachen Algorithmus. Wir hoffen, dass es das einfachste Programm für eine Nullstelle ist. Der Abschnitt endet mit Programmen für die Benotung von Schülern und Überlegungen, die eventuell auch für den Kindergarten wichtig sein können.

Programm 6.1: Berechne auf einfache Weise die Nullstelle der Sinusfunktion im Intervall [3, 4].	Ergebnis
X1:= 3 ... 4! 0.00001	X1
sel X sin <0	3.1416
sel X pos =1	

Programm 6.2: Berechne auf einfache Weise die Nullstelle der Sinusfunktion im Intervall [3, 4].	Ergebnis
X1:= 3 ... 4! 0.00001	X1
sel X sin * (X +0.00001 sin) <=0	3.14159

Programm 6.3: Berechne die ganzzahligen Nullstellen des Polynoms " $X^2-15X+56$ ".	Ergebnis (tabh)
X1:= -100 .. 100	X1
sel X poly [1 -15 56]=0	7 8
# oder sel X- 15 *X + 56=0	

Mit den folgenden Programmen wird gezeigt, dass Schüler, die die Integral- und Differentialrechnung nicht erlernt haben, gleichwohl in der Lage sind, ihre Schulanwendungen zu verstehen und zu nutzen, die im Wesentlichen darin bestehen:

1. Wie groß sind Flächen unter Kurven?
2. Was sind lokale Extrema von Funktionen?

Programm 6.4: Berechne die Fläche unter einem Kreisbogen mit Durchmesser 4 im Intervall [0, 2].	Ergebnis
X1:= 0 ... 2!0.000'1	AGG
HOEHE:= X*X - 4 abs sqrt	3.14169223791
RECHTECK:=HOEHE*0.0001	
gib AGG AGG:=RECHTECK!++	

Programm 6.5: Anfrage 6.4, aber kürzer und präziser.	
PI:=0 ... 2!0.000'01 poly [-1 0 4] sqrt*0.000'01 ++	
Ergebnis	
PI	
3.14160264044	

Programm 6.6: Ermittle pi durch Nullstellenbestimmung mit Intervallhalbierung.

```

MI,LI,RE l:= 1.,2.,4.  while RE - LI  >= 0.000'000'01 !
      LI pred +(RE pred) :2;
      (MI;RE pred)if MI sin > 0!(LI pred;MI)
sel MI pos- =1
gib  MI

```

Zwischenergebnis nach erster Programmzeile:

MI	,LI	,RE l
1.	2.	4.
3.	3.	4.
3.5	3.	3.5
3.25	3.	3.25
3.125	3.125	3.25
3.1875	3.125	3.1875
3.15625	3.125	3.15625
3.140625	3.140625	3.15625
3.1484375	3.140625	3.1484375
3.14453125	3.140625	3.14453125
3.142578125	3.140625	3.142578125
3.1416015625	3.140625	3.1416015625
3.14111328125	3.14111328125	3.1416015625
3.14135742188	3.14135742188	3.1416015625
3.14147949219	3.14147949219	3.1416015625
3.14154052734	3.14154052734	3.1416015625
3.14157104492	3.14157104492	3.1416015625
3.14158630371	3.14158630371	3.1416015625
3.14159393311	3.14158630371	3.14159393311
3.14159011841	3.14159011841	3.14159393311
3.14159202576	3.14159202576	3.14159393311
3.14159297943	3.14159202576	3.14159297943
3.14159250259	3.14159250259	3.14159297943
3.14159274101	3.14159250259	3.14159274101
3.1415926218	3.1415926218	3.14159274101
3.14159268141	3.1415926218	3.14159268141
3.14159265161	3.14159265161	3.14159268141
3.14159266651	3.14159265161	3.14159266651
3.14159265906	3.14159265161	3.14159265906

Ergebnis

MI

3.14159265906

Man erkennt, dass die ersten 8 Stellen von pi (3.14159265359...) nach dem Komma korrekt sind und dass diese Korrektheit lediglich 29 Schritte benötigt.

Programm 6.7: Berechne die ersten 7 Fibonacci-Zahlen.

```

Xl:= 1 .. 7

```

```

FIB1,FIB2 l:= 0,1  next  preds nth 2;preds ++ at X

```

Ergebnis (tab)

X	,FIB1	,FIB2	l
---	-------	-------	---

1	0	1	
2	1	1	
3	1	2	

4	2	3
5	3	5
6	5	8
7	8	13

Programm 6.8: Berechne das Pascaldreieck bis zum Exponenten 9.

```
X1 := 0 ..9
XTUP:= 1 next (0, XTUP pred)+(XTUP pred,0) at X
wort
```

Ergebnis (tab)

X ,XTUP	1
0	1
1	1,1
2	1,2,1
3	1,3,3,1
4	1,4,6,4,1
5	1,5,10,10,5,1
6	1,6,15,20,15,6,1
7	1,7,21,35,35,21,7,1
8	1,8,28,56,70,56,28,8,1
9	1,9,36,84,126,126,84,36,9,1

Jetzt stellen wir einen Brute-Force-Algorithmus für ein Maximum vor.

Programm 6.9: Finden Sie auf einfache Weise das lokale Maximum der Sinusfunktion im Intervall [1, 3].

```
1 ... 3.!0.00001 sin max
```

Ergebnis

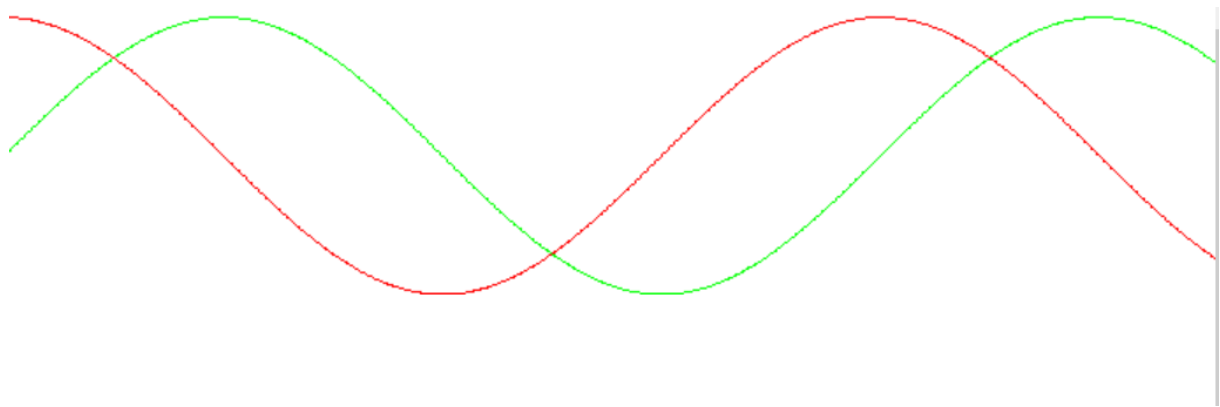
PZAHL

0.999999999993

Programm 6.10: Berechnen Sie die Sinusfunktion und eine Näherung der ersten Ableitung im Intervall [0,4].

```
X1:= 0 ... 10!0.01
SINUS := X sin
ABLEITUNG:=X+0.0001 sin -(X sin)*10000
RGBSIN:=green leftat SINUS
RGBABLEITUNG:=red leftat ABLEITUNG
```

Ergebnis (bild):



Ergebnis (tab): Es besteht aus 1001 Zeilen.

X,	RGBSIN,	SINUS,	RGBABLEITUNG,	ABLEITUNG 1
0.	0.,1.,0.	0.	1.,0.,0.	0.999999998333
0.01	0.,1.,0.	0.00999983333417	1.,0.,0.	0.999949498758
0.02	0.,1.,0.	0.0199986666933	1.,0.,0.	0.999799005067
0.03	0.,1.,0.	0.0299955002025	1.,0.,0.	0.999548532308
0.04	0.,1.,0.	0.0399893341866	1.,0.,0.	0.999198105529
0.05	0.,1.,0.	0.0499791692707	1.,0.,0.	0.998747759772
0.06	0.,1.,0.	0.0599640064794	1.,0.,0.	0.998197540071
0.07	0.,1.,0.	0.0699428473375	1.,0.,0.	0.997547501448
0.08	0.,1.,0.	0.0799146939692	1.,0.,0.	0.996797708907
0.09	0.,1.,0.	0.089878549198	1.,0.,0.	0.995948237425
...				
9.98	0.,1.,0.	-0.527131998452	1.,0.,0.	-0.849757059218
9.99	0.,1.,0.	-0.535603334614	1.,0.,0.	-0.844442914714
10.	0.,1.,0.	-0.544021110889	1.,0.,0.	-0.83904432662

Die folgenden Beispiele basieren auf einer fiktiven Notentabelle mit Prüfungen:

NAME,	(FACH,	EXAM1,	NOTE1 1)1
Einstein	Deutsch	1 3	1 2 1 3 1
	Physik	1 f	1 2 1 1 1 1 1
	Algebra	1 2	1 1 1 1 2
	Kunst	3	3 3 2 1
Gauss	Deutsch	2 3	1 2 1 2
	Algebra	1 1	1 1 1 1
Guericke	Physik	k 1	1 1 2 1
	Deutsch	2 1	1 2 1 1 1
	Algebra	1 1	2 1 1 1 2
Newton	Physik	1 1	2 1 1 2 1
Konfuzius	Philosophie	1 1	1 1 1 1 1
	Chinesisch	1 1	2 2 1 1 1
Marx	Ökonomie	1 1	1 2 1 2 1
	Philosophie	1 2	1 3 1
Brecht	Deutsch	1 1	1 1 1 1 1 1 1 1
	Philosophie	1 2	2 1 1 1 2
Cantor	Mengentheorie	1 1	1 1 1

Tabment 6.1: genies.tabh (k: krank, f: gefehlt)

Programm 6.11: Berechnen Sie die gewichteten Mittelwerte für jede Person und jedes Fach und den Gesamtwert. Sortieren Sie die Daten.

```
aus genies.tabh
gib DUR,(NAME,DUR,(FACH,DUR m)m)
    DUR:=EXAM1 ++: *0.6 + (NOTE1 ++: *0.4)! ++:
rnd 1
```

Ergebnis (tab):

DUR,(NAME,	DUR2,(FACH,	DUR3 1)1
1.4 Brecht	1.2 Deutsch	1.0
	Philosophie	1.5
Cantor	1.0 Mengentheorie	1.0
Einstein	1.7 Algebra	1.4
	Kunst	2.7

		Deutsch	1.8
		Physik	1.1
Gauss	1.6	Algebra	1.0
		Deutsch	2.1
Guericke	1.2	Algebra	1.2
		Deutsch	1.4
		Physik	1.1
Konfuzius	1.1	Chinesisch	1.2
		Philosophie	1.0
Marx	1.4	Ökonomie	1.2
		Philosophie	1.6
Newton	1.2	Physik	1.2

Wenn wir den Durchschnitt nach Abschluss der ersten Prüfung berechnen wollen, können wir die folgende Formel verwenden:

$DUR := EXAM1 \cdot nth(1) \cdot 0.3 + (NOTE1 + \dots + NOTE_n) \cdot 0.7$

Programm 6.12: Bestimmen Sie alle Fächer und Personen, die eine 1 und eine nachfolgende 3 erhalten haben.

```
aus genies.tabh
sel NAME FACH! NOTE=1 & NOTE succ=3      #succ: successor = Nachfolger
```

Ergebnis:

NAME,	(FACH,	EXAM1,	NOTE1	1)	1)
Einstein	Deutsch	1	3	1	2 1 3 1
Marx	Philosophie	1	2	1	3 1

Nun wenden wir uns noch weiteren "einfachen" Problemen zu. Es mag sein, dass diese Aufgaben nicht nur für die Schule, sondern auch für den Kindergarten von Bedeutung sind. Nun wird allgemein angenommen, dass die Addition von natürlichen Zahlen die einfachste und die Division die schwierigste der vier Grundrechenarten ist. Das könnte falsch sein. Wir haben Experimente mit einem 3- und einem 6-jährigen Kindergartenkind gemacht. Die Aufgabe bestand darin, 11 Äpfel auf vier Kinder zu verteilen. Die vier Kinder wurden durch Fotos dargestellt. Weder das 6-jährige Kind noch das 3-jährige Kind hatten ein Problem. Sie erzielten das gleiche Ergebnis bei der Division. Es wurde in einer Tabelle dargestellt:

KIND,	APFEL
Ernst	
Clara	
Sophia	
Claudia	

Tabment 6.1: 11 geteilt durch 4

Was können wir aus diesem Experiment lernen?

1. Kleine Kinder können einen Apfel nicht teilen. Sie haben noch kein klares Verständnis von $1/2$ oder $2/3$, ..., so dass die "gewöhnliche" Division nicht gelehrt werden kann.
2. Es gibt keinen Rest in der Division; es gibt keinen Grund, etwas zu verschwenden.

Betrachten wir die **Addition**, die nächst einfachste Operation. Die einfachste Darstellung der Zahl drei sind drei Striche. Das Gleiche gilt für beliebig viele andere natürliche Zahlen. Hier betrachten wir nur zwei Zahlen: 3 und 4.

Wir müssen diese durch Listen oder Bags (Multimengen) darstellen, weil die Menge $\{| \}$ dasselbe ist wie $\{|\}$. Das Ergebnis jeder Operation wäre eins.

APFEL 1
drei.tabh

APFEL 1
vier.tabh

Programm 6.13: drei plus vier	Ergebnis
aus drei.tabh,vier.tabh	APFEL1
gib APFEL1	

Auch hier wird deutlich, dass ein solcher Vorgang nur einen geringen Aufwand im Unterricht erfordern könnte. Aber was ist das Ergebnis von 4 Äpfeln und 3 Birnen? Da jedes Paar von zwei Tabmenten wieder ein Tabment ist, könnte das Ergebnis dieser "Addition" ein Tabment vom Typ APFEL1, BIRNEL sein.

Die Multiplikation kann auch auf sehr einfache Weise gehandhabt werden. Betrachten wir die sehr einfache Frage, wie viele Äpfel für 4 Kinder benötigt werden, wenn jedes Kind 3 Äpfel möchte:

Programm 6.14: vier mal 3	Zwischenergebnis nach Zeile 2
KIND1:= Ernst Clara Sophia Claudia	KIND, APFEL1 1
APFEL1:= at KIND	Ernst
gib APFEL1	Clara
	Sophia
	Claudia

Dieser Multiplikationsalgorithmus ist nicht nur einfacher, er macht auch deutlich, dass die Multiplikation im Wesentlichen die Berechnung der Fläche eines Rechtecks ist.

Das obige Zwischenergebnis erhalten wir auch durch das folgende Programm:

KIND1:=Ernst Clara Sophia Claudia
ÄPFEL:= at KIND
gib KIND,ÄPFEL 1 # ÄPFEL ist atomar, d.h. es wird jede Apfelliste als # Gesamtheit übernommen

Wenn wir die **Subtraktions**operation auf Kollektionen mit unterschiedlichen Elementen (Mengen) anwenden wollen, kann die Subtraktion durch eine Selektion ausgedrückt werden.

Programm 6.15: Subtraktion (Differenz) mit Mengen	Ergebnis
NAMEm := {Ernst Clara Ulrike}	NAME1
sel- NAME in {Ulrike Sophia}	Clara Ernst

Wir schließen diesen Abschnitt mit den folgenden Aussagen ab:

1. Das Ergebnis unserer "Rechenoperationen" sind keine Zahlen, sondern Tabellen.
2. Der Umgang mit Tabellen ist vermutlich einfacher als der mit Zahlen, weil die Abstraktionsebene niedriger ist.

7 Multiplikation, Schule und Digitalisierung

In diesem Kapitel soll gezeigt werden, dass der gängige Multiplikationsalgorithmus für Dezimalzahlen durch einfachere ergänzt werden könnte und sollte und dass ganz allgemein eine tiefe Digitalisierung (TD) angestrebt werden sollte. Die tiefe Digitalisierung kann wahrscheinlich nur durch mathematisches Verständnis umgesetzt werden. Im Gegensatz zur flachen Digitalisierung, bei der der Nutzer in der Regel durch einfaches Anklicken eines Buttons ein Computerergebnis präsentiert bekommt und oft unsicher ist, ob dieses Ergebnis richtig ist, sollte der Nutzer bei der tiefen Digitalisierung das Ergebnis genauso verstehen können wie die Berechnung von 132.66 mal 453.2 mit einem Taschenrechner. Das könnte die Fläche seines Grundstücks sein. Der große Unterschied zwischen der heutigen Verwendung von Taschenrechnern und der heutigen Verwendung von leistungsstarken Computern besteht darin, dass die Benutzer jahrelang damit verbracht haben, die Einzeldatenoperationen zu lernen: $+$ $-$ $*$ $:$ $\sin$ $\log$ Massendatenoperationen stehen noch nicht auf dem Lehrplan. Selektion - manchmal auch Filteroperation genannt - sowie Operationen zum Zusammenführen von Tabelleninhalten zur Umstrukturierung ... zählen wir zu den Massendatenoperationen. Diese werden nicht auf einzelne Zahlen angewendet, sondern auf möglicherweise sehr große strukturierte Tabellen, die neben Zahlen auch Wörter und Texte enthalten können. Wenn der Nutzer solche Massendatenoperationen verstanden hat und sie im Rahmen einer Programmiersprache implementiert wurden, kann er diese Ergebnisse auch interpretieren und im Zweifelsfall korrigieren, verändern oder verbessern.

7.1 Wer kann im Kopf multiplizieren?

Begebenheit 1

In einer Mathematikprüfung, die ein Pharmaziestudent aus Bologna im zweiten Studienjahr zu absolvieren hatte, musste der Student unter anderem 7 mal 8 berechnen. Der Pharmaziestudent: 59
Der Algebra-Professor: Aber 59 ist keine gerade Zahl. Der Pharmaziestudent: 64

Begebenheit 2

Wallerie - ein Erfurter Kindergartenmädchen der großen Gruppe - ist heute schon Studentin.
Ich habe ihr eine Aufgabe gestellt: Wie viele Brauseflaschen enthält ein Kasten mit 4 Reihen, wenn in jeder Reihe 5 Flaschen stehen?
Wallerie dachte eine Weile nach: Neunzehn
Ihr Vater - ein junger Ingenieur: Du rechnest nicht, du rätst.

Begebenheit 3

Ich frage Isabella, eine Schülerin der zweiten Klasse aus Gerwisch: Wie viel ist 3 mal 4? Nach einer Weile: Zwölf
Der Vater: Das hat aber lange gedauert.

Aus dem zweiten Vorkommnis schließe ich, dass Vorschulkinder bereits das Wesen der Multiplikation verstanden haben. Natürlich ist es möglich, dass einige Vorschulkinder... 4 mal 5 nicht genau im Kopf ausrechnen können. Aber 3 mal 4 würde ich jedem Kind zutrauen. Es steht außer Frage, dass sie niemals in der Lage sein werden, 12 mal 13 auf diese Art und Weise zu berechnen. Ich glaube sogar, dass kein Mensch in der Lage ist, 7 mal 8 in seinem Kopf zu berechnen. Ältere Erwachsene haben das Einmaleins in der Schule so oft ausrechnen(?) müssen, dass sie es nur noch auswendig können und sich nicht mehr daran erinnern, wie sie als Kind multipliziert haben. Früher war es sehr wichtig, das Einmaleins auswendig zu können, weil es eine Voraussetzung für die schriftliche Dezimalmultiplikation war.

Die Meinung eines Amateur-Neurologen: In den vielen Schuljahren, in denen das Einmaleins gelehrt wurde, wurden die ursprünglichen Neuronenverbindungen oder Gehirnzellen "überschrieben" und sind praktisch nicht mehr vorhanden.

Deshalb ist die überwiegende Mehrheit der Erwachsenen nicht in der Lage, die ursprüngliche Multiplikation in ihrem Kopf durchzuführen. Sie können das Einmaleins nur auswendig und die schriftliche Dezimalmultiplikation nicht im Kopf durchführen. Selbst bei der Multiplikation kleinerer Zahlen wie 29 mal 63 werden sie mit einfacher zu handhabenden Rechengesetzen arbeiten und nicht den Algorithmus verwenden, in den ihre Lehrer, Eltern und Großeltern viel Zeit und Mühe investiert haben. Anhand der Begebenheiten 1, 2 und 3 kann man sogar vermuten, dass viele Erwachsene gar nicht mehr wissen, dass Kinder rechnen müssen, um die Ergebnisse zu erhalten. Wenn man so jung ist, kann man es noch nicht auswendig, ohne zu rechnen. Eine fast richtige Antwort deutet darauf hin, dass gerechnet wurde, ebenso wie eine sehr schnelle Antwort darauf hindeutet, dass nicht gerechnet und somit nicht nachgedacht wurde.

7.2 Wer kann schriftlich multiplizieren?

7 mal 8 mit einem Bleistift rechnen, sollte jedes Kind der zweiten Klasse beherrschen. Die Voraussetzung ist, dass man sich die Zahlen bis hundert vorstellen kann. Das kann man, wenn man bis hundert zählen kann. Wenn man die Aufgabe veranschaulicht, sollten viele Kinder sie noch schneller lösen können:

Jedes der sieben Kinder möchte acht Bonbons haben. Wie viele Süßigkeiten müssen Sie kaufen?

1. Schreibe die Namen von sieben Kindern untereinander.
2. Setze acht Striche leserlich hinter jeden Namen.
3. Zähle alle Striche.

Jeder kann sich vorstellen, dass man mit diesem Algorithmus beliebig große Zahlen multiplizieren kann. Aber es wäre schön, wenn man das Ergebnis nicht nur aussprechen und als Wort schreiben könnte, sondern wenn man das Ergebnis kompakter als Dezimalzahl schreiben könnte.

4. Das Ergebnis in eine Dezimalzahl umwandeln.

Wenn jemand auf diese Weise 100 mal 100 berechnen will, ist die Wahrscheinlichkeit, ein richtiges Ergebnis zu erhalten, sehr gering. Außerdem würde es sehr, sehr lange dauern. Im Zeitalter leistungsfähiger Computer sollten diese Argumente jedoch unbedeutend sein. Entscheidend ist einen Algorithmus klar verstanden zu haben. Die Frage bleibt:

Ist die Strichlistenmultiplikation der einfachste Multiplikationsalgorithmus?

7.3 Wer kann die Multiplikation programmieren?

Begebenheit 4

Eine Ingenieurin aus meinem ehemaligen Institut erzählt mir, dass sie die erste war, die in der Staßfurter Fernsehfabrik Assembler lernte. Damit konnte sie nach der Wiedervereinigung Effizienzprobleme lösen, die ihr ein neuer Chef der Fernsehgerätefabrik nicht zugetraut hatte. Sie erinnert sich sehr genau, dass sie große Schwierigkeiten hatte, C zu lernen.

Begebenheit 5

Da ich kein Professor war, konnte ich 1992 am Gastwissenschaftlerprogramm von IBM Deutschland teilnehmen. Im IBM-Forschungszentrum - dem wissenschaftlichen Zentrum in Heidelberg - wurde schon seit vielen Jahren an dem Datenbankprojekt AIMP (Advanced Information Management Prototype) entwickelt. Im Wesentlichen handelte es sich dabei um die Datenbankanfragesprache HDBL (Heidelberg Database Language), mit der NF²-Relationen verarbeitet werden konnten. Diese Relationen verallgemeinern das Tabellenkonzept des Relationalen Datenmodells auf strukturierte Tabellen. Letztlich konnte die IBM Deutschland die Zentrale in den USA aber nicht davon überzeugen, dass ihr Prototyp auf den Markt gebracht werden soll. Noch bemerkenswerter war für mich, dass ein Mitarbeiter, der in einer sehr großen Anzahl von Publikationen über HDBL als Autor aufgeführt war, die einfachsten Fragen zu HDBL nicht beantworten konnte. Er erklärte mir dann, dass er jahrelang in

PASCAL für die HDBL-Dateiverwaltung programmiert habe und eigentlich gar kein Interesse daran habe, Anfragen in HDBL zu formulieren.

Begebenheit 6

Ich wollte UNIX verstehen und kaufte mir das Buch "UNIX und C" vom VEB Verlag und Technik. Ich las, markierte, las, markierte, las, markierte und wiederholte. Aber nur langsam und mit geringem Erfolg.

Meiner Meinung nach ist die wichtigste Schlussfolgerung aus diesen Begebenheiten, dass es sehr wichtig ist, welche Sprache man zuerst lernt. Das Umlernen dürfte in der Regel schwieriger sein als das Neuerlernen. Außerdem muss man davon ausgehen, dass Denkweisen aus prozeduralen oder objektorientierten Programmiersprachen für Anfragesprachen wenige Vorteile bieten. Ferner kann man programmieren sicher nicht durch lesen allein lernen. Man muss Fehler selber machen.

Ich denke, dass Operationen den meisten Menschen eher durch Algorithmen als durch deskriptive Formalisierungen beigebracht werden können. Wenn jemand einen Algorithmus programmieren kann, kann oder sollte er ihn in elementarere Schritte zerlegen, um ihn besser zu verstehen. Wenn geeignete Programmiersprachen zur Verfügung stehen, hat die Dezimalmultiplikation keine Chance gegen den Strichalgorithmus, was die Erlernbarkeit und Lesbarkeit betreffen. Ganz gleich, wie hoch das Abstraktionsniveau einer Programmiersprache ist. Ich glaube, weniger als ein Prozent der Weltbevölkerung kann heute die Dezimalmultiplikation in 30 Minuten programmieren. In der Tat sind weit weniger als 1 Prozent der Weltbevölkerung Softwareentwickler.

Kürzlich belegte China laut Hacker Rank durch gut organisierte Programmierolympiaden nicht nur den ersten Platz im Bereich der funktionalen Programmierung, sondern auch den ersten Platz in der Gesamtwertung der "besten Entwickler" der Welt, noch vor Russland. Für mich persönlich ist das besonders beeindruckend, weil ein chinesischer Student, der bei mir 2009 sein Diplom abgeschlossen hatte, mir erzählte, dass er China verlassen hat, weil er an seiner chinesischen Universität zunächst PASCAL lernen sollte. Diese (methodisch vielleicht gar nicht so schlechte) Sprache hielt er nicht für zeitgemäß.

Die wirtschaftlichen Auswirkungen, die sicherlich auch auf die chinesische Bildungspolitik zurückzuführen sind, sind schon heute deutlich zu sehen. Von 1990 bis heute hat sich der Anteil Deutschlands an den weltweiten Exporten von Hochtechnologiegütern fast halbiert. China hat seinen Anteil in dieser Zeit von einem auf 24 Prozent gesteigert (laut Handelsblatt).

In der funktionalen französischen Sprache OCaml kann der entsprechende Code sicherlich am elegantesten gelöst werden. Dass OCaml sehr gute Konzepte und effiziente Implementierungen hat, sieht man vielleicht auch daran, dass Microsoft OCaml kopiert hat. F# hat sogar die gleiche Syntax wie OCaml. Betrachten wir zunächst einen Multiplikationsalgorithmus auf der meiner Meinung nach hohen Abstraktionsebene der funktionalen Programmierung.

Lassen Sie uns zunächst jedoch kurz klären, wie die Multiplikation in OCaml praktisch durchgeführt wird:

Programm 7.3.1: Ganzzahlige Multiplikation auf Benutzerebene in OCaml	Ergebnis
<code>7*8;;</code>	<code>- : int = 56</code>

Für 7.1 mal 8.1 muss man ein anderes Operationssymbol wählen.

Programm 7.3.2: Multiplikation von Gleitkommazahlen auf Benutzerebene in OCaml	Ergebnis
<code>7.1 *. 8.1;;</code>	<code>- : float = 57.51</code>

OCaml hat auch einen Datentyp `bigint` für beliebig große ganze Zahlen.

Sophia								
Ulrike								
Claudia								
Käthe								
Endergebnis								
56								

Wenn man nur am Endergebnis interessiert ist, kann man auch alle Namen durch einen einzigen ersetzen. Man könnte auch eine gib-Anweisung für die letzten beiden Zeilen schreiben, die unabhängig von der gegebenen Struktur durchzählt:

Programm 7.3.5: Verkürzte Strichlistenmultiplikation in o++o								
NAME1:= otto *1 7 STRICH1:= *1 8 at NAME gib SIEBEN_MAL_ACHT SIEBEN_MAL_ACHT:= STRICH! ++1								
Ergebnis								
SIEBEN_MAL_ACHT 56								

Jeder kann neue Spaltennamen in o++o frei wählen. Es dürfen jedoch keine Kleinbuchstaben verwendet werden. Anstelle von SIEBEN_MAL_ACHT könnten Sie auch das kürzere PRODUKT_7_8 wählen. Der Spaltenname SIEBEN_MAL_ACHT im gib-Teil kommt in der mit den ersten beiden Zeilen erzeugten Tabelle nicht vor. Sie ist als neue Aggregations-Spalte gedacht. Die letzte Zeile drückt aus, dass diese einzelne Ausgabespalte durch die Zählaggregation (++1) berechnet werden soll. Aufgrund der Einrückung (4 Leerzeichen) gehört diese Zeile logisch gesehen noch zur gib-Anweisung.

Dieses Programm kann nur formuliert werden, weil o++o mit strukturierten Tabellen arbeitet.

Jeder sollte selbst beurteilen, welche Multiplikation kindlich-naiver und damit leichter zu verstehen ist. An dieser Stelle sollte erwähnt werden, dass die o++o-Multiplikation viel allgemeiner ist als in anderen Programmiersprachen. Damit ist aber nicht das obige Programm gemeint, sondern die Operation, die sich hinter dem Symbol * versteckt. Sie können zum Beispiel ein ganzes Tabment mit einer Zahl multiplizieren:

ZUTAT,	MENGE,	EINHEIT	1
Rinderfilet	750	g	
Wacholderbeeren	5		
Nelken	3		
Sellerie	50	g	
Möhren	50	g	
Fett	50	g	
Rauchspeck	50	g	
Essigwasser	300	ml	

Tabment: sauerbraten.tab

Programm 7.3.6: Rechne ein Rezept für 4 Personen auf 6 Personen um. um!								
sauerbraten.tab :4*6								
Ergebnis (tab)								
ZUTAT,	MENGE,	EINHEIT	1					
Rinderfilet	1125.	g						
Wacholderbeeren	7.5							
Nelken	4.5							
Sellerie	75.	g						
Möhren	75.	g						

Fett	75.	g
Rauchspeck	75.	g
Essigwasser	450.	ml

Ein Mathematiklehrer des Magdeburger Sportgymnasium bemerkte, dass die Schüler keine Probleme haben, schriftlich in anderen Positionssystemen als dem Dezimalsystem zu multiplizieren. Lassen Sie uns kurz über Binärzahlen sprechen. Aus meiner Sicht ist es sehr interessant, dass es wieder eines mathematischen Genies wie Leibniz bedurfte, um Dualzahlen salonfähig zu machen, obwohl das Rechnen mit Dualzahlen viel einfacher ist als das Rechnen mit Dezimalzahlen. Damit wäre natürlich auch die binäre Multiplikation viel einfacher zu lehren als die dezimale Multiplikation. Man muss nur eine kleine Multiplikationstabelle auswendig lernen:

- $0 \cdot 0 = 0$
- $1 \cdot 0 = 0$
- $0 \cdot 1 = 0$
- $1 \cdot 1 = 1$

Dieser Algorithmus hat weitere Vorteile: Er ist sehr effizient - sowohl in Bezug auf den Speicher als auch auf die Geschwindigkeit. Deshalb wird er in Computern verwendet. Nicht jeder muss sich dessen bewusst sein. Selbst wenn der Benutzer nur die dezimale Multiplikation kennt, kann es keine Probleme geben. Seine Dezimalzahlen können ohne sein Wissen in Binärzahlen umgewandelt werden, binäre Berechnungen werden durchgeführt, und wenn das Ergebnis ausgegeben wird, wird es wieder in Dezimalzahlen umgewandelt. Analog dazu kann man sich vorstellen, dass eine Dezimalzahl für ein Kind (Benutzer) in eine Strichzahl umgewandelt wird, die Strichmultiplikation, die es verstanden haben sollte, angewandt wird und die Strichzahl wieder in eine Dezimalzahl umgewandelt wird. Damit soll ausgedrückt werden, dass der Benutzer den einprägsamsten und nützlichsten Multiplikationsalgorithmus erlernen soll und darauf vertrauen können muss, dass die Computerexperten seinen Algorithmus korrekt "umsetzen".

Damit soll nur gesagt werden, dass die Ergebnisse korrekt sein müssen. Intern kann ein anderer, effizienterer Algorithmus verwendet werden. Natürlich gilt diese Idee der internen Optimierung nicht nur für Multiplikationsalgorithmen. Der Abschnitt schließt mit einem Multiplikationsalgorithmus, der an Pivot-Tabellen erinnert, aber dem heute in den Schulen gelehrt Algorithmus für die Multiplikation mit Dezimalzahlen sehr nahekommt.

Programm 7.3.7: Multiplikation einer Liste der Länge 2 mit einem Tripel mittels Matrix-Multiplikation in o++o			
10 3 *mat (100,20,4) pivot ++			
Zwischenergebnis nach der ersten Zeile im tab-Format			
ZAHL, ZAHL, ZAHL 1			
1000	200	40	
300	60	12	
Endergebnis im tab-Format			
ZAHL ,ZAHL ,ZAHL ,SUM 1			
1000	200	40	1240
300	60	12	372
1300	260	52	1612

Das Endergebnis der Multiplikation von 13 mal 124 ist 1612.

Ob diese Multiplikation leichter zu vermitteln ist als die heutige Multiplikation mit Dezimalzahlen, müssen Mathematiklehrer herausfinden, indem sie ganze Klassen mit beiden Methoden multiplizieren lassen.

Programm 7.3.8: Vollständiger (Matrix-) Multiplikationsalgorithmus in o++o

```

X:=4321
Y:=678
XX1:=X cut 1 zahl
YY1:=Y cut 1 zahl
XL:=XX1 ++1
YL:=YY1 ++1
BX1:= 10 *1 XL hoch (XL- 1 ... 0! -1) * XX1
BY1:= 10 *1 YL hoch (YL- 1 ... 0! -1) * YY1
MATRIX:= BX1 *mat (BY1 transpose)
gib MATRIX
cross ++
last
gib SUM
'3

```

Ergebnis (tab)

SU1

2'929'638

Die Operation cut überführt einen Text in die Liste seiner Zeichen. Diese und andere Operationen könnten auch für den Deutsch- oder Englischunterricht vorteilhaft eingesetzt werden. Jeder Buchstabe wird durch die Umrechnungsfunktion zahl in eine Ziffer umgewandelt. Die Vermittlung der vereinfachten Matrix-Multiplikation in der einen oder anderen Form würde die Schüler auch auf die Verwendung von Matrizen vorbereiten, die in der heutigen Praxis eine große Rolle spielen.

Mit dem letzten Programm dieses Abschnitts soll die Pivot-Multiplikation mit Hilfe einer nutzerdefinierten Operation realisiert werden, dass der Benutzer nur die ersten beiden Zahlen in den ersten beiden Zeilen des eigentlichen Programms ändern muss, um eine Multiplikation durchzuführen.

Programm 7.3.9: Detailliertere Matrix-Multiplikation der Faktoren 7653 und 322 in o++o

```

defop $X myop.powerlist = begin
X1l:=$X cut 1 zahl
gib X1 l-
XPOT:= (10 ^ (X1 pos - 1))
X2:= XPOT * X1
gib X2l-
end
aus 7'653 myop.powerlist
*mat 322 myop.powerlist transpose
cross ++
gib SUM
++
'3

```

Endergebnis (tab)

2'464'266

7.4 Strichlistenmultiplikation versus Dezimalmultiplikation

Die schriftliche Dezimalmultiplikation hatte eine große Bedeutung, weil viele Menschen damit sogar zwei 8-stellige Zahlen mit hoher Wahrscheinlichkeit richtig berechnen konnten. Die Korrektheit konnte durch die Neunerprobe noch verbessert werden. Außerdem war es in der früheren Praxis der mit Abstand schnellste Algorithmus.

Der spätere Rechenschieber war schneller, aber nicht so genau. Die Zahlentabelle mit Logarithmen war für manche zu anspruchsvoll. Im Zeitalter der Computer sind beide Techniken bereits eingemottet worden.

Der schriftliche Dezimalalgorithmus wird immer noch von vielen Menschen verwendet ..., und jeder, der ihn beherrscht, ist stolz auf seine Fähigkeiten. Die Frage ist: Können wir den dezimalen Multiplikationsalgorithmus in der Schule durch die Strichmultiplikation ersetzen oder/und ihn um andere Multiplikationsalgorithmen ergänzen?

Die Strichmultiplikation hat nicht nur den Vorteil, dass sie bereits in einer früheren Klasse gelehrt werden kann. Wenn dieser Algorithmus entsprechend wiederholt wird, merkt jeder, dass gerade diese Multiplikation die Standardanwendung - eine Rechteckflächenberechnung - realisiert. Diese Anwendung aus der dezimalen Multiplikation abzuleiten, erscheint mir zu schwierig. Angesichts der Programmierbarkeit beider Algorithmen sollte schnell klar werden, dass die Strichmultiplikation der Dezimalmultiplikation weit überlegen ist. Die obige Strichlistenmultiplikation erfordert 4 Schritte, die nacheinander abgearbeitet werden müssen. Insbesondere gibt es keine Schleife und keine Rekursion. Andererseits zeigt der obige Algorithmus, dass eine Programmiersprache in der Lage sein muss, mit strukturierten Tabellen zu arbeiten, um benutzerfreundliche Multiplikationsprogramme zu ermöglichen. Da diese Strichmultiplikation in gewissem Sinne Massendaten verarbeitet, bereitet sie besser auf die Digitalisierung der Gesellschaft vor als der heute in den Schulen gelehrt Algorithmus. Wenn gewünscht ist, dass jeder die Multiplikation programmieren können soll, müssen einfachere Multiplikationsalgorithmen gelehrt werden.

7.5 Wie könnte o++o den Lehrplan der Schule bereichern?

Wir haben ein Datenmodell mit einer dazugehörigen Programmiersprache o++o entwickelt, das nicht nur die Grundlage für Informationssysteme für die Wirtschaft sein soll, sondern auch viele Vorteile für den Schulunterricht bietet. Da die Sprache o++o auf mathematischen Konzepten basiert, sollte sie in den Mathematikunterricht integriert werden. Aber auch in den anderen Fächern kann o++o sinnvoll eingesetzt werden, denn die Extraktion und Visualisierung von Informationen aus der deutschen Wikipedia scheint für jedes Schulfach wichtig zu sein. Eine Programmiersprache soll die Schülerinnen und Schüler in die Lage versetzen, die vielfältigen Aufgaben, die ihnen in ihrer Zukunft gestellt werden, besser zu lösen. Das gilt besonders für die vor uns liegende Digitalisierung. Digital bedeutet eigentlich, dass alles auf zwei Dinge hinausläuft - Null und Eins. Ich kann mir nicht vorstellen, dass jemand die mächtige Strichlistenoperation, die zum Beispiel hinter der gib-Anweisung steht, auf ein solches Denken zurückführen kann. Unsere Axiome der Strichlistenoperation wurden auf einem hohen abstrakten algebraischen Niveau formuliert, wo das Denken in Nullen und Einsen nur hinderlich ist. Für die Mathematik ist die Abstraktion wichtiger als die Digitalisierung im eigentlichen Sinne des Wortes. Obwohl sich o++o bisher nur mit inhaltlichen Fragen beschäftigt hat, lassen sich mit CSS in o++o auch viele Formatfragen realisieren. Anhand des folgenden Beispiels wird deutlich, dass es auch sinnvoll ist, Formfragen direkt in o++o einzufangen:

Programm 7.5.1: Das Produkt aus 5 Zahlen im Tausenderformat in o++o
28911 5233 199 6311 6781 ** '3
Ergebnis
1'288'424'128'758'129'267

o++o könnte vielleicht in der Unterstufe gelehrt werden:

Programm 7.5.2: Die Summe der ersten hundert Zahlen (Gauß-Problem)
1 .. 100 ++
Ergebnis
5050

Ich denke, dass diese und viele andere schwierige Probleme auch in den unteren Klassenstufen unterrichtet werden könnten. Um an dieser Stelle nicht missverstanden zu werden. Wir sollten bei der Einführung der Digitalisierung in der Schule bereits auf Erfahrungen zurückgreifen, die vor Jahrzehnten mit dem Taschenrechner gemacht wurden. Auch der zu frühe und zu breite Einsatz von Taschenrechnern hat wohl dazu geführt, dass viele Schüler die Grundrechenarten schlechter beherrschen..., schlechter im Kopf rechnen als in früheren Klassenstufen. Bei falschen Eingaben in den Taschenrechner scheinen viele auch nicht in der Lage zu sein, die zu erwartenden Größenordnungen der Ergebnisse abzuschätzen. Aus diesem Grund ist der Taschenrechner hier erst ab der siebten Klasse erlaubt.

