Anagramme und Komposita

In dieser Übung geht es um die Extraktion von Anagrammen und Komposita aus einem Textkorpus.

Laden Sie zunächst das Textkorpus mit der URL `http://www.cis.uni-muenchen.de/~schmid/lehre/data/zeit-10M-tagged.gz` und dekomprimieren Sie die Datei.

Anagramme extrahieren

Schreiben Sie ein Programm, welches alle Wörter in der ersten Spalte der Datei einliest, die nur aus Buchstaben bestehen, und dann alle Anagramme (z.B. Lampe, Palme, Ampel) ausgibt.

Aufruf:

```
python anagramme.py zeit.tagged
```

Die Ausgabe erfolgt auf den Bildschirm und sollte so aussehen:

```
Ampel Lampe Palme
Ernst Stern
Stuben buntes
...
```

Die Liste der Wörter in jeder Zeile soll absteigend nach Korpushäufigkeit sortiert sein. Wörter mit einer Häufigkeit von unter 10 werden ignoriert. Listen mit nur einem Element sollen nicht ausgegeben werden. Wenn ein Wort in zwei Schreibungen auftritt (bspw. “Klein” und “klein”), geben Sie nur die häufigere aus.

Vorüberlegungen

- Welche Eigenschaft ist allen Wörtern eines Anagrammes gemeinsam?
- Wie kann man die Menge aller Anagramme schnell (in linearer Zeit) finden?
- Welche Datenstrukturen sind sinnvoll?

Implementieren Sie das Programm.

Komposita aufspalten

Hier sollen Sie die Komposita-Zerlegungsstrategie von Philipp Koehn implementieren (`https://aclanthology.org/E03-1076.pdf`). Die Grundidee von Koehns Methode besteht darin, ein gegebenes Kompositum auf alle möglichen Arten so zu zerlegen, dass jedes

Element der Zerlegung mindestens drei Buchstaben umfasst und in einer Wortliste (unter Ignorierung der Groß-/Kleinschreibung) enthalten ist. Jede mögliche Zerlegung eines Kompositums (inkl. des nicht zerlegten Wortes) wird dann mit dem geometrischen Mittel der Worthäufigkeiten seiner Elemente bewertet.

Ausnahme: Das Fugenelement 's' darf zwischen zwei Elementen auftauchen und wird bei der Scoreberechnung nicht berücksichtigt und auch nicht als Teil der Zerlegung ausgegeben.

Sie sollen ein Programm schreiben, welches aus dem Zeit-Korpus alle Nomen-Types (Wortart "NN"), die nur aus einem Großbuchstaben und einer Folge von Kleinbuchstaben bestehen, mit ihren Häufigkeiten extrahiert und diese dann mit der beschriebenen Methode in Teilwörter zerlegt. Auch die Teilwörter sollen Nomen aus dem Zeitkorpus sein.

Verwenden Sie für die Zerlegung eine rekursive Funktion, welche die Ergebnisse mit dem Python-Befehl *yield* zurückgibt.

Aufruf:

```
python split-compounds.py zeit.tagged
```

Die möglichen Zerlegungen jedes Wortes werden nach abnehmender Bewertung sortiert und im folgenden Format auf den Bildschirm ausgegeben:

```
Arbeitsamt 33.9 Arbeit Amt
Arbeitsamt 19 Arbeitsamt
Arbeitsamt 1.7 Arbeit Samt
Abteilungen 112 Abteilungen
Abteilungen 20.1 Abtei Lungen
Abteilungen 15.3 Abt Ei Lungen
...
```

Achtung: Dies sind nicht die echten Zerlegungen und Bewertungen, die sie generieren sollen! Sie sollen bspw. auch keine Teilwörter mit zwei Buchstaben ausgeben.

Vorüberlegungen

- Welche Schritte muss Ihr Programm ausführen?
- Welche Datenstrukturen verwenden Sie am besten?
- Wie werden die Zerlegungen berechnet und bewertet?
- Wie behandeln Sie am besten das Fugen-s (Beispiel: *Arbeit-s-amt)*?

Achtung: Bei der Programmierung dürfen nur die Standard-Bibliotheken von Python verwendet werden.

Bitte schicken Sie die beiden Programme bis zum angegebenen Abgabetermin an `schmid@cis.lmu.de`.