LÖSUNGSSTRATEGIE ZUM PROGRAMM

Rechner-Testkonfiguration Softwarevoraussetzungen:

l i 486
l 8 MB RAM
l Graphic-Accellerator-Karte
l Compileroptionen:
  - Stack: 16000
  - Heap: 8192

l Softwarevoraussetzungen:
  - MS-DOS 6.2
  - WINDOWS 3.1
  - Turbo-PASCAL für WINDOWS 1.5 oder Borland PASCAL 7.0 für WINDOWS

Compiler-Optionen:

Optionen	
– **Bereichsüberprüfung:**	**4**
– **Stack-Prüfung:**	**4**
– **I/O-Prüfung:**	**4**
– **FAR-Aufrufe erzeugen:**	**7**
– **Debug-Informationen:**	**4**
– **Lokale Symbole:**	**4**
– **286-Code:**	**4**
– **80x87-Code:**	**7**
– **Windows Stack Frame:**	**7**
– **Erweiterte Syntax:**	**4**
Datenausrichtung	**WORD**
Speichergrößen	
– **Stack:**	**16000**
– **HEAP:**	**8192**
Stringprüfung	**Streng**
Boolsche Auswertung	**Vollständig**

Vorbemerkung:

Nach Versuchen des Programmlaufes mit Schleifen wurde das Programm umgearbeitet und die EXIT-Prozedur eingearebeitet. Das ist nach Meinung des Autors genau dann zulässig, wenn die Wirkung die gleiche ist, wie unter Nutzung von Schleifen oder Rekursion, aber die logische Konstruktion dieser unlogisch kompliziert und der Programmlauf unübersichtlich ist. Natürlich wird dann gegen die Grundregeln der Programmierung verstoßen. Strukturiert programmiert wird aber in dem Augenblick, wo die Sprünge nach vor erfolgen (die wird mit EXIT eingehalten), und nicht zurück. Das Programm hat einen Vorwärtsablauf mit dem Zusatz, daß bequem unter bestimmmten Zuständen nicht gewünschte Programmteile ausgelassen werden.

Programmbeschreibung:

Konzeptioneller Gedanke des Programmes ist es, durch fortlaufendes Anfügen einzugebender Wortpaare zu einer Übereinstimmung zweier Zielworte zu gelangen. Dabei wird das Anfügen ausgehend von mindestens einem Wortpaar, welches mit gleichem Zeichen beginnt, solange vorgenommen, wie der gemeinsame Wortstamm noch übereinstimmend ist. Ist dies nicht der Fall, ist auch keine Korrespondenz möglich und die weitere Suche mit dieser Wortkombination wird abgebrochen. Die Stelle bis zur letzten Übereinstimmung des Wortstammes wird aufgesucht (das ist, da nach jedem Anfügen verglichen wird, immer die vorletzte), und mit dem Wort des nachfolgenden Index des wieder abgeschnittenen wird angefügt und erneut getestet. Im praktischen Sinne bedeutet dies, alle Wortpaarkombinationen mit vorgegebener Anzahl von Anfügeoperationen zu „probieren“, solange, bis alle Wortkombinationen, welche einen gemeinsamen Wortstamm haben und innerhalb der angegebenen Anfügetiefe liegen, getestet sind. Ist dies nicht der Fall, oder wird Korrespondenz gefunden, kommt es zur Auswertung!

Anliegen des Programmes ist es aus technischen Gründen nicht **(verfügbarer freier Speicherbereich in Extremfällen!)**, jede denkbare Korrespondenz zu finden (für die wohl größte Anzahl von Wortpaaren ergibt sich ohnehin keine solche, dagegen folgt aus einer bestimmten (sehr begrenzten!) Menge von Wortpaaren fast nach jedem Anfügen Korrespondenz!). Das Programm findet unter bestimmten Umständen (**wenn die ersten beiden gefundenen Worte schon übereinstimmen!**) Korrespondenz und quittiert den weiteren Dienst mit eben dieser Wortpaakombination. Dagegen sucht es für Wortpaarfolgen, wo **nur ein Wortppaar** mit gleichem Zeichen beginnt, in vorgegebenen Rahmen (Suchstring nicht größer als 65535!!!) nach den möglichen Korrespondenzen!

