Assignment 10.1.a

In [1]:
```
import re

def tokenize(sentence):
    # Remove basic punctuation using regular expression
    sentence = re.sub(r'[^\w\s]', '', sentence)

    # Split the sentence into words
    tokens = sentence.split()

    return tokens
```

In [2]:
```
sentence = "Hello, world! This is a sample sentence."
tokens = tokenize(sentence)
print(tokens)
```

```
['Hello', 'world', 'This', 'is', 'a', 'sample', 'sentence']
```

Assignment 10.1.b

In [3]:
```
def ngram(tokens, n):
    ngrams = []
    for i in range(len(tokens) - n + 1):
        ngram = tokens[i:i+n]
        ngrams.append(ngram)
    return ngrams
```

In [4]:
```
tokens = ['Hello', 'world', 'This', 'is', 'a', 'sample', 'sentence']
ngrams = ngram(tokens, 3)
print(ngrams)
```

```
[['Hello', 'world', 'This'], ['world', 'This', 'is'], ['This', 'is', 'a'], ['is',
'a', 'sample'], ['a', 'sample', 'sentence']]
```

Assignment 10.1.c

In [6]:
```
def one_hot_encode(tokens, num_words):
    token_index = {}
    results = []

    for token in tokens:
        if token not in token_index:
            token_index[token] = len(token_index) + 1

    for token in tokens:
        token_encoding = [0] * num_words
        token_encoding[token_index[token] - 1] = 1
        results.append(token_encoding)

    return results
```

In [7]:
```
tokens = ['Hello', 'world', 'This', 'is', 'a', 'sample', 'sentence']
num_words = len(set(tokens))
```

```
encoded_vectors = one_hot_encode(tokens, num_words)
for vector in encoded_vectors:
    print(vector)
```

```
[1, 0, 0, 0, 0, 0, 0]
[0, 1, 0, 0, 0, 0, 0]
[0, 0, 1, 0, 0, 0, 0]
[0, 0, 0, 1, 0, 0, 0]
[0, 0, 0, 0, 1, 0, 0]
[0, 0, 0, 0, 0, 1, 0]
[0, 0, 0, 0, 0, 0, 1]
```

In [ ]: