INF1820 V2017 – Oblig 2a
Korpora og ordklassetagger

Innleveringsfrist, fredag 3 mars

Lever inn svarene dine i Devilry i filer som angir brukernavnet ditt, slik: `oblig2a_brukernavn.py`. Pass på at fila di kan kjøres som et program; det skal ikke være en REPL-sesjon limt inn i en fil.
En perfekt besvarelse på denne oppgaven er verdt 100 poeng.

Merk at oppgavene her skal løses *uten* bruk av de NLTKs funksjonalitet for frekvens `nltk.FreqDist` og `nltk.ConditionalFreqDist`. I alle oppgavene skal vi jobbe med ordklassetaggede data fra nyhetsdelen i Brown-korpuset. Du får tilgang til korpuset i programet ditt slik:

```
import nltk
brown = nltk.corpus.brown.tagged_words(categories="news")
```

Variabelen `brown` er nå en liste med tupler, der det første elementet er ordformen og det andre elementet er taggen som er blitt tilordnet ordet. En liste av taggene i Brown og hva de betyr finner du på `http://www.comp.leeds.ac.uk/ccalas/tagsets/brown.html`.

Merk til slutt at det her er viktig å passe på at du ikke skiller mellom store og små bokstaver, slik at for eksempel *The* og *the* telles sammen, ikke hver for seg. Det kan du gjøre med methoden `lower()` slik:

```
>>> "The".lower()
'the'
```

1 Tagg- og ordfrekvens (40 poeng)

Ved hjelp av Python dictionaries og sortering, finn ut følgende:

1. Hva er det mest frekvente ordet i Brown?
2. Hva er den mest frekvente ordklassen i Brown?
3. Hva er den *minst* frekvente ordklassen?

2 Flertydighet (60 poeng)

1. Hvor mange ord er flertydige? Det vil si, hvor mange ord forekommer med mer enn én ordklassetagg?
2. Hvilket ord har flest tagger, og hvor mange distinkte tagger finner du?

3. Skriv en funksjon `freqs(w)` som tar et ord som argument og skriver ut hvor ofte ordet forekommer med hver av taggene. For eksempel forekommer *run* 20 ganger med NN, 11 ganger med NN, og 4 ganger med VBN.

4. Ved hjelp av funksjonen fra punkt 3, finn frekvenslisten for det mest flertydige ordet i Brown. Hva observerer du?

I de to første deloppgavene er et viktig poeng at du må passe på at for hvert ord telles hver distinkte tagg nøyaktig en gang; det vil si at for et ord med to forekomster med NP og tre med NN har vi listen `["NP", "NN"]` og ikke `["NP", "NN", "NN", "NN", "NP"]`. Dette kan løses på flere forskjellige måter, men en nyttig variant er å bruke Pythons innebygde støtte for mengder. En kort beskrivelse av hvordan du bruker mengder finner du i avsnitt A til slutt i oppgavesettet.

A Mengder

Mengder i Python konstrueres ved hjelp av funksjonen `set()`, og elementer legges til med metoden `add()`. Vi kan da gjøre:

```
>>> mengde = set()
>>> mengde.add("NP")
>>> mengde.add("NN")
>>> mengde.add("NN")
>>> mengde.add("NN")
>>> mengde
set(["NP", "NN"])
```

Som du ser kan vi legge til det samme elementet så mange ganger vi vil, men det vil bare forekomme én gang i mengden.