Ich bin zum Beispiel sogar der Meinung, dass Schreibprogramme wie WORD bis etwa zur siebten Klasse auch in der Schule nicht gelehrt werden sollten. Wenn ein Schüler am eigenen Leib erfahren hat, dass WORD fast alle seine Rechtschreibfehler korrigiert, ist es sehr schwer, ihm klar zu machen, ... dass seine eigenen Rechtschreibkenntnisse für seine Zukunft von Bedeutung sind. In ähnlicher Weise würde ich die Einführung von digitalen Whiteboards kritisch prüfen lassen.

Im vergangenen Dezember fiel mir an der Universität Halle auf, dass die Mathematikprofessoren immer noch mit Tafeln und gewöhnlicher Kreide arbeiteten.

Programm 7.5.3: Führe ein o++o-Programm in der zweiten Klasse an der Tafel aus!

```
4 3 1 12 ++
```

Dieses Rechnen an der Tafel mit Kreide oder Bleistift könnte auch auf die zukünftige Digitalisierung vorbereiten. Außerdem könnte die Lehrkraft aus Motivationsgründen schon den Unterstufenschülern demonstrieren, dass man mit dem Symbol ++ die Gaußsche Aufgabe oder noch größere Probleme lösen kann. Meiner Meinung nach wissen viele Menschen nicht aktiv, wie man eine Bedingung formuliert, obwohl das nicht schwer ist. Sie ist jedoch nicht Teil des Lehrplans. Die Bedingungen, die alle in Magdeburg lebenden Personen `ORT=Magdeburg` auswählen oder alle Flüsse herausfiltern, die länger als 1000 km `LÄNGE>1000` sind, sehen nicht kompliziert aus. Viele können das nicht, weil die heutigen Suchmaschinen das nicht verlangen bzw. nicht verarbeiten können. Aber wenn ich in einem zukünftigen Unternehmen spontan wichtige Informationen aus einer Unternehmensdatenbank extrahieren muss, dann muss ich das wissen.

Meiner Meinung nach kann die Problemlösungskompetenz der Schüler auf vielerlei Hinsicht verbessert werden. Sogar die Anwendungen der Differential- und Integralrechnung könnten in Sekundarschulen gelehrt werden, ohne die schwierigen Theorien von Leibniz und Newton verstehen zu müssen. Mit o++o können wir Flächen unter Kurven in einer kurzen Codezeile berechnen, ohne schwer lesbare Schleifen zu verwenden. Eine Näherung der Fläche unter dem ersten Sinusbogen kann mit dem 2000 Jahre alten Algorithmus von Archimedes in einer Zeile berechnet werden:

Programm 7.5.4: Dieses o++o-Programm benötigt keine Integralrechnung!

```
1 ... 2!0.0001 sin *0.0001 ++
```

Ergebnis

```
0.956536680039
```

Im Folgenden wird eine Anwendung der Differentialrechnung vorgestellt, die ohne Kenntnisse der Differentialrechnung durchgeführt werden kann.

Programm 7.5.5: Ein o++o-Programm zur Approximation des lokalen Minimums der Parabel (ein spezielles Polynom) " $3x^2 + 4x + 6$ "!

```
-10 ... 10!0.0001 poly [3 4 6] min
```

Ergebnis

```
4.66666667
```

Ich glaube, dass dies bereits in jeder 9. oder 10. Klasse unterrichtet werden kann, ohne hier ins Detail zu gehen. Wenn ich mit Schülern spreche, habe ich manchmal den Eindruck, dass es im Informatikunterricht mehr um Formfragen (HTML,...) als um Inhalte geht. Wir wissen, dass es sehr schwer ist, aber wir sollten trotzdem das von unserem Dr. A. Merkel vorgegebene Ziel erreichen: **Jeder soll lesen und rechnen lernen, aber auch programmieren.** Wenn man sich Programmiersprachen wie C, Java oder Python anschaut, ist das Ziel nicht realisierbar. Dafür braucht man einfachere Sprachen, die in der Lage sind, mit kurzen Programmen Endanwenderprobleme zu lösen. C und Co. hatten andere Ziele. Sie sollten dazu dienen, Systeme zu programmieren, an denen hunderte oder mehr Menschen arbeiten können, die viele Millionen Zeilen Code enthalten können und trotzdem performant arbeiten. o++o folgt dem neuen Paradigma der tabellenorientierten Programmierung und hat vor allem das von A. Merkel formulierte Ziel. Hätte o++o nicht von Anfang an methodische und pragmatische Fragen in den Vordergrund gestellt, wäre dieses Ziel auch mit o+o nicht realisierbar. Die Beherrschung von Operationen für Massendaten scheint für eine langfristige Digitalisierungsstrategie notwendig zu sein.

7.6 Kann die Strichlistenoperation bereits in der dritten Klasse unterrichtet werden?

Wie bereits erwähnt, ist die gib-Anweisung, die die Strichlistenoperation beinhaltet, ein mächtiges Werkzeug. Sie kann nicht nur zum Sortieren normaler flacher Tabellen, sondern auch beliebiger Tabellen verwendet werden. Gleichzeitig können Sie auch Aggregationen wie ++ (Summe), ++1 (Anzahl) usw. verwenden. Wenn die Schülerinnen und Schüler der dritten Klasse Schwierigkeiten mit einer formalen Syntax haben, bedeutet dies nicht unbedingt, dass der dahinterstehende Algorithmus nicht unterrichtet werden kann. Sie könnten zum Beispiel Tiere zählen. Dabei muss es sich nicht nur um eine Zahl handeln. Eine Tabelle, die die Anzahl für jede Tierart bestimmt, wäre sicher auch leicht zu vermitteln:

Programm 7.6.1: Tierarten zählen mit Strichen			
TIER1:=Esel Sau Eber Eber Sau Esel gib TIER,ANZ m ANZ:=TIER! ++			
Ergebnis (tabh)			
TIER, ANZ m			
Eber			
Esel			
Sau			

Das folgende Beispiel ist etwas anspruchsvoller, da die Ergebnistabelle strukturiert ist.

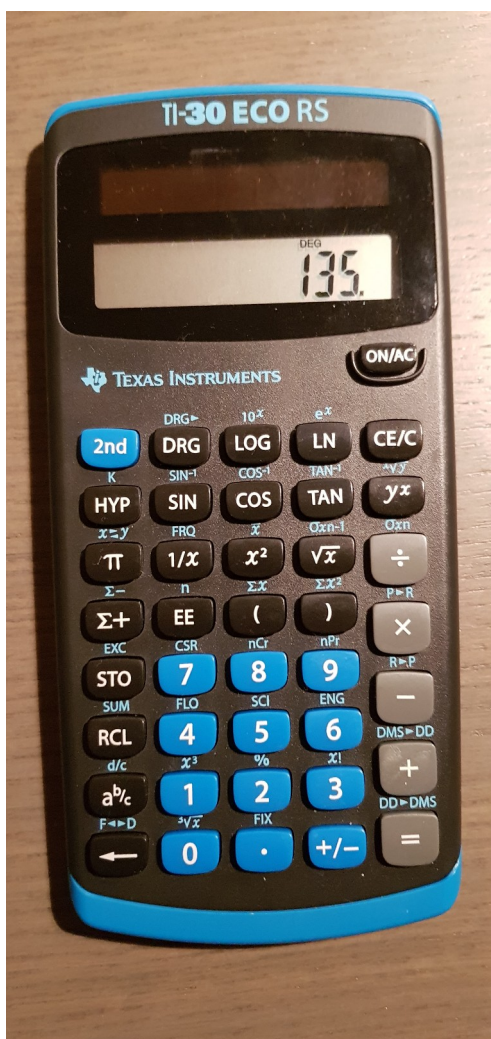
Programm 7.6.2: Zählen in strukturierten Tabellen			
<TAB!			
MARKE	,FARBE,	TYP,	GEWICHT 1
VW	Blau	Polo	1250
IFA	Papyrus	500	580
VW	Blau	Golf	1450
Audi	Gelb	Quattro	2070
VW	Blau	Polo	1380
IFA	Beige	601	620
VW	Rot	Golf	1400
Audi	Rot	Quattro	2100
IFA	Beige	601	620
VW	Beige	Polo	1300

!TAB> gib MARKE,ANZAHL, (FARBE,ANZAHL m) m ANZAHL:=TYP! ++			
Ergebnis:			
MARKE, ANZAHL,	(FARBE,	ANZAHL2 m) m	
Audi	Gelb		
	Rot		
IFA	Beige		
	Papyrus		
VW	Beige		
	Blau		
	Rot		

Man kann sich vielleicht vorstellen, wie Kinder an der Tafel zählen und sortieren und dabei diesen Algorithmus verwenden. In o++o wird eine Menge (m) oder ein Multimenge (bag) immer nach den ersten Spaltennamen sortiert. Im obigen Beispiel sind dies `MARKE` und `FARBE`.

Das heißt, Kinder können vermutlich Daten in strukturierten Tabellen sortieren. Die heutigen Informatikstudenten lernen jedoch keinen Sortieralgorithmus für strukturierte Tabellen. Ein Beitrag von mir in der deutschen Wikipedia, der insbesondere dieses Sortieren beinhaltete, wurde gelöscht, da er „nicht zum Grundwissen eines Informatikers gehört“.

7.7 Rechnet der Schultaschenrechner von Texas-Instruments falsch?



Der links abgebildete Taschenrechner TI-30 ECO RS, der von deutschen Bildungsministerien als Schultaschenrechner zugelassen wurde, gibt bei der Aufgabe

2 hoch 2 hoch 3

64 aus. Richtig entsprechend der Regeln der heutigen mathematischen Konventionen, die man auch in der Wikipedia unter Operatorrangfolge (rechts-assoziativ) nachlesen kann, wäre 256.

Für hoch muss man dort allerdings das Symbol y^x tippen.

Nun kann man natürlich sagen, jede Firma kann rechnen, wie sie will. Das machen sie auch. Beim Windows-taschenrechner (modus normal) ergibt $1 + 2 \times 3$ auch eine falsche Lösung im Sinne der Schulmathematik. Sachsen-Anhalt hat vielleicht nicht genug Geld, um den amerikanischen Tech- Giganten Microsoft zu verklagen. Aber wie soll man verhindern, dass viele Schüler durch diese "Vielfalt" die Orientierung verlieren?

Wie das obenstehende Bild suggeriert, macht der Taschenrechner einen sehr guten Eindruck. Er verhält sich jedoch auch in vielen anderen Punkten anders, als in

der Schule gelehrt wird und ist zudem schwer zu bedienen und dadurch auch bei einfachen Aufgaben fehleranfällig.

In der Mathematik schreibt man den "Sinus von 3.14" in der Regel wie folgt:

$\sin(3,14)$

Im Mathematiklehrbuch "Schlüssel zur Mathematik" (Sekundarstufe Sachsen-Anhalt Klasse 10 Cornelsen, ISBN 978-3-06-0044558-7) heißt es bedauerlicher sogar:

"Die Funktion $f(x) = \sin x$ heißt Sinusfunktion."

Der Texas Instruments Rechner akzeptiert das Komma nicht als Dezimalzahltrenner und man muss zunächst 3.14 und dann erst die sin Taste drücken. Wenigstens ist Texas Instruments konsequent im Tippen an dieser Stelle. Die Wurzel aus 4 ermittelt man auch, indem man zuerst die 4 tippt und anschließend das Wurzelzeichen. Wie jeder erwartet ergibt sich 2. . Denkbar wäre aber auch 2 ohne Dezimalpunkt? Bei $2+2$ ermittelt Texas Instruments auch 4. und nicht 4, obwohl jeder weiß, dass das Ergebnis dieser Addition eine ganze Zahl ist. Um an dieser Stelle Missverständnissen vorzubeugen: Wir kritisieren nicht, dass dieser Texas-Instruments-Taschenrechner die nutzerfreundlichere Tippvariante für einstelligen Operationen wählt, sondern dass sich Lehrplan und Schulpraxis hier wesentlich unterscheiden.

Man könnte den Begriff "Kommazahl" auch gleich durch "Punktzahl" ersetzen.

Verwunderlich sind auch Bezeichnungen auf der Tastatur.

$\Sigma+$ (EE, RCL, STO, ...).

Durch Vorgängerrechner sind viele bereits mit dem Symbol $M+$ - zum Speicher (Memory) addieren - vertraut. Soll hier Innovation vorgetäuscht werden?



Interessant ist in diesem Zusammenhang auch, dass bereits in den 70er Jahren Taschenrechner existierten, deren Funktionsumfang für die Nutzung im Mathematikunterricht und für eine Vielzahl von Anwendungen, insbesondere im naturwissenschaftlichen und technischen Bereich, vollkommen ausreichend war.

Diese Rechner zeichneten sich durch ein übersichtliches Tastatur-Layout aus, welches auf Mehrfachbelegungen der Tasten verzichtete. Die Rechnerarchitektur realisierte konsequent das Rechnen von links nach rechts und es konnte auf Klammerebenen verzichtet werden. Der Funktionsumfang beschränkte sich auf die notwendigen und häufig benutzten Funktionen. Somit wurden Probleme, die sich aus unterschiedlichen Designs ergeben, minimiert und eine einfache und intuitive Bedienung gewährleistet.

Ein Beispiel dafür ist der in der Abbildung dargestellte, 1975 in Japan entwickelte und produzierte wissenschaftliche Taschenrechner.

Wie löst man nun die obigen Probleme?

Aus o++o Sicht verhält sich der TI-30 ECO RS bei diesen Problemen jedoch zum großen Teil richtig. Er wählt beispielsweise bei $2^3 \cdot 4$ die Weise des Rechnens, die die Mehrheit der Menschen bevorzugt, nämlich von-links-nach-rechts zu rechnen. Das trifft auch auf Ingenieure zu, wie ich vielfach erfahren konnte. Dass einstellige Funktionen nach der Zahl (dem Argument) getippt werden, begrüßen wir auch, weil diese Rechenart ebenfalls dem Prinzip von-links-nach-rechts folgt:

3.14 sin cos

Der Rechner von Texas-Instruments berechnet hierbei zuerst den Sinus und auf das Ergebnis wendet er die Kosinusfunktion an. Das wird zwar nicht im Mathematikunterricht gelehrt, ist aber ebenfalls leichter verständlich. Bedauerlicherweise ist der Taschenrechner an der Stelle nicht vollständig konsequent. Bei $1 + 2 \times 3$ rechnet er nicht mehr von-links-nach-rechts. Jetzt rechnet er so, wie es angeblich Descartes wollte. Bloß damit man ein Polynom etwas eleganter schreiben konnte, habe die Menschen das allgemeine Prinzip von-links-nach-rechts zu rechnen aufgegeben. Da man heute auch eine Liste von Zahlen als Inputwert betrachten kann, hat diese Argumentation aus dem 17-ten Jahrhundert aus unserer Sicht keine Existenzberechtigung mehr. Anstelle von

$$X^3 + 2 \cdot X^2 + 3 \cdot X + 4$$

können wir heute kurz und bündig $X^{\text{poly } 1 \ 2 \ 3 \ 4}$ tippen.

Allgemein schätzen wir zudem ein, dass alle heutigen Taschenrechner moralisch verschlissen sind. Sie sollten gar nicht mehr in der Schule eingesetzt werden. Der erste elektronische, tatsächlich handflächengroße Taschenrechner wurde bereits 1967 entwickelt und hatte - wie heute noch üblich ein **sehr kleines Display**. Da heute im Jahr 2023 Handys mit wesentlich größeren Displays existieren

und wir auch wesentlich leistungsfähigere Apps mit weit größerem Anwendungsgebiet kennen, sollte man heute Taschenrechner generell aus der Schule verbannen oder im Schulmuseum ausstellen.

Betrachten wir einmal ein sehr einfaches Problem. Sie wollen mit dem Texas Instruments Rechner 10 Zahlen addieren. Wenn sie am Ende der Rechnung überschlagen, dass das Ergebnis nicht stimmen kann, können sie die Eingabe nicht noch einmal ansehen. Sie müssen alle Zahlen noch einmal eintippen. Ob sie das dann richtig machen, wenn alle diese Zahlen aus 10 Ziffern bestehen, ist unklar.

Betrachten wir weiter die Prozenttaste %. Spielt man mit dem Rechner und tippt beispielsweise

10 %

, so erhält man als Ergebnis 0.1.

Also könnte man vermuten, dass die Prozenttaste nur falsch beschriftet ist, und sie einfach nur durch 100 dividiert. Die Prozenttaste ist zudem bei diesem Taschenrechner schwer zu tippen, da man zuvor 2nd tippen muss. Wenn man das kleine blaue Prozentzeichen, das keine eigene Taste besitzt, gefunden hat, muss man sich zudem sehr stark konzentrieren, damit man erkennt, ob man die Taste darüber oder darunter drücken soll. Das sind natürlich potentielle Fehlerquellen. Man muss natürlich auch wissen, ob die Taste 2nd nur für die nächste Operation gilt oder solange, bis ich sie wieder drücke. Tippt man dann beispielsweise

10 + 10 %

, so erhält man zunächst 1. Erst wenn man weiter = drückt, erhält der heutige Nutzer die wahrscheinlich von ihm gewünschte Zahl 11.

Wenn man aber mathematisch denkt, kommt eigentlich nur eine der folgenden zwei Lösungen infrage:

10 + (10 %)

oder

(10 + 10) %

Man erhält im ersten Fall 10.1 und im zweiten 0.2.

Das heißt, mit diesem Symbol wird mathematischem Denken widersprochen. Wie soll man

10 + 10 %

anders als Term auffassen? Wozu müssen alle Schüler eine Termdefinition lernen, wenn sie in der Praxis des Taschenrechners in der Schule nicht angewandt wird?

Unseres Wissens gibt es nur eine Programmiersprache, die dieses Symbol überhaupt im Zusammenhang mit Prozentrechnung benutzt. Hier wird allerdings +% als ein zweistelliges Operationssymbol eingesetzt. Damit wird es auch mathematisch klar und sauber.

Wie die drei Buchstaben von sin ein Operationssymbol darstellen, ist auch +% eine Operation.

In o++o ergibt 10 +% 10 11. .

Tippt man beim Texas-Instruments-Rechner

10 sin x²

, so sieht man nie auf dem Display, welches Operationssymbol man gerade oder zuvor getippt hat. Ferner erschweren die Tastaturbezeichnungen die Regel "Punkt vor Strich" zu verstehen, wenn das Multiplikationszeichen aus 2 Strichen besteht und das Divisionszeichen einen Strich enthält. Wir beschließen den Abschnitt mit einer scheinbar sehr einfachen Vielfachaddition. Wir denken, dass kaum jemand eine Addition von sehr vielen Zahlen mit dem Taschenrechner korrekt bewerkstelligen kann.

Sophia hat einen Tiger aus Legobausteinen gebaut. Am Ende der Beschreibung des Baukastens werden alle verwendeten Bausteinarten mit Anzahl der verwendeten Steine aufgeführt. Opa möchte wissen, aus wie vielen Legosteinen besteht der Tiger und wieviele Bausteine gibt es von jeder Art?





Programm 7.7.1: Gesamtsumme uns Zwischennummen vieler Zahllisten.

Wahrscheinlich geht dieses horizontale Eingeben der Listen schneller als die Eingabe einer TAB-Struktur. Letztere wäre jedoch leichter zu erfassen

```

8 4 2 1 15 8 4 1 10 6 6 4
,4 1 1 2 14 4 2 2 4 1 2
,4 2 1 4 4 2 1 2 1 1 1
,6 2 1 2 1 2 4 2 2 1 4 1
,4 2 1 2 1 17 11 3 1 2 2
,2 3 4 4 4 11 2 1 2 1 2 4
,7 4 2 4 14 6 5 4 6 4 12 4
,2 2 2 1 6 1 1 2 2 1 1 2
,4 4 2 6 2 2 6 1 4 4 5 5 22
,4 14 4 8 5 8 4 1 6 12 6 2 4
,1 4 2 4 10 6 2 8 1 1 12 1
,1 2 1 1 8 6 6 6 2 1 1 4 4
,2 2 6 2 6 4 17 7 26 2 4 2
,8 12 2 6 6 4 16 8 6
,2 4 3 2 1 2 4
,2 2 1 1 1 2 1
=: X11,X21,X31,X41,X51,X61,X71,X81,X91,X101,X111,X121,X131,X141,X151,X161
cross ++

```

Ergebnis:

X11	X21	X31	X41	X51	X61	X71	X81	X91	X101	X111	X121	X131	X141	X151	X161	SUM?
8	4	4	6	4	2	7	2	4	4	1	1	2	8	2	2	755
4	1	2	2	2	3	4	2	4	14	4	2	2	12	4	2	
2	1	1	1	1	4	2	2	2	4	2	1	6	2	3	1	
1	2	4	2	2	4	4	1	6	8	4	1	2	6	2	1	
15	14	4	1	1	4	14	6	2	5	10	8	6	6	1	1	
8	4	2	2	17	11	6	1	2	8	6	6	4	4	2	2	
4	2	1	4	11	2	5	1	6	4	2	6	17	16	4	1	
1	2	2	2	3	1	4	2	1	1	8	6	7	8	18	10	
10	4	1	2	1	2	6	2	4	6	1	2	26	6			
6	1	1	1	2	1	4	1	4	12	1	1	2	68			
6	2	1	4	2	2	12	1	5	6	12	1	4				

4	37	1	1	46	4	4	2	5	2	1	4	2
69		24	28		40	72	23	22	4	52	4	80
								67	78		43	

Obiges Programm besteht aus einem Tupel von 16 Listen von Zahlen. Das ist sicher vorteilhafter als eine einzige Liste oder ein einziges Tupel zu tippen. Im letzten Fall müsste man anstelle der vielen leicht zu tippenden Leerzeichen Kommas tippen, was sicher kein Vorteil wäre.

7.8 Ist EXCEL moralisch verschlissen?

Programm 7.8.1: Ein o++o-Programm, für das EXCEL mehr als sechs Arbeitsblätter benötigt!

```
<TAB!
NAME,      LAENGE, (ALTER, GEWICHT m)m
Klaus      1.68      18      61
              30      65
              61      80
Rolf        1.78      40      72
Kathi       1.70      18      55
              40      70
Walleri     1.00      3       16
Viktoria    1.61      13      51
Bert        1.72      18      66
              30      70

!TAB>
sel NAME! ALTER>20
gib BMI,(ALTER,BMI,(NAME,BMI m) m) BMI:=GEWICHT:LAENGE:LAENGE!++:
rnd 2
```

Wenn man dieses o++o-Programm in EXCEL realisiert benötigt man mehr als 6 Arbeitsblätter. Kaum jemand übersieht diese EXCEL-Blätter, weshalb sie auch nur sehr schwer zu ändern sind. Mehr Details finden Sie unter [o++o versus EXCEL](#). Tabellenkalkulationsprogramme haben mehrere Vorteile und sind weit verbreitet, haben aber auch eine Reihe von Nachteilen, die wir aufzählen wollen:

1. Daten und Formeln werden vermischt. Aus diesem Grund und weil ein EXCEL-Arbeitsblatt Hunderte oder gar Tausende von Formeln enthalten kann, ist es kaum möglich, die Korrektheit der Programme zu überprüfen oder sie an Änderungen anzupassen.
2. EXCEL kennt keine Schemata für strukturierte Tabellen: z.B. beschreibt FACH,NOTE1 I ein strukturiertes Schema - hier wird eine Liste von Fächern beschrieben, wobei es für jedes Fach auch eine Liste von Noten gibt.
3. EXCEL kann strukturierte Tabellen zwar visuell darstellen, sie aber nicht direkt sortieren oder vernünftig verarbeiten.
4. Sie können EXCEL nicht verwenden, um Datenbanken, XML oder Wikipedia abzufragen. Dazu müssten Sie noch SQL, XQuery oder besser o++o lernen.
5. EXCEL-Formeln sind relativ kryptisch, weil sie z. B. oft einzelne Zellbezeichnungen enthalten. Die Summe über eine Spalte wird in EXCEL z.B. in der Form geschrieben:
=SUMME(F12:F75)
6. Eine einzige EXCEL-Formel kann mehr Analyse erfordern als ein komplettes o++o-Programm.
7. EXCEL enthält nur wenige mathematische Konzepte und erfordert daher einen übermäßigen Anteil an Detailwissen.
8. EXCEL bietet deutschen und anderen Benutzern das Dezimalkomma an. Dies erschwert jedoch den Austausch entsprechender Arbeitsblätter internationaler Unternehmen über Länder-grenzen hinweg, da viele Länder den Dezimalpunkt bevorzugen.

9. Da Daten und Programme in der Regel in o++o getrennt sind, können die Daten problemlos von mehreren Programmen verwendet werden. Das ist mit EXCEL schwieriger.
10. Für Aggregationen (Summen, Durchschnitte, Maxima, ...) pro Wert müssen Sie in EXCEL in der Regel vorsortieren oder gruppieren aber nicht in o++o.
11. In EXCEL müssen Sie jede Zahl in eine eigene Zelle schreiben. Das könnte einen Smartphone-Bildschirm schnell überfordern.
12. o++o basiert auf einem abstrakten Tabment-Konzept für Daten. Ein Tabment kann bereits standardmäßig auf viele Arten dargestellt werden: web tab xml bild Säule ... und auch kompakt (hsq). Mit CSS können die Ausgaben von o++o praktisch fast beliebig formatiert werden. EXCEL basiert dagegen auf einem konkreten Druckbild. Das erleichtert zunächst die Erstellung einfacher Anwendungen, ist aber für die Komplexität der heutigen Anwendungen eher ein Nachteil.
13. Nach dem Studium der oben genannten Kritikpunkte an EXCEL bemerkte ein VW-Ingenieur: Bei VW kann EXCEL von jedem Mitarbeiter nach Belieben genutzt werden. In der Regel sind es aber nur einfache Tabellen, die "schön" gemacht werden sollen. Manchmal sind ein paar einfache Rechenoperationen notwendig. Umfassende, komplexe Anwendungen finden in EXCEL nicht statt.

EXCEL kennt keine Massendatenoperationen und könnte allein deswegen moralisch verschlissen sein. Deshalb plädiere ich dafür, EXCEL-Programme auch aus dem Schulunterricht zu entfernen und durch leistungsfähigere und erfolgversprechendere Konzepte und Systeme zu ersetzen.

7.9 o++o Beweise

Beweise haben in der Schule und auch außerhalb der Welt der Berufsmathematiker bisher eine untergeordnete Rolle gespielt. Doch jeder möchte Vertrauen in eine Berechnung, ein System oder einen Taschenrechner haben. Wenn man mit einem neuen Rechnertyp oder System konfrontiert wird, versucht jeder zunächst, Aufgaben wie $2 \text{ mal } 3$ zu lösen. Wer weitere Probleme vermutet, testet z.B. $1 \text{ plus } 2 \text{ mal } 3$.

Der hochangesehene deutsche Wirtschaftswissenschaftler Professor Sinn sagt in seinem Vortrag **Energiewende ins Nichts** (siehe youtube), dass Berechnungen nur dann wirklich Sinn machen, wenn man sie verstehen kann. Dazu muss man alle Teilschritte im Einzelnen verstehen.

An dieser Forderung von Prof. Sinn arbeiten wir schon seit Jahrzehnten. Die SQL-Designer hatten diese Anforderung zu Beginn ihrer Entwicklung etwas anders formuliert:

SQL sollte eine Endbenutzersprache werden.

Daraus folgt unmittelbar, dass der OttoNormalVerbraucher in der Lage sein sollte, SQL-Programme zu verstehen. Heute kommen jedoch fast alle SQL-Programmierer aus der Informatik-Ecke.

Die Bedeutung der Statistik in den Schulen nimmt zu.

Wie kann man einem Schüler eine neue statistische Funktion beibringen, wie z. B. den Durchschnitt $+$ oder die Funktion `streu (mad)` von o++o. Wenn man die Funktion einfach auf mehrere Listen von Zahlen anwendet und das Ergebnis betrachtet, kann man ihre Bedeutung normalerweise nicht verstehen. Wenn der Lehrer jedoch weiß, dass die Schüler bereits die Funktionen $++$ (Summe) und $++1$ (Anzahl) verstanden haben, ist dies nicht mehr so schwierig.

Programm 7.9.1: Vorbereitung eines o++o Beweises für die $++$ Operation.

```
X1:= 3 5 4 2 1
SUM:=X1 ++
ANZAHL:=X1 ++1
MEINDURCHSCHNITT:=SUM:ANZAHL
OTTODURCHSCHNITT:=X1++:
```

Ergebnis (ment)

```
<TABM>
<SUM>15</SUM>
```

```

<ANZAHL>5</ANZAHL>
<MEINDURCHSCHNITT>3.</MEINDURCHSCHNITT>
<OTTODURCHSCHNITT>3.</OTTODURCHSCHNITT>
<X>3</X>
<X>5</X>
<X>4</X>
<X>2</X>
<X>1</X>
</TABM>

```

Trotz dieser (Doku)ment-Ausgabe ist klar, dass die beiden Durchschnittswerte übereinstimmen. Hier stimmt die ment-Ausgabe fast vollständig mit der xml-Ausgabe überein. Die Spalten MEINDURCHSCHNITT und OTTODURCHSCHNITT müssen aber bei allen anderen Eingabelisten übereinstimmen. Das Programm hat den Vorteil, dass es sehr einfach ist. Aber der Schüler muss trotzdem viele Daten eingeben. Am Beispiel von o++o-mad, das in Deutschland noch keine Rolle gespielt hat, wollen wir zeigen, dass ein erweitertes o++o-Programm uns einen Großteil der Tipparbeit abnehmen kann. Diese mad-Funktion ist eines der einfachsten und klarsten statistischen Funktionen, aber sie hat nicht so schöne mathematische Eigenschaften. Nun setzen wir die Kenntnis der Operationen ++:, ..x und abs voraus. Durch von ..x bis!anz wird eine Liste von anz Zufallszahlen zwischen von und bis erzeugt. abs berechnet den Absolutwert.

Programm 7.9.2: o++o Beweis für die ++: Operation.

```

ZUFALLSZAHL1:= 1 ..x 10!10
X1:= 1 ..x ZUFALLSZAHL!ZUFALLSZAHL
DUR:=X1 ++:
ABSTAND:=DUR - X abs
MYMAD:=ABSTAND1 ++:
OTTOMAD:=X1 mad
gib DUR,MYMAD,OTTOMAD 1

```

Ergebnis (tab)

DUR,	MYMAD ,	OTTOMAD 1
1.5	0.5	0.5
6.66666666667	2.	2.
2.75	0.875	0.875
2.33333333333	0.88888888889	0.88888888889
3.8	2.96	2.96
1.5	0.5	0.5
5.	1.75	1.75
6.66666666667	1.40740740741	1.40740740741
5.875	1.375	1.375
3.57142857143	0.65306122449	0.65306122449

Wir können die Ergebnistabelle leicht auf tausend Ausgabezeilen erweitern, indem wir die letzte Zahl 10 der ersten Zeile durch 1000 ersetzen. Wir haben nur die relevanten Spalten mit der Anweisung gib extrahiert.

7.10 Ein Beispiel für eine tiefe Digitalisierung

Vielleicht macht das folgende Beispiel das Konzept der Tiefendigitalisierung (TD) ein wenig deutlicher: Wenn Addition, Multiplikation, in der Schule **nicht** gelehrt **würden**, bräuchte man heute z.B. verschiedene Apps zum Lösen der beiden folgenden Probleme.

Eine Analogie zur Tiefen Digitalisierung aus dem Bereich der "Einzeldaten"-Operationen

1. Ich habe eine Ladung von 36.57 Tonnen Schüttgut erhalten und werde 31 weitere Ladungen dieser Art erhalten. Wie viel Schüttgut werde ich insgesamt haben?
2. Ich habe ein rechteckiges Grundstück von 32 m Breite und 36.57 m Länge. Wie groß ist mein Grundstück?

Jeder, der die Multiplikation verstanden hat, weiß, dass es sich um ein und dasselbe Problem handelt, das mit einem einfachen Taschenrechner sehr leicht gelöst werden kann. Für die heutige Digitalisierung bedeutet dies, dass eine TD weit weniger Computeranwendungen erfordern könnte als eine FD (flache Digitalisierung) und dass die Endnutzer (Manager, Politiker, ...) weit mehr herkömmliche Anwendungen (Apps) beherrschen könnten. Denn wenn alle Apps und Anwendungen auf einem (z.B. o++o) Datenmodell basieren, kann man natürlich auch die Schnittstellen dieser Apps standardisieren und eine solche Anwendung könnte viele herkömmliche FD-Applikationen ersetzen.

8 Schemata und strukturierte Tabellen

Alle Spaltennamen einer Tabelle werden oft als Schema der Tabelle betrachtet. Spaltennamen sind erforderlich, um entsprechende Spaltenwerte richtig zu verstehen. Betrachten wir strukturierte Tabellen, ist es vorteilhaft, die Spaltennamen um entsprechende Kollektionssymbole anzureichern; zum Beispiel I für Liste.

NAME,	GEBORENIN,	(TAT,	JAHR I) 1
Otto der Grosse	Altsachsen(De)	Zum König von Deutschland gewählt	936
		Die Ungarn auf dem Lechfeld geschlagen	955
		Erster Kaiser des Heil. Röm. Reichs	962
Otto von Mähren	Mähren	heiratete Euphemia von Ungarn	1086
Otto von Guericke	MD (De)	Erfinder der Luftpumpe	1649
		Halbkugelversuch vor dem Kaiser	1654
Otto von Bismarck	Preussen(De)	mit Zuckerbrot und Peitsche-Politik	1871
		Emser Depesche	1870
		Erster Reichskanzler von Deutschland	1871
Nicolaus Otto	Taunus (De)	Miterfinder des Ottomotors	1876
Otto Normalverbraucher De		erlernt Autofahren	1960
		erlernt eine Programmiersprache	2025

Tabment 8.1: ottos.tab

Die obige Tabelle (TABMENT=TABelle+dokuMENT) ottos.tab enthält eine Liste von 6 „Personen“ und für jede Person eine Wiederholgruppe (TAT, JAHR I) – eine Liste von (TAT, JAHR)-Paaren. Hier hat eine Person 4 Spalten, aber es ist ein Tripel (3-Tupel). Es ist eine strukturiertes Tupel, kurz Strupel (Bezeichnung von Prof. Schek). Die ersten beiden Komponenten sind vom Typ TEXT und die dritte Komponente ist eine Liste von Subtupeln (Paaren) (2-Tupeln). Wir werden die Attributwerte einer Ebene Segment nennen. Das NAME-Segment ist dasselbe wie das GEBORENIN-Segment.

Das erste NAME-Segment ist:

NAME,	GEBORENIN
Otto der Grosse	Altsachsen(De)

Das erste TAT-Segment von Otto dem Großen lautet:

TAT,	JAHR
Zum König von Deutschland gewählt	936

Die erste Person entspricht dem ersten Strupel; es ist ein NAME-Tupel (=GEBORENIN-Tupel):

NAME,	GEBORENIN,	(TAT,	JAHR I)
Otto der Grosse	Altsachsen(De)	Zum König von Deutschland gewählt	936
		Die Ungarn auf dem Lechfeld geschlagen	955
		Erster Kaiser des Heil. Röm. Reichs	962

Da die TAT-Tupel (= TAT-Sub-Tupel) keine weiteren Kollektionen enthalten, ist ein TAT-Segment dasselbe wie ein TAT-Tupel. Wenn wir die obige Tabelle durch eine gewöhnliche flache Tabelle darstellen würden, müsste jedes (NAME, GEBORENIN)-Paar in jeder Zeile erscheinen. Das heißt, (Otto der Große, Altsachsen(De)) müsste 3 Mal erscheinen (einmal für jedes TAT-Segment). Dann ist es zum Beispiel nicht so einfach, die Personen in der Tabelle zu zählen. Mit obiger Tabelle sieht das entsprechende Programm wie folgt aus:

Programm 8.1: Wie viele Ottos sind in der Tabelle enthalten?
Wie viele Elemente (Strupel) enthält die äußerste Kollektion ?

```
ottos.tab  
++1
```

Hier und im Folgenden verwenden wir diese Abkürzungen und Schlüsselwörter:

aus:

++1: zählen

gib: (entspricht dem SELECT von SQL)

sel: mit (Französisch) (für die Selektion)

sel-: ohne (für die Selektion)

:= : extension (erweitert die angegebene Tabelle um eine neue (komplexe) Spalte)

m: Menge: enthält verschiedene Elemente

b: Bag: ein Element darf mehrmals vorkommen

l: Liste: die Reihenfolge der Elemente ist von Bedeutung

Das Ergebnis von Programm 8.1 ist eine einfache Tabelle:

ZAHL
6

Das Schema dieser Tabelle enthält kein Kollektionssymbol, da die Tabelle genau ein Element enthält. Ebenso benötigen wir in den folgenden 2 Anfragen kein Kollektionssymbol. Wir wollen die folgenden Anfragen nicht im Detail erläutern. Wir verwenden die Anfragen, um zu verdeutlichen, welche verschiedenen Arten von Tabellen es gibt und welche Schemata dazu gehören.

Programm 8.2: Wie viele Personen und wie viele Taten sind in der Datei ottos.tab enthalten?

```
ottos.tab  
gib ANZPERSON, ANZTAT  
    ANZPERSON:= NAME! ++1  
    ANZTAT := TAT ! ++1
```

Ergebnis

ANZPERSON, ANZTAT
6 12

Programm 8.3: Nennen Sie mir den Namen des in Sachsen Geborenen.

```
aus ottos.tab  
sel Sachsen in GEBORENIN  
gib NAME
```

Ergebnis

NAME
Otto der Grosse

Programm 8.4: Nennen Sie mir den Namen einer adligen Person.

```
aus ottos.tab  
sel "von" in NAME  
gib NAME
```

Ergebnis

NAME

Otto von Mähren

Wenn in der künftigen o++o-Software Schlüsselworte wie „sel“ und „in“ farblich hervorgehoben werden, ist auch die zweite Programmzeile besser lesbar.

