Praktische Umsetzung von Algorithmen in Programme

Referat von:

Guberac Anita

Zivko Manuela

Was versteht man unter einem Algorithmus?

- Unter einem Algorithmus versteht man eine genau definierte Handlungsvorschrift zur

  Lösung eines Problems oder einer bestimmten Art von Problemen in endlich vielen Schritten

Eigenschaften eines Algorithmus:

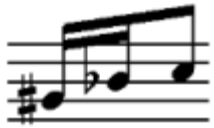
- ***KORREKTHEIT!!***
- ***FINITHEIT***
- ***AUSFÜHRBARKEIT***
- ***DYNAMISCHE FINITHEIT***
- ***TERMINIERUNG***
- ***DETERMINIERTHEIT***
- ***DETERMINISMUS***

Darstellungsformen von Algorithmen

- *Natürlich-sprachlich:*

✓ konventioneller Sprachstil (Kochrezepte: Man nehme…)

✓ in abgekürzter Form (Gebrauchsanweisungen: Stecker A in Position B stecken, dann Schrauben lösen)

Prozess	Ausführender	Algorithmus	Anweisung
Kuchenbacken	Bäcker	Rezept	Nimm 500 g Mehl
Spielen einer Melodie	Sänger, Instrumentalist	Ton	
Bedienung eines Handys	Anrufer	Bedienungs-anleitung	Drücke die Taste #
Kassieren im Supermarkt	Kassierer/in	Bedienungsplan für Registrierkasse	Eintippen von 237

Darstellungsformen von Algorithmen

- *als Diagramme:*

  - ✓Flußdiagramme
  - ✓Blockdiagramme

  - *in Programmen*

Textmanipulation

- Daten bzw. Symbole manipulieren heißt:

  Vergleichen, Löschen, Ersetzen, Umstellen, Einfügen

- Für das Manipulieren sind viele typische, häufig wiederkehrende Operationen vorprogrammiert
- Die Verarbeitungsanweisungen werden dem Computer per Programmiersprache mitgeteilt

Stringhandling unter TURBO PASCAL

Die wichtigsten Facts:

- Variablen zum Speichern → Datentyp STRING
- STRINGS können als Zeichenkette über entsprechende Funktionen angegangen werden → SUBSTR.
- oder direkte Ansteuerung
- Teile von variabel langen Strings → über SUBSTR, SUBSTR2, Procedure SUBS2

Segmentierung

- ***Zergliederungsalgorithmus:***

  vom H-Text zum V-Text:

