44. Bundeswettbewerb Informatik - 1. Runde / 2. Junioraufgabe

Luna Hagemann (Team-ID 00008 (Einzelteam), Teilnahme-ID 78245)

16. November 2025

Inhaltsverzeichnis

1 Lösungsidee

Wörter werden dass Zeichen für Zeichen durchgegangen, wobei für jede Regel eine oder mehrere if-Bedingungen angewendet werden, um herauszufinden, ob das gegebene Wort an der gegebenen Stelle getrennt werden muss. Ist dies der Fall, wird ein einfacher Bindestrich hinzugefügt, um die Trennung zu zeigen.

1.1 Interpretation der Regeln

Da die Regeln mit „kann“ etwas ungenau formuliert sind, habe ich die Annahme getroffen, dass „kann“ als „muss“ interpretiert wird, auch wenn eine andere Regel dies explizit ausschließt (z.B. „kann nicht“ oder „nur da“). Andere Interpretationen würden hier den expliziten Hinweis in der Aufgabenstellung die letzte Regel zuerst anzuwenden sinnlos machen.

2 Umsetzung

Nach der Input-Verarbeitung wird der Text in einzelne Wörter zerlegt, wobei Satzzeichen erst mal aus dem Text entfernt und separat gespeichert werden. Die Wörter werden dann Zeichen für Zeichen durchgegangen, wobei sich immer angeschaut wird, ob eine Trennung nach diesem Buchstaben stattfinden soll. Das Programm geht nun mit einer oder mehreren if-Bedingungen durch jede Regel von der 4. zur 1. (s. 1.1 warum dies der Fall ist) durch, wobei die Trennung oder nicht Trennung direkt angewendet wird. Um Ressourcen zu sparen und das Programm effizient zu gestalten, werden weitere Regeln in diesem Fall dann gar nicht geprüft. Trifft keine if-Bedingung zu, wird am Ende einfach der Buchstabe einzeln hinzugefügt, also ohne eine Silbentrennung.

3 Verbesserungsmöglichkeiten

Wie in den Beispielen (4) zu sehen, ist diese Silbentrennung bei weitem nicht ausreichend. Eine perfekte Silbentrennung ist wahrscheinlich nur zu erreichen, wenn es Integrationen mit Datenbanken gibt, welche für alle (oder die meisten) Wörter der deutschen Sprache die korrekte Silbentrennung gespeichert haben.

Als Verbesserung ohne Datenbank, hätte man eigene Regeln implementieren können, da Lars' Regeln bisher noch nicht ausreichend sind. Wie im Q&A gefordert, hätten diese Regeln zusätzlich zu Lars' Regeln angewendet werden müssen. Unter dieser Voraussetzung habe ich leider keine passenden Regeln gefunden, welche das Programm verbessert hätten und deswegen diesen Teil der Aufgaben nicht gelöst.

4 Beispiele

Beispiel	Text	Silbentrennung
silben01	Mein Name ist Lars und ich esse schrecklich gerne Sauerkraut.	Me-i-n Na-me ist Lars und ich es-se s-c-hrec-klich ger-ne Sa-u-er-kra-u-t.
silben02	Der Kapitän ist ebenfalls Präsident der örtlichen Dampfschifffahrtsgesellschaft.	De-r Ka-pi-tä-n ist e-ben-fal-ls Prä-si-dent de-r ör-tlic-he-n Dam-p-f-s-chif-ffah-r-t-sge-sel-l-s-chaft.
silben03	Es ist arschkalt!	Es ist ar-s-c-hkalt!
silben04	Ist das Audiosignal im Radio schlecht?	Ist da-s Au-di-o-sig-na-l i-m Ra-di-o s-c-hlec-ht?
silben05	Sein Vater ist Bauer und erntet Mais.	Se-i-n Va-te-r ist Ba-u-e-r und er-nte-t Ma-i-s.
silben06	Ich angle Karpfen.	Ich an-gle Kar-pfe-n.
silben07	Was sind acht Hobbits? Ein Hobbyte!	Wa-s sind ac-ht Hob-bits? Ei-n Hob-b-yte!
silben08	Freude, schöner Götterfunken, Tochter aus Elisium, Wir betreten feuertrunken, Himmlische, dein Heiligthum. Deine Zauber binden wieder, Was die Mode streng getheilt, Alle Menschen werden Brüder, Wo dein sanfter Flügel weilt.	Fre-u-de, s-chö-ne-r Göt-ter-fun-ke-n, Toc-hte-r a-u-s Eli-si-u-m, Wi-r bet-re-te-n fe-u-er-trun-ke-n, Him-mlis-che, de-i-n He-i-lig-thu-m. De-i-ne Za-u-be-r bin-de-n wi-e-de-r, Wa-s di-e Mo-de s-treng get-he-ilt, Al-le Men-s-che-n wer-de-n Brü-de-r, Wo de-i-n san-fte-r Flü-ge-l we-ilt.

Das 7. Beispiel ist extrem cool!

5 Quellcode

```
# INPUT HANDLING
# FUNCTION getitem(): safe get, returns empty string if out of Index
def getitem(list: list, i: int):
    try:
        return list[i]
    except IndexError:
        return ''

con = ['b', 'c', 'd', 'f', 'g', 'h', 'j', 'k', 'l', 'm', 'n', 'p', 'q', 'r', 's', 't', 'v', 'w', 'x', 'y', 'z', 'ß']
voc = ['a', 'e', 'i', 'o', 'u', 'ä', 'ö', 'ü']
zwi = ['ai', 'au', 'ei', 'eu', 'oi', 'ui', 'äu']

# Lars' rules
gen = ''
for w in txt.strip().split('-'):
    punctuation = ''.join([c for c in w if c in ['.', ',', '?', '!']])  # only punctuation
    w = w.strip('.,?!')
    wgen = ''
    w = [c for c in w]
    for i in range(len(w)):
        # quick access variables (i (item) + m (minus / plus if empty) + n (addition or su
        # imporves overall speed, as the safe get funtion will be called less.
        im2 = getitem(w, i - 2)
        im1 = getitem(w, i - 1)
        i0 = getitem(w, i)
        i1 = getitem(w, i + 1)
        i2 = getitem(w, i + 2)
        # no split after last character
        if i == len(w) - 1:
            wgen += i0
            continue
        # rule 4
        if i0 in voc and not (i1 in con and i2 in con):
            wgen += i0 + '–'
```

```
            continue
        elif i0 in voc: # "only"
            wgen += i0
            continue
        # rule 3
        if i0 in con and i1 in con and i2 in con: # split after first con
            wgen += i0 + '-'
            continue
        if im1 in con and i0 in con and i1 in con: # no split after second con
            wgen += i0
            continue
        # rule 2
        if i == 0 or i == len(w) - 2: # index 0 (split after first character) or second-hi
            wgen += i0
            continue
        # rule 1
        if i0 in con and i1 in con:
            wgen += i0 + '-'
            continue
        wgen += i0 # if no rule added a split
    gen += wgen + punctuation + ' '
```