**EXPERIMENT NO 7**

**AIM:** Chunking.

**RESOURCES REQUIRED:**

Python 3, NLTK toolkit, Text editor, 4 GB RAM and above, i5 processor and above

**THEORY:**

**Chunking:**

Chunking is used to add more structure to the sentence by following parts of speech (POS) tagging. It is also known as shallow parsing. The resulting group of words is called "chunks." In shallow parsing, there is maximum one level between roots and leaves while deep parsing comprises more than one level. Shallow Parsing is also called light parsing or chunking.

The primary usage of chunking is to make a group of "noun phrases." The parts of speech are combined with regular expressions.

**Rules for Chunking:**

There are no predefined rules, but you can combine them according to need and requirement.

For example, you need to tag Noun, verb (past tense), adjective, and coordinating junction from the sentence. You can use the rule as below

chunk:{<NN.?>\*<VBD.?>\*<JJ.?>\*<CC>?}

Following table shows what the various symbol means:

| **Name of symbol** | **Description** |
| --- | --- |
| . | Any character except new line |
| \* | Match 0 or more repetitions |
| ? | Match 0 or 1 repetitions |

**Use Case of Chunking**

Chunking is used for entity detection. An entity is that part of the sentence by which machine get the value for any intention

Example: Temperature of New York. Here Temperature is the intention and New York is an entity.

In other words, chunking is used as selecting the subsets of tokens. Please follow the below code to understand how chunking is used to select the tokens. In this example, you will see the graph which will correspond to a chunk of a noun phrase. We will write the code and draw the graph for better understanding.

**CONCLUSION:**

Chunking is the process of making noun phrases. It is applied after parts of speech tagging. Chunking has been studied and implemented on a text corpus.

**CODE:**

from random import choice

from nltk import pos\_tag

from nltk import word\_tokenize

from nltk.corpus import brown

from nltk import RegexpParser

samples = choice(brown.paras(categories="adventure"))

corpus = " ".join([" ".join(sample) for sample in samples])

print(f"Original corpus :\n{corpus}\n")

tokens = word\_tokenize(corpus)

print(f"Tokenized words : \n{tokens}\n")

tagged\_words = pos\_tag(tokens)

print(f"POS tagged words : \n{tagged\_words}\n")