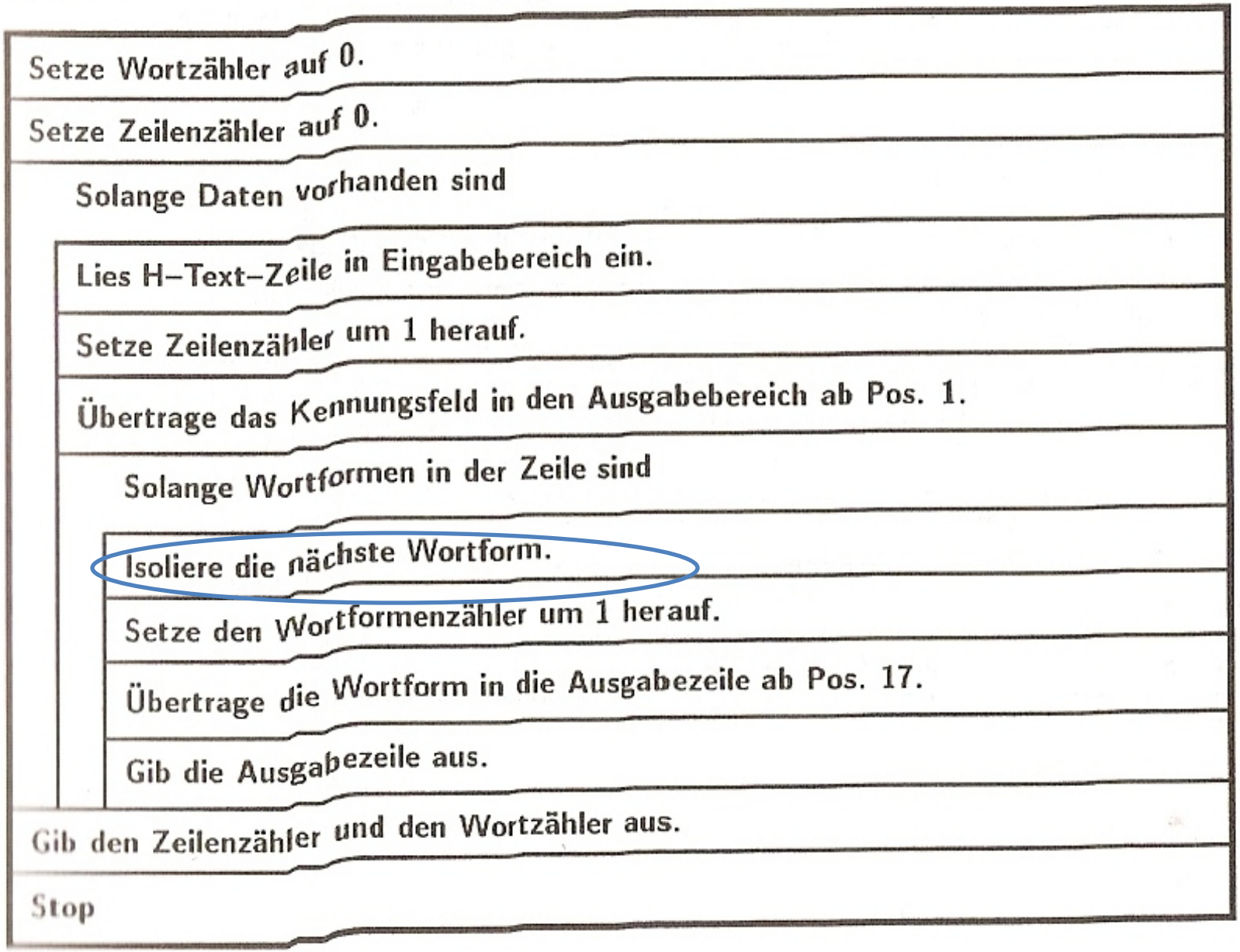
✓ H-Text → zeilenweise einlesen

✓ Abspeicherung der Zeilenkennung (Infos über Text, Seite und Zeile)

✓ Isolierung einzelner Segmente

✓ Eingabe der laufender Segmentnummer, Segment, mögl. Zusatzinfos → als V-Text-Zeile

Zergliedere



Unterblock Isoliere die nächste Wortform

Ist eine neue Zeile eingelesen worden?	
ja	nein
Setze Pointer1 und Pointer2 auf 10	
Solange Pointer2 nicht auf Blank zeigt oder größer als Zeilenlänge wird	
Erhöhe Pointer2 um 1.	
Übertrage Teilkette aus dem Eingabebereich ab der Pos. Pointer1 mit der Länge (Pointer2 − Pointer1) auf den Bereich Wortform des Ausgabebereichs.	
Setze Pointer1 und Pointer2 auf den Wert (Pointer2 + 1) .	
Falls Pointer2 größer Zeilenlänge, setze ein Signal, daß keine weiteren Wortformen vorhanden sind.	
Kehre in den aufrufenden Block zurück.	

```
PROCEDURE iso;        {Unterprogramm, das die eigentl. Segmentierung vornimmt}
BEGIN
        WHILE (s[pointer2] <> ´´) AND (pointer2 <= sl) DO
                pointer2 := pointer2 +1;
        u := Substr (s, pointer1, pointer2-pointer1);
        pointer1 .= pointer2 +1
        pointer2 := pointer2 +1
        IF pointer2 > sl THEN flag := FALSE
        ELSE flag := TRUE;
END;        {Ende von ISO}
```

Kodierung und Umkodierung

- Überführung von Texten von einem Kodierungssystem in ein anderes →

  oft anzutreffendes Problem in der CL:
- Graphievarianten
- Phonematische Transkription schriftsprachlicher Texte
- Umsetzung in Blindenschrift

Kodierung/Umkodierung (Beispiel)

- ‚sch‘ soll am Verbformenanfang durch ‚X‘ ersetzt werden

Beginnt die Verbform mit 'SCH'?	
ja	nein
Ersetze 'SCH' durch 'X'.	
Setze Verbform in Verbformenfeld linksbündig um.	
Vermindere die Längenangabe um 2.	
Setze 'SCH'–Zähler um 1 herauf.	

```
IF Substr (s, 1, 3) = ´sch‘ THEN
BEGIN
        u := ´X´+ Substr (S, 4, 18) +´  ´;
        Subs2 (s, 1, 20, u);
        t := Substr (s, 34, 2);
        VAL (t, fl, dummy);
        fl := fl – 2;
        STR (fl:2,u);
        Subs2(s, 34,2,u);
        schz := schz +1;
END;
```

