KMP-Algorithmus

- Der KMP-Algorithmus wird genutzt, um ein Muster (Pattern) in einem Text effizient zu suchen.
- Ein naiver Suchalgorithmus vergleicht das Muster mit dem Text bis eine Ungleichheit auftritt und verschiebt das Muster nur um ein Zeichen nach rechts.
- Oft ist jedoch an der Abbruchstelle bereits bekannt, dass der Vergleich nach einem einmaligen Verschieben erneut fehlschlagen wird.
- Der KMP-Algorithmus nutzt diese Eigenschaft, indem vor Beginn der Suche eine Verschiebetabelle berechnet wird, welche ausschließlich vom Pattern abhängig ist. Diese Tabelle enthält an jeder Patternposition j eine Zahl $\text{Tab}[j]$. Bei Ungleichheit an Position j wird das Pattern soweit nach rechts verschoben, dass die Patternposition $\text{Tab}[j]$ an der momentanen Textposition steht.

KMP-Algorithmus

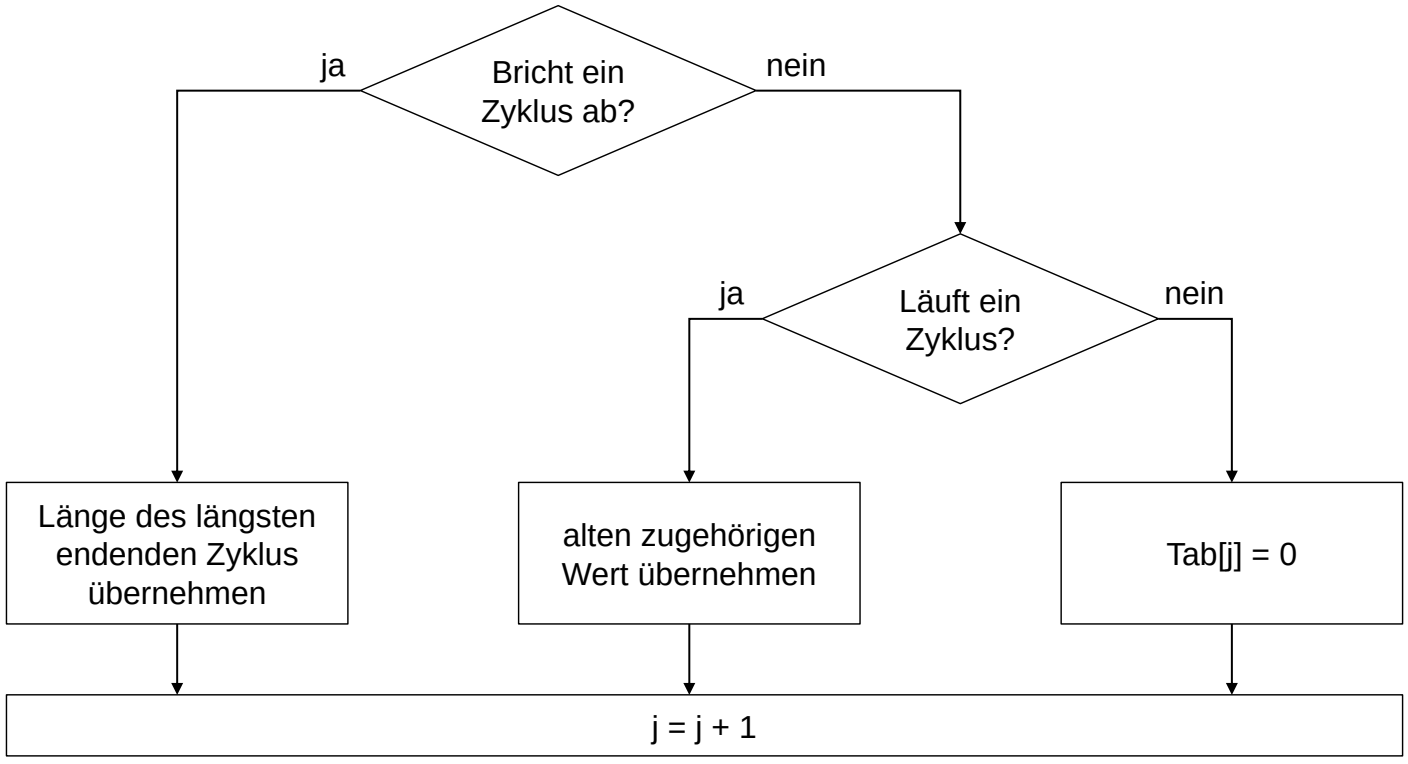
- Definition der Verschiebetabelle für ein Pattern b:

$$\begin{aligned}\text{Tab}[j] = \max(\{-1\} \cup \{m \,|\, & 0 \leq m \leq j-1 \\ & \wedge\ b_0 \ldots b_{m-1} = b_{j-m} \ldots b_{j-1} \\ & \wedge\ b_m \neq b_j\})\end{aligned}$$