Aber hier wäre es besser, die Namen aller Adligen auszugeben:

Programm 8.5: Sortieren Sie alle adligen Namen.

```
aus ottos.tab
sel von in NAME
gib NAMEm
```

Ergebnis (tab)

NAMEm

Otto von Bismarck
 Otto von Guericke
 Otto von Mähren

Um Platz auf dem Bildschirm oder Papier zu sparen, können wir die Elemente einer Liste oder einer anderen Kollektion auch horizontal anordnen:

Ergebnis (tabh)

NAMEm

"Otto von Bismarck" "Otto von Guericke" "Otto von Mähren"

Programm 8.6: Zähle alle Taten und alle Taten jedes Jahrhunderts. Füge jedem Jahrhundert die entsprechenden Personen hinzu.

```
aus ottos.tab
JAHRHUNDERT:=JAHR div 100 +1
gib ANZTAT,(JAHRHUNDERT,ANZTAT,NAMEm m)
  ANZTAT:= TAT ! ++1
```

Ergebnis: Tabelle mit 3 Segmenttypen: ANZTAT, (JAHRHUNDERT, ANZTAT2) und NAME (WIDTH 65)

ANZTAT, (JAHRHUNDERT, ANZTAT2, NAMEm m)

12	10	3	Otto der Grosse
	11	1	Otto von Mähren
	17	2	Otto von Guericke
	19	4	Nicolaus Otto
			Otto von Bismarck
	20	1	Otto Normalverbraucher
	21	1	Otto Normalverbraucher

Programm 8.7: Zähle alle Taten und die Taten jedes Jahrhunderts mit entsprechenden Personen, wobei für jede Tat die entsprechende Person erscheinen muss (mit Duplikaten).

```
aus ottos.tab
JAHRHUNDERT:=JAHR div 100 +1
gib ANZTAT,(JAHRHUNDERT,ANZTAT,NAMEb m)
  ANZTAT:= TAT ! ++1
```

Ergebnis (tab)

ANZTAT, (JAHRHUNDERT, ANZTAT2, NAMEb m)

12	10	3	Otto der Grosse
			Otto der Grosse
			Otto der Grosse

11	1	Otto von Mähren
17	2	Otto von Guericke
		Otto von Guericke
19	4	Nicolaus Otto
		Otto von Bismarck
		Otto von Bismarck
		Otto von Bismarck
20	1	Otto Normalverbraucher
21	1	Otto Normalverbraucher

Dabei steht b für bag (multiset). Bisher haben wir nur Tabellen mit verschachtelten Ebenen betrachtet. Aber eine strukturierte Tabelle darf auch „unabhängige“ Kollektionen enthalten:

NAME	,RESIDENZl,	FRAU1,	LAND1	1
Otto der Grosse	Magdeburg Memleben	Editha Adelheid	Sachsen	
			Thüringen	
			Bayern	
			Franken	
			Schwaben	
			Italien	
			Böhmen	
			Holland	
			Lothringen	
			Friesland	
Karl IV.	Prag Tangermünde	Margarethe	Böhmen	
		Anna	Schlesien	
		Anna	Brandenburg	
		Elisabeth	Italien	
			Ungarn	

Tabment 8.2: kaiser.tab

In dieser Tabelle stehen „Memleben“ und „Otto der Große“ in gleicher Beziehung zueinander wie „Adelheid“ und „Otto der Große“. Das bedeutet aber nicht, dass „Adelheid“ und „Memleben“ miteinander in Beziehung stehen, obwohl sie in derselben Zeile stehen. Daher ist die folgende Umstrukturierung sinnlos.

Programm 8.8: Anfrage mit leerem Ergebnis

aus kaiser.tab

gib NAME,RESIDENZ,FRAU m

gib NAME,RESIDENZm,FRAUm m ist jedoch sinnvoll.

9 Tabment-Typen (TTs) und strukturierte Dokumente

Für strukturierte Tabellen und Dokumente verwenden wir die Bezeichnung Tabment. Daher kürzen wir den Typ eines Tabments mit TT (Tabment Typ) ab. Das TT vervollständigt die durch ein Schema gegebenen Informationen. Es gibt für jedes Tag sein Schema an. Die TT für die obige Tabelle ottos.tab ist zum Beispiel:

```
TABMENT! OTTOS
OTTOS! NAME,GEBORENIN,(TAT,JAHR 1)m
NAME GEBORENIN TAT! TEXT
JAHR! ZAHL
```

TEXT und ZAHL sind elementare Typen, die nicht weiter erläutert werden müssen. Jedes benannte Tabment ist von einem Tag umgeben, das sich aus dem Dateinamen durch Weglassen des Typ-Suffix ergibt. Daher kann unsere erste Tabelle auch im Dokumentstil oder in einem Dokumentstil mit inneren Tabellen (ment oder xml) dargestellt werden.

zum Beispiel:

```
<OTTOS>
  <NAME>Otto der Grosse</NAME>
  <GEBORENIN>Altsachsen(De)</GEBORENIN>
  <TAT>Zum König von Deutschland gewählt</TAT>
  <JAHR>936</JAHR>
  <TAT>Die Ungarn auf dem Lechfeld geschlagen</TAT>
  <JAHR>955</JAHR>
  <TAT>Erster Kaiser des Heil. Röm. Reichs</TAT>
  <JAHR>962</JAHR>
  <NAME>Otto von Mähren</NAME>
  <GEBORENIN>Mähren</GEBORENIN>
  <TAT>heiratete Euphemia von Ungarn</TAT>
  <JAHR>1086</JAHR>
  <NAME>Otto von Guericke</NAME>
  <GEBORENIN>MD (De)</GEBORENIN>
  <TAT>Erfinder der Luftpumpe</TAT>
  <JAHR>1649</JAHR>
  <TAT>Halbkugelversuch vor dem Kaiser</TAT>
  <JAHR>1654</JAHR>
  <NAME>Otto von Bismarck</NAME>
  <GEBORENIN>Preussen(De)</GEBORENIN>
  <TAT>mit Zuckerbrot und Peitsche-Politik</TAT>
  <JAHR>1871</JAHR>
  <TAT>Emser Depesche</TAT>
  <JAHR>1870</JAHR>
  <TAT>Erster Reichskanzler von Deutschland</TAT>
  <JAHR>1871</JAHR>
  <NAME>Nicolaus Otto</NAME>
  <GEBORENIN>Taunus (De)</GEBORENIN>
  <TAT>Miterfinder des Ottomotors</TAT>
  <JAHR>1876</JAHR>
  <NAME>Otto Normalverbraucher</NAME>
  <GEBORENIN>De</GEBORENIN>
  <TAT>erlernt Autofahren</TAT>
  <JAHR>1960</JAHR>
  <TAT>erlernt eine Programmiersprache</TAT>
  <JAHR>2025</JAHR>
</OTTOS>
```

Tabment 9.1: Tabelle ottos.tab im XML-Dokumentstil

```

"Otto der Grosse" Altsachsen(De)
  "Zum König von Deutschland gewählt" 936
  "Die Ungarn auf dem Lechfeld geschlagen" 955
  "Erster Kaiser des Heil. Röm. Reichs" 962
"Otto von Mähren" Mähren
  "heiratete Euphemia von Ungarn" 1086
"Otto von Guericke" "MD (De)"
  "Erfinder der Luftpumpe" 1649
  "Halbkugelversuch vor dem Kaiser" 1654
"Otto von Bismarck" Preussen(De)
  "mit Zuckerbrot und Peitsche-Politik" 1871
  "Emser Depesche" 1870
  "Erster Reichskanzler von Deutschland" 1871
"Nicolaus Otto" "Taunus (De)"
  "Miterfinder des Ottomotors" 1876
"Otto Normalverbraucher" De
  "erlernt Autofahren" 1960
  "erlernt eine Programmiersprache" 2025

```

Tabment 9.2: Tabelle ottos.tab im hsq-Stil

Betrachten wir Teile eines kleinen, aber echten Dokuments: „Grundgesetz für die Bundesrepublik Deutschland“.

```

<META!
TABMENT! GRUNDGESETZ
GRUNDGESETZ! NR,TITEL,INHALT 1
NR ! ONR
TITEL INHALT! TEXT
!META>
<GRUNDGESETZ>
  <NR>1</NR>
    <TITEL>Die Grundrechte</TITEL><INHALT></INHALT>
  <NR>1.1</NR>
    <TITEL>Menschenwürde - Menschenrechte - Bindung der Grundrechte</TITEL>
    <INHALT></INHALT>
  <NR>1.1.1</NR>
    <TITEL></TITEL>
    <INHALT>Die Würde des Menschen ist unantastbar. Sie zu achten und zu schützen ist
Verpflichtung aller staatlichen Gewalt.</INHALT>
  <NR>1.1.2</NR>
    <TITEL></TITEL>
    <INHALT>Das Deutsche Volk bekennt sich darum zu unverletzlichen und unveräußerlichen
Menschenrechten als Grundlage jeder menschlichen Gemeinschaft, des Friedens und der Gerechtigkeit in
der Welt.</INHALT>
  <NR>1.1.3</NR>
    <TITEL></TITEL>
    <INHALT>Die nachfolgenden Grundrechte binden Gesetzgebung, vollziehende Gewalt und
Rechtsprechung als unmittelbar geltendes Recht.</INHALT>
  <NR>1.2</NR>
    <TITEL>Persönliche Freiheit</TITEL>
    <INHALT></INHALT>
  <NR>1.2.1</NR>
    <TITEL></TITEL>
    <INHALT>Jeder hat das Recht auf die freie Entfaltung seiner Persönlichkeit, soweit er
nicht die Rechte anderer verletzt und nicht gegen die verfassungsmäßige Ordnung oder das
Sittengesetz verstößt.</INHALT>
  <NR>1.2.2</NR>
    <TITEL></TITEL>
    <INHALT>Jeder hat das Recht auf Leben und ...</INHALT>
  <NR>1.14</NR>
    <TITEL>Eigentum Erbrecht Enteignung</TITEL>
    <INHALT></INHALT>
  <NR>1.14.1</NR>
    <TITEL></TITEL>
    <INHALT>Das Eigentum und das Erbrecht werden gewährleistet. Inhalt und Schranken werden
durch die Gesetze bestimmt.</INHALT>
  <NR>1.14.2</NR>

```

```

<TITEL></TITEL>
<INHALT>Eigentum verpflichtet. Sein Gebrauch soll zugleich dem Wohle der Allgemeinheit
dienen.</INHALT>
<NR>1.14.3</NR>
<TITEL></TITEL>
<INHALT>Eine Enteignung ist nur zum Wohle der Allgemeinheit zulässig. Sie darf nur durch
Gesetz oder auf Grund eines Gesetzes erfolgen, das Art und Ausmaß der Entschädigung regelt.
Die Entschädigung ist unter gerechter Abwägung der Interessen der Allgemeinheit und der
Beteiligten zu bestimmen. Wegen der Höhe der Entschädigung steht im Streitfalle der
Rechtsweg vor den ordentlichen Gerichten offen.</INHALT>
<NR>1.19</NR>
<TITEL>Einschränkung von Grundrechten</TITEL>
<INHALT></INHALT>
<NR>1.19.1</NR>
<TITEL></TITEL>
<INHALT>Soweit nach diesem Grundgesetz ein Grundrecht durch Gesetz ...</INHALT>
<NR>2.</NR>
<TITEL>Der Bund und die Länder</TITEL>
<INHALT></INHALT>
<NR>2.1</NR>
<TITEL>Grundlagen staatlicher Ordnung, Widerstandsrecht</TITEL><INHALT></INHALT>
<NR>2.1.1</NR><TITEL></TITEL>
<INHALT>Die Bundesrepublik Deutschland ist ein demokratischer und sozialer
Bundesstaat.</INHALT>
<NR>2.1.2</NR><TITEL></TITEL>
<INHALT>Alle Staatsgewalt geht vom Volke aus. Sie wird vom Volke in Wahlen und
Abstimmungen ...</INHALT>
</GRUNDGESETZ>

```

Tabment 9.3: grundgesetz1.ment

```

<META!
TABMENT! GRUNDGESETZ
GRUNDGESETZ! KAPITEL1
KAPITEL! KNR,KTITEL,ARTIKEL1
ARTIKEL! ANR,ATITEL,(PNR,PARAGRAPH 1)
KNR ANR PNR! ZAHL
KTITEL ATITEL PARAGRAPH! TEXT
!META>
<GRUNDGESETZ>
<KAPITEL>
<KNR>1</KNR>
<KTITEL>Die Grundrechte</KTITEL>
<ARTIKEL>
<ANR>1</ANR>
<ATITEL>Menschenwürde - Menschenrechte - Bindung der Grundrechte</ATITEL>
<PNR>1</PNR>
<PARAGRAPH>Die Würde des Menschen ist unantastbar. Sie zu achten und zu schützen ist
Verpflichtung aller staatlichen Gewalt.</PARAGRAPH>
<PNR>2</PNR>
<PARAGRAPH>Das Deutsche Volk bekennt sich darum zu unverletzlichen und unveräußerlichen
Menschenrechten als Grundlage jeder menschlichen Gemeinschaft, des Friedens und der Gerechtigkeit in
der Welt.</PARAGRAPH>
<PNR>3</PNR>
<PARAGRAPH>Die nachfolgenden Grundrechte binden Gesetzgebung, vollziehende Gewalt und
Rechtsprechung als unmittelbar geltendes Recht.</PARAGRAPH>
</ARTIKEL>
<ARTIKEL>
<ANR>2</ANR>
<ATITEL>Persönliche Freiheit</ATITEL>
<PNR>1</PNR>
<PARAGRAPH>Jeder hat das Recht auf die freie Entfaltung seiner Persönlichkeit, soweit er
nicht die Rechte anderer verletzt und nicht gegen die verfassungsmäßige Ordnung oder das
Sittengesetz verstößt.</PARAGRAPH>
<PNR>2</PNR>
<PARAGRAPH>Jeder hat das Recht auf Leben und ...</PARAGRAPH>
</ARTIKEL>
<ARTIKEL>
<ANR>14</ANR>
<ATITEL>Eigentum Erbrecht Enteignung</ATITEL>
<PNR>1</PNR>
<PARAGRAPH>Das Eigentum und das Erbrecht werden gewährleistet. Inhalt und Schranken werden
durch die Gesetze bestimmt.</PARAGRAPH>
<PNR>2</PNR>
<PARAGRAPH>Eigentum verpflichtet. Sein Gebrauch soll zugleich dem Wohle der Allgemeinheit
dienen.</PARAGRAPH>

```

```

<PNR>3</PNR>
<PARAGRAPH>Eine Enteignung ist nur zum Wohle der Allgemeinheit zulässig. Sie darf nur durch
Gesetz oder auf Grund eines Gesetzes erfolgen, das Art und Ausmaß der Entschädigung regelt.
Die Entschädigung ist unter gerechter Abwägung der Interessen der Allgemeinheit und der
Beteiligten zu bestimmen. Wegen der Höhe der Entschädigung steht im Streitfalle der
Rechtsweg vor den ordentlichen_Gerichten_offen.</PARAGRAPH>
</ARTIKEL>
<ARTIKEL>
<ANR>19</ANR>
<ATITEL>Einschränkung von Grundrechten</ATITEL>
<PNR>1</PNR>
<PARAGRAPH>Soweit_nach_diesem GrunGesetz ein Grundrecht_durch_Gesetz ...</PARAGRAPH>
</ARTIKEL>
</KAPITEL>
<KAPITEL>
<KNR>2</KNR>
<KTITEL>Der Bund und die Länder</KTITEL>
<ARTIKEL>
<ANR>20</ANR>
<ATITEL>Grundlagen staatlicher Ordnung, Widerstandsrecht</ATITEL>
<PNR>1</PNR>
<PARAGRAPH>Die Bundesrepublik Deutschland ist ein demokratischer und
sozialer_Bundesstaat.</PARAGRAPH>
<PNR>2</PNR>
<PARAGRAPH>Alle Staatsgewalt geht vom Volke aus. Sie wird vom Volke in Wahlen und Abstimmungen
...</PARAGRAPH>
</ARTIKEL>
</KAPITEL>
</GRUNDGESETZ>

```

Tabment 9.4 grundgesetz2.ment

grundgesetz1.ment und grundgesetz2.ment enthalten die gleichen Informationen, unterscheiden sich dennoch wesentlich in ihrer Art der Strukturierung. Die Metadaten der zweiten Version lehnen sich an die Bezeichnungen des Originalgrundgesetzes an. Hier liegt durch die Begriffe KAPITEL, ARTIKEL und PARAGRAPH eine Strukturierungstiefe von 3 vor, die eigentlich nur durch neue Namen erhöht werden könnte. Die Metadaten vom ersten Dokument können dagegen für beliebige strukturierte Dokumente genutzt werden. Obwohl lediglich 3 „Spaltennamen“ (NR, TITEL, INHALT) vorhanden sind, ist die Tiefe der Strukturierung unbeschränkt.

Dokumente, die das "neue Zeile" Zeichen (Return) enthalten, sind für tab- und hsq-Ausgaben nur bedingt geeignet. Strukturierte Dokumente werden in der Regel als ment oder xml ausgegeben. Für (NR,TITEL,INHALT I)-Dokumente bietet sich sogar die tabellarische web-Ausgabe an. Dass die web-Ausgabe für die Gegenüberstellung beider obiger Dokumenttypen vorteilhaft ist, soll jetzt demonstriert werden.

Programm 9.1: Gib ein Dokument in Form zweier unterschiedlicher Tabellen aus.
--

grundgesetz1.ment, grundgesetz2.ment

Ergebnis (web)

GRUNDGESETZ1			GRUNDGESETZ2																		
NR	TITEL	INHALT	KNR	KTITEL	ARTIKEL1																
1	Die Grundrechte		1	Die Grundrechte	<table><tr><th>ANR</th><th>ATITEL</th><th colspan="2">(PNR, PARAGRAPH 1)</th></tr><tr><td rowspan="3">1</td><td rowspan="3">Menschenwürde - Menschenrechte - Bindung der Grundrechte</td><td>PNR</td><td>PARAGRAPH</td></tr><tr><td>1</td><td>Die Würde des Menschen ist unantastbar. Sie zu achten und zu schützen ist Verpflichtung aller staatlichen Gewalt.</td></tr><tr><td>2</td><td>Das Deutsche Volk bekennt sich darum zu unverletzlichen und unveräußerlichen Menschenrechten als Grundlage jeder menschlichen Gemeinschaft, des Friedens und der Gerechtigkeit in der Welt.</td></tr><tr><td></td><td></td><td>3</td><td>Die nachfolgenden Grundrechte binden Gesetzgebung, vollziehende Gewalt und Rechtsprechung als unmittelbar geltendes Recht.</td></tr></table>	ANR	ATITEL	(PNR, PARAGRAPH 1)		1	Menschenwürde - Menschenrechte - Bindung der Grundrechte	PNR	PARAGRAPH	1	Die Würde des Menschen ist unantastbar. Sie zu achten und zu schützen ist Verpflichtung aller staatlichen Gewalt.	2	Das Deutsche Volk bekennt sich darum zu unverletzlichen und unveräußerlichen Menschenrechten als Grundlage jeder menschlichen Gemeinschaft, des Friedens und der Gerechtigkeit in der Welt.			3	Die nachfolgenden Grundrechte binden Gesetzgebung, vollziehende Gewalt und Rechtsprechung als unmittelbar geltendes Recht.
ANR	ATITEL	(PNR, PARAGRAPH 1)																			
1	Menschenwürde - Menschenrechte - Bindung der Grundrechte	PNR			PARAGRAPH																
		1			Die Würde des Menschen ist unantastbar. Sie zu achten und zu schützen ist Verpflichtung aller staatlichen Gewalt.																
		2			Das Deutsche Volk bekennt sich darum zu unverletzlichen und unveräußerlichen Menschenrechten als Grundlage jeder menschlichen Gemeinschaft, des Friedens und der Gerechtigkeit in der Welt.																
		3			Die nachfolgenden Grundrechte binden Gesetzgebung, vollziehende Gewalt und Rechtsprechung als unmittelbar geltendes Recht.																
1.1	Menschenwürde - Menschenrechte - Bindung der Grundrechte																				
1.1.1		Die Würde des Menschen ist unantastbar. Sie zu achten und zu schützen ist Verpflichtung aller staatlichen Gewalt.																			
1.1.2		Das Deutsche Volk bekennt sich darum zu unverletzlichen und unveräußerlichen Menschenrechten als Grundlage jeder menschlichen Gemeinschaft, des Friedens und der Gerechtigkeit in der Welt.																			
1.1.3		Die nachfolgenden Grundrechte binden Gesetzgebung, vollziehende Gewalt und Rechtsprechung als unmittelbar geltendes Recht.																			
1.2	Persönliche Freiheit																				
1.2.1.		Jeder hat das Recht auf die freie Entfaltung seiner Persönlichkeit, soweit er nicht die Rechte anderer verletzt und nicht gegen die verfassungsmäßige Ordnung oder das Sittengesetz verstößt.																			
1.2.2		Jeder hat das Recht auf Leben und ...																			
1.14	Eigentum Erbrecht Enteignung																				
1.14.1		Das Eigentum und das Erbrecht werden gewährleistet. Inhalt und Schranken werden durch die Gesetze bestimmt.																			
1.14.2		Eigentum verpflichtet. Sein Gebrauch soll zugleich dem Wohle der Allgemeinheit dienen.																			
1.14.3		Eine Enteignung ist nur zum Wohle der Allgemeinheit zulässig. Sie darf nur durch Gesetz oder auf Grund eines Gesetzes erfolgen, das Art und Ausmaß der Entschädigung regelt. Die Entschädigung ist unter gerechter Abwägung der Interessen der Allgemeinheit und der Beteiligten zu bestimmen. Wegen der Höhe der Entschädigung steht im Streitfalle der Rechtsweg vor den ordentlichen Gerichten offen.																			
1.19	Einschränkung von Grundrechten																				
1.19.1		Soweit nach diesem Grundgesetz ein Grundrecht durch Gesetz ...																			
2.	Der Bund und die Länder		2	Der Bund und die Länder	<table><tr><th>ANR</th><th>ATITEL</th><th colspan="2">(PNR, PARAGRAPH 1)</th></tr><tr><td rowspan="3">20</td><td rowspan="3">Grundlagen staatlicher Ordnung, Widerstandsrecht</td><td>PNR</td><td>PARAGRAPH</td></tr><tr><td>1</td><td>Die Bundesrepublik Deutschland ist ein demokratischer und sozialer Bundesstaat.</td></tr><tr><td>2</td><td>Alle Staatsgewalt geht vom Volke aus. Sie wird vom Volke in Wahlen und Abstimmungen ...</td></tr></table>	ANR	ATITEL	(PNR, PARAGRAPH 1)		20	Grundlagen staatlicher Ordnung, Widerstandsrecht	PNR	PARAGRAPH	1	Die Bundesrepublik Deutschland ist ein demokratischer und sozialer Bundesstaat.	2	Alle Staatsgewalt geht vom Volke aus. Sie wird vom Volke in Wahlen und Abstimmungen ...				
ANR	ATITEL	(PNR, PARAGRAPH 1)																			
20	Grundlagen staatlicher Ordnung, Widerstandsrecht	PNR			PARAGRAPH																
		1			Die Bundesrepublik Deutschland ist ein demokratischer und sozialer Bundesstaat.																
		2			Alle Staatsgewalt geht vom Volke aus. Sie wird vom Volke in Wahlen und Abstimmungen ...																
2.1	Grundlagen staatlicher Ordnung, Widerstandsrecht																				
2.1.1		Die Bundesrepublik Deutschland ist ein demokratischer und sozialer Bundesstaat.																			
2.1.2		Alle Staatsgewalt geht vom Volke aus. Sie wird vom Volke in Wahlen und Abstimmungen ...																			

Programm 9.2: Berechne für jedes Kapitel die Anzahl der Paragraphen zweifach.			
aus grundgesetz1.ment,grundgesetz2.ment KNR1:=NR subtext 1!1 zahl at NR KNR2:=KNR gib (KNR1,PANZ1 m),(KNR2,PANZ2 m) PANZ1:=INHALT!++1 PANZ2:=PNR!++1			
Ergebnis (tab)			
KNR1,PANZ1 m,(KNR2,PANZ2 m)			
1	14	1	9
2	4	2	2

Programm 9.3: Berechne zweimal die Anzahl der Zeichen (Buchstaben) jedes Paragraphen.					
grundgesetz1.ment,grundgesetz2.ment gib NR,ANZ1 m,(KNR,(ANR,(PNR,ANZ2 m)m)m) ANZ1 :=INHALT cut 1 ++1! ++ ANZ2 :=PARAGRAPH cut 1 ++1! ++					
Ergebnis (tab)					
NR ,ANZ1 m,(KNR ,(ANR ,(PNR ,ANZ2 m) m) m)					
1	1	1	1	1	113
1.1	1			2	187
1.1.1	113			3	122
1.1.2	187		2	1	206
1.1.3	122			2	37
1.2	1		14	1	116
1.2.1.	206			2	97
1.2.2	37			3	431
1.14	1		19	1	61
1.14.1	116	2	20	1	79
1.14.2	97			2	87
1.14.3	431				
1.19	1				
1.19.1	61				
2.	1				
2.1	1				
2.1.1	79				
2.1.2	87				

Programm 9.4: Berechne die Anzahl der Zeichen und Worte jedes Artikels.		
grundgesetz1.ment gib NR,ANZ1,ANZ2 m ANZ1 :=INHALT cut 1 ++1! ++ ANZ2 :=INHALT wort1 ++1! ++		
Ergebnis (tab)		
ANR, ANZ1, ANZ2 m		
1	1	0
1.1	1	0
1.1.1	113	17
1.1.2	187	24

1.1.3	122	13
1.2	1	0
1.2.1.	206	27
1.2.2	37	7
1.14	1	0
1.14.1	116	15
1.14.2	97	11
1.14.3	431	57
1.19	1	0
1.19.1	61	6
2.	1	0
2.1	1	0
2.1.1	79	7
2.1.2	87	14

Das Beispiel macht klar, dass ein Lehrer die Aufsätze einer Klasse auf diese Weise leicht durchzählen lassen kann. Man erkennt weiter, dass sich Anfragen an Dokumente nicht sehr stark von Anfragen an Tabellen unterscheiden, da man sich die Dokumente auch als Tabellen vorstellen kann.

Die hier beschriebenen Funktionen könnten unter anderem für typische Aufgaben von Redaktionen oder Übersetzungsbüros nützlich sein.

Programm 9.5: Gib mir Artikel 2 des Grundgesetzes.

```
aus grundgesetz1.ment
sel NR subtext 1!3="1.2"
gib TITEL,INHALT 1
```

Ergebnis (ment):

```
TABMENT ! TABM
TABM    ! TITEL,INHALT 1
TITEL   ! TEXT
INHALT  ! TEXT
```

```
<TABM>
  <TITEL>Persönliche Freiheit</TITEL>
  <INHALT></INHALT>
  <TITEL></TITEL>
  <INHALT>Jeder hat das Recht auf die freie Entfaltung seiner Persönlichkeit, soweit er
    nicht die Rechte anderer verletzt und nicht gegen die verfassungsmäßige Ordnung oder das
    Sittengesetz verstößt.</INHALT>
  <TITEL></TITEL>
  <INHALT>Jeder hat das Recht auf Leben und ...</INHALT>
</TABM>
```

Wir betrachten ein weiteres Dokument mit TT. Es verwendet Alternativen durch (or). Es stammt aus den XQuery-Anwendungsfällen (C+07).

```
<META!
TABMENT! REPORT1
REPORT1! ABSCHNITT1
ABSCHNITT! TITEL,INHALT
INHALT! TEXT | NARKOSE | VORBEREITUNG | SCHNITT | AKTION | BEOBACHTUNG 1
VORBEREITUNG! TEXT|AKTION 1
SCHNITT! TEXT|GEOGRAPHIE|INSTRUMENT 1
AKTION! TEXT|INSTRUMENT 1
TITEL NARKOSE BEOBACHTUNG GEOGRAPHIE INSTRUMENT! TEXT
!META>
<REPORT1>
  <ABSCHNITT>
    <TITEL>Verfahren</TITEL>
```

```

<INHALT>
  Die Patientin wurde in den Operationssaal gebracht, wo sie in Rückenlage platziert wurde und
<NARKOSE> induziert unter Vollnarkose. </NARKOSE>
<VORBEREITUNG>
<AKTION>Es wurde ein Foley-Katheter gelegt, um die Blase </AKTION> zu dekomprimieren und der Bauch
wurde dann steril vorbereitet und drapiert.
</VORBEREITUNG>
<SCHNITT>
Ein gekrümmter Einschnitt wurde gemacht
<GEOGRAPHIE> in der Mittellinie sofort infraumbilical </GEOGRAPHIE>
  und das subkutane Gewebe wurde geteilt
  <INSTRUMENT> Elektrokauterisation verwenden. </INSTRUMENT>
</SCHNITT>
Die Faszie wurde identifiziert und
<AKTION> # 2 Ø Maxon Nähte wurden auf jeder Seite der Mittellinie angeordnet.
</AKTION>
<SCHNITT>
Die Faszie wurde geteilt mit
<INSTRUMENT> electrocautery </INSTRUMENT>
  und das Peritoneum eingegeben.
</SCHNITT>
<BEOBACHTUNG>Der Dünndarm wurde identifiziert.</BEOBACHTUNG>
  und
  <AKTION> das <INSTRUMENT>Hasson-Trokar</INSTRUMENT>
  wurde unter direkter Visualisierung gelegt.
  </AKTION>
  <AKTION>Das <INSTRUMENT>Trokar</INSTRUMENT>unter Verwendung der
    Nähte wurde an der Faszie befestigt.
</AKTION>
</INHALT>
</ABSCHNITT>
</REPORT1>

```

Tabment 9.5: report1.ment

In report1.xml ist der INHALT eine Liste von Elementen, wobei jedes Element entweder vom Typ TEXT, ANÄSTHESIE, VORBEREITUNG, SCHNITT, AKTION oder BEOBACHTUNG ist. Im obigen Dokument ist das erste Element einfacher TEXT, das zweite ist vom Typ ANESTESIA, das dritte vom Typ VORBEREITUNG, Da unser Bericht auf die obige Weise getaggt wurde, sind die folgenden Beispielanfragen möglich.

Zum Beispiel:

Programm 9.6: Welche Instrumente wurden im zweiten Schnitt benutzt?
aus report1.ment
gib SCHNITT1
sel SCHNITT pos = 2
gib INSTRUMENT1
Ergebnis (tab)
INSTRUMENT1
electrocautery

Programm 9.7: Welches sind die ersten beiden Instrumente, die eingesetzt werden?
aus report1.ment
gib INSTRUMENT1
sel INSTRUMENT pos < 3
Ergebnis (tab)
INSTRUMENT1
Elektrokauterisation verwenden.
electrocautery

10 Eine Universitätsdatenbank

Wir betrachten eine nicht-relationale Datenbank, die aus einer flachen und zwei strukturierten Tabellen besteht:

faks! FAK, DEKAN, BUDGET, STUDKAPAZI m
studenten! STID, NAME, ORT?, STIP, FAK, (KURS, NOTE m), (PROJ, STUNDEN m) m
kurse! KURS, LEHRENDER, (ISBN, TITEL m)m

Die unterstrichenen Spaltennamen sind Schlüssel. Die letzten beiden Tabellen können durch die folgenden 5 flachen Relationen dargestellt werden:

studenten1: STID, NAME, ORT?, STIP, FAK m

examen1: STID, KURS, NOTE m

projekte1: STID, PROJ, STUNDEN m

kurse1: KURS, LEHRENDER m

kurs_buecher1: KURS, ISBN, TITEL m

FAK,	DEKAN,	BUDGET,	STUDKAPAZI m
Kunst	Sitte	2'000	600
Infor	Reichel	10'000	500
Mathe	Dassow	1'000	200
Philo	Hegel	1'000	10
Sport	Streich	8'000	150

Tabment 10.1: faks.tab

STID,NAME,	ORT?,	STIP,FAK,	(KURS,	NOTE m),	(PROJ,	STUNDEN m)m
1111 Ernst	Oehna	500 Mathe	Algebra	1	Fritz	4
			Logik	2	Otto	2
			Geschichte	1		
2222 Sophia	Berlin	400 Infor	Algebra	3	Ghandi	5
			Datenbanken	1	Ming	4
			Otto	1	Otto	6
3333 Clara	Oehna	450 Infor	Datenbanken	1		
			OCaml	2		
4444 Ulrike		400 Kunst			Monet	10
5555 Käthe	Gerwisch	600 Kunst	Repin	1	Monet	20
			Apel	1		
6666 Claudia	Berlin	600 Sport	Psycho	2	Matthes	8
			Ski	1	Witt	12

Tabment 10.2: studenten.tab

KURS,	LEHRENDER,	(ISBN,	TITEL m)m
Algebra	Reichel	0138-3019	Structural Induction on Partial Alg
		3-8244-2099-6	Strukturierte Tabellen
Datenbanken	Saake	0-321-31256-2	Database Systems an Application
		0-7167-8069-0	Principles of Database Systems
Otto	Benecke	0-7167-8069-0	Principles of Database Systems
		3-8244-2099-6	Strukturierte Tabellen

Tabment 10.3: kurse.tab

STID,	NAME,	ORT?,	STIP,FAK	m
1111	Ernst	Oehna	500	Mathe
2222	Sophia	Berlin	400	Infor
3333	Clara	Oehna	450	Infor
4444	Ulrike		400	Kunst
5555	Käthe	Gerwisch	600	Kunst
6666	Claudia	Berlin	600	Sport

Tabment 10.4: studenten1.tab

STID,	KURS,	NOTE	m
1111	Algebra	1	
1111	Geschichte	1	
1111	Logik	2	
2222	Algebra	3	
2222	Datenbanken	1	
2222	Otto	1	
3333	Datenbanken	1	
3333	OCaml	2	
5555	Apel	1	
5555	Repin	1	
6666	Psycho	2	
6666	Ski	1	

Tabment 10.5: examen1.tab

STID,	PROJ,	STUNDEN	m
1111	Fritz	4	
1111	Otto	2	
2222	Ghandi	5	
2222	Ming	4	
2222	Otto	6	
4444	Monet	10	
5555	Monet	20	
6666	Matthes	8	
6666	Witt	12	

Tabment 10.6: projekte1.tab

Die obigen Tabellen und die folgenden Programme beziehen sich auf tab-Dateien, obwohl wir im Hinterkopf behalten, dass die angegebenen Tabellen Datenbanktabellen sein könnten.

10.1 Selektion (sel sel-)

Eine Bedingung spezifiziert Tupel oder Subtupel. Bei einer sel-Klausel bilden die spezifizierten Tupel das Ergebnis, bei einer sel--Klausel werden die angegebenen Tupel weggelassen.

Folglich werden das Schema und der TT des betrachteten Tabments nicht durch eine Selektion verändert. Spaltennamen oder Tags werden in einem o++o-Programm in Großbuchstaben geschrieben. Sie müssen mit einem Buchstaben oder dem Zeichen „_“ beginnen. Ein WORT, das nicht von "-Symbolen eingeschlossen ist, muss daher einen Kleinbuchstaben verwenden. TEXTe können Leerzeichen enthalten; sie müssen dann aber in "-Symbolen eingeschlossen werden.

Programm 10.1.1: Finden Sie alle Studenten aus Berlin und Oehna mit schlechten Ergebnissen.
aus studenten.tab
sel ORT in "Berlin Oehna" # selektiert Studenten
sel NOTE > 2 # selektiert Prüfungen und Studenten
Ergebnis (tab)

STID,NAME ,ORT? ,STIP,FAK ,(KURS ,NOTE m),(PROJ ,STUNDEN m) m							
2222 Sophia Berlin 400 Infor Algebra 3 Ghandi 5							
Ming 4							
Otto 6							
Zwischenergebnis nach der ersten Bedingung							
1111 Ernst Oehna 500 Mathe Algebra 1 Fritz 4							
Geschichte 1 Otto 2							
Logik 2							
2222 Sophia Berlin 400 Infor Algebra 3 Ghandi 5							
Datenbanken 1 Ming 4							
Otto 1 Otto 6							
3333 Clara Oehna 450 Infor Datenbanken 1							
OCaml 2							
6666 Claudia Berlin 600 Sport Psycho 2 Matthes 8							
Ski 1 Witt 12							

Auf das Ergebnis der ersten Bedingung wird die zweite „Bedingung“ angewendet. Die zweite „Bedingung“ ist eine Abkürzung für die folgenden zwei Bedingungen:

```

sel STID! NOTE>2 # Auswahl STID-Tupel (NOTE>2 muss existieren)
sel KURS! NOTE>2 # Auswahl KURS-Tupel

```

Die erste dieser beiden Bedingungen drückt aus, dass wir (vollständige) Studententupel auswählen, für die ein (KURS,NOTE)-Subtupel mit einer Note 3 oder höher existiert. Wir schreiben den Existenzquantor nicht, weil sich hinter jeder Bedingung genau ein EXIST-Quantor verbirgt. „#“ ist das Kommentarsymbol. Es kann verwendet werden, um die Bedeutung eines Programmschritts zu beschreiben. Außerdem können Zeilen auskommentiert werden, um Zwischenergebnisse anzuzeigen.

Programm 10.1.2: Geben Sie für alle Studenten aus Oehna und Berlin alle Ergebnisse mit 3 oder schlechter an.

```

aus studenten.tab
sel ORT in "Berlin Oehna" # äquivalent: ORT in Berlin Oehna
sel KURS! NOTE>2 # wählt Examen und nicht Studenten
gib NAME,ORT,(KURS,NOTE m) b

```

Ergebnis (tab)

NAME ,ORT ,(KURS , NOTE m) b

Clara Oehna
Claudia Berlin
Ernst Oehna
Sophia Berlin Algebra 3

Nach Anwendung der beiden Bedingungen wurde die Restrukturierung (siehe Abschnitt 10.3) angewendet. Daher hat sich das Schema des Ergebnisses geändert und die Daten wurden sortiert.

Programm 10.1.3: Finden Sie alle Studierenden aus Oehna und Berlin mit der Note 3 oder einer schlechteren Note, mit allen Ergebnissen.

```

aus studenten.tab
sel ORT in "Berlin Oehna"
sel STID! NOTE>2 # wählt nur Studenten und nicht Examen
gib NAME,ORT,(KURS,NOTE m)b

```

Ergebnis (tab)			
NAME	ORT	(KURS	NOTE m) b
Sophia	Berlin	Algebra	3
		Datenbanken	1
		Otto	1

Programm 10.1.4: Finden Sie alle Studierenden, die nur die Note 1 und mindestens eine Note 1 haben.			
aus studenten.tab sel NOTE m = {1} # { } sind Mengenklammern			
Ergebnis (tab)			
STID,NAME	ORT?	STIP,FAK	(KURS ,NOTE m),(PROJ ,STUNDEN m) m
5555 Kaethe Gerwisch	600 Kunst	Apel	1 Monet 20
		Repin	1

Für die Bewertung der Bedingung wird für jeden Schüler die Liste seiner Noten in eine Menge überführt. Somit ist die Menge von Ernst $\{1\ 2\ 1\} = \{1\ 2\}$ und die Menge $\{1\ 1\}$ von Käthe gleich $\{1\}$. Zwei Mengen sind gleich, wenn sich jedes Element der linken Seite auch auf der rechten Seite und jedes Element der rechten Seite auf der linken Seite befindet. Mit anderen Worten: Zwei Mengen M1 und M2 sind gleich, wenn 'M1 inmath M2 & M2 inmath M1' gilt. Wenn wir alle Schüler mit genau zwei Noten 1 haben wollen, dann können wir Multimengen verwenden: NOTE b = $\{\{1\ 1\}\}$ (b kürzt Bag ab). Wenn auch die Reihenfolge der Noten wichtig ist, dann können wir Listen nehmen: NOTE l = $[1\ 2\ 1]$, ...

Programm 10.1.5: Finde alle Studenten, die im Algebra-Kurs eine Eins bekommen haben.			
aus studenten.tab sel STID! KURS=Algebra & NOTE=1 gib STID,NAME,(KURS,NOTE m)m			
Ergebnis (tab)			
STID,NAME	(KURS	NOTE m) m	
1111 Ernst	Algebra	1	
	Geschichte	1	
	Logik	2	

Programm 10.1.6: Finden Sie alle Studenten, die einen Algebrakurs belegt haben und eine 1 haben (nicht unbedingt im selben Kurs).			
aus studenten.tab sel STID! KURS=Algebra sel STID! NOTE=1 gib STID,NAME,(KURS,NOTE m)m			
Ergebnis (tab)			
STID,NAME	(KURS	NOTE m) m	
1111 Ernst	Algebra	1	
	Geschichte	1	
	Logik	2	
2222 Sophia	Algebra	3	
	Datenbanken	1	
	Otto	1	

Programm 10.1.7: Finden Sie alle Studenten, die bereits Prüfungen in Algebra und Datenbanken haben.

```
aus studenten.tab
sel STID! KURS=Algebra
sel STID! KURS=Datenbanken
```

Ergebnis (tab)

STID,NAME ,ORT? ,STIP,FAK ,(KURS ,NOTE m),(PROJ ,STUNDEN m) m
2222 Sophia Berlin 400 Infor Algebra 3 Ghandi 5
Datenbanken 1 Ming 4
Otto 1 Otto 6

Zwischenergebnis nach der ersten Bedingung

1111 Ernst Oehna 500 Mathe Algebra 1 Fritz 4
Geschichte 1 Otto 2
Logik 2
2222 Sophia Berlin 400 Infor Algebra 3 Ghandi 5
Datenbanken 1 Ming 4
Otto 1 Otto 6

Wenn wir beide Bedingungen durch & (and) verbinden würden, „enthält“ diese Bedingung nur einen EXIST-Quantor, von der Art, dass kein Untertupel existiert, das beide Unterbedingungen gleichzeitig erfüllt. Das Ergebnis wäre in jedem Fall leer.

