# Statistical Machine Translation
# Lab Exercise
# 4: Language Modelling

Please use Java as your programming language for this lab
Refer to the lecture slides for extra information

**1**- Given a file containing a number of sentences, please calculate the **frequency** (*p(w)*) of each word (*w*) in these sentences according to the formula:

$$p(w) = \frac{occurrences\ of\ word}{number\ of\ tokens} \quad (1)$$

**Hint:** The input file is usually called "corpus", which is used to calculate the word frequency for the following language modelling.

**Input:** a file (suppose that the input file only have one sentence "the cat sat on the mat with a cat")
**Output:**

The word "a" frequency is: **0.111111111111**
The word "on" frequency is: **0.111111111111**
The word "mat" frequency is: **0.111111111111**
The word "cat" frequency is: **0.222222222222**
The word "the" frequency is: **0.222222222222**
The word "with" frequency is: **0.111111111111**
The word "sat" frequency is: **0.111111111111**

**2**- Given a sentence (*s*) and a corpus, please calculate the **unigram probability** (language model) of the sentence according to the formula:

$$p(s = w_1, \ldots, w_n) = p(w_1) \times \ldots \times p(w_n) \quad (2)$$

**Hint:** The *P(w)* is calculated based the the corpus (i.e. Question 1), and then apply Equation 2 to calculate the unigram probability of the input sentence.

**Input:**
a corpus file (suppose that the input file only have one sentence "the cat sat on the mat with a cat")
an input sentence "a cat sat on the mat"
**Output:** **8.36300632515e-07**

**3**- Please write a program to compute **bigram probability of an input sentence**. The input to your program is a corpus file containing a number of sentences and an input sentence. The output is the probability of the input sentence. To compute **bigram relative frequency**, please use this formula:

$$p(w_2|w_1) = \frac{count\ (w_1,w_2)}{\sum_w count(w_1,w)} \quad (3)$$

To compute the **bigram probability of a sentence** use this formula:

$$p(s) = p(w_2|w_1) \times p(w_3|w_2) \ldots \times p(w_n|w_{n-1}) \quad (4)$$

**Hint**:
1, Interpolation of the n-gram function in Question 1 of Lab-3 could be a good idea.
2, Creating functions based on Question 1 and 2 could be a good idea.

**Input:**
a corpus file (containing a number of sentences)
an input sentence "**<s> a cat sat on the mat </s>**"
**Output:** The probability of the sentence "**<s> a cat sat on the mat </s>**" is **0.00097615576843**

**4**- First, try another sentence using your program of Question 3:
Please calculate the probability of the sentence "**<s> a cat sat on the car </s>**". What result/error do you get? Please think about what the reason is and why we need smoothing technique in language modeling.

Second, modify your function of **bigram relative frequency** according to add-one smoothing formula:

$$p(w_2|w_1) = \frac{count\ (w_1,w_2)+1}{\sum_w count(w_1,w)+v} \quad (5)$$

where *v* is vocabulary size (how many unique words in your file). Please use your smoothed function to calculate **bigram probability of a sentence** of the two sentences.

**Input:**
a corpus file (containing a number of sentences)
an input sentence "**<s> a cat sat on the mat </s>**"
**Output: 0.000140949604457**

**Input:**
a corpus file (containing a number of sentences)
an input sentence "**<s> a cat sat on the car </s>**"
**Output: 3.00170453936e-05**

**Optional-** In order to adapt your **bigram probability** program to **n-gram probability** program. Please add one more input to your program of Question 4**.**

**Input:**
a corpus file (containing a number of sentences)
an input sentence "**<s> a cat sat on the mat </s>**".
gram_number (1, 2, 3 and 4)

**Output:**
1-gram: **2.28175851587e-08**
2-gram: **0.000140949604457**
3-gram: **0.000263061746438**
4-gram: **0.000423106305459**