#!/usr/bin/python

# -\*- encoding: utf-8 -\*-

#Zack Larsen, CSC 594 (NLP) HW #2

from \_\_future\_\_ import division

import re, pprint, nltk, sys, datetime, os, collections, shelve

from nltk import bigrams, trigrams, word\_tokenize, sent\_tokenize

import math

text = sys.stdin.read()

#print text     #or, print text[:100]

d = shelve.open('bigrams.db')

bigrams = d['bigrams']

d.close()

#print bigrams

#d = shelve.open('unigrams.db')

#print d.keys()

#unigrams = d['unigrams']

#d.close()

#print unigrams

######### Number of sentences ##########

sent\_tokenize\_list = sent\_tokenize(text)

#print sent\_tokenize\_list[0]

#############Generate N-grams on the test set sentences##############

def find\_ngrams(input\_list, n):

    return zip(\*[input\_list[i:] for i in range(n)])

############## For each sentence in test set, compute probability using chain rule ############

N\_bigrams = len(bigrams)

#N\_unigrams = len(unigrams)

for i in range(0,len(sent\_tokenize\_list)):

    #tokenize the sentence

    token\_list=word\_tokenize(sent\_tokenize\_list[i])

    token\_list.insert(0,'<s>')

    token\_list.append('</s>')

    print token\_list

    #create the unigrams and bigrams from the sentence

    bigram\_list = find\_ngrams(token\_list,2)

    print bigram\_list

    unigram\_list = token\_list

    #compute probability by taking ln(sumPR), where sumPR is the sum of the probabilities of the tokens in the sentence.

    #convert this later using exp(ln(sumPR))

    bigramsum = 0

    for bigram in bigram\_list:

        prob = bigrams[bigram]

        #print prob

        bigramsum += math.log(prob)

    bigramprob = math.exp(bigramsum)

    #print bigramprob

    # To compute perplexity, we just take the probability raised to the pwer of (-1/N\_bigrams)

    bigram\_perplexity = bigramprob\*\*(-1/N\_bigrams)

    unigramsum = 0

#    for unigram in unigram\_list:

#        prob = unigrams[unigram]

#        unigramsum += math.log(prob)

#    unigramprob = math.exp(unigramsum)

    #unigram\_perplexity = unigramprob\*\*(-1/N\_unigrams)

#    print 'Sentence',i,':', sent\_tokenize\_list[i]

#    print '-unigram [Prob]'#,unigramprob, '[Perp]',unigram\_perplexity

#    print '-bigram  [Prob]',bigramprob, '[Perp]',bigram\_perplexity

#    print '\n'