Register

– Segmentierung des H-Textes in Wortformen

– Alphabetische Sortierung des V-Textes nach Wortformen

– Auszählen der verschiedenen Wortformen des V-Textes

– Herstellung der Rangliste

– Herstellung des alphabetischen Registers

– Herstellung des rückläufigen Registers

Lexikonvergleich

- Informationsentnahme:

✓ *sequentielle Durchsuchung*

✓ *Inhaltsverzeichnis → Direktzugriff*

- Algorithmus leistet Folgendes:

- Einlesen der Wortform
- Vergleich der Wortform mit eingelesenem Schlüsselwort aus Lexikon
- Übertragung des Info-Teils d. Lexikondatensatzes (im Fall einer Übereinstimmung)
- Anschließende Ausgabe des Wortformen-datensatzes
- Unveränderte Ausgabe der Wortformen ohne Entsprechung

Index und Konkordanz

- ***KWIC*** *(Key Word In Context)*

- *Algorithmus benötigt einen H-Text als Eingabe*

Unterblock Generiere KWIC–Zeilen

Setze den Bereich Kennung auf den Wert der Zeilenkennung im mittleren Arbeitsbereich.
Setze den Pointer1 auf den Beginn des Textteils im mittleren Arbeitsbereich.
Solange Pointer1 nicht größer als 160 ist
Setze Pointer2 auf den Wert des nächsten Blanks relativ zu Pointer1.
Übertrage Teilkette ab Pointer1 mit der Länge 50 Bytes in den rechten Teil des Ausgabebereichs.
Übertrage Teilkette mit der Länge 50 Bytes ab der Position (Pointer1 – 50) in den linken Teil des Ausgabebereichs.
Speichere den Ausgabebereich ab.
Erhöhe den KWIC–Zeilenzähler um 1.
Setze Pointer1 auf den Wert (Pointer1 + Pointer2).
Solange Pointer1 auf Blank zeigt
Erhöhe Pointer1 um 1.
Kehre in den aufrufenden Block zurück.

```
PROCEDURE Kwicy;{Proc. Zum Generieren von KWIC-Zeilen}
BEGIN
        kn := substr (u,81,10);
        pl := 91;
        While pl <= 160 DO
```

Textanalyse

- Algorithmen zur Analyse von Texten mithilfe von Regeln (sprachspezifische)
- 1:1 Umsetzungen: Ausnahme
- Operation auf Ebene der Morphologie, Syntax und Semantik
- Verwendung zur Bestimmung komplexer linguistischer Einheiten
- Algorithmen zu morphologischen und syntaktischen Analysen

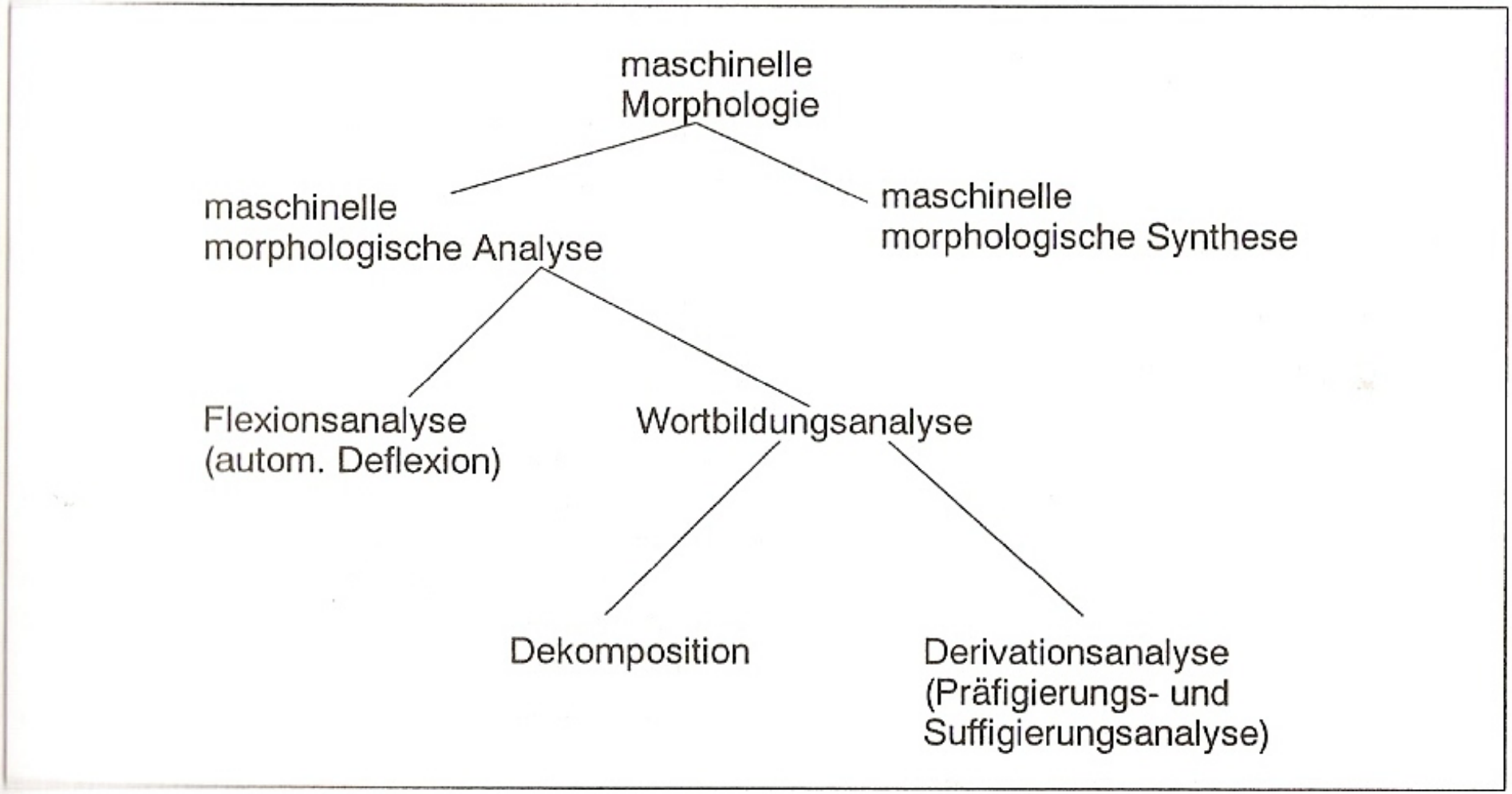
Morphologische Analyse

- Flexionsmorphologie: 2. Bereiche:

❖ **Wortbildungslehre:**
  - **Derivation**
  - **Komposition**

❖ **Flexionslehre**

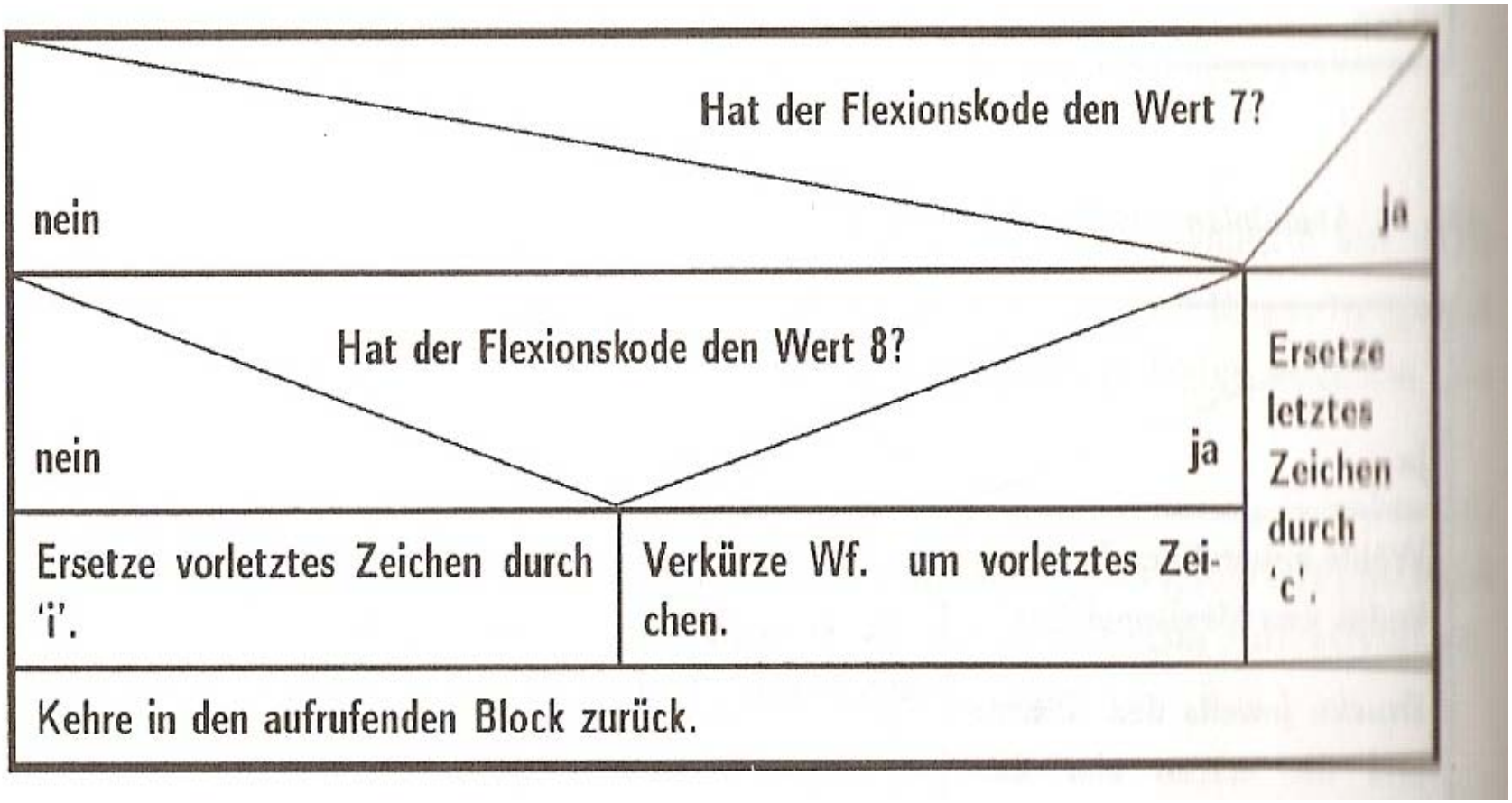
Teilbereiche der maschinellen Morphologie



Flexionsgenerierung

Klasse	Beispiel	Kode
A	SAGITTA	1
O–Mask.	LUPUS	2
O–Neutr.	OPPIDUM	3
A–O–Adj.	BONUS	4
Kons. –OR	FUROR	5
Kons. –EX	REX	6
Kons. –UX	CRUX	7
Kons. –ER	MATER	8
Kons. –EN	NOMEN	9

Ablaufplan


Hat der Flexionskode den Wert 7?
nein
ja
Hat der Flexionskode den Wert 8?
nein
ja
Ersetze letztes Zeichen durch 'c'.
Ersetze vorletztes Zeichen durch 'i'.
Verkürze Wf. um vorletztes Zeichen.
Kehre in den aufrufenden Block zurück.

sagitta	lupus	oppidum	furor	rex	crux
sagittae	lupi	oppidi	furoris	regis	crucis
sagittae	lupo	oppido	furori	regi	cruci
sagittam	lupum	oppidum	furorem	regem	crucem
sagitta	lupo	oppido	furore	rege	cruce
sagittae	lupi	oppida	furores	reges	cruces
sagittarum	luporum	oppidorum	furorum	regum	crucum
sagittis	lupis	oppidis	furoribus	regibus	crucibus
sagittas	lupos	oppida	furores	reges	cruces
sagittis	lupis	oppidis	furoribus	regibus	crucibus

mater	nomen
matris	nominis
matri	nomini
matrem	nominem
matre	nomine
matres	nomina
matrum	nominum
matribus	nominibus
matres	nomina
matribus	nominibus

(Auf den Vokativ wird verzichtet)

Inhaltsanalyse

- Daten werden inhaltlich analysiert
- Zuordnung von Kategorien an bestimmte lexikalische Einheiten im Text
- zunächst quantitative Auswertung der Daten (Wortlisten, Worthäufigkeiten, KWIC-Konkordanzen)

Automatische Textzusammenfassung

- Zwei Ansätze:
  - ✓ ***wissenschaftlicher Ansatz:***
  - Trennung von wichtigen und unwichtigen Teilen
  - Kernaussage wird ausfindig gemacht
  - muss alle Kernaussagen des ursprüngl. Textes enthalten

  - ✓ **statistischer Ansatz:**
  - o Sätze – hohe Anzahl an wichtigen Wörtern
  - o Korpora (Textsammlung)-Auswertung

Parser

- Zerlegen von Sätzen in ihre Konstituenten
- Entsprechende Algorithmen: PARSER
  - ***Bottom-up-parsing:***
  - setzt bei den Einzelelementen an
  - Ermittelt schrittweise die übergeordneten Konstruktionen → Endsymbol S

  - **Top-down-parsing:**
  - Ermittelt Satzstruktur
  - Schrittweise Zerlegung des Satzes in seine Konstituenten → terminalen Symbolen