```
import nltk
nltk.download('punkt_tab')
nltk.download('averaged_perceptron_tagger_eng')
```

```
[nltk_data] Downloading package punkt_tab to /root/nltk_data...
[nltk_data]   Unzipping tokenizers/punkt_tab.zip.
[nltk_data] Downloading package averaged_perceptron_tagger_eng to
[nltk_data]     /root/nltk_data...
[nltk_data]   Unzipping taggers/averaged_perceptron_tagger_eng.zip.
True
```

```
#Tokenization
sentence = "He sat in the city on a rainy Sunday."
tokens = nltk.word_tokenize(sentence)
print("Tokens",tokens)
```

```
Tokens ['He', 'sat', 'in', 'the', 'city', 'on', 'a', 'rainy', 'Sunday', '.']
```

```
#PoS Tagging
pos_tags = nltk.pos_tag(tokens)
print("Part of Speech Tags",pos_tags)
```

```
Part of Speech Tags [('He', 'PRP'), ('sat', 'VBD'), ('in', 'IN'), ('the', 'DT'), ('city', 'NN'), ('on', 'IN'), ('a', 'DT'), ('rainy', 'JJ'
```

```
#Chunking OR Shalow Parsing
grammar = r"""
  PP: {<IN><DT>?<JJ.*>*<NN.*>+}
"""

chunker = nltk.RegexpParser(grammar)  # This is where RegexpParser is used
tree = chunker.parse(pos_tags)

print(chunker)

print(tree)
```

```
chunk.RegexpParser with 1 stages:
RegexpChunkParser with 1 rules:
       <ChunkRule: '<IN><DT>?<JJ.*>*<NN.*>+'>
(S
  He/PRP
  sat/VBD
  (PP in/IN the/DT city/NN)
  (PP on/IN a/DT rainy/JJ Sunday/NNP)
  ./.)
```