Einschränkungen:

Bedingt durch die Größe des zur Verfügung stehenden Hauptspeichers und die extreme Laufzeit des Programmes müssen Einschränkungen hinsichtlich der Anzahl der einzugebenden Worte (max. 20), der Größe derselben (max. 5 Zeichen) sowie auch der Tiefe des Anfügens an Testworte (65000), vorgenommen werden. Die Entscheidung zugunsten von Turbo-PASCAL für WINDOWS fiel aus dem Grunde der dort verfügbaren C-Strings mit ihrer Länge von maximal 65535 Zeichen. Dies läßt eine größere Suchtiefe für das Anfügen an die Teststrings zu. Es ist nicht möglich, nach Abschluß des Suchvorganges insgesamt, alle gefundenen Korrespondenzen aufzulisten. Es erfolgt dies während des Programmlaufes als „Zwischenergebnis“.

Kurzbeschreibung der wichtigsten Konstanten:

1. hilfstext Typ **BOOLEAN** ist die Verwaltungsgröße (vordefinierte Variable) zur Einarbeitung von Hilfetexten.
2. abbruch Typ **BOOLEAN** ist die vordefinierte Variable, die über den Programmabbruch entscheidet. Sie wird als Variablenparameter durch alle Prozeduren, die eine Ausschrift auf dem Monitor beinhalten, durchgereicht.
3. index_dimension Typ **WORD** ist die Konstante, die über die Dimension des Indexfeldes der Ergebnisindizies entscheidet. Zu große Dimensionen ergeben Compilerfehler wegen Segmentüberschreitung.
4. c_dimension Typ **WORD** ist die Konstante, welche die Dimension der verwendeten C-Strings beinhaltet. Alle im Programm verwendeten C-Strings werden mit eben dieser Größe dimensioniert.

Kurzbeschreibung der wichtigsten Variablen:

Hinweis:

- **Lokale Variablen werden mit einem vorangestellten „l“ gekennzeichnet also: l_taste**
- **formale Parameter in Prozeduren werden mit einem vorangestellten „f“ gekennzeichnet also: f_exitcode**

1. wortpaar [Wortnummer,x- oder y-Wort] Typ **STRING[5]** ist das Eingabefeld von PASCAL-Strings der Länge 5 (max. Zeichenzahl) mit der Dimension 20 (Menge der Worte),2 (Index für das Wort x oder y).
2. anfangsindex [Wortnummer] Typ **BOOLEAN** ist ein Feld von Wahrheitswerten, in denen die Indizies der Wortpaare TRUE gesetzt werden, die mit gleichem Zeichen beginnen. In der Folge wird nur mit diesen Worten beginnend nach Korrespondenz gesucht.
3. geht Typ **BOOLEAN** ist ein Wahrheitswert, in dem festgehalten wird, ob es ein Wortpaar gibt, welches mit gleichem Zeichen beginnt, und damit Korrespondenz generell möglich wird. Die Entscheidung dazu übernimmt die Prozedur **gehts_denn_ueberhaupt**.
4. testwort_x und testwort_y Typ **PCHAR** sind C-Strings, die die, auf Korrespondenz zu prüfenden Worte aufnehmen.
5. indiezies [wortdimension] Typ **BYTE** ist das Registraturfeld der Indizies der Worte, mit denen erfolgreich getestet wurde.
6. control Typ **BOOLEAN** Steuerungsvariable, welche über die Anzeige von Zwischenergebnissen entscheidet.
7. abbruch Typ **BOOLEAN** Steuerungsvariable, welche über den Programmabbruch entscheidet. Da sie in alle Prozeduren als Variablenparameter übergeben wird, kann ein Programmabbruch von jeglicher Programmtiefe heraus erreicht werden.
8. exitcode Typ **BOOLEAN** Steuerungsvariable, die erkennt, ob der gemeinsame Wortstamm der Testworte nach einem Vergleich noch übereinstimmt. Die Verwaltung der notwendigen Steueroperationen wird dann von der Prozedur **indexverwaltung** übernommen.
9. gleich Typ **BOOLEAN** Steuerungsvariable, die erkennt, ob es Worte innerhalb des Eingabefeldes gibt, die bei gleichem Index auch mit gleichem zeichen beginnen. Das erkennt dann die Prozedur **gehts_denn_ueberhaupt**.

