Wordnet, synsets

1. Vérifier en exécutant les premières manipulations du *howto* nltk que l'environnement est correctement installé.
2. Mesurer, pour chaque texte fourni, le nombre de synsets moyen par *type* (on suppose qu'on manipule un *bag-of-words*).
3. En prenant en compte cette fois le nombre d'occurrences de chaque type, et si possible aussi la morphologie, déterminer si les mots (pleins) les plus fréquents sont les plus "ambigus". [La structure de données la plus appropriée est sans doute un dictionnaire python.]
4. Faire les mêmes deux manipulations précédentes avec les 10 000 premiers tokens du *Brown corpus*, sans tenir compte des étiquettes, puis en en tenant compte.

```
from nltk.corpus import brown
for w in brown.tagged_words()[:10000]:
   ...

nltk.help.brown_tagset()
```

Mesures de similarité

1. Définir une fonction qui calcule pour un mot donné son "vecteur contexte" : un dictionnaire dont les clés sont tous les mots du corpus et dont les valeurs sont 1 si les deux mots co-occurrent dans une fenêtre de 10 mots, et 0 sinon.
2. À partir de la fonction précédente, on peut définir la *distance* entre deux vecteurs-contextes : on choisira la distance euclidienne entre vecteurs.
3. La mesure précédente permet de définir une distance entre *textes* (vus comme des sacs de mots) : on choisira la moyenne des distances (ou des similarités) pour tous les couples (x_i, y_j) où x_i est dans le premier texte et y_j dans le second.

Application : appariement de textes

En prenant au hasard quelques extraits distincts des trois textes fournis, vérifier que la distance est la plus courte quand les extraits viennent du même fichier initial.
On pourra aussi utiliser les mesures de distances déjà définies par nltk pour répliquer cette expérience.

On fournira un script auto-suffisant qui réalisera dans l'ordre les manipulations demandées.