- Zwei-Finger-Methode:
  - Da das Maximum gesucht ist, betrachten wir m in der Reihenfolge $j-1, j-2, ..., 0$.
  - Wir werten zuerst die Eigenschaft $b_m \neq b_j$ aus und dann erst $b_0 \ldots b_{m-1} = b_{j-m} \ldots b_{j-1}$.

KMP-Algorithmus

- Zyklenmethode:
  - Zyklus: Wiederholtes Auftauchen des Patternanfangs
  - Für $j = 0$: Tab$[j] = -1$
  - Für $j > 0$:



Übung 3

Position	0	1	2	3	4	5	6	7	8	9
Pattern	a	a	b	a	a	a	c	a	a	b
Tabelle	-1	-1	1	-1	-1	2	2	-1	-1	1

Position	0	1	2	3	4	5
Pattern	c	b	c	c	b	a
Tabelle	-1	0	-1	1	0	2

Levenshtein-Distanz

- Die Levenshtein-Distanz gibt die Kosten zur Überführung eines Wortes w in ein Wort v an.
- Die möglichen Operationen sind Insertion (Kosten: 1), Deletion (Kosten: 1) und Substitution (Kosten: 1).
- Zur Berechnung wird eine Matrix erstellt, welche jeweils die Levenshtein-Distanzen aller Kombinationen der Präfixe von w bzw. v enthält. In der Berechnungsmatrix soll das Quellwort w am linken Rand und das Zielwort v am oberen Rand stehen.
- Minimale Alignments können aus der Berechnungsmatrix abgelesen werden. In den Alignments soll das Quellwort w oben und das Zielwort v unten stehen.

Levenshtein-Distanz

- Bildungsvorschrift der Matrix:

Abkürzung: $d(w_1 \dots w_j, v_1 \dots v_i) \rightsquigarrow d(j, i)$

$$d(0, i) = i$$

$$d(j, 0) = j$$

$$d(j,i) = \min \begin{cases} d(j, i-1) + 1 & \text{Insertion } \rightarrow \\ d(j-1, i) + 1 & \text{Deletion } \downarrow \\ d(j-1, i-1) + \begin{cases} 1 & \text{wenn } w_j \neq v_i \\ 0 & \text{sonst} \end{cases} & \text{Substitution bzw. keine Operation } \searrow \end{cases}$$

Übung 4

$d(j,i)$		**D**		**i**		**s**		**t**		**a**		**n**		**z**
	0	→ 1	→	2	→	3	→	4	→	5	→	6	→	7
	↓	↘												
D	1	0	→	1	→	2	→	3	→	4	→	5	→	6
	↓	↓	↘											
i	2	1		0	→	1	→	2	→	3	→	4	→	5
	↓	↓		↓	↘		↘		↘		↘			
n	3	2		1		1	→	2	→	3		3	→	4
	↓	↓		↓	↘		↘		↘		↘	↓	↘	
s	4	3		2		1	→	2	→	3	→	4		4
	↓	↓		↓		↓	↘							
t	5	4		3		2		1	→	2	→	3	→	4
	↓	↓		↓		↓		↓	↘					
a	6	5		4		3		2		1	→	2	→	3
	↓	↓		↓	↘	↓		↓		↓	↘		↘	
s	7	6		5		4		3		2		2	→	3

Generierung der Matrix möglich unter:
`https://users.ifsr.de/~peine/levenshtein`

Übung 4

d(Dinstas, Distanz) $= 3$

Alignments:

D	i	n	s	t	a	*	s
|	|	|	|	|	|	|	|
D	i	*	s	t	a	n	z
		d				i	s

D	i	n	s	t	a	s	*
|	|	|	|	|	|	|	|
D	i	*	s	t	a	n	z
		d				s	i

Zusatzaufgabe 1

Position	0	1	2	3	4	5	6	7
Pattern	a	b	b	a	b	b	a	a
Tabelle	-1	0	0	-1	0	0	-1	4

Position	0	1	2	3	4	5
Pattern	b	a	b	a	b	c
Tabelle	-1	?	?	0	?	3