<u>***Machine Translation Assignment 1***</u> *Spring 2013*
<u>***IBM Model 1***</u> *s0946868*

The implementation of the EM algorithm here is done in Python 2.7.1.
The input on the command line is two text files, the first one should be the English file, followed by one in a foreign language. (eg. $ python em.py toy.en toy.de)

<u>***Data Structures***</u>

For the easier manipulation of the data, I decided to use python lists to store the unique words in both languages, as well as the pairs of sentences through which the EM algorithm will loop through later. For the possible alignments and probabilities I used several python dictionaries, as follows:
An alignments dictionary -> Key: foreign word, Value: list of all possible translations of the foreign word in English
A probabilities dictionary -> Key: foreign word, Value: dictionary with possible translations in English as keys and probabilities as values (initialized at 0)
Count (e|f) dictionary -> Key: foreign word, Value: dictionary
Total_f/total_s dictionaries > Key: foreign/English word, Value: sum of probabilities (initialized at 0). These are used as helpers during update.

<u>***Processing***</u>

Before loading the text file information into the data structures, some processing occurs. The input text is turned to all lowercase to avoid confusion; all punctuation is stripped and replaced with whitespace (this is to prevent the possessive form in English, for example, to stick to the previous word, or in the case of dashes, the words that contain a dash in between should be split as well, and not put together as one word).

<u>***The EM Algorithm***</u>

Before running the algorithm, as a starting point, an uniform probability is distributed to all English words for each possible alignment to a foreign word in the probabilities dictionary, while all values in the rest of the dictionaries, are assigned to zero at the beginning of each iteration of the EM algorithm.
For each pair of foreign-English sentences, the probability of each English word translation is stored as 0, and then for each foreign word in the sentence, the probability of the English translation is updated (expectation step of the EM algorithm). Then a second step loop runs through all English words in the sentence pairs and then through all foreign ones, in order to collect the counts as evidence that *e* is a translation of *f* (maximisation step) $\text{count}(e|f) \mathrel{+}= \frac{t(e|f)}{\text{s-total}(e)}$

In the end the algorithm goes through the dictionary of all unique foreign words in order to compute the conditional translation probability of the English word given the foreign one, according to the formula: $t(e|f) = \frac{\text{count}(e|f)}{\text{total}(f)}$
The last step effectively gives us the model estimation. This implementation of the algorithm could iterate over the probabilities a 1000 times or until convergence.

<u>***Output***</u>

The output of the algorithm is structured in 2 different files - one is the Viterbi alignments file, where each foreign word is associated with only one English word, considered the translation in this case. The second file is the word translation table for each foreign word along with the considered translations of the word in English and their probabilities. In the name of readability only the 3 translations with highest probability are shown in the latter table.