10. l_key Typ **CHAR** ist eine lokale Steuerungsvariable, die in jeder Prozedur den Zustand der Tastatur überwacht und auswertet.
11. index_lesen Typ **BOOLEAN** ist eine Steuerungsvariable, die in der Prozedur **indexverwaltung** entscheidet, ob das Indexfeld gelesen, oder geschrieben wird.
12. index_lesen Typ **BOOLEAN** ist eine Steuerungsvariable, die in der Prozedur **indexverwaltung** entscheidet, ob das Indexfeld gelesen, oder geschrieben wird.
13. aktueller_index Typ **BYTE** ist die Nummer des jeweils aktuellen Index'. Da die Vorgabe des Programmes nicht mehr als 20 Worte vorsieht, ist der Datentyp BYTE vollkommen gerechtfertigt.
14. aktuelle_indexnummer Typ **WORD** ist die Nummer des jeweiligen Feldelements, welches als Inhalt die Nummer des Wortpaares verwaltet, daß Korrespondenz erbringen soll.
15. ermittlungstiefe Typ **WORD** ist die Variable, welche darüber entscheidet, wie oft Anfügeoperationen wiederholt werden und wie groß die Suchstrings theoretisch werden können. Angewandt wird sie mit zwei Zielstellungen:

– Anzahl der Anfügeoperationen an einen Ausgangsstring (der mit gleichem Zeichen beginnen muß)
– Wiederholungstiefe für fehlgeschlagene Anfügeversuche (wenn es keinen gemeinsamen Wortstamm nach dem Anfügen gibt)

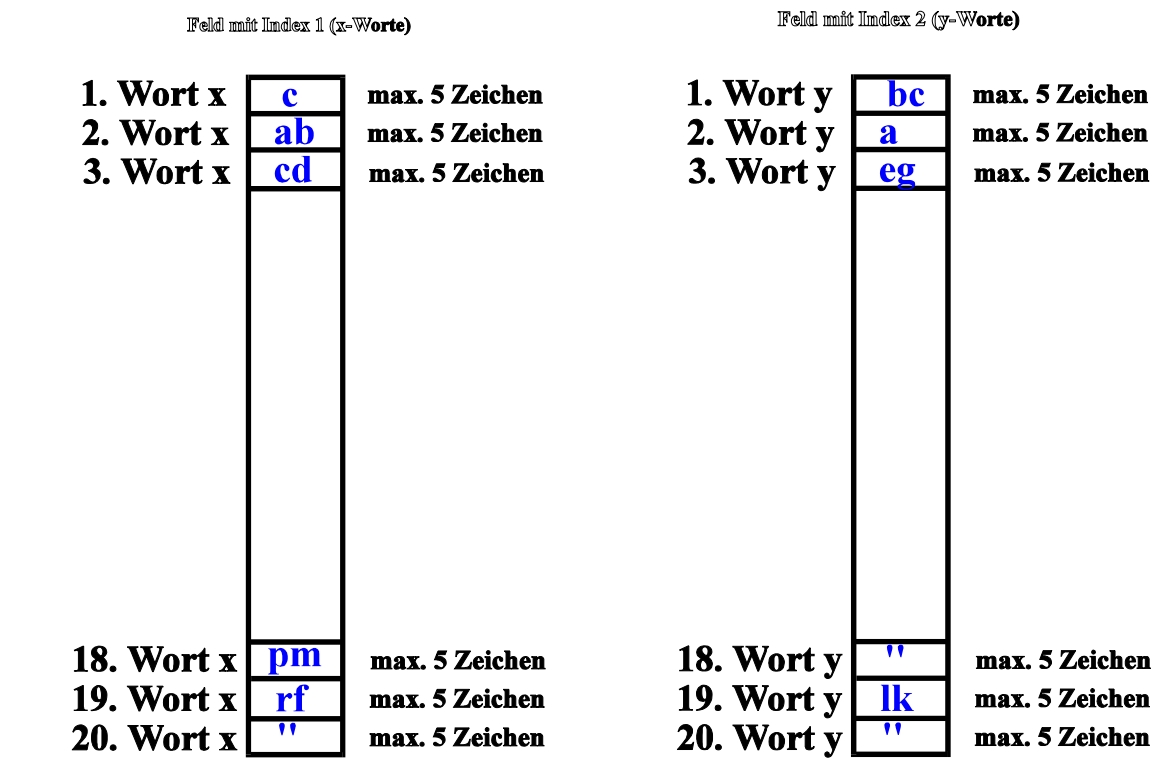
Da das Programm zwei Betriebsmodi besitzt, hat auch diese Variable verschiedene Inhalte. Bei automatischer Abarbeitung wird 65535 gesetzt, bei Protokollführung begrenzt auf 50!