Programm 10.1.8: Geben Sie für jeden Studenten, der Algebra beendet hat, alle weiteren Kurse an, die er beendet hat.

```
aus studenten.tab
sel STID! KURS=Algebra # wählt Studenten
sel- KURS! KURS=Algebra # wählt Examen
gib NAME,KURSB m
```

Ergebnis (tabh)

NAME, KURSB m

Ernst Geschichte Logik
Sophia Datenbanken Otto

Programm 10.1.9: Gesucht sind alle Segmente, die einen Bezug zu Otto haben.

```
aus studenten.tab
sel Otto
```

Ergebnis (tab)

STID,NAME ,ORT? ,STIP,FAK ,(KURS ,NOTE m),(PROJ ,STUNDEN m) m
1111 Ernst Oehna 500 Mathe Otto 2
2222 Sophia Berlin 400 Infor Otto 1 Otto 6

Programm 10.1.10: Drucke aus allen Tupeln der Universitätsdatenbank (auf die ich Zugriff habe) die Segmente, die das Wort Apel enthalten.

```
aus studenten.tab,kurse.tab
sel Apel
```

Ergebnis (xml)

```
<TABM>
  <STUDENTEN>
    <STID>5555</STID>
    <NAME>Käthe</NAME>
    <ORT>Gerwisch</ORT>
```

```

<STIP>600</STIP>
<FAK>Kunst</FAK>
<KURS>Apel</KURS>
<NOTE>1</NOTE>
</STUDENTEN>
<KURSE/>
</TABM>

```

Bisher haben wir in diesem Abschnitt nur „Selektion nach Inhalt“ betrachtet, aber fast die gleiche Bedeutung hat „Selektion nach Position“. Dies ist nicht nur für Listen sinnvoll, sondern kann auch im Rahmen von „relationalen Anwendungen“ genutzt werden. Wir betrachten hier nur zwei Beispiele.

Programm 10.1.11: Gib für jeden Studenten aus Oehna mit Prüfungen, die letzte Prüfung.

```

aus studenten.tab
sel  ORT=Oehna
sel  NOTE pos- = 1
gib  STID,NAME,(KURS,NOTE m)m

```

Ergebnis (tab)

```
STID,NAME ,(KURS ,NOTE m) m
```

```

1111 Ernst  Logik    2
3333 Clara  OCaml    2

```

Die Funktion pos (pos-) liefert die Positionsnummer (Positionsnummer rückwärts) des (Unter-) Elements in der entsprechenden Menge. Daher ist NOTE pos dasselbe wie KURS pos.

Programm 10.1.12: Gib für die 3 besten Studenten die 2 besten Prüfungen an. Wir lassen Ulrike weg, da wir für sie keinen Durchschnitt berechnen können. Sie hat noch keine Noten.

```

aus studenten.tab
sel- NAME=Ulrike
sel  NOTE=NOTE
gib  AVGM,NAME,FAK,(NOTE,KURS m)m
      AVGM:= NOTE! ++:
sel  NAME pos < 4
sel  NOTE pos < 3
rnd  2

```

Ergebnis (tab)

```
AVGM,NAME ,FAK ,(NOTE,KURS m ) m
```

```

1.00 Käthe  Kunst  1    Apel
                1    Repin
1.33 Ernst  Mathe  1    Algebra
                1    Geschichte
1.50 Clara  Infor  1    Datenbanken
                2    OCaml

```

Hier genügt es zu wissen, dass durch die gib-Klausel die Studenten nach AVGM sortiert werden, die Prüfungen nach NOTE sortiert sind und AVGM der Durchschnitt für jeden Studenten ist. Die gib-Klausel wird in Abschnitt 10.3 näher erläutert.

Obwohl die folgende Anfrage kein *sel* oder *sel-* erfordert, handelt es sich bei *first* und *last* dennoch um Selektionen. *last* wählt aus jeder Kollektion das letzte Element. Diese Operationen kann man benutzen, um schnell einen ersten Eindruck von Struktur und Inhalt eines Tabments zu bekommen.

Programm 10.1.13: Finde die letzten Elemente jeder Kollektion der Studentendatei.									
studenten.tab last									
Ergebnis (tab)									
STID, NAME, ORT?, STIP, FAK, (KURS, NOTE m),(PROJ, STUNDEN m) m									
6666	Claudia	Berlin	600	Sport	Ski	1		Witt	12

Programm 10.1.14: Finde die Studenten mit dem höchsten Stipendium.									
aus studenten.tab									
STIPMAX:=STIP1 max # eine neue Spalte mit einem Wert wird erzeugt									
sel STIP=STIPMAX									
Ergebnis (tab)									
STIPMAX,STID,NAME ,ORT? ,STIP,FAK ,(KURS ,NOTE m),(PROJ ,STUNDEN m) m									
600	5555	Käthe	Gerwisch	600	Kunst	Apel	1	Monet	20
						Repin	1		
6666	Claudia	Berlin		600	Sport	Psycho	2	Matthes	8
						Ski	1	Witt	12

10.2 Berechnungen (:=)

Programm 10.2.1: Berechne die Bruttowerte mehrerer Preise.									
3.18 55.88 17.90 * 1.19									
Ergebnis (hsqh und tabh)									
PZAHL1									
3.7842	66.4972	21.301							

Programm 10.2.2: Berechne die Bruttowerte mehrerer Preise und belasse die eingegebenen Werte in der Ausgabe.									
NETTO1:=3.18 55.88 17.90									
BRUTTO:=NETTO*1.19									
Ergebnis (tab)									
NETTO,BRUTTO 1									
3.18	3.7842								
55.88	66.4972								
17.9	21.301								

Programm 10.2.3: Berechne die Bruttowerte mehrerer Preise.									
3.18 55.88 17.90 +% 19									
Ergebnis (tabh)									
PZAHL1									
3.7842	66.4972	21.301							

Programm 10.2.4: Wandle alle Nettopreise einer kleinen Tabelle in Bruttopreise um.									
<TAB!									
ARTIKEL, PREIS 1									
OttoRAMDB 500									
OttoWiki 10									
OttoCalc 20									

!TAB> +% 19
Ergebnis
ARTIKEL ,PREIS 1
OttoRAMDB 595. OttoWiki 11.9 OttoCalc 23.8

Zu jedem Wert der Tabelle werden 19% hinzugefügt. Textwerte werden durch arithmetische Operationen mit Zahlen nicht verändert.

Programm 10.2.5: Berechne den Bruttowert jeder Position und die Summe aller Bruttowerte.				
<TAB! ARTIKEL, PREIS, ANZAHL m OttoRAMDB 500 20 OttoWiki 10 200 OttoCalc 20 4000 !TAB> TOTAL:=PREIS*ANZAHL +% 19 TOTALSUM:=TOTAL1 ++				
Ergebnis				
TOTALSUM,(ARTIKEL ,PREIS, ANZAHL,TOTAL m)				
109480.	OttoCalc	20	4000	95200.
	OttoRAMDB	500	20	11900.
	OttoWiki	10	200	2380.

Programm 10.2.6: Gib das Stipendium jedes Informatikstudenten in Dollar an.							
aus studenten.tab sel FAK=Infor DOL:=STIP*1.02							
Ergebnis							
STID,NAME ,ORT?	,STIP,FAK	,DOL ,(KURS	,NOTE m),	(PROJ ,STUNDEN m)	m		
2222 Sophia Berlin	400 Infor	408.	Algebra	3	Ghandi	5	
			Datenbanken	1	Ming	4	
			Otto	1	Otto	6	
3333 Clara Oehna	450 Infor	459.	Datenbanken	1			
			OCaml	2			

Programm 10.2.7: Zahle jedem Studenten für jedes seiner Projekte 100 Euro.	
aus studenten.tab BONUS:= PROJ1 ++1 *100 gib STID,NAME,BONUS m	
Ergebnis	
STID,NAME	,BONUS m
1111 Ernst	200
2222 Sophia	300
3333 Clara	0
4444 Ulrike	100
5555 Kaethe	100
6666 Claudia	200

Programm 10.2.8: Zahle jedem Studenten aus Oehna einen zusätzlichen Bonus, abhängig von seinem Notendurchschnitt.

```
aus studenten.tab
sel ORT=Oehna
AVG1:= NOTE1 ++:
BONUS3:=1000 : AVG1
gib STID,NAME,AVG1,BONUS3 m
rnd 2
```

Ergebnis

STID,NAME	AVG1	BONUS3
1111 Ernst	1.33	750.00
3333 Clara	1.50	666.67

Mit Hilfe von rnd (runden) wird jeder Wert einer Tabelle auf 2 Stellen nach Punkt (ehemals Komma) gerundet. Bei Texten bleibt der Wert wieder unverändert.

Programm 10.2.9: Die Studenten der Mathe Fakultät bekommen einen Bonus von 900 Euro, der Informatik von 800 Euro und alle anderen bekommen 700 Euro.

```
aus studenten.tab
BONUS:= 900 if FAK=Mathe !
        800 if FAK=Infor !
        700
gib STID,NAME,FAK, BONUS m
```

Ergebnis

STID,NAME	FAK	BONUS
1111 Ernst	Mathe	900
2222 Sophia	Infor	800
3333 Clara	Infor	800
4444 Ulrike	Kunst	700
5555 Kaethe	Kunst	700
6666 Claudia	Sport	700

Programm 10.2.10: Berechne den BMI (body mass index) für jedes Gewicht jeder Person.

```
<TAB!
NAME, LAENGE, (ALTER, GEWICHT 1)1
Klaus 1.68      18      61
              30      65
              56      80
              61      75
Kathi 1.70      18      55
              40      70
!TAB>
BMI:= GEWICHT : LAENGE : LAENGE
rnd 2
```

Ergebnis (tab)

NAME	LAENGE	(ALTER,GEWICHT,BMI 1)
Kathi	1.70	18 55 19.03
		40 70 24.22
Klaus	1.68	18 61 21.61
		30 65 23.03
		56 80 28.34
		61 75 26.57

Beachte, dass die angegebene Formel nicht nur auf die Zeilen angewendet wird, in der eine Länge steht, sondern auch auf die folgenden Zeilen. Dies ist möglich, weil die Tabelle ein bestimmtes Schema hat und unser System das Schema verstehen kann.

10.3 Restrukturierung (gib)

Die Restrukturierungsoperation (stroke) erlaubt die Restrukturierung eines beliebigen Tabments in ein anderes beliebiges Tabment nur durch Angabe des Schemas oder des TT des Ziel-Tabments. Zusätzlich können Aggregationen, Beseitigung von Duplikaten, Union, Sortierung und bestimmte Joins realisiert werden.

Programm 10.3.1: Illustrieren der Kollektions-Symbole						
aus studenten.tab gib FAKm,FAKb,FAKl,FAKm-,FAKb-,FAKl-,FAK?						
Ergebnis						
FAKm ,FAKb ,FAKl ,FAKm- ,FAKb- ,FAKl- ,FAK?						
Infor	Infor	Mathe	Sport	Sport	Sport	Mathe
Kunst	Infor	Infor	Mathe	Mathe	Kunst	
Mathe	Kunst	Infor	Kunst	Kunst	Kunst	
Sport	Kunst	Kunst	Infor	Kunst	Infor	
	Mathe	Kunst		Infor	Infor	
	Sport	Sport		Infor	Mathe	

Die STID-Segmente (Typ: (STID, NAME, ORT?, STIP, FAK)) werden nacheinander in jede der gegebenen FAK-Kollektionen eingefügt. KURS- und PROJ-Segmente werden nicht beachtet. Aus Zwecken der Übersichtlichkeit haben wir die Kollektionssymbole der gib-Klausel beibehalten.

Programm 10.3.2: Sortiere Studenten nach FAK und NAME.	
aus studenten.tab gib FAK,NAMEb m	
Ergebnis (tabh)	
FAK, NAMEb m	
Infor	Clara Sophia
Kunst	Kaethe Ulrike
Mathe	Ernst
Sport	Claudia

STID-Segmente werden zuerst in die FAK-Ebene und dann tiefer in die NAME-Ebene eingefügt, Segment für Segment. KURS- und PROJ-Segmente werden nicht mehr berührt.

Programm 10.3.3: Sortiere Studenten nach FAK und NAME, mit dem Ergebnis einer flachen Tabelle.	
aus studenten.tab gib FAK,NAME m	
Ergebnis (tab)	
FAK ,NAME m	
Infor	Clara
Infor	Sophia
Kunst	Käthe
Kunst	Ulrike
Mathe	Ernst
Sport	Claudia

Wenn wir m durch b ersetzen, ändern sich die Ergebniselemente nicht.

Programm 10.3.4: Sortiere die Fakultäten abwärts nach BUDGET und zweitens nach Studentenkapazität.

aus faks.tab

gib BUDGET,STUDKAPAZI,FAK m-

Ergebnis (tab)

BUDGET,STUDKAPAZI,FAK m

10000	500	Infor
8000	150	Sport
2000	600	Kunst
1000	200	Mathe
1000	10	Philo

Programm 10.3.5: Sortiere die Fakultäten nach Budget und zusätzlich nach Studentenkapazität (Duplizierung der Tabelle).

aus faks.tab

gib BUDGET,FAK m-, (STUDKAPAZI,FAK m-)

Ergebnis (tab)

BUDGET,FAK m- , (STUDKAPAZI,FAK m-)

10000	Infor	600	Kunst
8000	Sport	500	Infor
2000	Kunst	200	Mathe
1000	Philo	150	Sport
000	Mathe	10	Philo

Programm 10.3.6: Packen Sie die Prüfungsdaten jedes Studenten nach Abteilung. (Gruppieren Sie bereits gruppierte Daten neu.).

aus studenten.tab

gib FAK,(KURS,NOTE b)m

Ergebnis (tab)

FAK , (KURS ,NOTE b) m

Infor	Algebra	3
	Datenbanken	1
	Datenbanken	1
	OCaml	2
	Otto	1
Kunst	Apel	1
	Repin	1
Mathe	Algebra	1
	Geschichte	1
	Logik	2
Sport	Psycho	2
	Ski	1

Hier werden die STID-Segmente in die FAK-Ebene eingefügt. Sie können nicht tiefer eingefügt werden, da sie weder KURS- noch NOTE-Werte enthalten. Die entsprechenden Prüfungsmultimengen sind dann zunächst stets leer. Dann wird jedes KURS-Segment ((KURS, NOTE)-Paar) um ihr übergeordnetes STID-Segment verlängert. Diese verlängerten Segmente können Schritt für Schritt in die entsprechende Multimenge eingefügt werden. Das verlängerte Segment hat den Typ: (STID, NAME, ORT?, STIP, FAK, KURS, NOTE) PROJ Segmente werden nicht benötigt.

Programm 10.3.7: (Spezielle Selektion mit gib-Klausel) Gib alle Studenten, für die ein ORT-Eintrag existiert, mit diesem Eintrag. Gib zusätzlich die gegebene Kollektion für Vergleichszwecke.

aus studenten.tab

gib NAME,ORT m,(NAME,ORT? m)

Ergebnis (tab)

NAME	ORT	m	(NAME	ORT?	m)
------	-----	---	-------	------	----

Clara	Oehna		Clara	Oehna	
Claudia	Berlin		Claudia	Berlin	
Ernst	Oehna		Ernst	Oehna	
Käthe	Gerwisch		Käthe	Gerwisch	
Sophia	Berlin		Sophia	Berlin	
			Ulrike		

In der ersten Menge fordert der Nutzer vollständige Paare. Da für Ulrike kein Paar existiert, kann sie nicht im ersten Ergebnis erscheinen.

Programm 10.3.8: (Selektion nur mit gib-Klausel) Geben Sie alle Studenten mit nichtleeren Prüfungs-Kollektionen an.

aus studenten.tab

gib STID,NAME,FAK,KURS,NOTE m

gib STID,NAME,FAK,(KURS,NOTE m)m

Ergebnis (tab)

STID	NAME	FAK	(KURS	NOTE	m) m
------	------	-----	-------	------	------

1111	Ernst	Mathe	Algebra	1	
			Geschichte	1	
			Logik	2	
2222	Sophia	Infor	Algebra	3	
			Datenbanken	1	
			Otto	1	
3333	Clara	Infor	Datenbanken	1	
			OCaml	2	
5555	Kaethe	Kunst	Apel	1	
			Repin	1	
6666	Claudia	Sport	Psycho	2	
			Ski	1	

Zwischenresultat nach der ersten gib-Klausel:

1111	Ernst	Mathe	Algebra	1
1111	Ernst	Mathe	Geschichte	1
1111	Ernst	Mathe	Logik	2
2222	Sophia	Infor	Algebra	3
2222	Sophia	Infor	Datenbanken	1
2222	Sophia	Infor	Otto	1
3333	Clara	Infor	Datenbanken	1
3333	Clara	Infor	OCaml	2
5555	Käthe	Kunst	Apel	1
5555	Käthe	Kunst	Repin	1
6666	Claudia	Sport	Psycho	2
6666	Claudia	Sport	Ski	1

Um das Zwischenergebnis zu erhalten, wird zunächst versucht STID-Segmente einzufügen. Dies ist nicht möglich, da auf dieser Ebene keine Prüfungsdaten vorhanden sind. Dann wiederum wird jedes KURS-Segment um seine übergeordneten Daten der ersten Ebene verlängert. Diese Daten werden Prüfung für Prüfung und Student für Student eingefügt.

Programm 10.3.9: (Unkontrollierte Selektion) Gib für jeden Namen den „ersten“ NOTE-Eintrag oder "Nullwert" aus, wenn kein Prüfungseintrag vorhanden ist. Drucke die weiteren Kollektionen für Vergleichszwecke.

```
aus studenten.tab
```

```
gib (NAME,NOTE? m),(NAME,NOTE m),(NAME,NOTE b),(NAME,NOTEb m)
```

Ergebnis (tabh)

NAME,	NOTE?	m,	(NAME,	NOTE m),	(NAME,	NOTE b),	(NAME,	NOTE1 m)
Clara	1		Clara	1	Clara	1	Clara	1 2
Claudia	2		Clara	2	Clara	2	Claudia	1 2
Ernst	1		Claudia	1	Claudia	1	Ernst	1 1 2
Käthe	1		Claudia	2	Claudia	2	Käthe	1 1
Sophia	3		Ernst	1	Ernst	1	Sophia	1 1 3
Ulrike	0		Ernst	2	Ernst	1	Ulrike	0
			Käthe	1	Ernst	2		
			Sophia	1	Käthe	1		
			Sophia	3	Käthe	1		
			Ulrike	0	Sophia	1		
					Sophia	1		
					Sophia	3		
					Ulrike	0		

In die erste und letzte Kollektion können STID-Segmente eingefügt werden, da nur Namen gefordert sind.

Programm 10.3.10: (Restrukturierung) Kehre die gegebene Strukturierung um. D.h., vertausche KURS und NAME.

```
aus studenten.tab
```

```
gib KURS,(NAME,NOTE b)m
```

Ergebnis (tab)

KURS	,	(NAME	,NOTE b) m
Algebra		Ernst	1
		Sophia	3
Apel		Kaethe	1
Datenbanken		Clara	1
		Sophia	1
Geschichte		Ernst	1
Logik		Ernst	2
OCaml		Clara	2
Otto		Sophia	1
Psycho		Claudia	2
Repin		Käthe	1
Ski		Claudia	1

Hier wird zunächst versucht, die STID-Segmente einzufügen. Da kein KURS-Attribut existiert, können sie nicht eingefügt werden. Daher werden die verlängerten KURS-Segmente zuerst auf der KURS-Ebene und dann auf der NAME-Ebene eingefügt.

Programm 10.3.11: (Restrukturierung mit zusätzlichen Tags) Kehre die gegebene Strukturierung um, indem Sie KURSE von innerer zu äußerer Kollektion und NAME von äußerer zu innerer ändern. Erstelle zusätzlich Tupel- und Sub-Tupel-Tags.

```
aus studenten.tab
```

```
gib KURSE
```

```
    KURSE      = KURSTUPELm
```

```
    KURSTUPEL = KURS,EXAMSTUPELb
```

```
    EXAMSTUPEL= NAME,NOTE
```

Ergebnis (ment)

```
<KURSE>
  <KURSTUPEL>
    <KURS>Algebra</KURS>
    <EXAMSTUPEL>
      <NAME>Ernst</NAME>
      <NOTE>1</NOTE>
    </EXAMSTUPEL>
    <EXAMSTUPEL>
      <NAME>Sophia</NAME>
      <NOTE>3</NOTE>
    </EXAMSTUPEL>
  </KURSTUPEL>
  <KURSTUPEL>
    <KURS>Apel</KURS>
    <EXAMSTUPEL>
      <NAME>Käthe</NAME>
      <NOTE>1</NOTE>
    </EXAMSTUPEL>
  </KURSTUPEL>
  <KURSTUPEL>
    <KURS>Datenbanken</KURS>
    <EXAMSTUPEL>
      <NAME>Clara</NAME>
      <NOTE>1</NOTE>
    </EXAMSTUPEL>
    <EXAMSTUPEL>
      <NAME>Sophia</NAME>
      <NOTE>1</NOTE>
    </EXAMSTUPEL>
  </KURSTUPEL>
  <KURSTUPEL>
    <KURS>Geschichte</KURS>
    <EXAMSTUPEL>
      <NAME>Ernst</NAME>
      <NOTE>1</NOTE>
    </EXAMSTUPEL>
  </KURSTUPEL>
  <KURSTUPEL>
    <KURS>Logik</KURS>
    <EXAMSTUPEL>
      <NAME>Ernst</NAME>
      <NOTE>2</NOTE>
    </EXAMSTUPEL>
  </KURSTUPEL>
  ...
</KURSE>
```

Nun wollen wir die mengentheoretischen Operationen Vereinigung, Durchschnitt und Mengendifferenz veranschaulichen. Da die STID-Spalte in studenten.tab bereits eine Vereinigung ist, veranschaulichen wir die Vereinigung mit den Dateien exams1 und projects1.

Programm 10.3.12: Konstruiere die Vereinigung zweier Dateien, wobei jeder Wert jeder Datei im Ergebnis erscheinen soll.

```
aus examen1.tab,projekte1.tab # ein Paar von Tabellen
gib STIDb
```

Ergebnis (tabh)
STIDb
111 1111 1111 1111 1111 1111 2222 2222 2222 2222 2222 2222 3333 3333 3333 3333 5555 5555 5555 5555 6666 6666 6666 6666
Wenn wir in der gib-Anweisung b durch m ersetzen, werden Duplikate eliminiert.
Ergebnis
STIDm
1111 2222 3333 5555 6666
Wenn wir wissen möchten, aus welcher Datei jeder STID stammt, können wir entsprechende Informationen hinzufügen
aus examen1.tab,projekte1.tab # ein Paar von Tabellen gib STID,KURS?,PROJ? b
Ergebnis
STID, KURS?, PROJ? b
1111 Fritz 1111 Otto 1111 Algebra 1111 Geschichte 1111 Logik 2222 Ghandi 2222 Ming 2222 Otto 2222 Algebra 2222 Datenbanken 2222 Otto 3333 Datenbanken 3333 OCaml 4444 Monet 5555 Monet 5555 Apel 5555 Repin 6666 Matthes 6666 Witt 6666 Psycho 6666 Ski

Programm 10.3.13: Konstruiere die Schnittmenge (Durchschnitt) zweier Dateien mit unterschiedlichen Schemata.
aus examen1.tab,projekte1.tab gib STID,KURS?,PROJ? m gib STID,KURS,PROJ m # diese Restrukturierung kann auch durch Selektionen realisiert werden gib STIDm
Ergebnis (tabh, width 50))
STIDm
111 1111 1111 1111 1111 1111 2222 2222 2222 2222 2222 2222 3333 3333 3333 3333 5555 5555 5555 5555 6666 6666 6666 6666
Wenn wir in der gib-Anweisung b durch m ersetzen, werden Duplikate eliminiert.
Ergebnis
STIDm
1111 2222 5555 6666

Zwischenergebnis nach der ersten gib-Anweisung			
aus examen1.tab,projekte1.tab # ein Paar von Tabellen gib STID,KURS?,PROJ? b			
Ergebnis			
STID, KURS?,	PROJ?	m	
1111	Algebra	Fritz	
2222	Algebra	Ghandi	
3333	Datenbanken		
4444		Monet	
5555	Apel	Monet	
6666	Psycho	Matthes	
Programm 10.3.14: Mengendifferenz: Gib alle STIDs von examen1.tab an, die nicht in projekte1.tab enthalten sind.			
aus examen1.tab rename STID ! STUDID ,projekte1.tab # wiederum ergibt sich ein Tupel (Paar) von Tabellen sel- STUDID in STIDm gib STUDIDm			
Ergebnis (tab)			
STUDIDm			
3333			

Programm 10.3.15: analog zu 10.3.14, aber mit verschachtelter Anfrage			
aus examen1.tab sel- STID in begin projekte1.tab;; gib STIDm end gib STIDm			
Ergebnis (tab)			
STUDIDm			
3333			

Programm 10.3.16: (Gruppierung mit Aggregation) Berechne die Anzahl der Studenten und die Anzahl für jede Fakultät. Sortiere die Studenten nach FAK und NAME.			
aus studenten.tab gib ANZ,(FAK,ANZ,(NAME,STID b)m) ANZ:= STID! ++1			
Ergebnis (tab)			
ANZ,FAK	ANZ2,(NAME	STID b)	m
6 Infor	2	Clara	3333
		Sophia	2222
Kunst	2	Käthe	5555
		Ulrike	4444
Mathe	1	Ernst	1111
Sport	1	Claudia	6666

Programm 10.3.17: (Restrukturierung mit Aggregation) Gib Sie die Gesamtsumme aller Stipendien und die Summe für jeden Kurs an. Sortiere die Datensätze nach Kurs.			
aus studenten.tab gib SU,(KURS,SU m) SU:= STIP! ++			
Ergebnis (tab)			
SU	(KURS	SU2	m)

2950	Algebra	900
	Apel	600
	Datenbanken	850
	Geschichte	500
	Logik	500
	OCaml	450
	Otto	400
	Psycho	600
	Repin	600
	Ski	600

Interessant ist hier, dass das ++ der inneren SU-Werte generell größer ist als der äußere SU-Wert. Dies liegt daran, dass eine bestimmte Lehrveranstaltung in der Regel in mehr als einem Studierendendatensatz vorkommt.

Programm 10.3.18: Suche den Namen des Studenten mit der ID 2222

```
aus studenten.tab
sel STID = 2222
gib NAME
```

Ergebnis (tab)

NAME

Sophia

Programm 10.3.19: Teile die Studierenden der beiden Fakultäten Informatik und Kunst in zwei unabhängige Tabellen auf.

```
aus studenten.tab
sel- Sport
gib FAK,(NAME,ORT? m)m
cut 1
```

Ergebnis (tab)

FAK ,(NAME ,ORT? m) 1,(FAK ,(NAME ,ORT? m) 1),(FAK ,(NAME ,ORT? m)1)

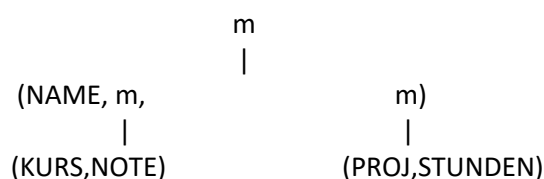
Infor Clara Oehna	Kunst Käthe Gerwisch	Mathe Ernst Oehna
Sophia Berlin	Ulrike	

Der Begriff des hierarchischen Weges ist wichtig für alle Operationen. Seine Definition basiert auf "schmalen" Schemen. Alle Kollektionssymbole außer '?' sind echte Kollektionssymbole.

Ein **Schema s ist schmal**, wenn für 2 beliebige echte Kollektionssymbole c und c' gilt, entweder c ist in c' enthalten oder c' in c. Felder f1 und f2 eines Schemas s befinden sich auf einem **hierarchischen Pfad** (kurz **HP**) **bzgl. s**, wenn das Schema, das durch Vergessen aller Felder außer f1 und f2 entsteht, schmal ist.

X,Ym,Zm m ist nicht schmal, aber X,Y?,Z? m. PROJ und KURS befinden sich in

NAME, (KURS, NOTE m), (PROJ, STUNDEN m)m, nicht auf einem hierarchischen Pfad; im Gegensatz zu PROJm and KURS. Das wird in der graphischen Repräsentation des Schemas sichtbar.



Programm 10.3.19: Bringe zwei Felder, die nicht auf einem HP sind, auf einen HP
aus studenten.tab gib KURS,PROJ m # in jedem Fall leer
Ergebnis (tabh)
KURS, PROJ m

Programm 10.3.20: Sortiere und gruppier die Studenten, die einen Algebra KURS abgelegt haben, nach den entsprechenden Noten und sortiere sie nach Namen und drucke alle ihre Projekte.
aus studenten.tab sel KURS=Algebra gib NAME,NOTE?,(PROJ,STUNDEN m)m gib NOTE,(NAME,(PROJ,STUNDEN m)b) m
Ergebnis
NOTE, (NAME, (PROJ, STUNDEN m) b) m
1 Ernst Fritz 4 Otto 2 3 Sophia Ghandi 5 Ming 4 Otto 6

Obwohl PROJ und NOTE nicht auf einem HP liegen, ist die Projekt-Kollektion nicht leer. Das konnte durch eine etwas spitzfindige Formulierung mit 2 gib-Anweisungen realisiert werden. Wir werden im Folgenden eine leichter verständliche Formulierung solcher Anfragen kennenlernen.

10.4 Ein einfacher Join und verschachtelte Anfragen

Das horizontale Zusammenführen oder Verbinden der Informationen zweier Tabellen wird als „Join“ bezeichnet. Bei unserem Ansatz ist die Verknüpfung zweier flacher Tabellen nicht unbedingt eine flache Tabelle. Wir brauchen keine zusätzliche Join-Operation. Mit := lassen sich sinnvolle Strukturen erstellen.

Programm 10.4.1: Erweitere die Studierendendaten um Prüfungsdaten.
aus studenten1.tab EXAMEN:= examen1.tab at FAK
Anfangsteil des Ergebnisses (tab)
STID,NAME ,ORT? ,STIP,FAK ,(STID,KURS , NOTE m) m
1111 Ernst Oehna 500 Mathe 1111 Algebra 1 1111 Geschichte 1 1111 Logik 2 2222 Algebra 3 2222 Datenbanken 1 2222 Otto 1 3333 Datenbanken 1 3333 OCaml 2 5555 Apel 1 5555 Repin 1 6666 Psycho 2 6666 Ski 1 2222 Sophia Berlin 400 Infor 1111 Algebra 1 1111 Geschichte 1 1111 Logik 2 2222 Algebra 3 2222 Datenbanken 1 2222 Otto 1

				3333	Datenbanken	1	
				3333	OCaml	2	
				5555	Apel	1	
				5555	Repin	1	
				6666	Psycho	2	
				6666	Ski	1	
3333	Clara	Oehna	450	Infor	1111	Algebra	1
					1111	Geschichte	1
					1111	Logik	2
					2222	Algebra	3
...							

Das Ergebnis enthält $5 \cdot 10 = 50$ Subtupel. Um die 10 gewünschten Subtupel zu erhalten, müssen wir eine Bedingung hinzufügen.

Programm 10.4.2: Fügen Sie jedem Studentendatensatz alle entsprechenden Prüfungsdatensätze hinzu („strukturierter Left Outer Join“).

aus studenten1.tab

EXAMEN:=examen1.tab at FAK

sel KURS! STUDENTEN1/STID=EXAMEN/STID

Ergebnis (tab)

STID,NAME	,ORT?	,STIP,FAK	, (STID,KURS	,NOTE m) m
1111 Ernst	Oehna	500 Mathe	1111 Algebra	1
			1111 Geschichte	1
			1111 Logik	2
2222 Sophia	Berlin	400 Infor	2222 Algebra	3
			2222 Datenbanken	1
			2222 Otto	1
3333 Clara	Oehna	450 Infor	3333 Datenbanken	1
			3333 OCaml	2
4444 Ulrike		400 Kunst		
5555 Käthe	Gerwisch	600 Kunst	5555 Apel	1
			5555 Repin	1
6666 Claudia	Berlin	600 Sport	6666 Psycho	2
			6666 Ski	1

Wenn wir Ulrike weglassen wollen, müssen wir nur den Level-Bezeichner 'KURS:' weglassen. Jedes Tabment mit dem Namen xyz.tab hat das äußerste Tag xyz. Daher ergibt sich durch obige Erweiterung folgender TT:

TT (Tabmenttyp des Ergebnisses)

TABMENT! STUDENTEN1

STUDENTEN1! STID,NAME,ORT?,STIP,FAK,EXAMEN m

EXAMEN! STID,KURS,NOTE m

NOTE STIP STID! ZAHL

KURS FAK ORT NAME! TEXT

Dieser TT ermöglicht eine genaue Angabe von Spaltennamen trotz doppeltem Namensvorkommen. EXAMEN/KURS ist dasselbe wie KURS, da KURS nur einmal auf der rechten Seite des TT erscheint.

Außerdem befindet sich STUDENTEN1 auf der rechten Seite von EXAMEN, sodass der Tag-Pfad STUDENTEN1/EXAMEN/KURS auch mit KURS identisch ist. Der „Tag-Pfad“ STID spezifiziert jedoch

nicht genau, da er zweimal vorkommt. STUDENTEN1/STID ist die Studentenkennung der Tabelle studenten1 und EXAMEN/STID=STUDENTEN1/EXAMEN/STID der Prüfungstabelle.

In einem Markierungspfad X/Y/Z muss im TT Z auf der rechten Seite von Y und Y auf der rechten Seite von X vorkommen. X ist die Vatermarkierung von Y und Y die Vatermarkierung von Z. Es gibt keine Markierung zwischen X und Y und Y und Z (in der XML- oder ment-Darstellung). Wenn wir nicht alle Zwischentags kennen, können wir auch die Schreibweise X//Z verwenden. Dabei können beliebig viele Tags zwischen X und Z liegen. Das heißt, dieser Tag-Pfad entspricht einem vollständigen Tag-Pfad X/X1/X2/.../Xn/Z für passende Tags X1,...,Xn. Daher beschreibt STUDENT1//KURS KURS genauso wie der vollständige Tag-Pfad STUDENT1/EXAMEN/KURS.

Programm 10.4.3: Programm mit verschachtelter Anfrage						
aus studenten1.tab EXAMEN:=begin aus examen1.tab;;sel STID=STID~ end at FAK						
Ergebnis (tab)						
STID,NAME	,ORT?	,STIP,FAK	, (STID,KURS	,NOTE m)	m	
1111 Ernst	Oehna	500 Mathe	1111 Algebra	1		
			1111 Geschichte	1		
			1111 Logik	2		
2222 Sophia	Berlin	400 Infor	2222 Algebra	3		
			2222 Datenbanken	1		
			2222 Otto	1		
3333 Clara	Oehna	450 Infor	3333 Datenbanken	1		
			3333 OCaml	2		
4444 Ulrike		400 Kunst				
5555 Kaethe	Gerwisch	600 Kunst	5555 Apel	1		
			5555 Repin	1		
6666 Claudia	Berlin	600 Sport	6666 Psycho	2		
			6666 Ski	1		

Verschachtelte Anfragen sind in begin und end enthalten. Wenn wir auf einen Spaltennamen außerhalb der inneren Anfrage verweisen wollen, müssen wir ein „~“ hinzufügen. Daher ist STID~ die Kennung von STID von studenten1.

Programm 10.4.4: Versuche, die gegebene Studententabelle aus drei gegebenen flachen Relationen zu generieren.						
aus studenten1.tab PR:=begin aus projekte1.tab sel STID=STID~ gib PROJ,STUNDEN m end at FAK EX:=begin aus examen1.tab sel STID=STID~ gib KURS,NOTE m end at FAK						
Ergebnis (tab)						
STID,NAME	,ORT?	,STIP,FAK	, (KURS	,NOTE m)	, (PROJ	,STUNDEN m)
1111 Ernst	Oehna	500 Mathe	Algebra	1	Fritz	4
			Geschichte	1	Otto	2
			Logik	2		
2222 Sophia	Berlin	400 Infor	Algebra	3	Ghandi	5
			Datenbanken	1	Ming	4
			Otto	1	Otto	6
3333 Clara	Oehna	450 Infor	Datenbanken	1		
			OCaml	2		

4444	Ulrike	400	Kunst	Monet	10		
5555	Käthe Gerwisch	600	Kunst	Apel	1	Monet	20
				Repin	1		
6666	Claudia Berlin	600	Sport	Psycho	2	Matthes	8
				Ski	1	Witt	12

Das Ergebnis entspricht studenten.tab.

Programm 10.4.5: Generiere eine Tabelle mit drei verschachtelten Ebenen.

```

aus faks.tab
proj- STUDKAPAZI
ST:=begin aus studenten1.tab
    sel  FAK=FAK~
    proj- FAK end          at BUDGET
EX:=begin aus examen1.tab
    sel  STID=STID~
    proj- STID end        at STIP

```

Ergebnis (tab)

FAK	,DEKAN	,BUDGET,	(STID,NAME	, (ORT?	,STIP,((KURS	,NOTE m) m) m
Infor Reichel	10000	2222	Sophia	Berlin	400	Algebra 3
						Datenbanken 1
						Otto 1
		3333	Clara	Oehna	450	Datenbanken 1
						OCaml 2
Kunst Sitte	2000	4444	Ulrike		400	
			5555	Kaethe Gerwisch	600	Apel 1
						Repin 1
Mathe Dassow	1000	1111	Ernst	Oehna	500	Algebra 1
						Geschichte 1
						Logik 2
Philo Hegel	1000					
Sport Streich	8000	6666	Claudia	Berlin	600	Psycho 2
						Ski 1

Wenn wir nur wenige Spalten löschen wollen, können wir statt einer gib-Klausel die proj--Klausel verwenden. Wir bemerken, dass wir eine Struktur mit der Verschachtelungstiefe 3 erhalten, obwohl die tiefste Verschachtelungsebene im Programm 2 ist.

10.5 Ein nutzerfreundlicher Join (ext ext2)

Anhand von Beispiel 10.3.20 wurde deutlich, dass das Problem des Ladens von Daten auf eine HP (Hierarchischen Pfad) in der Quellstruktur, wenn diese noch nicht auf einer HP vorhanden sind, in manchen Fällen durch eine zusätzliche gib-Anweisung ohne Verwendung des kartesischen Produkts gelöst werden kann. Dieses Problem ist besonders relevant bei relationalen Datenbanken mit flachen Strukturen. In einem Tupel solcher Dateien befindet sich nichts auf einer HP außer den Feldern, die sich in derselben Tabelle befinden. Daher ist eine gewöhnliche gib-Anweisung wenig ausdrucksstark. Relationale Systeme lösen dieses Problem mit Joins. Der Join ist jedoch mit dem kartesischen Produkt verknüpft. Außerdem müssen Join-Bedingungen verwendet werden. In [Gol08] wurden Experimente mit Studenten beschrieben. Diese zeigten, dass fehlende Join-Bedingungen der häufigste semantische SQL-Fehler sind. Bei Verwendung beider Konstrukte aus diesem Abschnitt müssen die Join-Bedingungen in der Regel nicht geschrieben werden. Im ersten Teil dieses Abschnitts werden einige typische Abfragen für ext (Extension) vorgestellt. Die Verwendung von ext ist einfach, die Definition scheint jedoch etwas komplizierter als die Anwendung.

