Autor: Piotr Puszkarski

Laboratoria: 3

1. Podziel tekst wszystkich orzeczeń z określonego roku na słowa.

```
2. words = []


   def tokenize(data):
       for text in data['items']:
           if not reader.in_2008(text):
               continue
           txt = re.sub(r'<[^>]*>', '', text['textContent'])
           txt = re.sub(r'-\n', '', txt)
           found = re.findall(r'\b[a-zA-ZąęłćżźóńśĄĘŁĆŻŹÓŃ]+\b', txt)
           found = [x.lower() for x in found]
           words.extend(found)
```

3. Oblicz i utrwal listę frekwencyjną słów (liczbę wystąpień posortowaną względem malejącej liczby wystąpień słów). Przykładowo, jeśli korpus zawiera zdanie "Ala Ala, kot", to lista frekwencyjna dla tego korupusu jest następująca:
   i. Ala: 2
   ii. kot: 1

```
frequency_list = Counter(words)
sorted_list = sorted(frequency_list.items(),
key=operator.itemgetter(1), reverse=True)
```

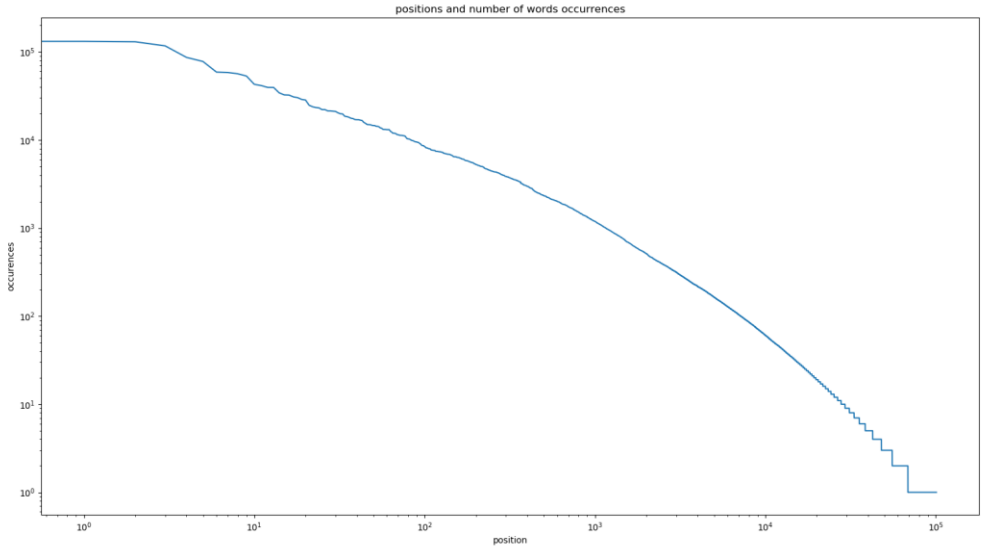
4. Z listy frekwencyjnej usuń pozycje, które nie stanowią słów (np. są to pojedyncze litery występujące w zanonimizowanej wersji nazw własnych). Usuń również wszystkie wartości, które nie są zapisane literami (w szczególności liczby).

```
ready_list = remove(sorted_list)
```

iteruję i wyrzucam pojedyncze słowa, a następnie zwracam listę bez pojedynczych słów.

```
def is_word(word):
    if len(word) == 1:
        return False
    return True
```

5. W skali logarytmicznej przedstaw wykres, w którym na osi X są pozycje na liście frekwencyjnej, a na osi Y liczba wystąpień słowa zajmującego określoną pozycję.



6. Korzystając z pliku polimorfologik.zip znajdź wszystkie słowa, które nie znajdują się w tym słowniku.

   Nie znaleziono 10863 słów

7. Przedstaw 30 przykładowych słów, które nie należą do słownika. (na końcu)
8. Korzystając z algorytmu korekty słów oraz listy frekwencyjnej, określ najbardziej prawdopodobną korektę słów, które nie występują w słowniku.

```
9. def levenshtein(ready_list, not_in_dict):
       candidates = []
       for new_word in not_in_dict:
           best_match = None
           dist = 99999
           for candidate in ready_list:
               if candidate[0] == new_word: #the same word
                   continue
               new_dist = Levenshtein.distance(candidate[0], new_word)
               if new_dist < dist: #distance
                   best_match = candidate
                   dist = new_dist
           candidates.append(best_match)
       return candidates
```

rrrr rur

peep peek

xxxxxxxxxxxxxxxxxxxxx xxxxxxxxxxxxxxxxxxxx

foxconn kocon

kjas kas

promulgatus promulgacji

luenen nuenen

tradax tradex

manufacturing factoring

tue te

protection production

nastajki nastawki

rogalę rolę

ustla ustala

darmota darmowa

jednol jedno

tpp typ

samogaśnięcie samogaśniecie

opo po

hiz hiv

risk ris

średniotygodniowej średniotygodniowy

geomembrany geomembrana

energa energia

prejurysdykcyjnych prejurysdykcyjnym

wewnątrzspółdzielczą wewnątrzspółdzielcze

welding holding

eurpfolms eurofilms

spochacza spochacz

retail metal