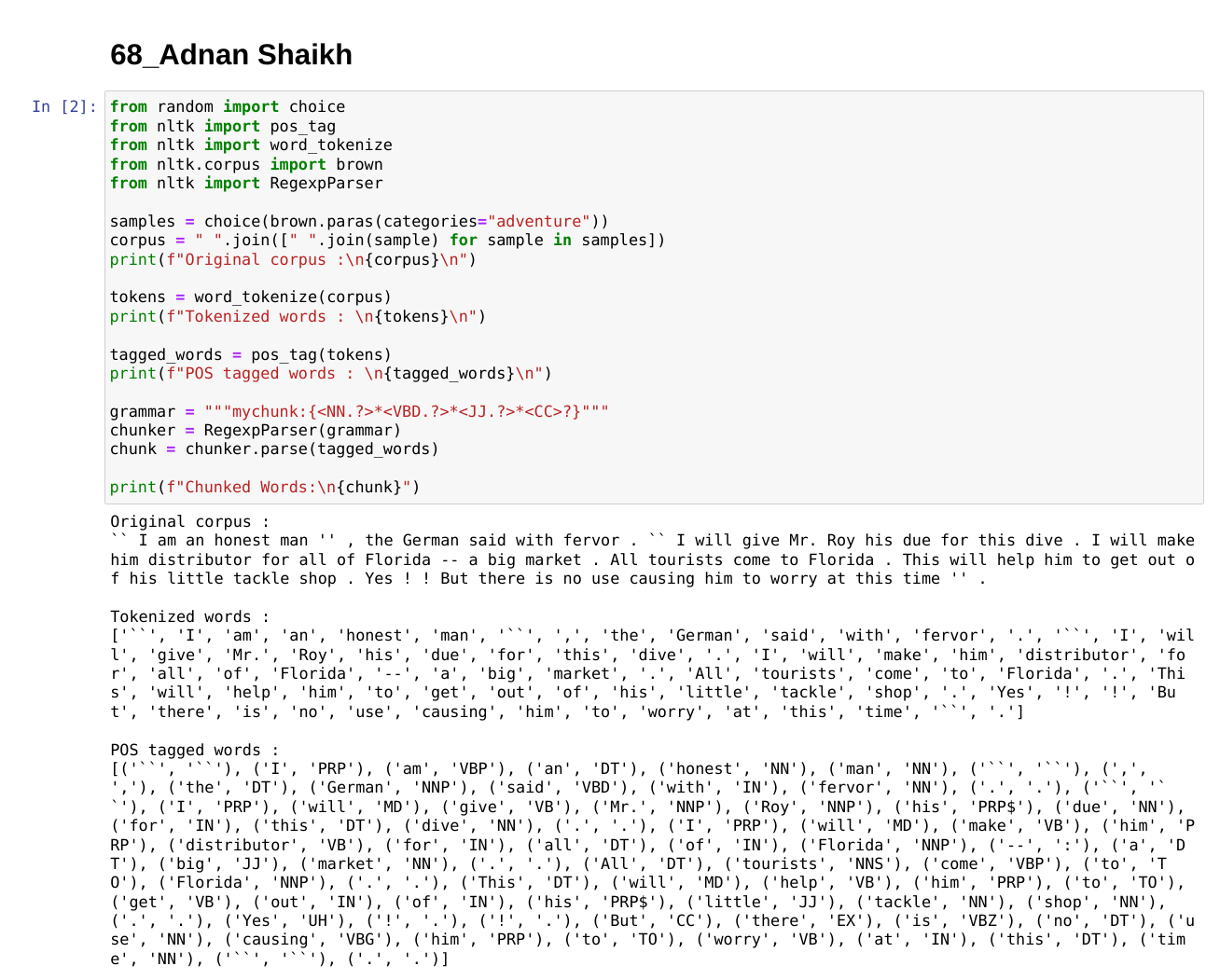
grammar = """mychunk:{<NN.?>\*<VBD.?>\*<JJ.?>\*<CC>?}"""

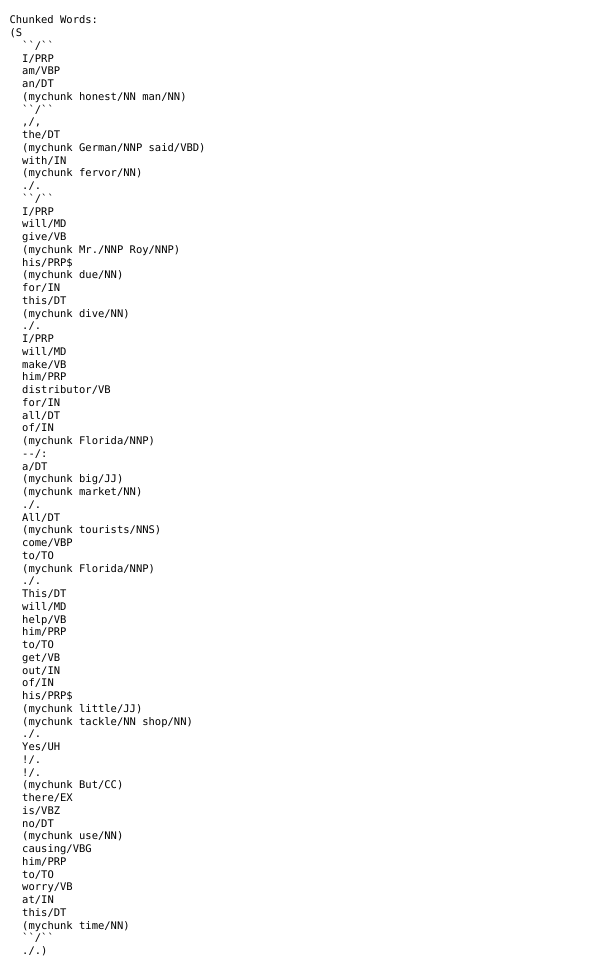
chunker = RegexpParser(grammar)

chunk = chunker.parse(tagged\_words)

print(f"Chunked Words:\n{chunk}")

**OUTPUT:**

****

****