Programm 10.5.1: Die sehr guten Prüfungen und die zeitintensiven Projekte sollen allen Studierenden gewährt werden, die nicht in Gerwisch wohnen, einen Kurs mit der Note 1

abgeschlossen haben und über die entsprechenden Projekte verfügen. Die Studierenden sollen nach ihrem Wohnort gruppiert werden.

```
aus studenten1.tab ext examen1.tab ext projekte1.tab
sel- ORT=Gerwisch
sel NOTE=1
sel STUNDEN>2
gib ORT,(NAME,(KURS,NOTE m),(PROJ,STUNDEN m)b)m
```

Result (tab)

ORT , (NAME , (KURS , NOTE 1), (PROJ , STUNDEN m) b) m

Berlin	Claudia	Ski	1	Matthes	8
				Witt	12
	Sophia	Databases	1	Ghandi	5
	Otto	1		Ming	4
				Otto	6
Oehna	Ernst	Algebra	1	Fritz	4
		History	1		

Programm 10.5.2: Group and sort student names with bad grades by faculty and output the students' bad courses. Omit STUDCAPACITY.

```
aus faks.tab,studenten1.tab,examen1.tab ext2
sel NOTE>2
proj- STUDKAPAZI
```

Result (tab)

FAC ,DEAN ,BUDGET ,(STID ,NAME ,LOC? ,STIP ,(COURSE ,MARK m) m) m

Infor Reichel 10000 2222 Sophia Berlin 400 Algebra 3

Programm 10.5.3: Gib alle Studenten von Oehna mit Dekan, Kursen und Projekten.

```
aus faks.tab ext studenten.tab
sel ORT=Oehna
gib STID,NAME,FAK,DEKAN,KURSm,PROJm m
```

Result (tab)

STID, NAME, FAK, DEKAN, KURSm, PROJm m

1111	Ernst	Math	Dassow	Algebra	Fritz
				History	Otto
				Logic	
3333	Clara	Infor Reichel	Databases		
			OCaml		

Programm 10.5.4: Finde alle Studierenden großer Fakultäten mit guten Algebra-Noten. FAK, (ORT, NAME) m. Strukturiere die Studierenden nach FAK und ORT und sortiere sie nach NAME.

```
aus faks.tab,studenten1.tab,examen1.tab ext2
sel STUDKAPAZI>300
sel NOTE<4 & KURS=Algebra
gib FAK,(ORT,NAMEb m)m
```

Result (tab)

FAK ,(ORT ,NAMEl m) m

Infor Berlin Sophia

11 Anfragen an die Wikipedia (keys)

Wie andere Wikipedia auch, stellt die deutsche Wikipedia einen großen Schatz an Wissen über Deutschland und die Welt dar. Nach der englischen Wikipedia ist sie die größte der Welt. Sie hat bereits 2,6 Millionen Einträge und die Anzahl, der Umfang und die Qualität der Inhalte werden ständig verbessert. Heute sind die meisten Anfragen an die Wikipedia einfacher Natur. Gib mir den Eintrag mit dem Schlüssel xyz. Diese Anfragen sind schnell beantwortet und schön gestaltet. Wenn der Schlüssel nicht existiert, wird eine Volltextsuche durchgeführt.

Da die Dokumente der Wikipedia sehr gut strukturiert sind, könnten viele weitere, interessante Anfragen realisiert werden. Darüber hinaus können viele einfache Anfragen ermöglicht werden, wenn viele Dokumente durch weitere Metadaten (z. B. Infoboxen) auf geeignete Weise erweitert werden. Metadaten sind Daten über Daten. So ist z. B. der "Spaltenname" LÄNGE (oder FLÄCHE) bereits in jedem FLUSS- (oder LÄNDER-) Dokument enthalten. Daher wäre es möglich, die 10 größten Flüsse der Welt oder Europas durch eine einfache Anfrage zu extrahieren, wenn entsprechende Anfragemöglichkeiten bestehen. Dies sind typische Datenbankanfragen. Neben Bildern nimmt strukturierter Text den größten Umfang in der Wikipedia ein. Das bedeutet, dass wir eine Sprache brauchen, die Datenbanken und Dokumentenanfragen benutzerfreundlich verbinden kann. Sie muss fein granulierte Selektionsmöglichkeiten besitzen und die Möglichkeit bieten, Teile von Dokumenten zu selektieren und diese Teile zu neuen Dokumenten zusammenzufügen. Auch enthält die Wikipedia eine erhebliche Menge an Zahldaten. Daher muss auch Möglichkeiten zum Rechnen bestehen. Da z.B. das Alter einer Person in der Regel nicht in der Wikipedia enthalten ist, sind Berechnungsmöglichkeiten notwendig. Im obigen Fall muss lediglich die Differenz zwischen dem Todesjahr bzw. dem aktuellen Datum und der Geburt realisiert werden.

Im Vergleich zu den anderen Dokumenten des Internets (HTML) ist die Wikipedia sehr gut strukturiert. Dennoch haben Wikipedia-Dokumente keinen so festen Typ wie XML-Dokumente. Es soll erreicht werden, dass die bestehenden und zukünftigen Dokumente weiter vereinheitlicht werden (geeignete Infoboxen definieren und Dokumente daran anpassen), um zu gewährleisten, dass Anfragen so einfach wie möglich formuliert werden können. Das bedeutet, dass neben der Vergrößerung der Quantität auch die Qualität im obigen Sinne verbessert werden sollte.

Der folgende TT stellt ungefähr entsprechende Teile der Struktur der deutschen Wikipedia dar. Diese Metadaten muss der Benutzer kennen oder zumindest zur Verfügung haben. Ansonsten ist er nicht in der Lage, Anfragen zu formulieren oder zu verstehen. Die Metadaten sind in deutscher Sprache verfasst, um entsprechende Anfragen leichter zu verstehen.

Ein kleiner aber ein wesentlicher Teil der Metadaten der deutschen Wikipedia.

```
WIKI! TITEL, (ANR, ATITEL, INHALT m), INFOBOX, INFO s
TITEL,ATITEL,INHALT! TEXT
ANR! ONR
BILD! BTITEL, JPG
URL, WIKILINK! WORT
TITEL ATITEL FETT KURSIV BTITEL! TEXT
Each infobox has its own scheme. We give some examples:
STAAT! EINWOHNER, FLAECHE, OFFIZIELLESPRACHE1,...
STADT! STAAT?, BUNDESLAND?, PROVINZ?, LANDKREIS?,
GEMEINDE?, HOEHE, FLAECHE, EINWOHNER, ...
FLUSS!          GEWAESSERKENNZEICHEN?,          LAGEm,          FLUSSSYSTEM?,
LAENGE?,EINZUGSGEBIET1,QUELLE,LINKERNEBENFLUSS,
RECHTERNEBENFLUSS1, GROSSSTAEDTE, MITTELSTAEDTE,...
LAENGE GROSSSTAEDTE MITTELSTAEDTE ...! TEXT
...
```

Aus der Sicht des Benutzers besteht Wikipedia nur aus einer einzigen Datei. Sie enthält jedoch viele verschiedene Objekte unterschiedlichen Typs. Die Namen der Typen stehen in der INFOBOX-Spalte. Für Elbe, Rhein und Weichsel, ... wird dort Fluss geschrieben. Eine Bedingung INFOBOX=Fluss wählt also alle Flüsse der Wikipedia aus. Bei der Übertragung in Datenbanken wäre dies eine einzige große Datei mit eigenen umfangreichen Metadaten wie LÄNGE, QUELLE, GROSSSTÄDTE,..... . Der Nachteil ist, dass alle größeren Städte eines Flusses in einem Feld zusammengefasst werden. Im Programm o+ +o muss man daher zunächst eine Wiederholungsgruppe GROSSSTADTI anlegen. Es wäre besser, wenn die Wikipediadatenbank hier bereits eine Wiederholgruppe vorsehen würde. Allein die Infobox FLUSS hat mehr als 20 Spaltennamen. Damit ist bereits klar, dass durch die vielen verschiedenen Infoboxnamen eine Unzahl von Spaltennamen existieren. Das erhöht die Komplexität und ist daher ein Nachteil. Allerdings können dadurch auch vielfältigere Anfragen realisiert werden, so dass Anfragen an Wikipedia nicht nur für Endnutzer, sondern auch für Spezialisten große neue Möglichkeiten eröffnen.

Ein weiteres Merkmal der oben genannten Metadaten ist, dass die strukturierten Texte nicht rekursiv gespeichert sind. Dies erleichtert Textanfragen.

Da unsere Implementation bisher nur auf der deutschen Wikipedia funktioniert, sind alle Beispiele in deutscher Sprache. Die folgende Beschreibung basiert auf Anfragen an einen Teil der Wikipedia mit etwa 28'000 Strupeln (strukturierte Tupel = komplexe Datensätze). Das Laden der Strupel hat etwa 60 Minuten gedauert. Im Allgemeinen gehen wir davon aus, dass wir die gesamte Wikipedia in den Arbeitsspeicher laden. Das Laden des oben erwähnten Teils der Wikipedia erforderte einen Gesamtspeicherverbrauch von 6.9 GByte. Wir rufen unseren Teil der deutschen Wikipedia mit dem Wort wiki oder 'aus wiki' auf.

s ist die Abkürzung für set. Im Gegensatz zu m ist für s kein eindeutiges Schema erforderlich.

Programm 11.1: Wie viele Strupel (strukturierte Tupel=Einträge) enthält unsere Miniwikipedia?
wiki ++1 '3
Ergebnis
28'819

Programm 11.2: Gesucht ist der erste Teil des Inhaltsverzeichnisses der Stadt Magdeburg.		
wiki keys [Magdeburg] gib TITEL,(ANR,ATITEL m) sel ANR <=12		
Ergebnis (tab):		
TITEL	,(ANR	,ATITEL m)
Magdeburg	0	Einleitung
	1	Geographie
	1.1	Schutzgebiete
	1.2	Klima
	1.3	Nachbargemeinden
	1.4	Stadtgliederung
	2	Geschichte
	2.1	Bedeutung und Herkunft des Namens
	2.2	Ur- und Frühgeschichte
	2.3	Mittelalter
	2.4	Frühe Neuzeit
	2.5	19. Jahrhundert

- 2.6 Weimarer Republik und Nationalsozialismus
- 2.7 Nachkriegs- und DDR-Zeit
- 2.8 1990 bis zur Gegenwart
- 2.9 1200. Stadtjubiläum
- 2.10 Eingemeindungen
- 2.11 Bevölkerungsentwicklung
- 3 Religionen
 - 3.1 Kirchengeschichte
 - 3.2 Evangelische Kirchen
 - 3.3 Römisch-katholische Kirche
 - 3.4 Zeugen Jehovas
- 4 Politik
 - 4.1 Stadtrat
 - 4.2 Oberbürgermeister seit 1808
 - 4.3 Wappen, Flagge und Dienstsiegel
 - 4.4 Städtepartnerschaften
 - 4.5 Stadtkampagne
- 5 Kultur und Sehenswürdigkeiten
 - 5.1 Bauwerke
 - 5.1.1 Sakralbauten
 - 5.1.1.1 Altstadt und alte Neustadt
 - 5.1.1.2 Außenbezirke
 - 5.1.2 Festungsanlagen
 - 5.1.3 Profanbauten und weitere Bauwerke
 - 5.1.4 Denkmäler und Skulpturen
 - 5.2 Friedhofsanlagen
 - 5.3 Plätze und Straßen
 - 5.4 Brunnen
 - 5.5 Brücken
 - 5.6 Museen
 - 5.7 Galerien
 - 5.8 Theater und Oper
 - 5.9 Kabarett
 - 5.10 Freizeit und Tourismus
 - 5.11 Zoologischer Garten
 - 5.12 Parks und Gärten
 - 5.13 Veranstaltungsorte
 - 5.13.1 Nachtleben
 - 5.14 Sport
 - 5.15 Regelmäßige Veranstaltungen
 - 5.15.1 Frühling
 - 5.15.2 Sommer
 - 5.15.3 Herbst
 - 5.15.4 Winter
 - 5.16 Musik
 - 5.17 Kulinarische Spezialitäten
 - 5.18 Große Einkaufsmöglichkeiten
 - 5.19 Stolpersteine
 - 5.20 Halbkugeln
- 6 Wirtschaft und Infrastruktur
 - 6.1 Industrie
 - 6.2 Verkehr
 - 6.2.1 Schienenverkehr
 - 6.2.2 Straßenverkehr
 - 6.2.2.1 Straßennamen

6.2.3	Öffentlicher Personennahverkehr
6.2.4	Fahrradverkehr
6.2.5	Schifffahrt
6.2.6	Flugverkehr
6.3	Medien
6.4	Öffentliche Einrichtungen
6.5	Bildung und Forschung
6.5.1	Hochschulen
6.5.2	Weitere Wissenschaftseinrichtungen und Institute
6.5.3	Projekte und Veranstaltungen
6.5.4	Bibliotheken und Archive
6.5.5	Schulwesen
6.6	Gesundheitswesen
7	Garnison Magdeburg
8	Persönlichkeiten
8.1	Ehrenbürger
8.2	Söhne und Töchter der Stadt
8.3	Persönlichkeiten der Stadt
8.4	Magdeburger Originale
9	In Kunst und Kultur
10	Filme
11	Siehe auch
12	Literatur
Programm 11.3 Gib Auskunft über das Nachtleben von Magdeburg.	
wiki	
keys [Magdeburg]	
sel ATITEL=Nachtleben	
gib INHALT	
Ergebnis (web)	
Magdeburgs Nachtleben setzt sich neben Live-Konzerten vorwiegend aus Tanzveranstaltungen in größeren Diskotheken und kleineren Clubs zusammen. Markant für Magdeburg ist, dass viele dieser Veranstaltungsorte in ehemaligen Festungs- und seit der Wende leerstehenden Industrieanlagen lokalisiert sind. Einige größere Diskotheken sind die Festung Mark, die neben elektronischen Musikveranstaltungen auch für kulturelle Veranstaltungen sorgt, und das "Alte Theater" am Jerichower Platz. Ebenfalls ein Industriegefühl bietet die ehemalige Fabrikhalle "Factory" im Süden der Stadt, in der regelmäßig deutsche und internationale Pop-, Rock-, Metal-, Indie-Bands spielen und Diskoveranstaltungen stattfinden. Zu den edleren Clubs der Stadt gehören der "Prinzzclub" und das "First", welche einen Mix aus Lounge und Club bieten. Mit 45 Jahren ist der "Studentenclub Baracke" ältester Club der Stadt und befindet sich direkt auf dem Gelände der Otto-von-Guericke-Universität. Als Äquivalent besteht am Campus des Universitätsklinikums die "Kiste" für Studenten der Medizinischen Fakultät. Daneben existieren weitere mittlere und kleinere Diskotheken und Clubs, wie das "Boys'n'Beats", die "Alte Feuerwache", die "Kunstkantine" oder das "Triebwerk". Zu erwähnen sind außerdem noch die einem Strand nachempfundenen Clubs "Strandbar" direkt an der Elbe, mit einem der ersten Citybeachkonzepte Deutschlands, und der "Montego Beachclub" im Stadtpark Rotehorn mit Volleyballfeldern und großem Pool. In den Jahren 2016 und 2017 wurden einige Diskotheken geschlossen. So befand sich im Süden der Stadt bis 2016 die Großraumdiskothek "Music Hall", der ehemalige "Funpark", welche neben den Mainstream-Genres auch spezielle Musikrichtungen bediente. Außerdem wurde 2016 der "Discoturm Nautica (Pearl Club)" geschlossen, nachdem das Spaßbad nach Insolvenz den Betreiber gewechselt hat und umgebaut wurde. Anfang 2017 wurde schließlich das "Kulturwerk Fichte", eine denkmalgeschützte Industriehalle aus den Gründerzeiten, wo Szenepartys und andere Großveranstaltungen stattfanden geschlossen[[Ref]][[Ref]][[Ref]]. Der	

Hasselbachplatz an der südlichen Innenstadt hat sich in den letzten Jahren zum Magdeburger Kneipenzentrum entwickelt. Auf Grund der hohen Frequentierung tagsüber, aber vor allem in den Abendstunden durch die Besucher der zahlreichen Clubs, Bars und Kneipen ist der Platz als Kriminalitätsschwerpunkt eingestuft und wird mittels Videotechnik überwacht.

Programm 11.4 wie Anfrage 11.3 nur etwas effizienter

```
wiki
keys [Magdeburg, ["5.13.1"]]
gib INHALT
```

Ergebnis

gleiches Ergebnis wie 11.3

Programm 11.5 Gib den Abschnitt 5.15 von Magdeburg mit allen Unterabschnitten beliebiger Tiefe aus.

```
wiki
keys [Magdeburg]
sel ANR like "5.15*"
gib ANR, ATITEL, INHALT m
```

Ergebnis (web)

ANR	ATITEL	INHALT
5.15	Regelmäßige Veranstaltungen	Die " Magdeburger Frühjahrsmesse", ein dreiwöchiger Rummel zum Frühlingsanfang, findet jährlich auf dem Messeplatz „Max Wille“ am Kleinen Stadtmarsch direkt am Elbufer statt. Seit 2010 werden im März/April die RoboCup German Open in den Magdeburger Messehallen ausgetragen [[Ref]]. Tausende Besucher verfolgen in verschiedenen Disziplinen, unter anderem Roboterfußball, internationale Teams, die mit ihren Robotern gegeneinander antreten. Jährlich findet seit 2000 der "Day of Thunder" auf dem Magdeburger Flugplatz statt. In 15 verschiedenen Rennklassen treten unterschiedliche Bauarten von Mopeds, Motorrädern, Autos und Quads bei einem 1/8-
5.15.1	Frühling	Meilen-Rennen gegeneinander an. Außerdem finden auf einer Showbühne Wettbewerbe wie „Best of Show“, „schönstes Gesamtkonzept“, „Beste Lackierung“, „Bester Innenraum“, „Best of Sound“, „Best of Exhaust“ und ein „dB-Sound-Contest“ statt. Das Magdeburger Historienspektakel Spectaculum Magdeburgense im Mai im Bereich der alten Festungsanlagen ist eine Mittelalterveranstaltung. Zahlreiche Veranstaltungen und Aktionen, darunter zum Beispiel Fakirshows, Theaterveranstaltungen, ein Mittelaltermarkt und musikalische Klänge aus dieser Zeit, unterhalten die Besucher. Jedes Jahr an Christi Himmelfahrt findet das "Fest der Begegnungen" gegen Fremdenfeindlichkeit im Rotehornpark statt.
5.15.2	Sommer	Neben den Kreuzgangserenaden im Dom findet im Juli/August das überregional bekannte Sommer-Open-Air des Theater Magdeburg auf dem Magdeburger Domplatz statt. Die "BallonMagie"-Tage im Elbauenpark werden jährlich im August veranstaltet. Mehrere Heißluftballons starten gleichzeitig und bereichern den Magdeburger

	Himmel. Sonderformen wie Eistüten, Würstchendosen oder Luftschiffe sind unter den Ballons vertreten. Der Christopher Street Day, ebenfalls im August, ist ein Festtag von Lesben, Schwulen, Bisexuellen und Transgendern. Demonstriert wird für die Rechte und gegen die Ausgrenzung dieser Gruppen. Er findet in zahlreichen Städten Deutschlands statt. Die Parade erstreckt sich vom Bahnhof Neustadt durch die Innenstadt, den Hasselbachplatz in die Liebigstraße. Rund um das historische Viertel am Magdeburger Dom findet seit 2011 jährlich das dreitägige "Kaiser Otto Fest" statt, bei dem Gebäude und Plätze wie die Bastion Cleve, das Kloster Unser Lieben Frauen, der Möllenvogteigarten, der Fürstenwall oder der Dom selber Veranstaltungsorte mittelalterlicher Attraktionen, Auftritte und Festumzüge wie zum Beispiel die Kaiserkrönung Ottos I., Ritterturniere, Falknershows und mittelalterliche Gesänge werden. Das Fest soll an die Bedeutung der Stadt Magdeburg als Wiege der deutschen Nation und der europäischen Geschichte erinnern. Einmal jährlich findet der Fahrradaktionstag statt. Nach einer Sternfahrt zum Sammelpunkt führt eine große Fahrraddemonstration quer durch die Stadt und über den Magdeburger Ring. Damit wollen die Radfahrer Flagge zeigen und für eine fahrradfreundlichere Stadt eintreten[[Ref]]. 2014 fand er am 28.&#160;Juni zum vierten Mal statt. Im September wird in Magdeburg das Landeserntedankfest, mit über 35.000&#160;[[Ref]];Besuchern die größte öffentliche Veranstaltung des landwirtschaftlichen Berufsstandes in Sachsen-Anhalt, im Elbauenpark gefeiert. Dazu kommen das Jazzfestival DIAGONALE, die Literaturwochen, eine Veranstaltung für Literaturfreunde mit vielen Angeboten und Ausstellungen, Vorlesungen und Aufführungen, das Kunstfestival Magdeburg, das OMMMA ("Ostmobil-Meeting Magdeburg") und die "Magdeburger Herbstmesse" (früher "Herrenmesse"), ein zum Herbstanfang stattfindender dreiwöchiger Rummel auf dem „Kleinen Stadtmarsch“. Im Jahr 2010 feierte sie ihr 1000-jähriges Bestehen, denn sie findet im heiligen Fest der Thebäischen Legion von Erzbischof Tagino, welches am 22.&#160;September 1010 gefeiert wurde, ihren Ursprung. Ab dem Jahr 1220 verschmolz das Fest des Mauritius und seiner heiligen Mitstreiter mit dem großen Magdeburger Jahrmarkt, welcher seiner Zeit noch auf dem Domplatz stattfand. Somit ist die Herbstmesse Magdeburg heute das älteste Volksfest Deutschlands[[Ref]].
5.15.3 Herbst	
5.15.4 Winter	Die größte Veranstaltung des Jahres ist der Magdeburger Weihnachtsmarkt mit rund 135&#160;Ständen. Er lockt jedes Jahr über 1,5&#160;Millionen Besucher an, wird auf dem Alten Markt ausgetragen und bietet viele Attraktionen, zum Beispiel tägliche Live-Musik, eine Weihnachtsmannsprechstunde, Märchenaufführungen und den historischen Weihnachtsmarkt. Er gilt als einer der kinderfreundlichsten Weihnachtsmärkte Deutschlands und ist der täglich am längsten geöffnete Weihnachtsmarkt in Deutschland[[Ref]]. Im Januar findet jährlich die sogenannte "Meile der Demokratie" mit über 10.000 Besuchern statt, wobei der Breite Weg bis zum Hasselbachplatz Austragungsort mit zahlreichen Aktionen, Infoständen, Gesprächsstunden und umfangreichem Bühnenprogramm für diese Veranstaltung ist. Sie entstand um dem zeitgleich stattfindenden Aufmarsch von Rechtsextremen den Raum zu nehmen. Diese nutzen

den Jahrestag der Luftangriffe auf Magdeburg am 16. Januar 1945 als Anlass für einen Trauerzug, und um die Opfer mit dem Holocaust und den Ermordeten in den Konzentrations- und Vernichtungslagern gleichzusetzen und somit die nationalsozialistischen Massenmorde zu verharmlosen.

Programm 11.6 Gib die zusammenfassenden Informationen von drei großen Flüssen.

wiki

keys [Mekong Jangtsekiang Amazonas]

sel ANR! ANR=0

gib TITEL,LAENGE,INHALT m-

Ergebnis (web)

TITEL	LAENGE	INHALT
Mekong	4350	Der "'Mekong"' ([] oder []) ist ein Strom in Südostasien, der sechs Länder durchquert. Seine Länge wird mit 4350 km bis 4909 km angegeben. Damit ist er einer der zwölf längsten Flüsse der Erde.
Jangtsekiang	6380	Der "'Jangtsekiang"', kurz "'Jangtse"' (, kurz:), ist der längste Fluss Chinas. Mit 6380 Kilometern, von denen 2800 Kilometer schiffbar sind, ist er auch der längste Fluss Asiens und nach dem Nil und dem Amazonas der drittlängste Strom der Welt. Sein Quellgebiet liegt im Hochland von Tibet in Qinghai. An seiner Mündung ins Ostchinesische Meer führt er im Jahresdurchschnitt 31.900 m ³ Wasser pro Sekunde. Der Jangtsekiang spielt im Selbstverständnis der Chinesen eine große Rolle. Er teilt das Land in Nord- und Südchina und war Ort zahlreicher wichtiger Ereignisse der chinesischen Geschichte. Dazu zählt etwa seine Überquerung durch die Volksbefreiungsarmee während des Chinesischen Bürgerkrieges am 21. April 1949 und das bis Mitte des 20. Jahrhunderts bestehende Recht westlicher Mächte, den Fluss mit Kanonenbooten zu befahren.
Amazonas	6992	Der "'Amazonas"' (auch , , in Brasilien oberhalb der Einmündung des Rio Negro bei Manaus [[Ref]], früher) ist ein Strom im nördlichen Südamerika. Etwa 300 km südlich des Äquators durchquert er das im Westen von den Anden umrahmte, von tropischem Regenwald geprägte Amazonasbecken ostwärts bis zum Atlantik. Der Amazonas ist mit einer mittleren Wasserführung von 206.000 m ³ /s der mit Abstand wasserreichste Fluss der Erde und führt an der Mündung mehr Wasser als die sechs nächstkleineren Flüsse zusammen und ca. 70-mal mehr als der Rhein[[Ref]]. Seinen Namen führt der Strom erst ab dem Zusammentreffen seiner beiden Quellflüsse Marañón und Ucayali in Peru, unterbrochen allerdings durch den brasilianischen Abschnitt oberhalb der Stadt Manaus mit dem Namen "Rio Solimões". Der in Brasilien meist mehrere Kilometer breite Fluss hat eine relativ ausgeglichene Wasserführung, da die Hochwasserphasen der Nebenflüsse jahreszeitlich verschoben auf den äquatornahen Hauptstrom treffen. Dennoch kann er die angrenzenden bewaldeten Alluvialflächen ("Várzea") auf einer Breite von bis zu 60 km überschwemmen. In zwei Hauptarmen durchströmt er die Inselwelt des fast 200 km breiten Mündungsbereichs, der zudem über Tidengewässer mit dem Pará-Ästuar verbunden ist und so die große Insel Marajó abtrennt.

Die nächste Anfrage identifiziert alle mittleren und großen Städte, die an der Spree liegen. Das Anfrageergebnis kommt relativ schnell, da nicht alle 30'000 Datensätze durchsucht werden müssen. Das liegt daran, dass hier Schlüssel und nicht die allgemeine sel-Operation verwendet wird. Über die Titel der Einträge hat man direkten Zugriff auf die einzelnen Strupel. Wenn die Anfrageoptimierung realisiert ist, sollte keys wieder aus der Benutzerebene verschwinden und durch sel ersetzt werden.

Die beiden Zuweisungen für STADTI sind notwendig, weil Wikipedia die großen und mittleren Städte als kommagetrennte Listen speichert. Das bedeutet, dass ein GROSSSTAEDTE-Eintrag eine Liste von STADT ergibt. Die Wikipedia basiert auf dem relationalen Datenbankmanagementsystem MariaDB. Diese Systeme haben allgemeine Probleme mit Listen (Wiederholgruppen):

Programm 11.7: Gesucht sind alle Groß- und Mittelstädte an der Spree in einer Spalte.

```
wiki
keys [Spree]
STADTI:=GROSSSTAEDTE split “,” at GROSSSTAEDTE
STADTI:=MITTELSTAEDTE split “,” at MITTELSTAEDTE
gib TITEL,STADTI
```

Ergebnis (tabh)

TITEL, STADTI

Spree Cottbus Berlin Bautzen Spremberg Fürstenwalde/Spree

Programm 11.8: Ermittle die Höhen über dem Meeresspiegel von drei Städten.

```
wiki
keys Berlin Stuttgart Heidelberg
gib TITEL,HOEHE 1
```

Ergebnis (tab)

TITEL, HOEHE 1

Heidelberg 114

Stuttgart 247

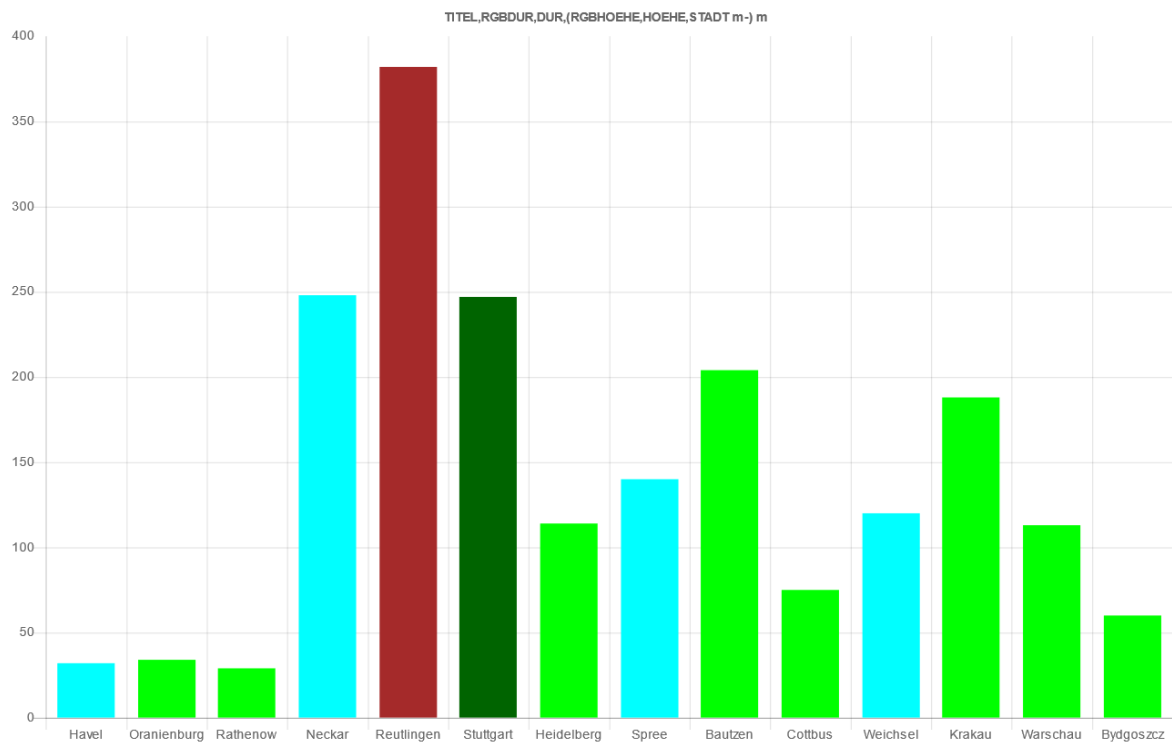
Aus obiger Anfrage kann man eine wichtige Lehre ziehen. Auch wenn man eine Anfragesprache beherrscht, bekommt man nicht immer die gewünschten Informationen. Da obige Miniwiki keinen Eintrag für Berlin enthält, kann Berlin nicht im Ergebnis dieser Anfragesprache vorkommen.

Programm 11.9: Ermittle für 4 Flüsse die anliegenden Städte mit ihren Höhen über dem Meeresspiegel. Für jeden Fluss ist die Durchschnittshöhe der Städte über NN zu bestimmen.

```
wiki
keys [Neckar Havel Spree Weichsel]
STADTI:=GROSSSTAEDTE split “,” at GROSSSTAEDTE
STADTI:=MITTELSTAEDTE split “,” at MITTELSTAEDTE
gib TITEL,STADTI m
=: $STAEDTE
aus $STAEDTE
gib STADTM
=: $STADTM
aus wiki
keys $STADTM
gib TITEL,HOEHE? 1
rename TITEL!STADT
, $STAEDTE
gibjoin TITEL,(STADT,HOEHE m)m
HOEHE::=HOEHE nthzahl 1
```

```
gib TITEL,DUR,(HOEHE,STADT m-) m DUR:=HOEHE!++:
rnd 0
```

Ergebnis (Diagramm-Säulen)



Ergebnis (tab)

TITEL	DUR	HOEHE	STADT
Havel	32.	34	Oranienburg
		29	Rathenow
Neckar	248.	382	Reutlingen
		247	Stuttgart
		114	Heidelberg
Spree	140.	204	Bautzen
		75	Cottbus
Weichsel	120.	188	Krakau
		113	Warschau
		60	Bydgoszcz

Die Farben im Säulendiagramm sind nutzerdefiniert. Lediglich zwei zusätzliche Zuweisungen waren erforderlich:

RGBDUR :=cyan leftat DUR

RGBHOEHE:=brown if HOEHE>300 !

darkgreen if HOEHE>220 !

green leftat HOEHE

12 Spezielle Umstrukturierungsoperationen (onrs)

Die Operation onrs wurde eingeführt, um o++o-Nummern für die Lösung von Stücklistenproblemen bereitzustellen. Das gegebene Tabment muss vom Typ

$X_1, \dots, X_n, (Y_1, \dots, Y_k \text{ m}) \text{ m}$

, wobei X_1 und Y_1 die Schlüssel der jeweiligen Kollektionen sind. Wir setzen voraus, dass beide Kollektionen Mengen sind. Dadurch haben wir im Arbeitsspeicher direkten Zugriff auf die entsprechenden Tupel bzw. Subtupel. Wir betrachten ein Beispiel:

Programm 12.1: Bilde die Stückliste für einen Minigolf.

```
<TAB!
OT,      EIGENSCHAFT,(UT,      ANZAHL m) m
Buchse   zylindrisch
Felge    glatt
Polo     modern      Rad      4
                        Motor    1
                        Karosse  1
Golf     schnell     Rad      4
                        Karosse  1
                        Klimaanlage 1
                        Motor    1
Kolben   leicht     KolbRing 2
                        Buchse   1
Motor    schwer     Kolben   6
                        Schraube 8
Rad      rund       Schraube 5
                        Reifen   1
                        Felge    1

!TAB>
onrs OTTONR ! Golf
```

Ergebnis (tab)

OT	EIGENSCHAFT	(OTTONR,UT,	ANZAHL m) m
Golf	schnell	1	Rad 4
		1.1	Schraube 5
		1.2	Reifen 1
		1.3	Felge 1
		2	Motor 1
		2.1	Schraube 8
		2.2	Kolben 6
		2.2.1	KolbRing 2
		2.2.2	Buchse 1
		3	Klimaanl 1
		4	Karosse 1

Man erkennt, dass allen direkten Unterteilen vom Golf eine otto-Nummer, die nur aus einer Zahl besteht, zugeordnet wird. Der Motor ist ein solches Teil. Die direkten Unterteile vom Motor (Schraube und Kolben) erhalten otto-Nummern mit zwei Zahlen. Analog erhalten die direkten Unterteile vom Kolben otto-Nummern mit 3 Zahlen. Für den Golf wird also eine nichtrekursive Menge ohne Redundanz gebildet. Auf diese Menge könnte jetzt die otto-Rekursion angewandt werden, um die Vielfachheit des Enthaltenseins eines Unterteils zu berechnen.

Neben dem Inputtabment benötigt onrs noch den Namen der ONR-Spalte (hier OTTONR) und ein oder mehrere Teile (hier nur Golf) für die die ONR-Auflösung vorgenommen werden soll. Demnach wäre auch die Programmzeile

onrs STRUK_NR ! [Polo Golf]

korrekt und sinnvoll.

13 Textverarbeitung mit o++o (+ - -+ cut satzl)

Mit dem Symbol + kann auch Text verknüpft und manipuliert werden. Auch hier wird ein kleiner Unterschied zwischen TEXT und WORT gemacht. Beispiel:

Programm 13.1: Es gibt kleine Unterschiede zwischen WORT- und TEXT-Konkatenation
WORTERGEBNIS:=otto + " o++o"
TEXTERGEBNIS:=otto text + " o++o"
Ergebnis (tab)
WORTERGEBNIS, TEXTERGEBNIS
otto_o++o otto o++o

Da der erste Eingabewert von WORTERGEBNIS ein Wort ist, ist das Ergebnis auch vom Typ WORT. Analoges gilt für den zweiten Fall, in dem das Ergebnis ein Text ist. Wörter können keine Leerzeichen enthalten.

-+ ist eine Operation mit 3 Eingabewerten

Der TT des ersten Eingabewertes bleibt erhalten. Jedes Vorkommen des zweiten Eingabewertes wird durch den dritten ersetzt.

Programm 13.2: -+text
<TAB! X, Y 1 1 Heute ist Montag. 2 Gestern ist Sonntag. !TAB> -+text "ist S" ! "war S"
Ergebnis (tab)
X,Y 1
1 Heute ist Montag. 2 Gestern war Sonntag.

Programm 13.3:
TEXTPLUS:="Heute ist ein schöner " + Tag TEXTMINUS:=Donnmmerwetter - "m" TEXTMINUSPLUS:="Heute ist ein schoener Tag." -+text oe ! ö
Ergebnis (tab)
TEXTPLUS, TEXTMINUS , TEXTMINUSPLUS
Heute ist ein schöner Tag Donnerwetter Heute ist ein schöner Tag.

Eine zweistellige Funktion cut, die alle Buchstaben aus einem Tabment extrahiert, wurde ebenfalls implementiert. Eine analoge Funktion für Sätze (satzl) wurde implementiert, die bisher eine relativ rudimentäre Satzende-Erkennung verwendet.

Programm 13.4: "Codiere" einen Text.
"Heute ist Dienstag. Morgen ist Mittwoch." cut 1 sel- WORT inmath ["a" "e" "i" "o" "u"] WORT:="t" if WORT="m"! "m" if WORT="t"! WORT;; ++text
Ergebnis (tab)

TEXT
Hm sm Dnsmg. Mrgn sm Mmmwch.

Die Operation cil ist besonders nützlich, um Anfragen an Dokumente zu stellen, deren Schema dem Benutzer nicht vollständig bekannt ist.

Mit dem Begriff "sel Schulze zurück" kann er alle Elemente einer Kollektion auswählen, die das Wort "Schulze" und das Wort "zurück" enthalten, ohne dass er die Attributnamen im Detail kennen muss. Wenn keine Auswahlebene angegeben wird, wird immer in den obersten Kollektionen ausgewählt.

Programm 13.5: Illustration von "in" in Texten
<TAB! NR,ABSCHNITT 1 1 Heute ist S. Schulze in Hongkong. Morgen ist S. Schulze in Beijing. 2 S. Schulze ist zurück. 3 S. Meier lebt in Magdeburg. !TAB> sel (Schulze zurück) in ABSCHNITT
Ergebnis (tab)
NR,ABSCHNITT 1
2 S. Schulze ist zurück.

Programm 13.6: Eliminiere aus Zahlen einer Liste jeweils den Punkt.
X1:= 3 hoch (1.1 ..15) Y := X cut 1 sel- WORT="." Y::=Y ++text
Ergebnis (tab)
X, Y 1
3.3483695221 33483695221
10.0451085663 100451085663
30.1353256989 301353256989
90.4059770967 904059770967
271.21793129 27121793129
813.653793871 813653793871
2440.96138161 244096138161
7322.88414484 732288414484
21968.6524345 219686524345
65905.9573035 659059573035
197717.871911 197717871911
593153.615732 593153615732
1779460.8472 17794608472
5338382.54159 533838254159

14 Formatieren mit o++o ('3 '4 norm3e norm3m mant rnd)

Analog zu SQL hatte sich o++o zunächst auf inhaltliche Probleme beschränkt. Allerdings benutzt o++o wesentlich reichere Strukturen als SQL. Formatierungen wurden dann von SVG übernommen. Dadurch konnte man eine Tabelle rahmen, Buchstaben fett oder farbig schreiben, Jetzt haben wir weitere Möglichkeiten umgesetzt. Zahlen mit einer größeren Mantisse sind schwer lesbar, wenn sie nicht gruppiert werden. Da für Zahlendarstellungen in der Welt viele verschiedene Varianten benutzt werden, haben wir Darstellungen gewählt, die möglichst nicht mit den existierenden kollidieren und trotzdem noch besser lesbar sind.

Gruppieren von Ziffernfolgen ('3 '4)

Nach Schweizer Vorbild verwendet o++o das Hochkomma um Zahlen besser lesbar zu gestalten. Dreierblöcke sind neben Viererblöcken die wichtigsten.

Programm 14.1: Verbessere die Lesbarkeit mehrerer Zahlen durch Gruppierung.		
12345678, 1234567.87654 '3 ;1234567890 '4		
Ergebnis (tab)		
ZAHL,	PZAHL,	ZAHL
12 '345 '678	1 '234 '567.876 '54	12 '3456 '7890

Solche Darstellungen werden durch die (einstelligen) Operationen '3 und '4 erzeugt. Der Nutzer kann das Hochkomma auch beliebig setzen, um beispielsweise Telefonnummern besser lesbar zu gestalten:

0176'84'208'408

Operationen, die dergleichen erzeugen, sind zu kompliziert, so dass o++o sie nicht erzeugen kann.

International wird sowohl das Komma als auch der Punkt als Dezimaltrenner benutzt. Dadurch wird auch der Punkt und das Komma zum Gruppieren verwendet. Wir hoffen diese Uneinheitlichkeit durch diese Vereinbarungen zu beseitigen.

Exponent zuerst Schreibweise (norm3m) und norm3e

PZAHLen-Zahlen mit langer Mantisse sind nicht schnell genug zu erfassen, da der wesentlichere Exponent erst am Ende der Zeichenfolge angegeben wird. Ferner denken die Menschen in Tausender, Millionen, Milliarden, Ein Exponent 7 oder 8 muss als erst 10- bzw. 100-Millionen erkannt werden. Diese Denkweise reflektiert o++o, indem nur Vielfache von 3 als Exponent zugelassen werden. Ferner können die Exponenten auch zuerst angegeben werden:

6m12.345 (12 Millionen ...)

9m123.4 (123 Milliarden ...)

die alte Mantisse zuerst Schreibweise kennt o++o dennoch. Allerdings werden hier auch nur Vielfache von 3 als Exponent benutzt:

12.3456789e6 (12 Millionen ...)

123.456789e9 (123 Milliarden ...)

Diese Formatierungen können durch die einstelligen (unären) Operationen norm3m (für die Darstellung mit m) und norm3e (für die letztere) erzeugt werden. Die 10 drückt die dezimale Schreibweise der Zahlen aus.

Programm 14.2: Verbessere die Lesbarkeit von Zahlen durch Normierung der Exponenten.	
12345678.9 norm3m;12345678.9 norm3e	
Ergebnis (tab)	
PZAHL	, PZAHL
6m12.3456789	12.3456789e6

Die Reduktion der Ziffern (mant)

Die meisten Menschen interessieren sich nicht für die vielen Dezimalstellen, wenn auf einem Taschenrechner die Wurzel aus 2 oder 3 mit mehr als 10 Stellen ausgegeben werden. Die Überhäufung mit irrelevanten Informationen lässt uns das Wesentliche schwerer erkennen. Daher ist ein Weglassen unnötiger Stellen (Informationen) wichtig.

Die zweistellige Funktion mant realisiert dies und wandelt das Ergebnis gleich in die m-Darstellung um. D.h. die Operation norm3m wird gleich mit angewandt. Das zweite Argument von mant gibt die Anzahl der gewünschten Ziffern an. Beispiel:

Programm 14.3: Reduziere die Anzahl der Stellen auf vier.	
12345678.98765 , 1234567890	
mant 4	
Ergebnis (web)	
PZAHL	ZAHL
6m12.34	9m1.234

Man erkennt bereits an diesem Beispiel, dass die Funktion mant gerade für das Handy wichtig ist. Dadurch werden mehr Spalten einer Ergebnistabelle gleichzeitig auf dem Display sichtbar.

15 Strukturierte Diagramme

Mit o++o kann man auf einfache Weise Diagramme erstellen. Wenn man ein o++o-Programm erstellt haben, kann man mit dem Diagramm-Button ein neues Browser-Fenster öffnen, das eine Auswahl an verschiedenen Diagrammtypen bietet. Säulendiagramme sind sicherlich die am häufigsten verwendeten. Für Diagramme gelten die folgenden Regeln:

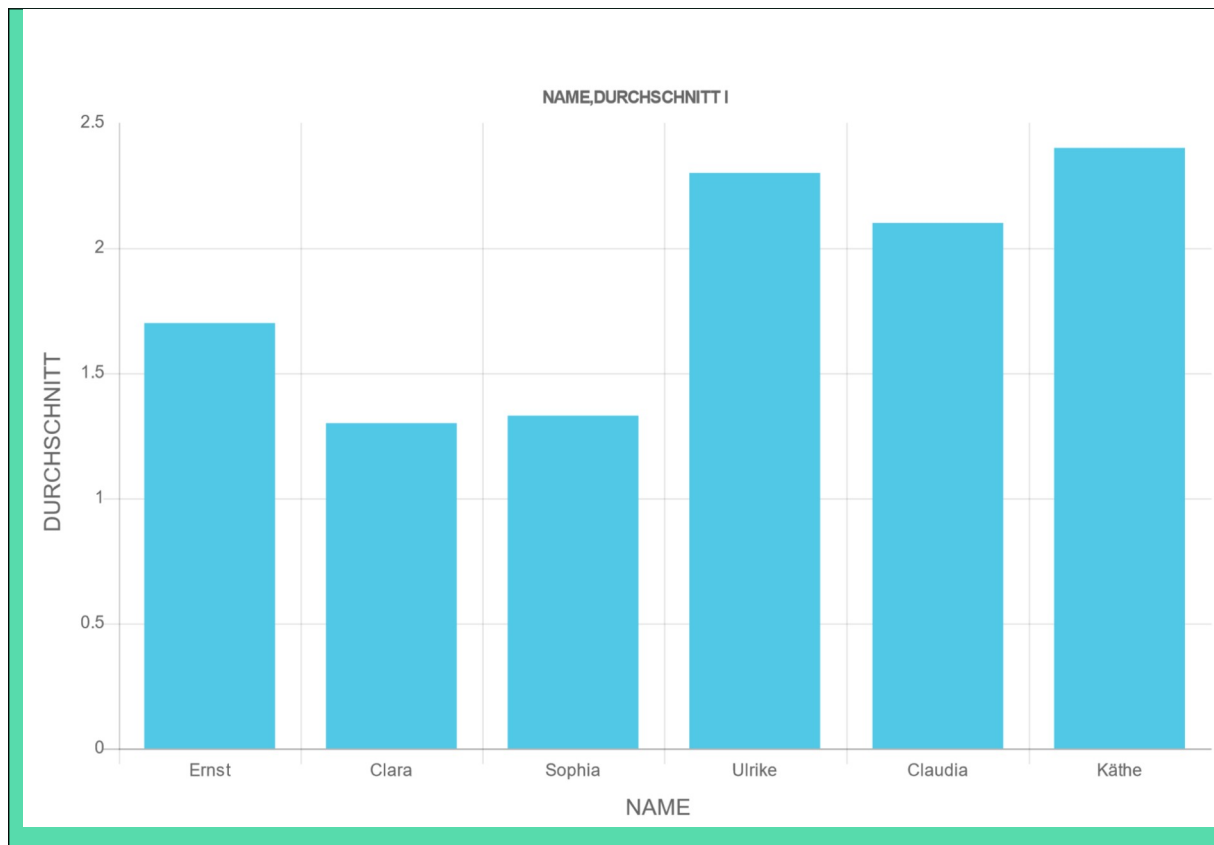
1. TEXTe werden vom System in Wörter umgewandelt, indem jedes Leerzeichen durch einen Unterstrich ersetzt wird.
2. Numerische Spalten (ZAHL, PZAHL, RATIO) werden als Säulen dargestellt.
3. Die erste Wort-Spalte jeder Hierarchieebene wird als Unterschrift für die Säulen benutzt. Die weiteren Wortspalten der Ebene werden ignoriert. Existiert keine Wortspalte, so fungiert ein Strich als Unterschrift.
4. Sind keine RGB-Werte vorgegeben, so setzt das System Standardfarben. Wenn der Nutzer die Farben wählen will, muss jede numerische Spalte eine RGB-Spalte in derselben Ebene oder übergeordnet besitzen. Wenn ein RGB-Wert direkt vor einer Zahlspalte steht, bestimmt er die Farbe der Spalte.
5. Wenn die anzuzeigende Tabelle mit dem Spaltennamen TITEL beginnt, wird der Inhalt der Spalte als Überschrift des gesamten Diagramms interpretiert.

Wir wissen bereits, dass eine einfache Liste von Zahlen ein o++o-Programm ist, das als Diagramm dargestellt werden kann. Wenn in jeder Zeile noch ein Wort steht, dient es als Unterschrift:

Programm 15.1: Erzeuge ein Säulendiagramm mit Unterschriften

```
<TAB!  
NAME,    DURCHSCHNITT 1  
Ernst    1.7  
Clara    1.3  
Sophia   1.33  
Ulrike   2.3  
Claudia  2.1  
Käthe    2.4  
!TAB>
```

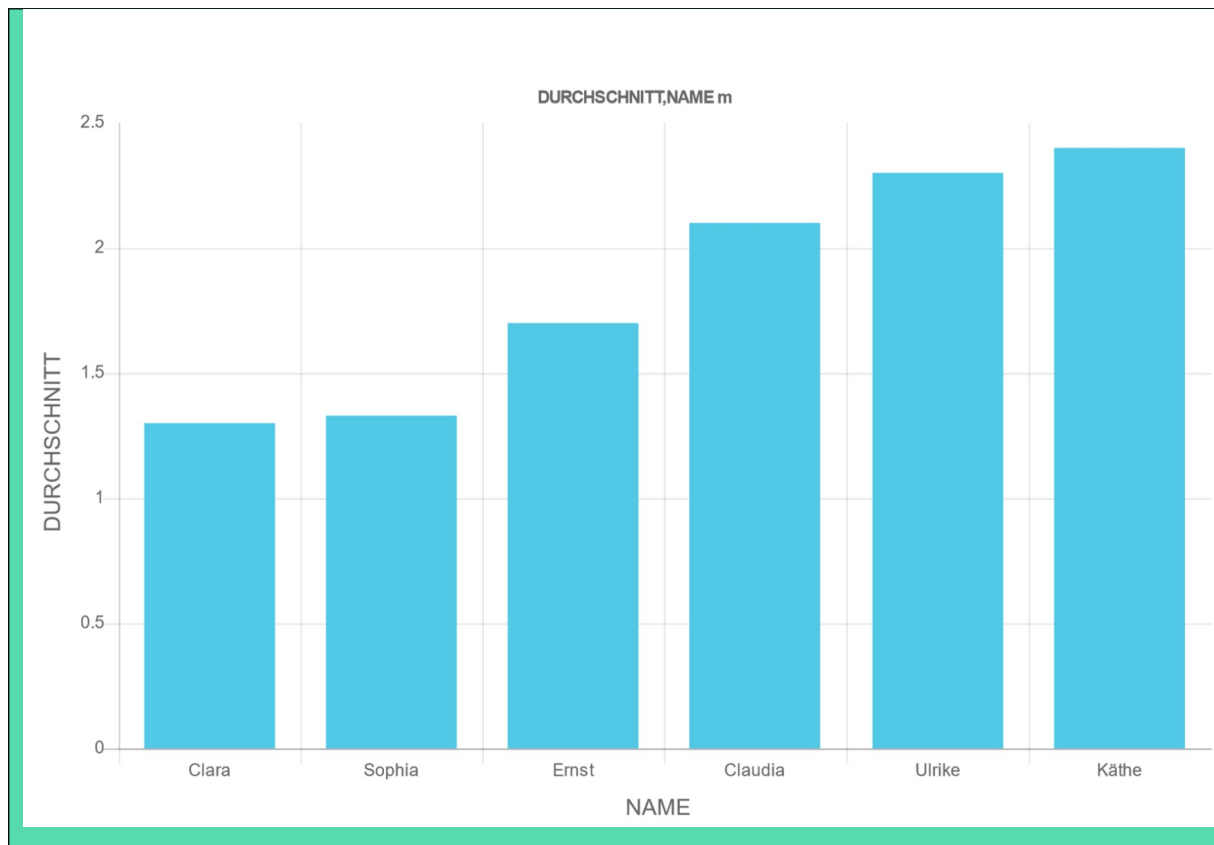
Ergebnis (struc. diagram- bar)



Programm 15.2: Sortiere die Säulen mit Unterschriften der Größe nach

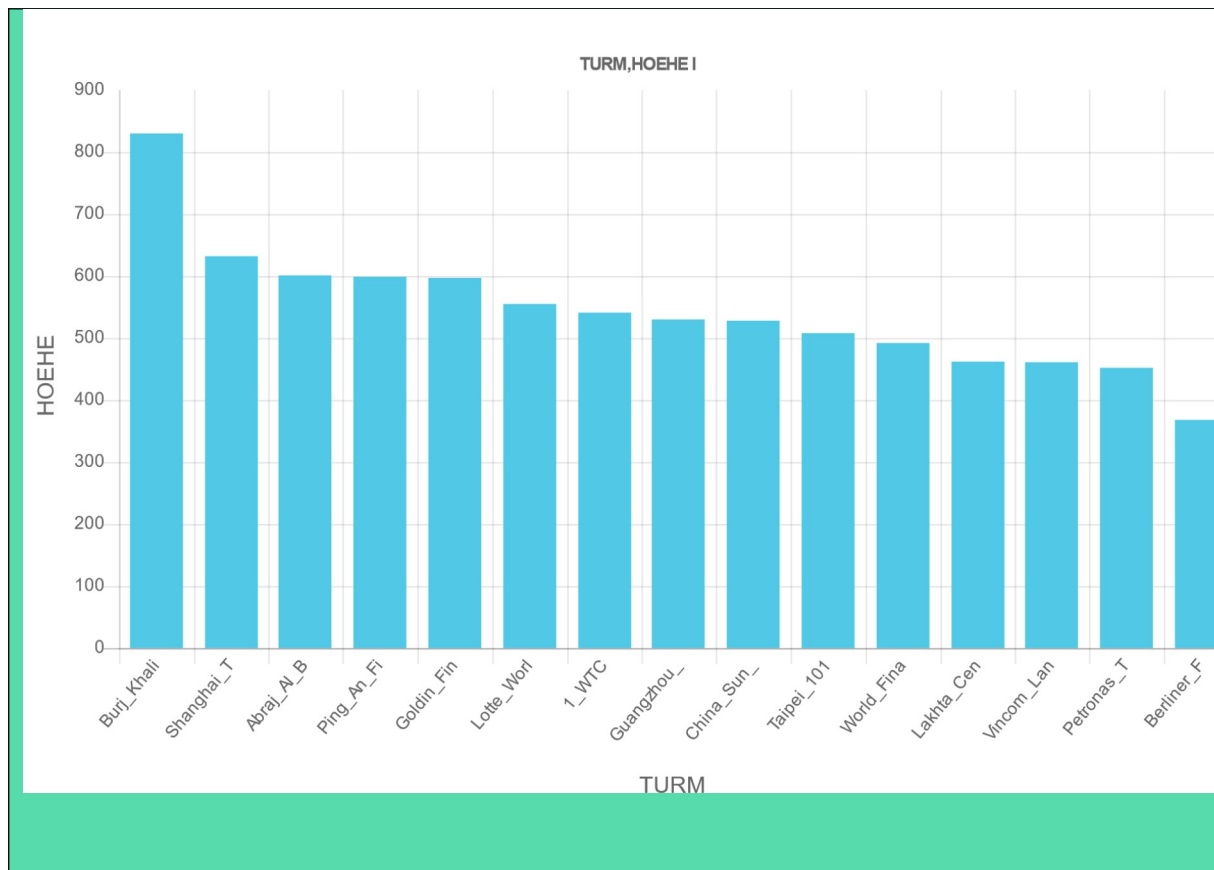
```
<TAB!
NAME,   DURCHSCHNITT 1
Ernst   1.7
Clara   1.3
Sophia  1.33
Ulrike  2.3
Claudia 2.1
Käthe   2.4
!TAB>
gib DURCHSCHNITT,NAME m
```

Ergebnis (struc diagram - bar)



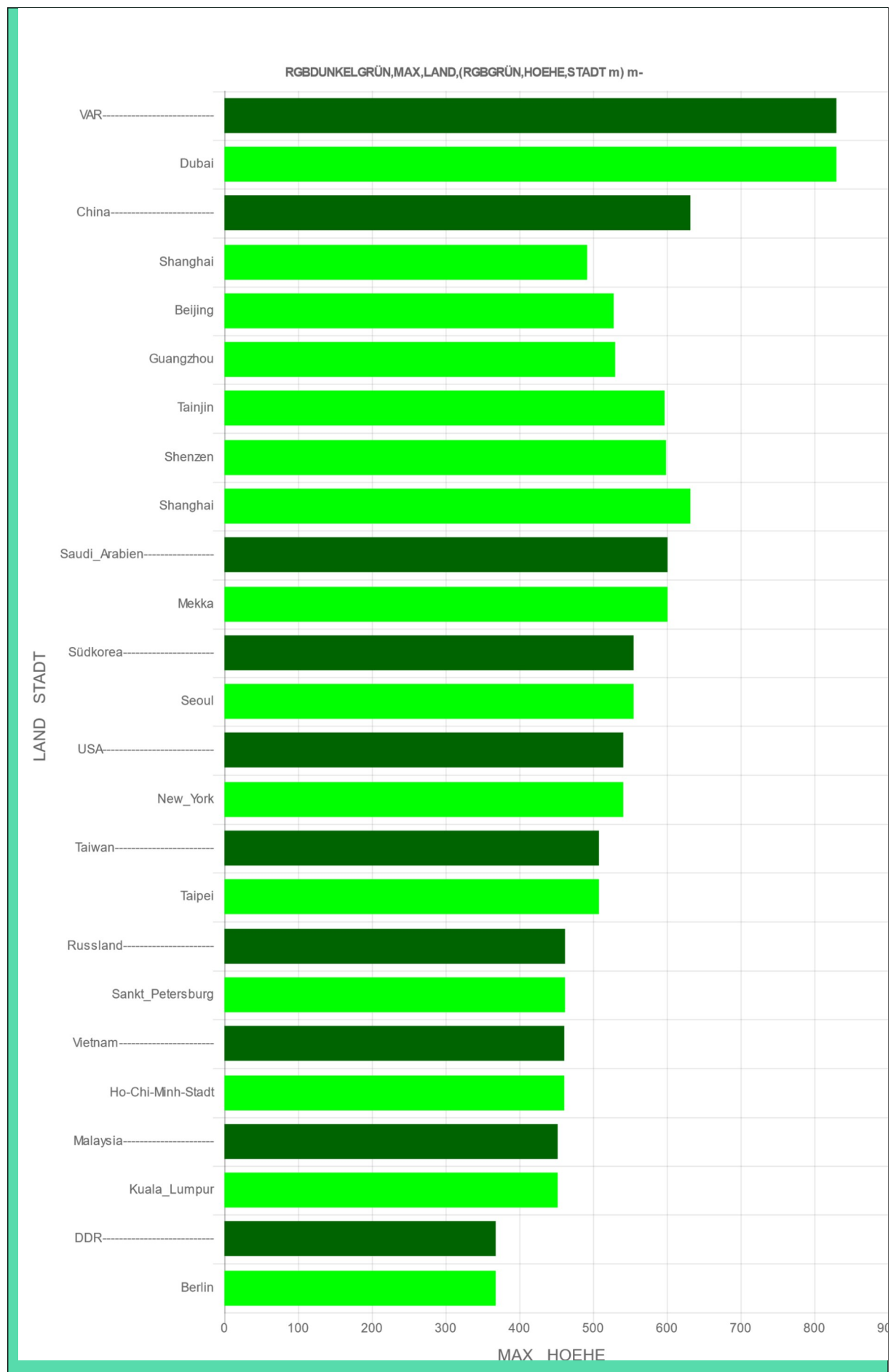
türme.tab			
TURM	,STADT	,LAND	,HÖHE l
Burj Khalifa	Dubai	VAR	830
Shanghai Tower	Shanghai	China	632
Abraj Al Bait	Mekka	Saudi Arabien	601
Ping An Finance Center	Shenzen	China	599
Goldin Finance	Tainjin	China	597
Lotte World Tower	Seoul	Südkorea	555
1 WTC	New York	USA	541
Guangzhou CTF Finance Center	Guangzhou	China	530
China Sun Tower	Beijing	China	528
Taipei 101	Taipei	Taiwan	508
World Finance Center	Shanghai	China	492
Lakhta Center	Sankt Petersburg	Russland	462
Vincom Landmark 81	Ho-Chi-Minh-Stadt	Vietnam	461
Petronas Towers	Kuala Lumpur	Malaysia	452
Berliner Fernsehturm	Berlin	DDR	368

Programm 15.3: Repräsentiere jeden Turm durch eine Säule.	
türme.tab	
TURM:=TURM subtext 1!10	# Durch diese Verkürzung des Namens sind auch
gib TURM,HÖHE l	# im Säulendiagramm alle Namen gleichzeitig
	# sichtbar; aus Platzgründen werden ansonsten
	# manchmal einige ausgeblendet
Ergebnis (struc.diagramm-bar)	



Programm 15.4: Repräsentiere jeden Turm durch einen Balken und gib die Balken länderweise aus. Länder mit den höchsten Türmen sind zuerst auszugeben (abwärts sortieren). Für jedes Land sind die Türme aufwärts zu sortieren. Die Länder sind optisch abzuheben.

```
aus türme.tab
gib MAX, LAND, (HÖHE, STADT m) m-    MAX:=HÖHE!max
LAND:==LAND wort + "-----" subtext 1!30
RGBDUNKELGRÜN:=darkgreen leftat MAX
RGBGRÜN:=green leftat HÖHE
Ergebnis (struc.diagram- bar(horizontal))
```





gewichte.tabh				
NAME,	LÄNGE,	(ALTER,GEWICHT1	1)	1
Bert	1.72	18	66 65	
		30	70 71	
Kathi	1.7	18	55 52	
		40	70 71	
Klaus	1.68	18	61 60 62	
		30	65 63 67	
		61	80 82 79	
Rolf	1.78	40	72 70 74	
Viktoria	1.61	13	51 50	
Walleri	1.	3	16 15	

Programm 15.5: Berechne die BMI Durchschnitte für alle Erwachsenen, für jede Altersstufe und insgesamt. Um die ersten 5 Zeilen zu realisieren benötigt EXCEL mehr als 6 Arbeitsblätter.

```
aus gewichte.tabh
sel NAME! ALTER>20
gib BMI,(ALTER,BMI,(NAME,BMI m) m)
    BMI:=GEWICHT:LÄNGE:LÄNGE ! ++:
rnd 2
ALTER:=ALTER wort
RGBROT:=red leftat BMI
RGBDUNKELGRÜN:=darkgreen leftat BMI2
RGBGRÜN:=green leftat BMI3
TITEL:="BMI-Durchschnitte insgesamt (rot), pro ALTER (dunkelgrün), "
    + "pro Person und Alter (grün)" leftat RGBROT
Ergebnis (struc.diagram bar)
```

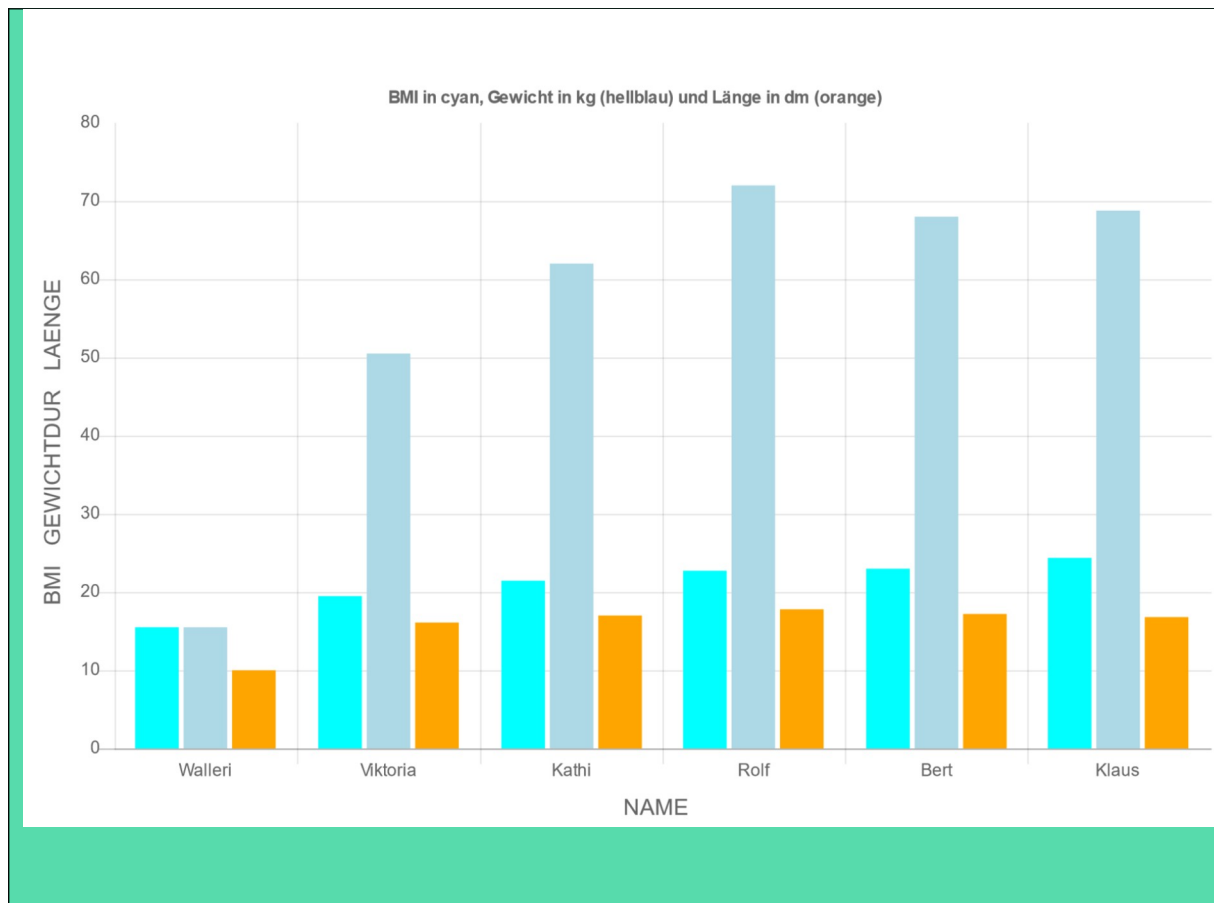


Programm 15.6: Vergleiche die Gewichte, Längen und die BMI aller Personen. Die Personen sind nach BMI zu sortieren.

```

aus gewichte.tabh
TITEL:="BMI in cyan, Gewicht in kg (hellblau) und Länge in dm (orange)"
gib TITEL,(BMI,NAME,GEWICHTDUR,LÄNGE m)
  GEWICHTDUR:=GEWICHT!++:
  BMI:=GEWICHT:LÄNGE:LÄNGE ! ++:
LÄNGE::=LÄNGE*10
ALTER::=ALTER wort
RGBHELLBLAU:=lightblue leftat GEWICHTDUR
RGBORANGE:=orange leftat LÄNGE
RGBCYAN:=cyan leftat BMI
Ergebnis ( struc. Diagram bar)

```



Programm 15.7: Repräsentiere 2 Funktionen durch Säulendiagramme.

```
X1:=0 ...10!0.05
```

```
SINUS :=X sin
```

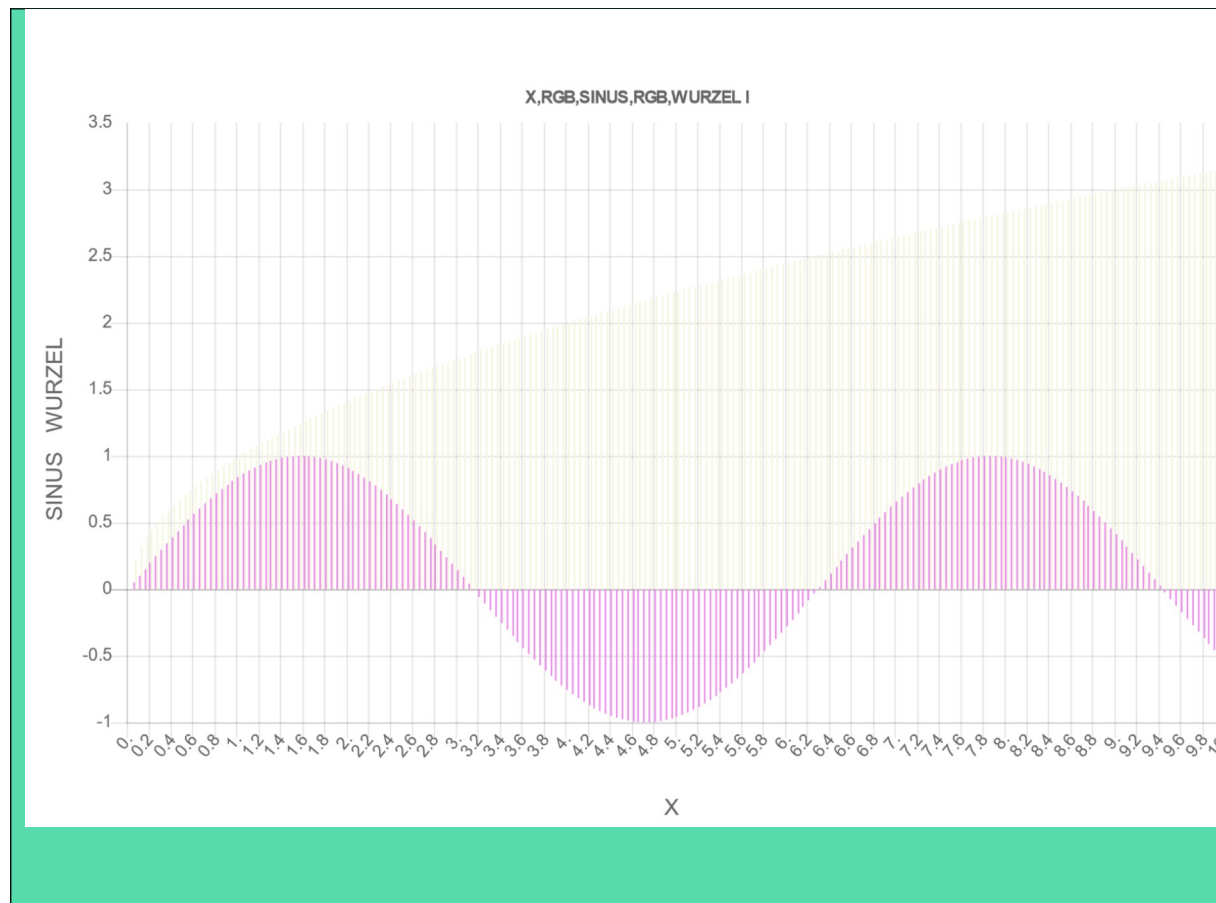
```
WURZEL:=X sqrt
```

```
X::=X wort
```

```
RGB:=violet leftat SINUS
```

```
RGB:=beige leftat WURZEL
```

```
Ergebnis (struc diagram - bar)
```

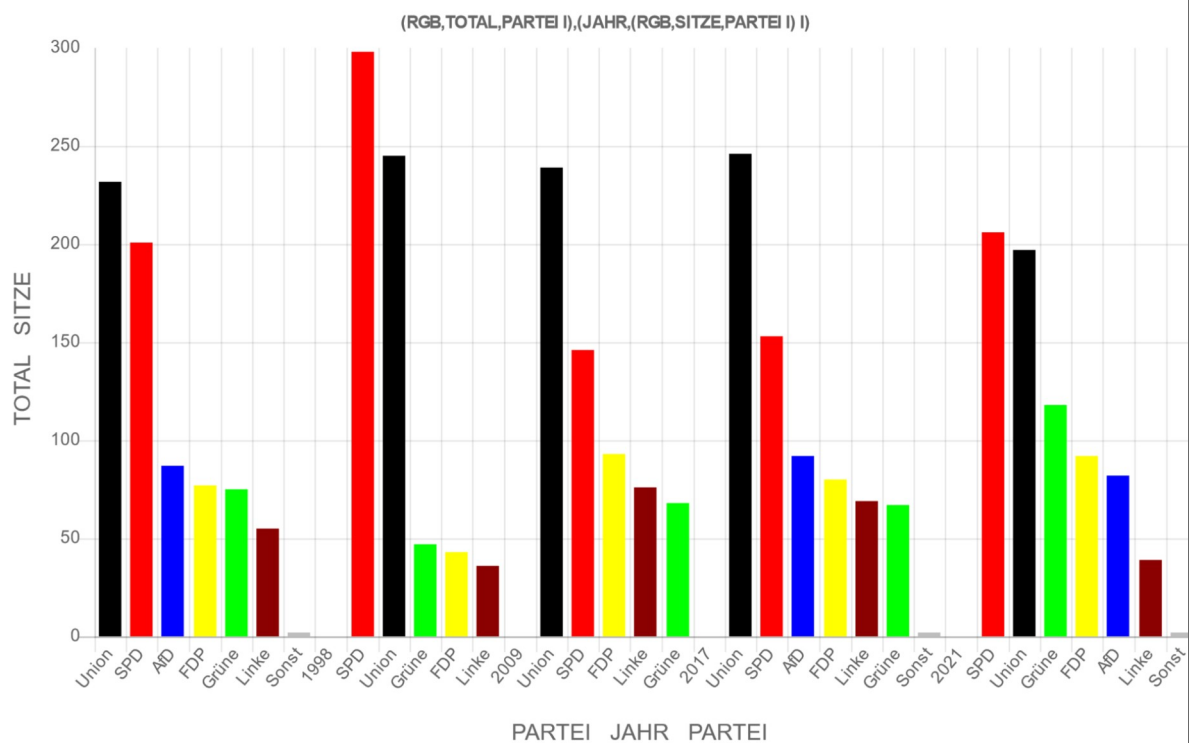


wahlen.tab		
JAHR, (PARTEI, SITZE 1) 1		
1998	PDS	36
	SPD	298
	Grüne	47
	Union	245
	FDP	43
2009	Linke	76
	SPD	146
	Grüne	68
	Union	239
	FDP	93
2017	Linke	69
	SPD	153
	Grüne	67
	Union	246
	FDP	80
	AfD	92
	Sonst	2
2021	Linke	39
	SPD	206
	Grüne	118
	FDP	92
	Union	197
	AfD	82
	Sonst	2

Programm 15.8: Visualisiere 4 Wahlergebnisse und ermittle die Durchschnittsstimmenanzahl der 4 Wahlen.

```
wahlen.tab
JAHR:=JAHR wort
PARTEI:=Linke if PARTEI="PDS" ! PARTEI
gib TOTAL,PARTEI m-,(JAHR,(SITZE,PARTEI m-)m) TOTAL:=SITZE! ++:
tocoll 1
RGB:=red      if PARTEI="SPD" !
  yellow      if PARTEI="FDP" !
  darkred     if PARTEI="Linke" !
  blue        if PARTEI="AfD" !
  black       if PARTEI="Union" !
  green       if PARTEI="Grüne" !
  grey        leftat TOTAL SITZE
```

Ergebnis (struc.diagramm-bar)



Ergebnis ohne RGB-Werte (tab)

```
TOTAL, PARTEI m-,(JAHR, (SITZE, PARTEI m-) m)
231.75 Union      1998   298   SPD
200.75 SPD                245   Union
87.   AfD                47   Grüne
77.   FDP                43   FDP
75.   Grüne              36   Linke
55.   Linke      2009   239   Union
2.    Sonst                146  SPD
                        93   FDP
                        76   Linke
                        68   Grüne
```

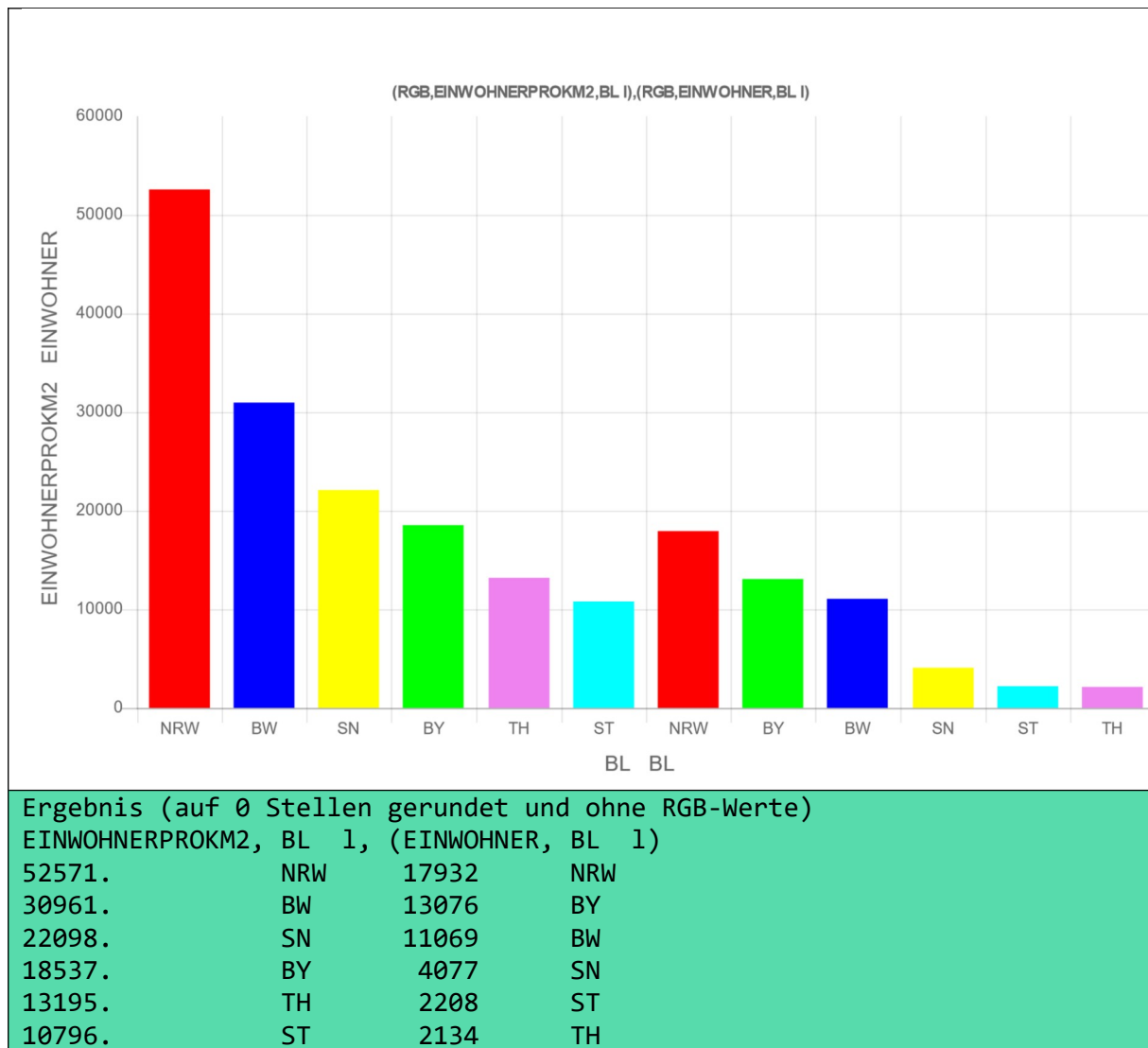
2017	246	Union
	153	SPD
	92	AfD
	80	FDP
	69	Linke
	67	Grüne
	2	Sonst
2021	206	SPD
	197	Union
	118	Grüne
	92	FDP
	82	AfD
	39	Linke
	2	Sonst

bundesländer6.tab				
BUNDESLAND,	BL,	FLÄCHE,	EINWOHNER,	BIP 1
Sachsen	SN	18449.99	4077	121.74
Sachsen-Anhalt	ST	20451.58	2208	60.1
Thüringen	TH	16172.5	2134	61.91
Bayern	BY	70541.57	13076	625.1
Baden-Württemberg	BW	35751.46	11069	493.27
Nordrhein-Westfalen	NRW	34110.26	17932	705.66

Programm 15.9: Sortiere und visualisiere die Bundesländer nach Einwohner pro Quadratkilometer und nach Einwohner. (Visualisiere 2 unabhängige Tabellen.)

```
bundesländer6.tab
EINWOHNERPROKM2:=EINWOHNER : FLÄCHE *100000
gib EINWOHNERPROKM2,BL m- , (EINWOHNER,BL m-)
tocoll 1
RGB:= red    if BL="NRW"!
      blue   if BL="BW" !
      yellow if BL="SN" !
      green  if BL="BY" !
      violet if BL="TH" !
      cyan leftat EINWOHNERPROKM2 EINWOHNER
```

Ergebnis (struc.diagram-bar)



16 Multiple Diagramme

Im vorangehenden Kapitel haben wir gesehen, dass sich eine strukturierte Tabelle in der Regel auch als strukturiertes Diagramm darstellen lässt. Das Programm 15.6 demonstriert, dass das auch für größere Tabellen funktioniert. Kreisdiagramme (=Tortendiagramme) werden jedoch schnell unübersichtlich, wenn ein Kreis zu viele Zahlen repräsentiert.

