Aufgabe 1: Wörter aufräumen

Team-ID: ?????

Team-Name: ?????

Bearbeiter/-innen dieser Aufgabe:
Katharina Libner , Christopher Besch

27. Oktober 2020

Inhaltsverzeichnis

1 Lösungsidee

Der Eingangstext besteht aus unvollständigen Wörtern. Diese Wörter müssen mithilfe von Ersatzwörtern aus der Wörterbank ersetzt werden.

Ein Ersatzwort kann ein Wort ersetzten, wenn

1. die Länge beider Wörter gleich ist und
2. jeder Buchstabe aus dem Ersatzwort mit dem gegenüberliegenden Buchstaben, der Buchstabe aus dem Wort an gleicher Stelle, übereinstimmt, oder der gegenüberliegende Buchstabe eine freie Stelle („_“) ist. Eine freie Stelle stimmt quasi mit jedem Buchstaben überein (=Joker).

Wenn ein Ersatzwort verwendet wurde, kann es nicht erneut für ein folgendes Wort benutzt werden.

Die Aufgabe des Programmes ist es, zu erkennen, ob eine Lösung möglich ist und welches Wort mit welchem Ersatzwort zu ersetzten ist.

Dafür werden für jedes Wort alle passenden Ersatzwörter gesucht. Wenn zu manchen Wörtern mehrere Ersatzwörter passen, gibt es mehrere Möglichkeiten, diese passenden Ersatzwörter auszuwählen. Einige dieser Möglichkeiten sind unmöglich, da sie ein und dasselbe Ersatzwort mehrmals verwenden. Es muss also eine Möglichkeit gefunden werden, die diese Regel nicht verletzt.

2 Umsetzung

Die Lösungsidee wird in Python implementiert. Zuerst wird das Rätsel aus der Datei gelesen. Hierbei werden die einzelnen Wörter im Text, der ersten Zeile in der Datei, voneinander getrennt. Die Nicht-Wörter vor den Wörtern, nach den Wörtern und zwischen den Wörtern werden ebenfalls für die finale Ausgabe gespeichert. In einer while-Schleife wird in dem Text nach dem ersten Wort mithilfe einer Regular Expression gesucht. Nach jedem Treffer wird der Text gekürzt, sodass das nächste Wort gefunden werden kann.

Nun wird mit der rekursiven Funktion replace_incomplete_words eine Lösung gesucht, sie nimmt einen Text und eine Wörterbank entgegen.

Bei jeder Ausführung dieser Funktion werden alle möglichen Ersatzwörter für das erste Wort im Text gesucht. Für jedes mögliche Ersatzwort wird die Funktion erneut aufgerufen, hier wird die Rekursion deutlich. Dieser Ausführung wird allerdings nicht der ganze Text und die ganze Wörterbank gegeben, das erste Wort wird aus dem Text und das verwendete Ersatzwort aus der Wörterbank weggelassen.

Wenn der Funktion ein leerer Text, ein Text ohne Wörter, gegeben wird, wird der Rekursionsanker getroffen, eine Lösung wurde gefunden. Da aus der Aufgabenstellung hervorgeht, dass es immer nur eine Lösung geben kann, wird, wenn die erste funktionierende Möglichkeit gefunden wurde, die Suche abgebrochen. Die Funktion gibt den Text mit allen Ersatzwörtern in korrekter Reihenfolge zurück.

Zum Schluss wird der finale Text mit allen Nicht-Wörtern ausgegeben.

3 Beispiele

4 Quellcode

```
def load_file(filepath):
    with open(filepath, "r", encoding="utf-8") as file:
        # first line is text
        text = file.readline().strip()
        # second line is word bank
        word_bank = file.readline().strip().split(" ")
    return text, word_bank
```

```
def does_match(word, possible_replacement):
    """
    is possible_replacement able to replace incomplete_word?
    """
    # does the length match?
    if len(word) != len(possible_replacement):
        return False
    # check every character
    for char_word, char_replacement in zip(word, possible_replacement):
        # when the current char in the replacement word is not unknown (isn't "_"),
        # it has to match with the current char of the word otherwise there is no chance of rep
        if char_word != "_" and char_word.upper() != char_replacement.upper():
            return False
    return True
```