Störung

❔ A1 👥 00122 👴 Leonhard Masche 🗓️ 3.09.2022

Lösungsidee

Die Idee zur Lösung ist, die Wörter im Buch (Alice im Wunderland) und ihre zugehörigen Zeilennummern in ein Array von Tuples zu laden. Nun kann man im Array mithilfe eines linken und rechten 'pointers' Treffer finden und diese zusammen mit ihren Zeilennummern ausgeben.

Umsetzung

Das Programm ist in Python umgesetzt und mit einer Umgebung ab der Version `3.6` ausführbar. Das Programm befindet sich in der Datei `program.py` in diesem Ordner.

Zuerst wird das Buch mithilfe einer RegEx (`(\\w+?)(?:\\W|$)`) in eine Liste geladen und mithilfe der `str.lower()` Methode normalisiert. In dieser Liste wird zusammen mit den Wörtern auch ihre Zeilennummer im Buch gespeichert.

Jetzt wird in einer verschachtelten `for`-Schleife über die Indizes im Buch (über die Wörter) iteriert. Dies stellt den linken Pointer dar. Der rechte Pointer wird nun auf den Wert des Linken gesetzt. In einer weiteren Schleife wird über die Wörter im zu findenden Satz iteriert. Wenn das nächste Wort mit dem nächsten im Buch übereinstimmt, das am rechten Pointer zu finden ist, wird dieser um eins inkrementiert. Stimmt das Wort nicht überein, wird durch `break` die innere Schleife gebrochen und der linke Pointer wird weiter vergrößert. Durch das `for ... else` Konstrukt kann geprüft werden, ob es eine volle Übereinstimmung der Textstelle und dem zu findenden Satz gibt. Ist das der Fall, wird sie der Ergebnisliste hinzugefügt.

Zum Schluss werden die Ergebnisse formatiert ausgegeben.

Big O

Für die beiden verschachtelten `for`-Schleifen ergibt sich schlimmstenfalls eine Komplexität von `O((n-k)*k)`, wobei `n` die Anzahl von Wörtern im Buch und `k` die Nummer von Wörtern im Zielsatz ist.

Praktisch ist dieser Maximalwert jedoch kaum zu erreichen, (vor allem nicht bei diesem Buch), da dazu das ganze Buch, sowohl als auch der Zielsatz aus demselben wiederholten Wort bestehen müssten. Es gäbe dann `n-k` Treffer.

Beispiele

Hier wird das Programm auf die sechs Beispiele von der Website angewendet:

`stoerung0.txt`

```
das _ mir _ _ _ vor

1 Ergebnis gefunden: (6.51ms)

> das kommt mir gar nicht richtig vor (l. 440)
```

---

stoerung1.txt

```
ich muß _ clara _

2 Ergebnisse gefunden: (6.08ms)

> ich muß in clara verwandelt (l. 425)
> ich muß doch clara sein (l. 441)
```

---

stoerung2.txt

```
fressen _ gern _

3 Ergebnisse gefunden: (6.06ms)

> fressen katzen gern spatzen (ll. 213-214)
> fressen katzen gern spatzen (l. 214)
> fressen spatzen gern katzen (l. 214)
```

---

stoerung3.txt

```
das _ fing _

2 Ergebnisse gefunden: (6.17ms)

> das spiel fing an (l. 2319)
> das publikum fing an (l. 3301)
```

---

stoerung4.txt

```
ein _ _ tag

1 Ergebnis gefunden: (6.05ms)

> ein sehr schöner tag (l. 2293)
```

`stoerung5.txt`

```
wollen _ so _ sein

1 Ergebnis gefunden: (5.86ms)

> wollen sie so gut sein (l. 2185)
```

Quellcode

`program.py`

```python
from itertools import chain
from os import path
from re import findall
from time import time
from typing import List, Tuple


def r_path(path_: str) -> str:
    return path.join(
        path.dirname(path.abspath(__file__)),
        path_
    )


# die wörter-liste
# ((lineno, word), ...)
book: Tuple[Tuple[int, str]]

with open(r_path('beispieldaten/Alice_im_Wunderland.txt'), 'r') as f:
    book = tuple(chain(*map(
        lambda x: ((x[0], word.lower()) for word in findall(r'(\w+?)(?:\W|$)', x[1])),
                ((i, line) for i, line in enumerate(f.read().split('\n'))))))


# hauptprogramm
def main():
    print()
    bsp_nr = int(input('Bitte die Nummer des Beispiels eingeben [0; 5]: '))
```

```
    print()

    start_time = time()  # variable für die zeitmessung

    # laden des beispielsatzes
    sentence: List[str]
    with open(r_path(f'beispieldaten/stoerung{bsp_nr}.txt'), 'r') as f:
        text = f.read()
        print(text)
        sentence = text.split()
    print()

    # iterieren über das wörterarray und vergleichen mit dem beispielsatz
    results: List[str] = []
    for left in range(len(book) - len(sentence)):
        right = left
        for next_word in sentence:
            if (book[right][1] == next_word) or next_word == '_':
                right += 1
            else:
                break
        else:
            line_l, line_r = book[left][0] + 1, book[right - 1][0] + 1
            results.append((
                ' '.join(map(lambda x: x[1], book[left:right])),
                f'l. {line_l}' if line_l == line_r else f'll. {line_l}-{line_r}'))

    # ergebnisse ausgeben
    if len(results) == 0:
        print(f"Kein Ergebnis gefunden! ({format((time() - start_time)*1000, '.2f')}ms)")
    else:
        print(f"{len(results)} Ergebnis{'se' if len(results) > 1 else ''} gefunden:"
              f" ({format((time() - start_time)*1000, '.2f')}ms)")
        print()
        for result, line_ref in results:
            print(f'> {result} ({line_ref})')


# programmschleife
print('Lieblingsbuchzitatfinder')
print('(Drücke ^C um das Programm zu beenden)')
print()
print('Buch: Alice im Wunderland')

while True:
    try:
        main()
    except Exception as e:
        print(e)
    except KeyboardInterrupt:
        print('\n')
        exit()
```