Strukturierte Tabellen enthalten in der Regel mehrere Teiltabellen. Diese enthalten in natürlicher Weise weniger Elemente als die Ausgangstabelle daher soll in diesem Kapitel jede Teiltabelle durch ein Diagramm repräsentiert werden. Mit multiplen Diagrammen werden strukturierte Tabellen noch direkter visualisiert als mit strukturierten Diagrammen.

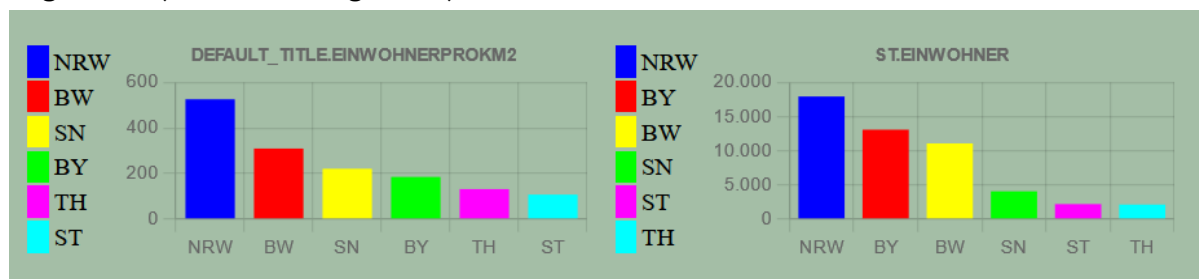
Programm 16.1: Sortiere und visualisiere die Bundesländer nach Einwohner pro Quadratkilometer und nach Einwohner. (Visualisiere 2 unabhängige Tabellen.)

bundesländer6.tab

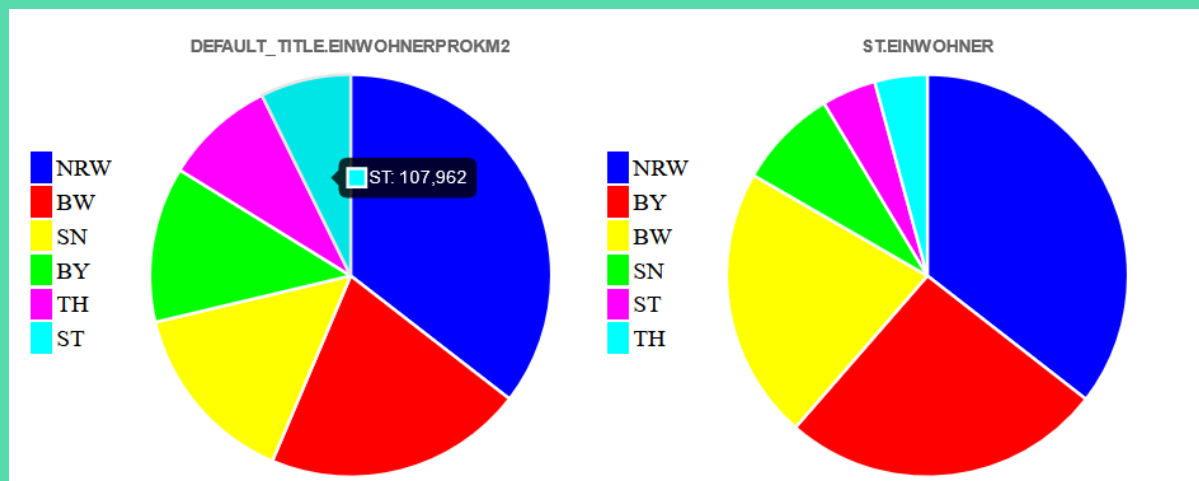
EINWOHNERPROKM2:=EINWOHNER*1000 : FLÄCHE

gib EINWOHNERPROKM2,BL m- , (EINWOHNER,BL m-)

Ergebnis (2 Säulendiagramme)



Ergebnis (2 Kreisdiagramme)



Im Programm 15.9 wurden die Einwohner pro Quadratkilometer mit 100'000 multipliziert, damit die Säulen gut sichtbar werden. Diese Anpassung ist bei multiplen Diagrammen nicht erforderlich.

Programm 16.2: wie 15.8

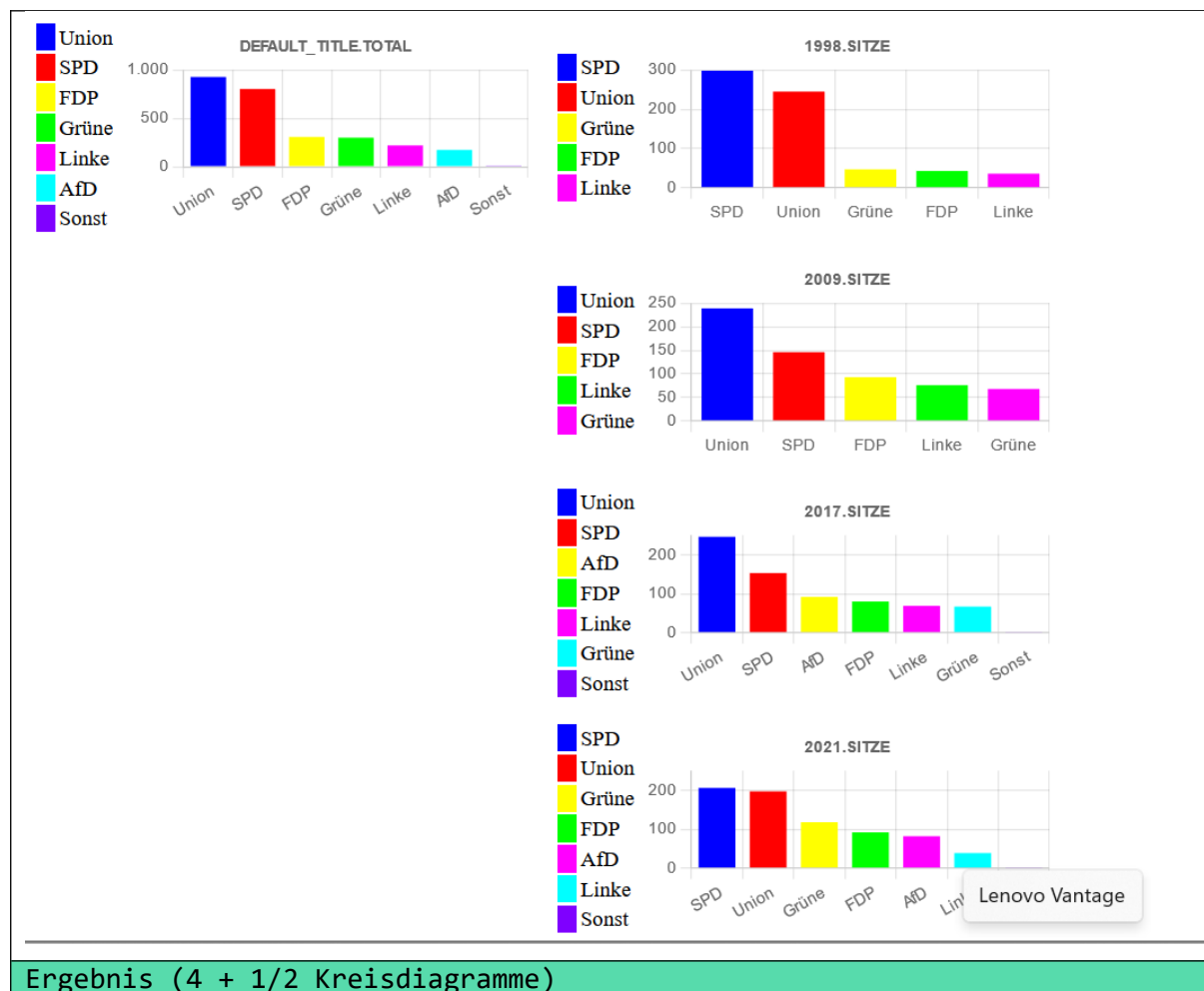
wahlen.tab

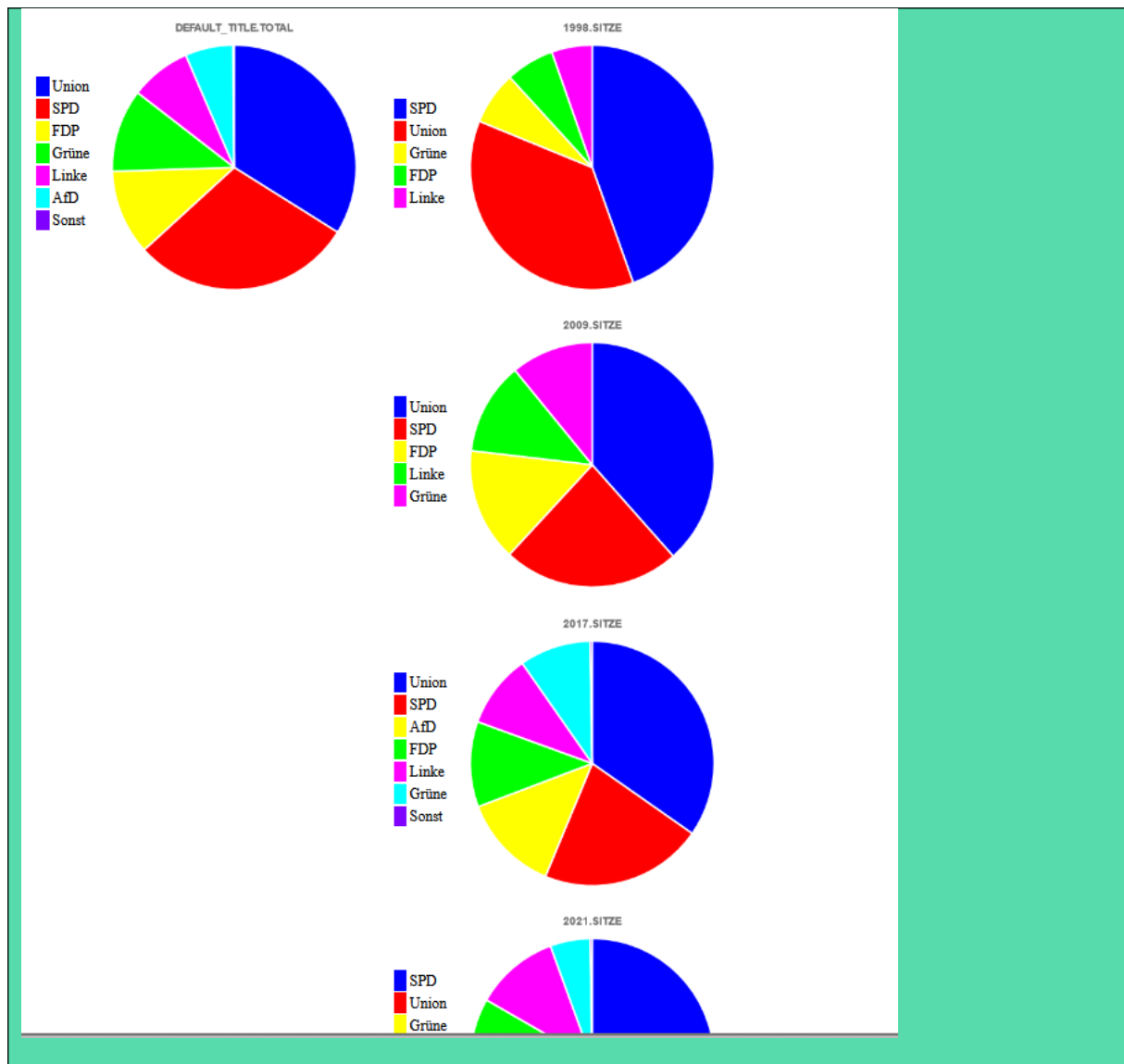
JAHR::=JAHR wort

PARTEI::=Linke if PARTEI="PDS" ! PARTEI

gib TOTAL,PARTEI m-,(JAHR,(SITZE,PARTEI m-)m) TOTAL:=SITZE! ++

Ergebnis (5 Säulendiagramme)





Man beachte den Unterschied zwischen obigem Programm und Programm 15.8. In 15.8 wird jeweils die Summe der vier Dateien berechnet und hier wird der Durchschnitt. Daher unterscheidet sich auch die Reihenfolge der Parteien in der Gesamtbilanz. Die Reihenfolge ändert sich auch dann nicht, wenn durch Division durch 4 ein "Durchschnitt" berechnet wird. Hier erkennt man wie wichtig es ist, dass der Endnutzer das Programm lesen können muss, um die erhaltenen Informationen richtig zu verarbeiten.

17 Bilderzeugung

Da mit o++o Zahlen auf einfache Weise generiert werden können, kann man auch ganze Bilder generieren. Durch `Xl:= 0 .. 4` werden zum Beispiel die Zahlen `0 1 2 3 4` generiert. Diesen Zahlen kann man Diagramme zuweisen, aber um ein Bild mit o++o zu erzeugen, benötigt man eine Liste oder eine Menge von Zahlenpaaren (X,Y). Der Punkt erhält eine Farbe, wenn sich ein RGB (RED,GREE,BLUE) Tripel vor dem X oder vor dem Y-Wert befindet:

(X, RGB, Y)

Für RGB-Werte hat o++o 3 Darstellungsmöglichkeiten.

Englische Farbnamen

red, silver, cyan, ...

Tripel ganzer Zahlen zwischen 0 und 255

(255,0,0) (=rot), (192,192,192) (=silber), (0,255,255) (=cyan)

Zahlentripel zwischen 0 und 1:

(1.,0.,0.) (=rot),(0.752941,0.752941;0.752941)(=silber),(0.,1.,1.) (=cyan)

Eigentlich kann man nur mit letzteren Tripeln richtig rechnen.

Wir beginnen mit 2 Funktionen, definieren sie aber zunächst nur für 10 X-Werte. Man muss genau hinschauen, um die Punkte zu erkennen:

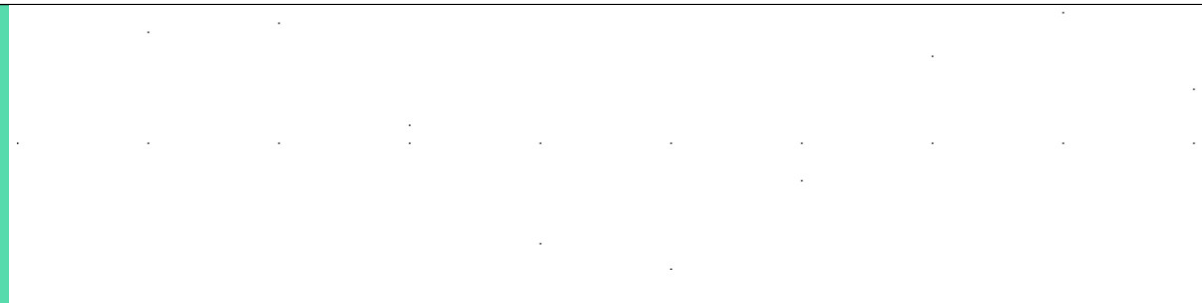
Programm 17.1: Erzeuge zweimal 10 Punkte.

```
Xl:=0 ..9
```

```
Y :=X sin
```

```
Y0:=X*0
```

Ergebnis (image - new window)



Durch die Einführung einer Schrittweite von 0.1 wird die Punktezahl verzehnfacht.

Programm 17.2: Erzeuge zweimal 100 Punkte.

```
Xl:= 0 ...9!0.1
```

```
Y:=X sin
```

```
Y0:=X*0
```

Ergebnis (image - new window)



Nun fügen wir der Schrittweite eine weitere 0 hinzu.

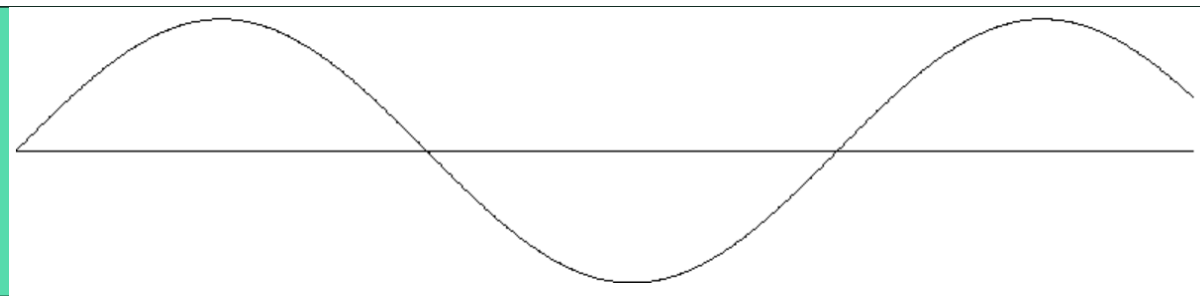
Programm 17.3: Erzeuge zweimal 1000 Punkte.

```
X1:=0 ...9!0.01
```

```
Y :=X sin
```

```
Y0:=X*0
```

Ergebnis (image - new window)



Die Sinusfunktion wird nun rot und die X-Achse grün. Die Tatsache, dass ein Spaltenname zweimal vorkommt (RGB), bereitet an dieser Stelle keine Probleme.

Programm 17.4: Stelle 2 Funktionen farblich dar.

```
X1:= 0 ...9!0.01
```

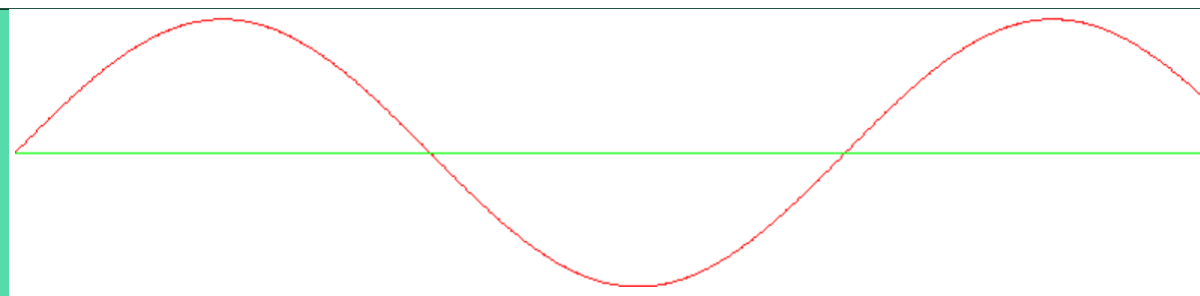
```
Y :=X sin
```

```
Y0:=X*0
```

```
RGB :=red leftat Y
```

```
RGB2:=green leftat Y0
```

Ergebnis (image - new window)



Dass es auch möglich ist, "Vollbilder" zu erzeugen, zeigt sich zunächst anhand der deutschen Flagge. Man erkennt, dass alle Punkte, die einem Farbwert folgen, in dieser Farbe ausgegeben werden. Somit werden nur drei Farbwerte für die deutsche Flagge benötigt. Der Begriff Pixel hat hier seine Bedeutung verloren oder muss neu definiert werden.

Programm 17.5: Generiere die deutsche Flagge

```
X1:= 0 ...9!0.01
```

```
Y1:= 0 ...2!0.01 at X
```

```
=: $RECHTECK
```

```
aus RGB:=gold
```

```
,$RECHTECK
```

```
RGB:=red
```

```
,$RECHTECK+(0,2)
```

```
RGB:=black
```

```
, $RECHTECK+(0,4)
```

```
Ergebnis (bild + new window)
```



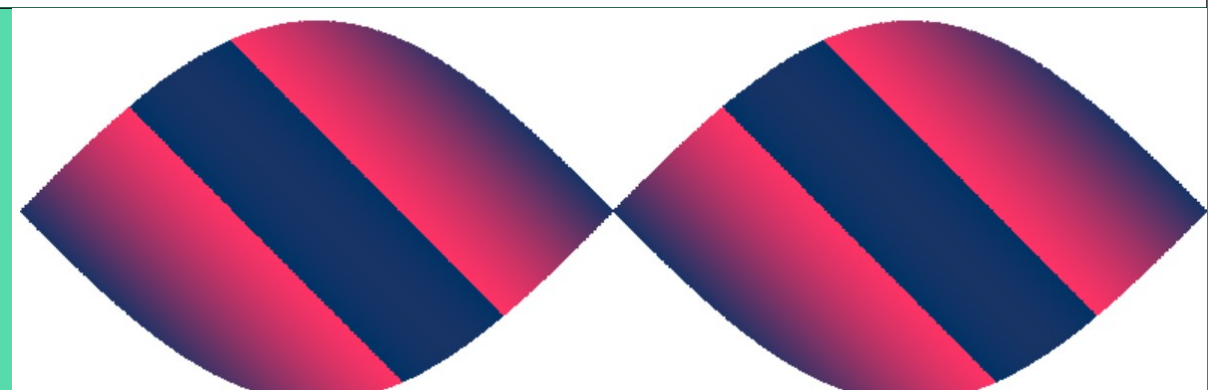
Programm 17.6: Entwirf einen Bikini. Färbe die Funktionen zwischen Sinus und Sinus gespiegelt.

```
X1:=pi * -1 ...pi!0.005
```

```
Y1:=X sin abs *-1 ...(X sin abs)! 0.005
```

```
RGB:= 0.1+(X+Y sin abs),0.2,0.4 leftat Y
```

```
Ergebnis (image + new window)
```



18 Bildbearbeitung

Die folgenden Programme basieren alle auf dem Foto eines Inders:



Programm 18.1: Stelle ein Foto dar!
inder.jpg
Ergebnis (image)
(siehe oben)

Obiges Bild nimmt lediglich 42.2 KByte ein.

Im Folgenden wird deutlich, dass die Datei 160 Tausend Datensätze (Tupel) enthält.

Programm 18.2: Zähle die Zeilen (Bildpunkte) des Fotos.
inder.jpg ++1 '3
Ergebnis (tab)
160'001

In der Regel ist es nicht möglich, alle Punkte eines Bildes zu betrachten. Daher geben wir im Folgenden nur die ersten 14 5-Tupel aus.

Programm 18.3: Selektiere die ersten 14 Bildpunkte.
inder.jpg sel X pos < 15
Ergebnis (tab)
X, RGB, Y 1 0. 228,232,220 0. 0. 227,231,219 0.01 0. 227,231,219 0.02 0. 226,230,218 0.03 0. 227,231,219 0.04 0. 228,232,220 0.05 0. 229,233,221 0.06 0. 230,234,222 0.07 0. 229,233,221 0.08 0. 229,234,221 0.09 0. 229,234,221 0.1 0. 229,234,221 0.11 0. 229,233,221 0.12 0. 229,233,220 0.13

Programm 18.4: Berechne die Extremwerte der Koordinaten der Bildpunkte
inder.jpg gib XMAX,XMIN,YMAX,YMIN XMAX:=X!max XMIN:=X!min YMAX:=Y!max YMIN:=Y!min
Ergebnis (tab)
XMAX, XMIN, YMAX, YMIN 4.02 0. 3.99 -0.05

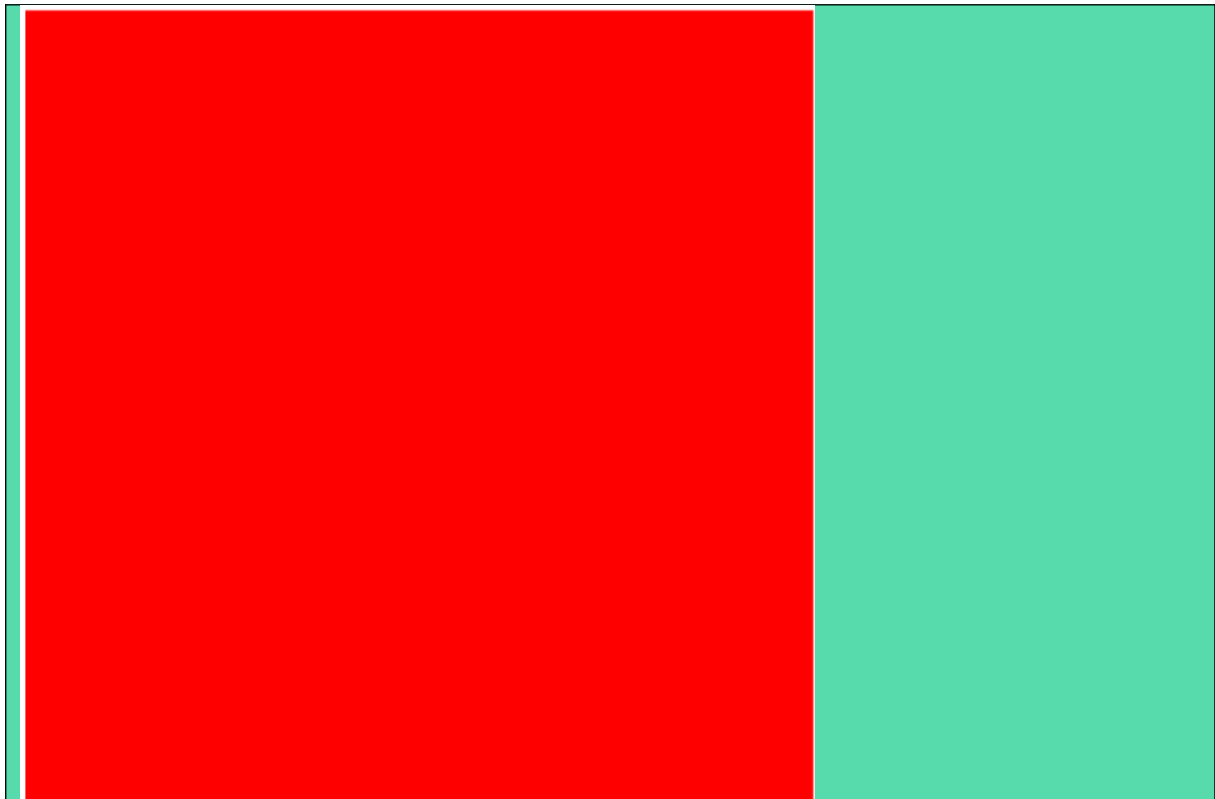
Man erkennt hierbei, dass das Foto ein 4 x 4 Quadrat darstellt. Die X- und Y-Werte liegen daher zwischen 0 und 4.

Programm 18.5: Gib den linken Teil des Fotos aus.
inder.jpg sel X pos < 50'000
Ergebnis (bild)



Ein einfaches rotes Rechteck kann auch mit einem beliebigen Foto erstellt werden, indem einfach jeder Farbwert überschrieben wird.

Programm 18.6: Streiche das Foto rot an.
<code>index.jpg</code> <code>RGB := red</code>
Ergebnis (image-new window)



Auch sehr einfach, die deutsche Flagge aus dem Foto zu erstellen:

Programm 18.7: Transformiere ein Foto zur deutschen Flagge.

```
index.jpg  
RGB := black if Y>8:3 !  
      red   if Y>8:6 !  
      gold
```

Ergebnis (image - new window)



Programm 18.8: Verstecke den Kopf hinter einen braunen Balken.

inder.jpg

```
RGB := brown if 1.5 < X & X < 2.5 ! RGB
```

Ergebnis (image - new window)



Programm 18.9: Wandle jedes stärkere Grün in reines Grün um.

inder.jpg

```
RGB := green if RGB nth 2 > 100 ! RGB
```

Ergebnis (image - new window)



Programm 18.10: Wandle ein Farbfoto in ein schwarz-cyan Foto um.

inder.jpg

```
RGB:= cyan if RGB ++ <220!black
```

Ergebnis (image - new window)



Programm 18.11: Gib das Foto einschließlich einer Verschiebung des Fotos aus.

inder.jpg

Y:=inder.jpg+(8.,0,0,0,0.)

Ergebnis (image - new window)



Programm 18.12: Ordne den Kopf des Inders in einem 2*4 Rechteck an.

inder.jpg

sel X- 2 hoch 2 +(Y - 2.3 hoch 2) <1

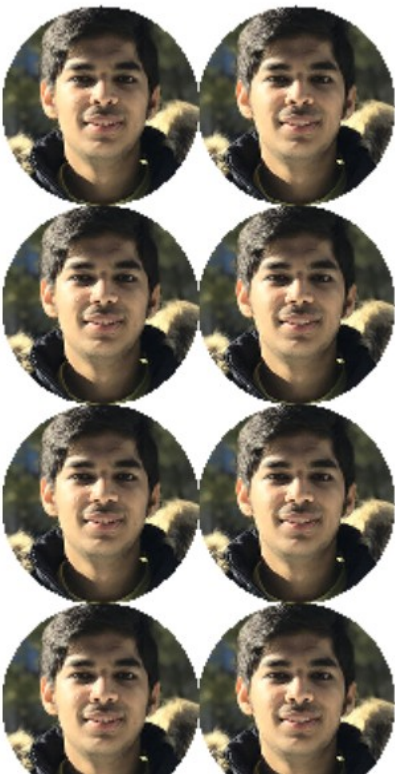
=: \$KOPF

aus Z1:= 1 ..4

W1:= \$KOPF + (0.;0;0;0;2*Z)

W2:= \$KOPF + (2.;0;0;0;2*Z)

Ergebnis (image - new window)



Im Folgenden wird klar, dass der Inder noch deutlich sichtbar ist, obwohl wir nur seinen roten Anteil übernommen haben. Eine blaue Welle beginnt oben rechts im folgenden Bild, unabhängig von den gegebenen Farben.

Programm 18.13: Erzeuge eine blaue Welle über die rot-Anteile des Fotos.

inder.jpg

```
RGB:=RGB nth 1;0 ;X+Y sin abs
```

Ergebnis (image)



Auch wenn wir hier nur wenige Beispiele gegeben haben, dürfte klar geworden sein, dass man mit o++o aus einem einzigen Foto tausende oder Millionen verschiedener Bilder erzeugen kann.

19 Eine Bäckeranwendung (CSS)

In diesem Kapitel nehmen Formfragen etwas mehr Raum ein als die inhaltlichen Fragen des otto-Programms. Der Bäcker will Rechnungen erstellen und die gesammelten Rechnungsdaten anschließend auswerten können. Das folgende Programm erzeugt die Rechnung und speichert die wesentlichen Daten aller Rechnungen gleich in eine Datei rechnungen22.hsq, die unmittelbar nach einer Rechnungserstellung neu ausgewertet werden kann.

Programm 19.1 Rechnungen für Kunden ausdrucken und in einer Datei abspeichern.

```
$RECHNNR := "25#22"
$KUNDE := "Senioren"
$RECHNVOM := "19.11.2022"
$DATUM := "21.11.2022"
$BEZAHLT := nein
$BESTELLUNG:= <TAB!
PRODUKT,          ANZAHL m
Hanfbrot          6
Baguette          6
Brötchen          77
Kürbisbrötchen   20
Partystange       5
Mischbrot 1 kg    7
Milchbrötchen    26
!TAB>
##### ENDE DER EINGABE #####
$FALSCHEPRODUKTE:=begin aus $BESTELLUNG;; gib PRODUKTm
-coll begin aus produkte.tab;;gib PRODUKTm end end
$KUNDEANZ:=begin aus kunden.tab;; sel KUNDE = $KUNDE;; ++1 end

aus ($KUNDEANZ = 0 or $FALSCHEPRODUKTE ++1>0) dann (($KUNDEANZ=0) dann
"Kunde inkorrekt" ! ("falsche Produkte:" ;; $FALSCHEPRODUKTE)) !
begin
aus $BESTELLUNG,produkte.tab
gibjoin (PRODUKT,STEUER,PREIS,ANZAHL) m
=: $BESTELLUNGGESAMT
aus $BESTELLUNGGESAMT
,absender.tab
,begin aus kunden.tab;;sel KUNDE = $KUNDE;;gib KUNDENADRESSE
KUNDENADRESSE=NAME,STRASSE,ORT end
GESAMTPREIS := STEUER :100 +1 *PREIS *ANZAHL
gib ABSENDER,KUNDENADRESSE,TABELLEN
TABELLEN= ZEILEN,ZAHLUNGSBETRAG
ZEILEN= PRODUKT,ANZAHL,STEUER,PREIS,GESAMTPREIS 1
ZAHLUNGSBETRAG:=GESAMTPREIS!++
rnd 2
,bankverbindung.tab
BILD:= "bilder/brotraucht.png" at ABSENDER
$L:="Re.Nr: " + ($RECHNNR text)
$M:="Rechn. vom " + ($RECHNVOM text)
$R:="Gerwisch, " + ($DATUM text )
BETREFF:= $L,$M,$R leftat TABELLEN
ZAHLUNGSBETRAG:="ZAHLUNGSBETRAG: " + (ZAHLUNGSBETRAG text) + " €"
tag0 RECHNUNG
FORMAT:=format2.tab
=: $ERGEBNIS
```

```
aus $RECHNNR,$DATUM,$KUNDE,$BEZAHLT,$BESTELLUNGSGESAMT
gib RECHNNR,DATUM,KUNDE,BEZAHLT,(PRODUKT,ANZAHL,STEUER,PREIS m) m
,rechnungen.hsq
gib RECHNNR,DATUM,KUNDE,BEZAHLT,(PRODUKT,ANZAHL,STEUER,PREIS m) m
rnd 2
save rechnungen22.hsq

aus $ERGEBNIS
end
```

Ergebnis (web) (new window)

Backhaus Magdeburg
 Inhaber: Manuel Schmidt
 Breiter Weg 128
 391 Magdeburg
 Tel: 0391 22444
 e-mail: backhausschmidt@web.de
 Steuer-Nr. 103/2662/08839



Seniorenheim Magdeburg
 Vechelder Weg 10
 39175 Biederitz

Re.Nr: 25#22

Rechn. vom 19.11.2022

Gerwisch, 21.11.2022

PRODUKT	ANZAHL	STEUER	PREIS	GESAMTPREIS
Baguette	6	7	2.20	14.12
Brötchen	77	7	0.34	28.01
Hanfbrot	6	7	3.50	22.47
Kürbisbrötchen	20	7	0.45	9.63
Milchbrötchen	26	7	0.43	11.96
Mischbrot 1 kg	7	7	1.90	14.23
Partystange	5	7	2.40	12.84

Zahlungsbetrag:
 113.27€

Bankverbindung
 Kontoinhaber: Manuel Schmidt
 DE68 8106 3238 7777 7764 66
 Volksbank Harzer Land

Nachdem der Bäcker seine Bestelldaten eingegeben hat, wird untersucht, ob alle Produktnamen auch in der Datei produkte.tab enthalten sind. Anschließend werden alle für die Kosten relevanten Daten der Bestellung in der Tabmentvariablen \$BESTELLUNGESAMT zusammengefasst. Diese Variable besitzt ein Schema vom Typ PRODUKT,STEUER,PREIS,ANZAHL m. D.h., sie enthält eine ganze Tabelle. Zu diesen Daten werden die Kundendaten und der Absender hinzugefügt. Es wird der Gesamtpreis berechnet. Die Rechnung erfordert zusätzliche Tags, damit die Daten in gewünschter Weise formatiert werden können. Die gesamte Rechnung wird in der Tabmentvariablen \$ERGEBNIS zwischengespeichert, damit die Rechnung zuvor noch zu rechnungen22.hsq hinzugefügt werden kann.

