Challenge

Q1

In [4]:
```
def predict(word, topk, count=count, next_word_count = next_word_count):

    possibleNextWords = next_word_count[word]
    ans = []
    pWord = getWordProbability(word)
    for nextWord in possibleNextWords.keys():
        pNextWord = getWordProbability(nextWord)
        bayesEstimate =  getConditionalProbability(word, nextWord) * getWordProbability(nextWord)/ pWor
d
        ans.append((nextWord, bayesEstimate))
    topk = min(len(possibleNextWords), topk)
    return [(k,v) for k, v in sorted(ans, key=lambda item: item[1], reverse = True)][:topk]
```

In [99]:
```
seedWord = 'alice'
prev = seedWord
paragraph = []
for i in range(100):
    k = 3
    nextWordsPossible = predict(prev,k)
    if len(nextWordsPossible) < k:
        k = len(nextWordsPossible)

    nextWord = nextWordsPossible[random.randint(0, k-1)][0]
    paragraph.append(prev)
    prev = nextWord
print(" ".join(paragraph))
```

```
alice the queen the mock turtles all said the queen and a little the king said to alice to alice and
she had the king said to be the queen said to the mock turtle and the queen and she said to be the ki
ng the queen said alice to be said to the queen said to the mock turtle to alice and the queen said t
he queen the queen the queen the king said the queen and she said the king the queen and a great or t
he king said to the mock turtles heavy sobbing of the mock
```