16. anfuegetiefe Typ **WORD** registriert, wie oft Anfügeoperationen schon wiederholt wurden.
17. tiefenregistratur Typ **WORD** registriert die Anzahl der fehgeschlagenen Anfügeversuche.

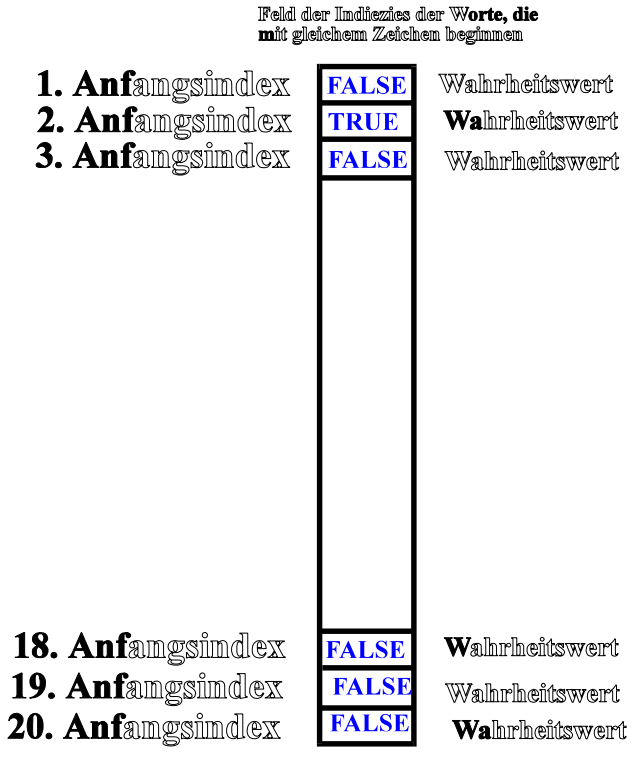
Datenstruktur der wichtigen Variablen:

wortpaar [Wortnummer,x- oder y-Wort] - **Eingabefeld von PASCAL-Strings:**

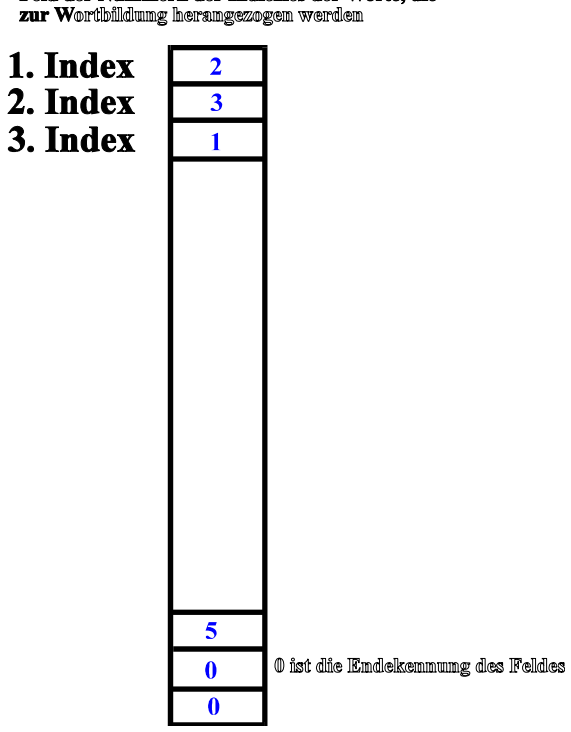
Zweidimensionales Feld von PASCAL-Strings



anfangsindex [Wortnummer] - **Feld von BOOLEAN-Variablen, die im Inhalt das Feststellen eines gemeinsamen Anfanges in´m Wortstamm von Worten aus x und x wiederspiegeln:**



indizies [aktuelle_indexnummer] - **Feld von BYTE-Variablen, die im Inhalt die Nummer des jeweils zur Wortbildung herangezogenen Index speichern:**



testwort_x,testwort_y - **C-Strings:**

C-String mit der maximalen Dimension 65536

a	b	c		d	c	#0

Beschreibung der Prozeduren:

Legende:

Þ Variable geht als Werteparameter in die Prozedur ein. Wird sie dort verändert, geht die Änderung nicht in die rufende Prozedur zurück.

Û **VAR** Variable geht als Variablenparameter in die Prozedur ein und verändert aus der Prozedur heraus.

```
PROGRAM main_routine;
```

Übergabeparameter:

Þ keine

l Steuert und kontrolliert den gesamten Programmablauf bis zum Schluß des Programmes.

```
PROCEDURE init;
```

Übergabeparameter:

Þ keine

l übernimmt die Fensterinitialisierung.
l Verwendet werden die Parameter einer S-VGA-Karte mit einer Aflösung von 1024 ´ 768 Pixeln.

```
PROCEDURE gehts_denn_ueberhaupt;
```

Übergabeparameter:

keine

l erkennt, ob Korrespondenz überhaupt möglich ist (das ist genau dann der Fall, wenn es mindestens ein Wortpaar gibt, welches mit gleichem Zeichen beginnt).
l es wird die Menge der Indizies ermittelt, für die oben genanntes zutrifft.

```
PROCEDURE indexverwaltung;
```

Übergabeparameter:

```
·   f_index_loeschen:BOOLEAN;
·   f_index_lesen:BOOLEAN;
·   VAR f_aktuelle_indexnummer:WORD;
·   f_aktueller_index:BYTE;
·   f_control:BOOLEAN;
·   VAR f_abbruch:BOOLEAN;
```

l verwaltet ein Feld der Indiezies der Worte, die zumindest am Versuch der Korrespondezfindung beteiligt sind. Dies geschieht aus zweierlei Hinsicht:

1. So kann das Indexfeld am Schluß ausgewiesen werden
2. Registriert dieses Feld vor allem den letzten Index, bei dem vor Wortanhängen der Wortstamm noch gleich und damit Korrespondenz möglich war. Dieses letzte Wort muß bei Erreichen von Ungleichheit im Wortstamm wieder abgeschnitten und mit dem nachfolgenden Wort erneut die Suche nach Korrespondenz aufgenommen werden.

l das Indexfeld kann beschrieben (erweitert), der letzte Index gelöscht (wenn mit der aktuellen Wortfolge nach anhängen des letzten Wortes keine Korrespondenz möglich ist) und das Feld kann nach erfolgter Auswertung zwecks Anzeige gelesen werden.
l vor Aufruf dieser Prozedur müssen die entsprechenden Wahrheitswerte richtig gesetzt sein, da sonst eine Fehlsteuerung der Indizies erfolgt.

```
PROCEDURE text_ausgabe;
```

Übergabeparameter:

```
Þ   f_dateiname:STRING;
Û   VAR f_abbruch:BOOLEAN;
```

- verwaltet die Textausgabe, wenn dies durch Nutzeranfoderung (Variable control=TRUE) oder durch das Hauptmenü angefordert wird.
- Dabei erfolgt die Steuerung seitenweise (je 18 Zeilen) bis zum Textende. Es sei denn, der Nutzer bricht mit ESCAPE ab.

```
PROCEDURE wortpaar_anzeigen;
```

Übergabeparameter:

```
Þ   f_abbruch:BOOLEAN;
Û   VAR f_richtige_eingabe:BOOLEAN;
```

- Listet die eingegebenen Wortfolgen nochmals auf.
- Die Liste wird mit der Möglichkeit der Korrektur abgeschlossen. In diesem Falle werden alle Parameter zurückgesetzt und die Eingabeprozedur erneut aufgerufen.

```
PROCEDURE wortpaare_einlesen;
```

Übergabeparameter:

```
Û   VAR f_abbruch:BOOLEAN;
```

- Liest die Wortfolgen mit Registrierung des jeweiligen Index (einfach die Nummer des aktuellen Wortpaares) ein.
- Das Einlesen erfolgt aus Kapazitäts- und Laufzeitgründen jedoch nur bis zu 20 Wortpaaren, das Einlesen wird danach zwangsläufig abgebrochen.

```
PROCEDURE wortstamm_bilden;
```

Übergabeparameter:

Þ keine

- erzeugt in den Testworten x und y je einen String, welche mindestens einen gemeinsamen Wortstamm haben.
- Der Wortstamm ist dabei das Abblild des kürzeren Strings im längeren - und diese müssen gleich sein!.

```
PROCEDURE wortvergleich;
```

Übergabeparameter:

```
Þ   f_control:BYTE;
Û   VAR f_gleich:BOOLEAN;
Û   VAR f_abbruch:BOOLEAN;
Þ   f_testwort_x:PCHAR;
Þ   f_testwort_y:PCHAR;
Þ   f_cstring_x:PCHAR;
Þ   f_cstring_y:PCHAR;
```

- Diese Prozedur findet nach jedem Anfügevorgang das jeweils kürzere Wort, worauf in je eine Hilfsvariable dieses und der Teilstring des längeren Wortes eingetragen werden. Daraufhin werden die beiden so entstandenen Strings verglichen und das Ergebnis (gleich oder ungleich) in die BOOLEAN-Variable **f_gleich** eingetragen.

```
PROCEDURE anhaengen;
```

Übergabeparameter:

```
Þ   f_i:BYTE;
Þ   f_j:BYTE;
Þ   f_control:BOOLEAN;
Û   VAR f_exitcode:BOOLEAN;
Û   VAR f_abbruch:BOOLEAN;
```

l Diese Prozedur hängt an die jeweiligen Ausgangsstrings in den Testworten für x und y die aktuellen Strings aus den Worten mit dem aktuellen Index an.
l Danach ruft diese Prozedur die Prozedur **wortvergleich** auf.

```
PROCEDURE heureka;
```

Übergabeparameter:

```
Þ   f_rekursionstiefe:WORD;
Þ   f_control:BOOLEAN;
Û   VAR f_exitcode:BOOLEAN;
Û   VAR f_abbruch:BOOLEAN;
```

l Ist generell Korrespondenz möglich, was die Prozedur **gehts_denn_ueberhaupt** feststellt, sucht diese Prozedur nach der Korrespondenz.

```
PROCEDURE einlesen_und_organisieren;
```

Übergabeparameter:

```
Û   VAR f_abbruch:BOOLEAN;
```

l Die Prozedur **einlesen_und_organisieren** verwaltet, ausgehend vom Hauptmenü, alle nachfolgenden Programmschritte.

```
PROCEDURE seitenende;
```

Übergabeparameter:

```
Û   VAR f_abbruch:BOOLEAN;
```

l Die Prozedur **seitenende** verwaltet Ausgabe von >ESCAPE< bzw. >ENTER< sowie die Beendigung des Programmes mit den aktuellen Parametern.

Rufstruktur der Prozeduren:

Anmerkung: Die Prozedur **seitenende** wird von fast allen Prozeduren gerufen und ist für den Programmablauf nicht von Bedeutung

→`main_routine`

ruft:

1. `init`
2. `main_menu`

→`init`

ruft keine weitere Prozedur

→`main_menu`

ruft bei eingegebener 1

1. `textausgabe` und danach
2. `einlesen_und_organisieren`

ruft bei eingegebener 2

`einlesen_und_organisieren`

ruft bei eingegebener 3

`main_menu` (zur Abfrage auf Abbruchbestätigung)

→`einlesen_und_organisieren`

ruft:

1. wortpaare_einlesen
2. wortpaar_anzeigen
3. pruefen

→wortpaare_einlesen

ruft keine weitere Prozedur

→wortpaar_anzeigen

ruft keine weitere Prozedur

→pruefen

ruft:

1. gehts_denn_ueberhaupt
2. wortstamm_bilden
3. heureka
4. indexverwaltung

→gehts_denn_ueberhaupt

ruft:

1. indexverwaltung

→wortstamm_bilden

ruft keine weitere Prozedur

→heureka

ruft:

1. text_ausgabe (auf Nutzeranforderung)
2. anhaengen
3. indexverwaltung

→anhaengen

ruft:

1. indexverwaltung
2. wortvergleich
3. indexverwaltung (in Abhängigkeit vom Programmlauf)
4. wortstamm_bilden (in Abhängigkeit vom Programmlauf)

→indexverwaltung

ruft keine weiteren Prozeduren

→wortvergleich

ruft keine weiteren Prozeduren

→wortstamm_bilden

ruft keine weiteren Prozeduren