Die Formatdaten wurden in der Datei format2.tab zusammengefasst:

Formatdatei format2.tab aus Programm 16.1		
SELECT,	TYPE,	STYLE1 1
RECHNUNG	div	vertikaleausrichtung:oben
BETREFF	span	rand-links:15mm
MITTE	span	
UNTEN	div	position:absolute oben:230mm rand-links:10mm border-width:6px border-style:groove border-color:darkgreen
ABSENDER	div	rand-oben:10mm rand-links:15mm rand-rechts:110mm border-width:6px border-style:groove border-color:darkgreen
BILD	img	position:absolute oben:10mm rechts:15mm border-width:6px border-style:groove border-color:violet
KUNDENADRESSE	div	rand-links:15mm rand-oben:10mm rand-unten:10mm rand-rechts:110mm border-width:6px border-style:groove border-color:cyan
L	span	border-width:6px border-style:groove border-color:darkgreen
M	span	border-width:6px border-style:groove border-color:darkgreen
R	span	border-width:6px border-style:groove border-color:darkgreen
TABELLEN	div	ausrichtung:rechts rand-rechts:10mm
ZEILEN	tabelle	float:right rand-oben:10mm rand-rechts:10mm schriftgroesse:gross border-width:6px border-style:groove border-color:firebrick
ZAHLUNGSBETRAG	tabelle	float:right clear:both rand-oben:10mm rand-rechts:10mm schriftgroesse:gross

GESAMTPREIS		border-width:6px border-style:groove border-color:red ausrichtung:rechts white-space:nowrap
STEUER		ausrichtung:rechts
EINZELPREIS		ausrichtung:rechts
MENGE		ausrichtung:rechts
NAME	span	fett
BANKVERBINDUNG	div	position:absolute oben:257mm rand-links:15mm border-width:6px border-style:groove border-color:darkgreen

Inputdatei **produkte.tab** für Programm 16.1

PRODUKT,	STEUER,	PREIS	m
Baguette	7	2.20	
Brötchen	7	0.34	
Brötchen belegt halb	7	1.70	
Cake Pops	7	0.39	
Kuchen Stk.	7	0.45	
Mini Berliner	7	0.55	
Mini Burger	7	0.35	
Mischbrot 1 kg	7	1.90	
Mischbrot 1.5 kg	7	3.20	
Fahrt	19	6.08	
Hanfbrot	7	3.50	
Kaviarstangen	7	4.30	
Knüppel	7	0.40	
Körnerbaguette	7	2.50	
Körnerbrot	7	3.20	
Kürbisbrötchen	7	0.45	
Mehrkornbrötchen	7	0.55	
Milchbrötchen	7	0.43	
Mini-Fladenbrot	7	0.60	
Mischbrot	7	3.30	
Olivebrot	7	2.90	
Partystange	7	2.40	
Pizzastange	7	3.20	
Roggenbrötchen	7	0.50	
Roggenkastenbrot	7	2.60	
Semmeln	7	0.55	
Sonnenblumenbrötchen	7	0.55	
Tomatenbrot	7	2.00	

Inputdatei **kunden.tab** für Programm 16.1

KUNDE,	NAME,	INHABER,	STRASSE,	ORT	m
Backschwein	Backschwein-Tanne	Günigk	Dorfstrasse 180	17822 Brack	
AFF	AFF Burg		Kapellenweg 22	39288 Burg	
JA	JA Sachsen-Anhalt		Schönebecker Str. 11-13	39104 Magdeburg	
Senioren	Seniorenheim Magdeburg		Vechelder Weg 10	39175 Biederitz	
DRK	DRK Seniorenwohnpark		An der alten Eiche	39175 Gerwisch	
Landgasthof	Landgasthof Hermsdorf		Möser Str. 22	39185 Hermsdorf	

Exquisit	Exquisit Catering GmbH	Gareisstraße 10	39106 Magdeburg
Inputdatei bankverbindung.tab für Programm 16.1			
NAME, Bankverbindung	KONTOINHABER, Kontoinhaber: Manuel Schmidt	IBAN, DE68 8106 3238 7777 7764 66	BANK Volksbank Harzer Land
Inputdatei absender.tab für Programm 16.1			
NAME, Backhaus Magdeburg	INHABER, Inhaber: Manuel Schmidt	STRASSE, Breiter Weg 128 391	ORT, Magdeburg
TEL, Tel: 0391 22444	EMAIL, e-mail: backhausschmidt@web.de	STEUERNUMMER, Steuer-Nr. 103/2662/08839	

Für die untenstehende Rechnung wurde das Bild (die .png-Datei) und die Formatdatei durch format.tab ersetzt:

Backhaus Magdeburg
 Inhaber: Manuel Schmidt
 Breiter Weg 128
 391 Magdeburg
 Tel: 0391 22444
 e-mail: backhausschmidt@web.de
 Steuer-Nr. 103/2662/08839



Landgasthof Biederitz
 Möser Str. 27
 39175 LOSTAU

Re.Nr: 18#22
 Rechn. vom 19.10.2022
 Gerwisch, 21.10.2022

PRODUKT	ANZAHL	STEUER	PREIS	GESAMTPREIS
Baguette	6	7	2.20	14.12
Brötchen	77	7	0.34	28.01
Hanfbrot	6	7	3.50	22.47
Kürbisbrötchen	20	7	0.45	9.63
Milchbrötchen	26	7	0.43	11.96
Mischbrot 1 kg	7	7	1.90	14.23
Partystange	5	7	2.40	12.84

Zahlungsbetrag: 113.27€

Bankverbindung
 Kontoinhaber: Manuel Schmidt
 DE68 8106 3238 7777 7764 66
 Volksbank Harzer Land

Die gleiche Ausgabe wird für Demonstrationszwecke mit 2 Hintergrundfarben angedeutet:

Backhaus Magdeburg
 Inhaber: Manuel Schmidt
 Breiter Weg 128
 391 Magdeburg
 Tel: 0391 22444
 e-mail: backhausschmidt@web.de
 Steuer-Nr. 103/2662/08839

Seniorenheim Magdeburg
 Vechelder Weg 10
 39175 Biederitz



Re.Nr: 25#22
 Rechn. vom 19.11.2022
 Gerwisch, 21.11.2022

PRODUKT	ANZAHL	STEUER	PREIS	GESAMTPREIS
Baguette	6	7	2.20	14.12
Brötchen	77	7	0.34	28.01
Hanfbrot	6	7	3.50	22.47
Kürbisbrötchen	20	7	0.45	9.63
Milchbrötchen	26	7	0.43	11.96
Mischbrot 1 kg	7	7	1.90	14.23
Partystange	5	7	2.40	12.84

Zahlungsbetrag: 113.270

Bankverbindung
 Kontoinhaber: Manuel Schmidt
 DE68 8106 3238 7777 7764 66
 Volksbank Harzer Land

Die Datei rechnungen22.hsq kann nach jeder Rechnungserstellung beispielsweise mit folgendem kurzen Programm ausgewertet werden. Es könnten vor dem gib noch Bedingungen eingefügt werden:

Programm 19.2: Berechne für jeden Kunden und jedes Produkt die Umsätze insgesamt und pro Monat.

```
rechnungen22.hsq
```

```
MON:=DATUM text subtext 4!2
```

```
gib UMSATZ,(KUNDE,(UMSATZ,(MON,UMSATZ m)m)m),  
      (PRODUKT,UMSATZ,(MON,UMSATZ m)m)m
```

```
UMSATZ:=ANZAHL*PREIS +% STEUER ! ++
```

```
rnd 2
```

Ergebnis (tab-Ausgabe):

UMSATZ	(KUNDE	(UMSATZ2,	(MON,	UMSATZ3	l) l) l),	(PRODUKT	,UMSATZ4,	(MON,	UMSATZ5	l) l) l
2066.10	AFF	254.21	11	195.42		Baguette	656.77	10	14.12	
			12	58.79				11	157.72	
	Backschwein	773.03	11	210.23				12	484.92	
			12	562.80		Brötchen	200.82	10	28.01	
	Landgasthof	113.27	10	113.27				11	107.32	
	Senioren	925.58	11	634.88				12	65.48	
			12	290.70		Brötchen belegt halb	231.01	11	120.05	
								12	110.96	
						Hanfbrot	827.65	10	22.47	
								11	554.26	
								12	250.92	
						Kürbisbrötchen	28.89	10	9.63	
								11	19.26	
						Milchbrötchen	35.89	10	11.96	
								11	23.93	
						Mischbrot 1 kg	46.56	10	14.23	
								11	32.32	
						Partystange	38.52	10	12.84	
								11	25.68	

20 Anhang A: Liste der Operationen und Schlüsselworte von o++o

Die meisten der bekannten Operationen haben eine Arität. Die Quadratwurzel zum Beispiel benötigt nur einen Eingabewert oder ein Argument - dies ist normalerweise eine Zahl. Im o++o-Datenmodell kann dies auch eine Liste von Zahlen sein. Dann wird die Quadratwurzel aus jeder der Zahlen gezogen. Die Liste wird dann als **ein** Eingabewert betrachtet, obwohl sie zehn oder sogar zehntausend Zahlen enthalten kann. Das heißt, sqrt bleibt auch in diesem Fall unär.

In der o++o-Syntax muss das Symbol sqrt dem Argument folgen (Postfix). Das bedeutet, dass keine zusätzlichen Klammern erforderlich sind. Es ist in o++o nicht zulässig sqrt([2 4 7]) zu schreiben. Stattdessen kann man aber

[2 4 7] sqrt

oder auch

2 4 7 sqrt

tippen.

In beiden Fällen erhält man das gleiche Ergebnis. Sie können sqrt sogar auf jedes Tabment anwenden.

Ein weiteres Beispiel ist die Addition. Der Operator + ist sogar noch besser bekannt als die Wurzeloperation. Er hat die Arität 2, d. h. er benötigt zwei Eingabewerte. Die Addition ist binär. Die Anwendung der falschen Anzahl von Argumenten führt zu einem syntaktischen Fehler und einer entsprechenden Fehlermeldung.

3 +

sowie

3 4 +

führen zu Fehlermeldungen.

In dem Term

3 + 4

ist 3 das erste Argument und 4 ist der zweite Inputwert. Auch hier kann als erstes Argument eine Liste oder ein anderes Tabment verwendet werden. Die Operation und das zweite Argument werden dann auf alle Elemente der Liste/Tabelle angewendet.

1 3 7 + 4

ergibt

5 7 11

Hier und bei vielen anderen Operationen entspricht der Typ des Ergebnisses dem Typ der ersten Eingabetabelle. Das obige Ergebnis ist also auch eine Liste von Zahlen. Binäre Operationen werden in o++o immer zwischen die beiden Eingabetabellen geschrieben. Man kann es auch so sehen, dass sie wie die unären Operationen nach dem ersten Eingabetabment erscheinen müssen. Das Gleiche gilt für dreistellige Operationen in o++o. "!" wird als Trennzeichen zwischen dem zweiten und dritten Eingabewert verwendet.

Hadmersleben subtext 4!5

hat zum Beispiel das Ergebnis:

mersl

Der erste Eingabewert ist "Hadmersleben". Der zweite Eingabewert (4) gibt die Position des Anfangsbuchstabens des Teilwortes an und der dritte Eingabewert (5) gibt die gewünschte Länge an.

5 if X>3 ! 6

erfordert ebenfalls 3 Eingabewerte (hier: die 5, einen Wahrheitswert und die 6). Ersetzen wir X durch 10, ist die Bedingung erfüllt und die verbesserte "if-then-else"-Operation liefert 5. Für X=1 ergibt sich jedoch der Wert 6.

Im Folgenden werden die Eingabe- und Ausgabedaten noch einmal anhand von typischen Beispielen illustriert.

(erstes) Inputtabment	unäre Operation		Outputtabment
pi	sin	ergibt	0.
1 4 9	sqrt	ergibt	1. 2. 3.
1 4 9	++:	ergibt	4.666666666666
123456789	'3	ergibt	123'456'789

erstes Inputtabment	binäre Operation	zweites Inputtabment		Outputtabment
7	+	8	ergibt	15
7.	+	8	ergibt	15.
1 5 3	+	4	ergibt	5 9 7
1 5 3	+	1.2	ergibt	2.2 6.2 4.2
1 5 3	+	3 7 8	ergibt	4 12 11
1 5 3	+	3 7	nicht definiert	
7 9	divrest	3	ergibt	2,1 3,0

erstes Inputtabment	ternäre Operation	zweites Inputtabment		drittes Inputtabment		Outputtabment
"Georg Cantor"	subtext	1	!	5	ergibt	Georg
5	if	3=4 (no)	!	6	ergibt	6
1	...	2.9	!	0.5	ergibt	1. 1.5 2. 2.5
1	..x	2	!	4	ergibt	1 2 1 2

An dieser Stelle sei angemerkt, dass in vielen Fällen das Ergebnis der vorherigen Zeile als erstes Eingabetabment einer Operation zählt:

noten.tab

++:

ergibt den Durchschnitt aller Zahlen, die in der ersten Zeile noten.tab vorkommen. Das Programm

xx.tab

+ 2

addiert 2 zu jeder Zahl in Tabelle xx.tab. xx.tab ist die erste Eingabetabelle und 2 ist die zweite. In analoger Weise extrahiert

namen.tab

subtext 3!4

aus jedem Textwert (TEXT oder WORT oder ONR) von names.tab einen Text der Länge 4, der an der dritten Position beginnt. Hier hat die ternäre Subtext-Operation die Eingabetabmente names.tab, 3 und 4.

In einer Zuweisung oder Bedingung können mehrere Operationen hintereinander angewandt werden. Sind alle Operationen unär (einstellig), so hat jeder entsprechende einzeilige Term die Gestalt

tbt op₁₁ op₁₂ op₁₃ ... op_{1n}

oder konkreter:

1 2 3 sin abs sqrt ++text

Sind alle Operationen binär (zweistellig) so ergibt sich die Form

$tbt_1 op_{21} tbt_2 op_{22} tbt_3 op_{23} tbt_4 \dots op_{2n} tbt_{n+1}$

oder konkreter

1 2 3 + 4 * 5 - 9

Terme mit ausschließlich 3 stelligen Operationen kommen sicher selten vor. Hier lediglich ein konstruiertes Beispiel:

Magdeburg subtext 2!6 subtext 2!2

ergibt gd

Wenn Klammern gesetzt werden, müssen diese zuerst ausgerechnet werden:

abcdefghijkl subtext 2!(2+3) ergibt bcdef

abcdefghijkl subtext 2!2+3 ergibt dagegen bc

(bc + 3 ergibt bc)

Wenn man sich nicht ganz sicher ist, kann man vorsichtshalber Klammern setzen.

+ 3

ist kein Term, da die Operation + hier keinen ersten Inputwert besitzt. Es würde daher eine Fehlermeldung erscheinen. Das trifft jedoch nicht zu, wenn sich der obige Code nicht in der ersten Zeile befinden würde. Der Wert der vorangehenden Zeile ist dann der erste Inputwert von + und 3 ist dann der zweite Inputwert.

Im Folgenden werden untenstehende Bezeichnungen benutzt:

num = ZAHL oder PZAHL oder RATIO oder BAR (|) oder Strichliste

nonum = TEXT oder WORT oder ONR

mixe = text und zahl kommen in einer Spalte vor

tbt steht für einen beliebigen Tabmenttyp

Bei den Typen geben wir häufig nur diejenigen an, die auch durch die Operation verändert werden.

Operations-symbol	Schreibweise: Input → Ergebnistyp	Beispiele	Bedeutung
+	tbt1 + tbt2 → tbt1	1 1 3 + 2.1 ergibt 3.1 3.1 5.1 xy ab + de ergibt xyde abde	Addition von Zahlen und verbinden von Text
*	tbt1 * tbt2 → tbt1	2 3 5 * 2 ergibt 4 6 10	Multiplikation
-	tbt1 - tbt2 → tbt1	3 - 2 ergibt 1 1'234 - 345 ergibt 889	Subtraktion
:	tbt1 : tbt2 → tbt1	3:4 ergibt 0.75	Division
++	tbt ++ → num	2 3 6 ++ ergibt 11	Summe
**	tbt ** → num	1 3 5 ** ergibt 15	Produkt
--	tbt -- → num	20 5 4 -- ergibt 11	Vielfachsubtraktion
::	tbt :: → num	64 2 2 :: ergibt 16	Vielfachdivision
++:	tbt ++: → PZAHL	1 2 3 2 ++: ergibt 2.0	Durchschnitt
++1	tbt ++1 → ZAHL	3 4 7 9 ++1 ergibt 4	Anzahl
++text	tbt ++text → text	[ab cde fg] ++text ergibt abcdefg	zu Text verbinden
++textsep	TEXT1 + +textsep "sep" → TEXT	ab cde fg ++textsep “;“ ergibt “ab;cde;fg“	zu Text unter Einschluss eines Separators verbinden
,	tbt1,tbt2 → tbt	1 2,3 ergibt (tab) ZAHL1,ZAHL 1 3 2	Paarbildung

;	tbt1;tbt2 → tbt	2,3 * 2 ergibt 4,6 2;3 * 2 ergibt 2,6	auch eine Paarbildung. aber ; trennt schärfer als ,
=	tbt1,tbt2 → BOOL	1 = 2 ergibt si	Gleichheit
<=	tbt1 <= tbt2 → BOOL	2 <= 2 ergibt si	kleiner gleich
>=	tbt1 >= tbt2 → BOOL	2 >= 4 ergibt no	größer gleich
+coll	coll1 +coll coll2 → coll1	{{1 2}} +coll {1} ergibt {{1 1 2 }}	"mengentheoretische" Vereinigung
-coll	coll1 -coll coll2 → coll1	[2 4 3 2] -coll [2] ergibt 4 3 2	Mengendifferenz
*coll	coll1 *coll coll2→ coll1	{1 2 3} *coll {4 5} ergibt ZAHL,ZAHL m 1 4 1 5 2 4 2 5 3 4 3 5	kartesisches Produkt
:coll	coll1:coll coll2→ coll1	{1 2 3}:coll [2 3 4] ergibt {2 3}	Durchschnitt
*l	tbt *l ZAHL → tbt l	Auto *l 3 ergibt Auto Auto Auto oder xx.tab *l 3	Element zu Liste vervielfachen
*mat	coll1 *mat coll2→ coll	(1,2) *mat [2 3] ergibt 8	Matrizenmultiplikation
-1mat	coll -1mat → coll	<TAB! X1,X2,X3 l 1 0 2 0 2 0 0 0 8 !TAB> -1mat ergibt X1, X2, X3 l 1. -0. -0.25 -0. 0.5 -0. 0. -0. 0.125	inverse Matrix

&	BOOL & BOOL -> BOOL	si & no ergibt no	Konjunktion (logisches und)
or	BOOL or BOOL -> BOOL	1=1 or 1=2 ergibt si	Disjunktion (logisches oder)
&&	coll1 && → BOOL	si,66,si && ergibt si	für alle
ex	coll1 ex → BOOL	1=2,no ex ergibt no	es existiert
..	zahl1.. zahl2 →zahl1	1 .. 4 ergibt 1 2 3 4	von .. bis generieren
...	zahl1 ... zahl2 ! zahl3 →zahl1	0 ... 6!2 ergibt 0 2 4 6	von ... bis ! schritt
..x	zahl1 ... zahl2 ! ZAHL →zahl1	1 ..x 6!3 ergibt 5 3 2	Zufallszahlen von ..x bis ! anzahl
'3	tbt '3 → tbt	1234567890 '3 ergibt 1'234'567'890	formatiere in 3er Blöcke
'4	tbt '4 → tbt	12345.67898 '4 ergibt 1'2345.6789'8	formatiere in 4er Blöcke
^ hoch	tbt ^ tbt2 → tbt	4 ^ 1/2 ergibt 2. 10 *1 4 hoch (0 ..3) ergibt 1 10 100 1000	Potenz
abs	tbt abs → tbt	-3 7 abs ergibt (tabh) 3 7	absoluter Betrag
arctan	tbt arctan -> tbt	1 arctan ergibt 0.785398163397 (= pi:4)	arcusTangens
at		GROSS:=NET +% 19 at NET	neue Spalte rechts neben der angegebenen stellen
aus	tbt1 aus tbt2 → tbt2	aus fluesse.tabh	Neuanfang eines Programms
comp	tbt name → tbt	<TAB! NAME,VORNAME,ORT Mill Paul Halle	Komponente

		!TAB> comp ORT ergibt ORT Halle (siehe auch nth)	
cos	PZAHL cos → PZAHL	pi cos ergibt -1.	Kosinus
cut	text cut ZAHL -> list	qwertzui cut 3 ergibt qwe rtz ui	Einen Text in gleichgroße Stücke schneiden
det	coll det → coll	<TAB! X1,X2,X3 1 1 0 2 0 2 0 0 0 8 !TAB> det ergibt 16.	Determinante
div	ZAHL div ZAHL →ZAHL	11 div 5 ergibt 2	ganzzahlige Division
divrest	ZAHL divrest ZAHL → Paar	11 divrest 5 ergibt 2,1 (nicht 2.1)	ganzzahlige Division mit Rest
e	e	e ^ 3 ln ergibt 3.	Euler'sche Konstante
next		X:= 100 next X pred *1.03 at Y	Einfache rekursive Operation
nextonr		X:=100 foronr X pred *1.03 at Y	for für onr-Rekursion
gib	tbt1 gib schema → tbt2	aus studenten.tab gib FAK,(ORT,NAME m)m	Umstrukturierung eines Tabments bzgl. gegebenen Schemas und Aggregationen
gibagg	tbt gibagg aggscheme -> tbt	aus studenten.tab proj- STUNDEN STID gibagg KURS,++ m Hier werden die Summen für STIP und NOTE pro KURS berechnet.	Für alle Zahlen- Spalten wird in der geforderten Struktur die gewünschte Aggregation durchgeführt
giball	tbt1 giball scheme2 → tbt2	giball X Y 1 Liste aller X- und Y- Subtabmente (beliebige Tiefe); korrespondiert zu ...//X Y of XPath	Extraktion aller entsprechenden Werte

gibtop	tbt1 gibtop scheme2 → tbt2	gibtop X1 korrespondiert zu: t/X: Liste of all X- Subtabmente of t, aus der höchsten Ebene von t.	Extraktion der obersten Werte
if	tbt if bedingung ! tbt2	1 if 1=2 ! 3 ergibt 3	3-stellige Operation, die "if_then-else" und "case" ersetzt
in	tbt1 in tbt2 → BOOL	"1 2 1" in "1 2" ergibt si "1 2 3" in "1 1 2" ergibt no	Jedes Wort der linken Seite ist Wort der rechten Seite
inmath	tbt1 inmath tbt2 → BOOL	[1 3] inmath [1 4 3] ergibt si [3 1] inmath [1 4 3] ergibt no 2 inmath {6 7 2} ergibt si	mathematische Inklusion; die linke Seite legt fest, ob das Enthaltensein bzgl. Listen, Mengen oder Multimengen zu interpretieren ist
keys	tbt1 keys tbt2 → tbt1	X1:= 1 ..40 Y:=X*X gib X,Y m keys [7 34] ergibt tab-Format: X, Y m 7 49 34 1156 or keys [yy,[y2] zz]	effiziente Selektion in Mengen oder Listen
keyslike	tbt1 keyslike tbt2 → tbt1	<TAB! NAME, ORT m Clara Oehna Claudia Dallgow Sophia Dallgow !TAB> keyslike ["*ia"] results NAME, ORT m Claudia Dallgow Sophia Dallgow	effiziente Selektion in Mengen oder Listen mit teilweiser Übereinstimmung
leftat		GROSS:=NET +% 19 leftat NET	neue Spalte links neben der angegebenen stellen
like	term like "term?*" → BOOL	Hadmersleben like "? admers*" ergibt si '?': repräsentiert ein Zeichen '*': repräsentiert 0 oder mehr Zeichen	ähnlich

linreg	tbt linreg → num, num	<TAB! PREIS,VERKAUFT 1 20 0 16 3 15 7 16 4 13 6 10 10 !TAB> linreg ergibt 19.73214,-0.98214	lineare Regression
lists	tbt lists zahl → tbt 1	X1:= 1 2 lists 2 ergibt (tabh-Format) X1 1 1 1 1 2 2 1 2 2	Erzeugen einer Liste von Listen vorgegebener Länge
ln	tbt ln → PZAHL	euler ln ergibt 1.	natürlicher Logarithmus
log	tbt1 log tbt2 → PZAHL	100 log 10 ergibt 2.	allgemeiner Logarithmus
lower	tbt lower → tbt	AsdRRGee34 lower ergibt asdrree34	in Kleinbuchstaben verwandeln
mad	tbt mad → PZAHL	[1 2 5 3 5 1] mad ergibt 1.5	Streuung
max	tbt max → num	12.21,2,Hallo max ergibt 12.21	Maximum aller Zahlen
median	tbt median → num	1 2 4,9.9 median ergibt 3.0	Median
min	tbt min → num	12.21,2,Hallo min ergibt 2	Minimum aller Zahlen
no	no → BOOL	no or si ergibt si	Wahrheitswert falsch entspricht der Antwort nein (spanisch no)
not	BOOL not → BOOL	si not ergibt no	Negation
nth	tbt nth ZAHL → tbt'	1 3 5 nth 2 ergibt 3	n-te Komponente

nthpred	Name nthpred ZAHL → term	X1:= 1 2 3 4 Y:= X nthpred 2 ergibt X ,Y? 1 1 2 3 1 4 2	n-ter Vorgänger
nthsucc	tbt nthsucc ZAHL → tbt'	X1:=2 4 7 4 3 4 4 sel X nthsucc 2=4 ergibt (tabh) 4 4 3	n-ter Nachfolger
nthzahl	tbt nthzahl ZAHL →tbt'	"2023.03.26" nthzahl 3 ergibt 26	n-te Zahl in einem Text
onr	tbt onr → tbt	1 3 5.2 "4.5.5" onr ergibt (tabh): 1 3 5.2 4.5.5	Umwandlung in o++o Nummer
onrs	tbt onrs Name ! element → tbt'	<TAB! X,Ym m k y z y w !TAB> onrs OTTO!k results X, (OTTO, Y m) 1 k 1 z 2 y 2.1 w	erzeugt o++o Nummern in einer Tabelle; das ist eine wichtige Komponente der Stücklistenauflösung.
permutations	list permutations -> list1	2 4 9 permutations ergibt (tabh) ZAHL1 1 2 4 9 2 9 4 4 2 9 4 9 2 9 2 4 9 4 2	permutations ist eine Abkürzung für das Programm: X1:= 2 4 9 lists 3 sel Xm= {2 4 9}
pi	pi → PZAHL	KREISFLÄCHE:=R*R*pi	Kreiszahl
poly	num poly list → num	3 poly [1 2 3] ergibt 18	Polynom
pos	Name pos → ZAHL	sel X pos < 10	Position
pos-	Name pos- → ZAHL	sel X pos- > 5	Position von hinten

posjoin	tbt1 posjoin tbt2 → tbt	1 3 posjoin xx yy ergibt 1 xx 3 yy	Vielfachpaarbildung
posjoin			
pred	Name pred → term	X:= 100 for X pred *1.03	Vorgänger (predecessor)
preds	preds → Vorgängertupel		Abkürzung für X pred, Y pred, ... in for-Operation
proj	tbt proj Namen -> tbt'	<TAB! X,Y,Z m 1 2 3 4 5 6 !TAB> proj X Xm 1 4	Projektion: Nur die angegebenen Spaltennamen verbleiben
proj-	tbt proj- Namen → tbt'	<TABH! X,Ym m 1 2 3 4 5 !TABH> proj- Y ergibt (tab-format) Xm 1 4	Streichen von Spalten
pzahl	tbt pzahl → PZAHL	1/5 6 9.7 pzahl ergibt (tabh) 0.2 6. 9.7	Konvertierung in eine PZAHL
pzahl1de	tbt pzahl1de → PZAHL	"Heute bekomme ich 356,88 Euro und nicht 66,8 ." pzahl1de ergibt 356.88	Erste PZAHL eines deutschen Textes
rat	ZAHL rat ZAHL → RATIO	<TAB! X,Y1 1 1 2 3 !TAB> Z:= X rat Y ergibt X,(Y, Z 1) 1 1 2 1/2 3 1/3	Umwandlung zweier Zahlen in eine RATIO Zahl
ratio	num ratio → RATIO	1/5 6 9.7 ratio ergibt (tabh)	Konvertierung in rationale Zahl

		1/5 6/1 97/10	
rename	tbt rename Name1 ! Name2 →tbt'	rename X!Y	Umbenennung von Spaltennamen
rest	ZAHL rest ZAHL → ZAHL	13 rest 5 ergibt 3	Rest bei ganzzahliger Division
rnd	PZAHL rnd ZAHL → PZAHL	17.678 3.45 zz 8 rnd 1 ergibt 17.7 3.5 zz 8	runden
route	tbt route → tbt	<TAB! X,Y m 0 0 1 1 0 1 !TAB> route erzeugt 2 Linien von (0,0) zu (1,1) und von (1,1) zu (0,1)	aus Punktfolge Geradenfolge generieren
satzl	TEXT satzl TEXT1	"Es ist prima. Toll. Morgen feiern wir." satzl ergibt (tabh-format): SATZ1 Es ist prima. Toll. Morgen feiern wir.	Liste von Sätzen
seg	NAME seg → term	<TAB! X,Y,Z,U,V1 1 1 2 3 4 5 6 6 7 8 9 10 11 !TAB> SUM:= X seg ++ ergibt (tabh) X ,SUM ,Y ,Z ,U ,V1 1 1 10 2 3 4 5 6 6 30 7 8 9 10 11	Vollständiges Segment
sel	tbt sel bed → tbt	fluesse.tabh sel LÄNGE >800	Selektion (mit)
sel-	tbt sel- bed → tbt	sel- LOC=Magdeburg sel- Magdeburg sel-: ohne die spezifizierten Strupel	Selektion (ohne)
sepl	konstante Liste,	"," ";" " " " !" "?" "(" ")" "@ " #" "\n" "-" " "	alle Trennzeichen, die in der Operation cil benutzt werden

	nützlich um die Operation split (in Worte splitten) exakt zu verstehen		
selnat	tbt selnat →tbt	students.tab,exams.tab sel NAME=Ernst selnat exams enthält dann ebenfalls nur Prüfungen von Ernst	natürliche Selektion (bzgl. gemeinsamer Spaltennamen)
si	si → BOOL	si & no ergibt no	Wahrheitswert wahr (Antwort ja (=si))
sin	PZAHL sin → PZAHL	3.14159 sin ergibt 2.65358979335e-06	Sinus Funktion
split			
sqrt	num sqrt → PZAHL	4 sqrt ergibt 2.	Quadratwurzel
subtext	text subtext ZAHL ! ZAHL → TEXT	aBCdE subtext 2 ! 3 ergibt BCd	Teiltext (substring)
subtext2	text subtext2 text ! text → TEXT	aBCdEfgh subtext2 "B"! fg ergibt CdE	Teiltext des ersten Textes der zwischen den beiden anderen gegebenen Texte liegt.
succ	Name succ → term	NOTE1:= 3 1 2 1 sel NOTE >NOTE succ ergibt NOTE1 3 2	Nachfolger (successor)
tag	tbt tag NAME! scheme → tbt'	ORT:=Magdeburg STRASSE:=Beims tag ADRESSE!ORT,STRASSE ergibt (metadaten) TABMENT ! ADRESSE ADRESSE ! ORT,STRASSE ORT ! WORT STRASSE ! WORT	umschlieÙe Daten eines Schemas mit einem Tag
tag0	tbt tag0 Name → tbt'	11 13 tag0 XX ergibt (ment) <XX> 11 13 </XX>	Einen Tag um das gesamte Tabment setzen
tan	num tan → PZAHL	3.14 tan ergibt	Tangens Funktion

		-0.00159265493641	
text	mixe text → TEXT	3.14 ttt 8 text ergibt TEXT1 3.14 ttt 8	Jeden elementaren Typ in TEXT transformieren.
textend	mixe textend ZAHL → TEXT	asdfgh textend 4 ergibt fgh abcde textend -2 ergibt de	hinterer Teiltext
textindex	text mixe text → ZAHL	"Heute ist Dienstag." textindex Di ergibt ZAHL 11	Position
time	time → PZAHL	time könnte ergeben: 1.557021	Systemzeit (nur die Differenz zweier solcher Zeiten ist für Effizienzbetrachtungen bedeutungsvoll)
trim	text trim → text	" Hi o++o " trim ergibt (ment) <TABM>Hi o++o</TABM>	hinten und vorn Leerzeichen entfernen
tup	NAME tup → term	<TAB! X,Y,Z,U,V1 1 1 2 3 4 5 6 6 7 8 9 10 11 !TAB> SUM:= Z tup ++ ergibt (tabh) X ,Y ,Z ,SUM ,U ,V1 1 1 2 3 21 4 5 6 6 7 8 51 9 10 11	Gesamtes Strupel
untag0	tbt untag0 → tbt'	X:=1 untag0 ergibt ZAHL 1	den äußersten Tag entfernen
upper	text upper → text	1.2,aw upper ergibt (tab-format) PZAHL, WORT 1.2 AW	in Großbuchstaben umwandeln
variance	tbt variance → PZAHL	[1 2 4 6] variance ergibt 4.91666666667	Varianz

vlists	tbt vlists ZAHL → tbt 1	variabel lange Listen; die Operation stimmt mit lists überein nur dass alle kürzeren Listen ebenfalls im Resultat erscheinen.	Liste von Listen variabler Länge
while	Scheme:= expr while expr ! expr -> tbt		Generiert Tabelle mit allen berechneten Werten
wort	mixe wort → wort	"Ich bin gut.Du auch." wort ergibt WORT Ich_bin_gut.Du_auch.	in Worte konvertieren
zahl	num zahl → ZAHL	"12" zahl ergibt 12 3.14 zahl ergibt 3	Konvertierung
zahltrip	text zahltrip -> Tripel von Zahlen	TAG,MON,JAHR:= 26.03.1963 zahltrip ergibt TAG, MON, JAHR 26 3 1963	Die ersten 3 Zahlen eines Textes
cut	tbt cut int → tbt'	123,Heute cut 1 ergibt (tabh) 1 2 3, H e u t e	Liste aller Zeichen (Buchstaben+Ziffern+ Sonderzeichen) vorgegebener Länge)
zahlratio	RATIO zahlratio -> ZAHL,RATIO	33/7 zahlratio ergibt 4 5/7	in ganzzahligen Anteil und echten Bruch umwandeln
zahl1de	text zahl1de → ZAHL	"Heute bekomme ich 66.356,11 Euro" zahl1de ergibt 66356	erste Zahl aus einem deutschen Text extrahieren

21 Anhang B: Liste der o++o Farbnamen

"aliceblue",(0.941176470588,0.972549019608,1.);
"antiquewhite",(0.980392156863,0.921568627451,0.843137254902);
"aquamarine",(0.498039215686,1.,0.83137254902);
"azure",(0.941176470588,1.,1.);
"beige",(0.960784313725,0.960784313725,0.862745098039);
"bisque",(1.,0.894117647059,0.76862745098);
"black",(0.,0.,0.);
"blanchedalmond",(1.,0.921568627451,0.803921568627);
"blue",(0.,0.,1.);
"blueviolet",(0.541176470588,0.16862745098,0.886274509804);
"brown",(0.647058823529,0.164705882353,0.164705882353);
"burlywood",(0.870588235294,0.721568627451,0.529411764706);
"cadetblue",(0.372549019608,0.619607843137,0.627450980392);
"chartreuse",(0.498039215686,1.,0.);
"chocolate",(0.823529411765,0.411764705882,0.117647058824);
"coral",(1.,0.498039215686,0.313725490196);
"cornflowerblue",(0.392156862745,0.58431372549,0.929411764706);
"cornsilk",(1.,0.972549019608,0.862745098039);
"cyan",(0.,1.,1.);
"darkgoldenrod",(0.721568627451,0.525490196078,0.043137254902);
"darkgreen",(0.,0.392156862745,0.);
"darkkhaki",(0.741176470588,0.717647058824,0.419607843137);
"darkolivegreen",(0.333333333333,0.419607843137,0.18431372549);
"darkorange",(1.,0.549019607843,0.);
"darkorchid",(0.6,0.196078431373,0.8);
"darkred",(0.5450,0.,0.);
"darksalmon",(0.913725490196,0.588235294118,0.478431372549);
"darkseagreen",(0.560784313725,0.737254901961,0.560784313725);
"darkslateblue",(0.282352941176,0.239215686275,0.545098039216);
"darkslategray",(0.18431372549,0.309803921569,0.309803921569);
"darkturquoise",(0.,0.807843137255,0.819607843137);
"darkviolet",(0.580392156863,0.,0.827450980392);
"deeppink",(1.,0.078431372549,0.576470588235);
"deepskyblue",(0.,0.749019607843,1.);
"dimgrey",(0.411764705882,0.411764705882,0.411764705882);
"dodgerblue",(0.117647058824,0.564705882353,1.);
"firebrick",(0.698039215686,0.133333333333,0.133333333333);
"floralwhite",(1.,0.980392156863,0.941176470588);
"forestgreen",(0.133333333333,0.545098039216,0.133333333333);
"gainsboro",(0.862745098039,0.862745098039,0.862745098039);
"ghostwhite",(0.972549019608,0.972549019608,1.);
"gold",(1.,0.843137254902,0.);
"goldenrod",(0.854901960784,0.647058823529,0.125490196078);
"green",(0.,1.,0.);
"greenyellow",(0.678431372549,1.,0.18431372549);
"grey",(0.745098039216,0.745098039216,0.745098039216);
"honeydew",(0.941176470588,1.,0.941176470588);
"hotpink",(1.,0.411764705882,0.705882352941);
"indianred",(0.803921568627,0.360784313725,0.360784313725);
"ivory",(1.,1.,0.941176470588);
"lavender",(0.901960784314,0.901960784314,0.980392156863);

"lavenderblush",(1.,0.941176470588,0.960784313725);
 "lawngreen",(0.486274509804,0.988235294118,0.);
 "lemonchiffon",(1.,0.980392156863,0.803921568627);
 "lightblue",(0.678431372549,0.847058823529,0.901960784314);
 "lightcoral",(0.941176470588,0.501960784314,0.501960784314);
 "lightcyan",(0.878431372549,1.,1.);
 "lightgoldenrod",(0.933333333333,0.866666666667,0.509803921569);
 "lightgray",(0.827450980392,0.827450980392,0.827450980392);
 "lightpink",(1.,0.713725490196,0.756862745098);
 "lightsalmon",(1.,0.627450980392,0.478431372549);
 "lightseagreen",(0.125490196078,0.698039215686,0.666666666667);
 "lightskyblue",(0.529411764706,0.807843137255,0.980392156863);
 "lightslateblue",(0.517647058824,0.439215686275,1.);
 "lightslategray",(0.466666666667,0.533333333333,0.6);
 "lightsteelblue",(0.690196078431,0.76862745098,0.870588235294);
 "lightyellow",(1.,1.,0.878431372549);
 "limegreen",(0.196078431373,0.803921568627,0.196078431373);
 "linen",(0.980392156863,0.941176470588,0.901960784314);
 "ltgoldenrodyello",(0.980392156863,0.980392156863,0.823529411765);
 "magenta",(1.,0.,1.);
 "maroon",(0.690196078431,0.188235294118,0.376470588235);
 "mediumaquamarine",(0.4,0.803921568627,0.666666666667);
 "mediumblue",(0.,0.,0.803921568627);
 "mediumorchid",(0.729411764706,0.333333333333,0.827450980392);
 "mediumpurple",(0.576470588235,0.439215686275,0.858823529412);
 "mediumseagreen",(0.235294117647,0.701960784314,0.443137254902);
 "mediumslateblue",(0.482352941176,0.407843137255,0.933333333333);
 "mediumturquoise",(0.282352941176,0.819607843137,0.8);
 "mediumvioletred",(0.780392156863,0.0823529411765,0.521568627451);
 "medspringgreen",(0.,0.980392156863,0.603921568627);
 "midnightblue",(0.0980392156863,0.0980392156863,0.439215686275);
 "mintcream",(0.960784313725,1.,0.980392156863);
 "mistyrose",(1.,0.894117647059,0.882352941176);
 "moccasin",(1.,0.894117647059,0.709803921569);
 "navajowhite",(1.,0.870588235294,0.678431372549);
 "navyblue",(0.,0.,0.501960784314);
 "oldlace",(0.992156862745,0.960784313725,0.901960784314);
 "olivedrab",(0.419607843137,0.556862745098,0.137254901961);
 "orange",(1.,0.647058823529,0.);
 "orangered",(1.,0.270588235294,0.);
 "orchid",(0.854901960784,0.439215686275,0.839215686275);
 "palegoldenrod",(0.933333333333,0.909803921569,0.666666666667);
 "palegreen",(0.596078431373,0.98431372549,0.596078431373);
 "paleturquoise",(0.686274509804,0.933333333333,0.933333333333);
 "palevioletred",(0.858823529412,0.439215686275,0.576470588235);
 "papayawhip",(1.,0.937254901961,0.835294117647);
 "peachpuff",(1.,0.854901960784,0.725490196078);
 "peru",(0.803921568627,0.521568627451,0.247058823529);
 "pink",(1.,0.752941176471,0.796078431373);
 "plum",(0.866666666667,0.627450980392,0.866666666667);
 "powderblue",(0.690196078431,0.878431372549,0.901960784314);
 "purple",(0.627450980392,0.125490196078,0.941176470588);
 "red",(1.,0.,0.);

"rosybrown",(0.737254901961,0.560784313725,0.560784313725);
 "royalblue",(0.254901960784,0.411764705882,0.882352941176);
 "saddlebrown",(0.545098039216,0.270588235294,0.0745098039216);
 "salmon",(0.980392156863,0.501960784314,0.447058823529);
 "sandybrown",(0.956862745098,0.643137254902,0.376470588235);
 "seagreen",(0.180392156863,0.545098039216,0.341176470588);
 "seashell",(1.,0.960784313725,0.933333333333);
 "sienna",(0.627450980392,0.321568627451,0.176470588235);
 "silver",(0.898039215686, 0.898039215686, 0.898039215686);
 "skyblue",(0.529411764706,0.807843137255,0.921568627451);
 "slateblue",(0.41568627451,0.352941176471,0.803921568627);
 "slategrey",(0.439215686275,0.501960784314,0.564705882353);
 "snow",(1.,0.980392156863,0.980392156863);
 "springgreen",(0.,1.,0.498039215686);
 "steelblue",(0.274509803922,0.509803921569,0.705882352941);
 "tan",(0.823529411765,0.705882352941,0.549019607843);
 "thistle",(0.847058823529,0.749019607843,0.847058823529);
 "tomato",(1.,0.388235294118,0.278431372549);
 "turquoise",(0.250980392157,0.878431372549,0.81568627451);
 "violet",(0.933333333333,0.509803921569,0.933333333333);
 "violetred",(0.81568627451,0.125490196078,0.564705882353);
 "wheat",(0.960784313725,0.870588235294,0.701960784314);
 "white",(1.,1.,1.);
 "whitesmoke",(0.960784313725,0.960784313725,0.960784313725);
 "yellow",(1.,1.,0.);
 "yellowgreen",(0.603921568627,0.803921568627,0.196078431373)