CREATING PARSE TREES.

Activity Guidance

Create a constituency-based parse tree for the following phrases:

- The government raised interest rates.
- The internet gives everyone a voice.
- The man saw the dog with the telescope.

Grammar:

S	→	*NP VP*	[0.90]	I + feel a breeze
	|	*S Conj S*	[0.10]	I feel a breeze + and + It stinks
NP	→	*Pronoun*	[0.25]	I
	|	*Name*	[0.10]	Ali
	|	*Noun*	[0.10]	pits
	|	*Article Noun*	[0.25]	the + wumpus
	|	*Article Adjs Noun*	[0.05]	the + smelly dead + wumpus
	|	*Digit Digit*	[0.05]	3 4
	|	*NP PP*	[0.10]	the wumpus + in 1 3
	|	*NP RelClause*	[0.05]	the wumpus + that is smelly
	|	*NP Conj NP*	[0.05]	the wumpus + and + I
VP	→	*Verb*	[0.40]	stinks
	|	*VP NP*	[0.35]	feel + a breeze
	|	*VP Adjective*	[0.05]	smells + dead
	|	*VP PP*	[0.10]	is + in 1 3
	|	*VP Adverb*	[0.10]	go + ahead
Adjs	→	*Adjective*	[0.80]	smelly
	|	*Adjective Adjs*	[0.20]	smelly + dead
PP	→	*Prep NP*	[1.00]	to + the east
RelClause	→	*RelPro VP*	[1.00]	that + is smelly

Figure 24.2 The grammar for $\mathcal{E}_0$, with example phrases for each rule. The syntactic categories are sentence (*S*), noun phrase (*NP*), verb phrase (*VP*), list of adjectives (*Adjs*), prepositional phrase (*PP*), and relative clause (*RelClause*).

Picture 1: The grammar E0

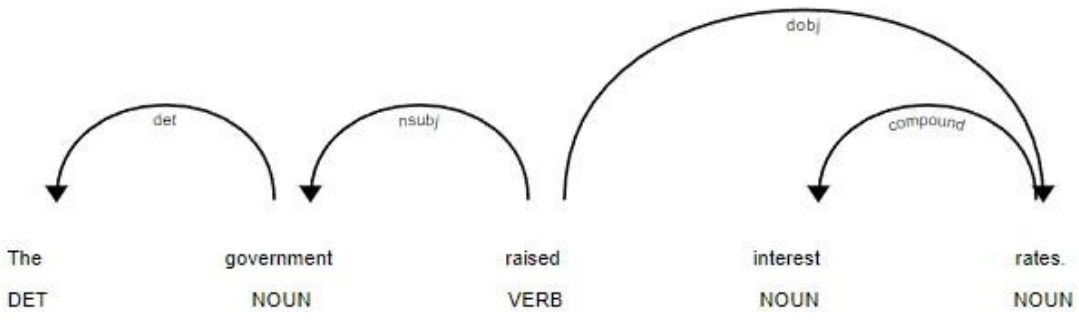
Sentences:

1. **The government raised interest rates.**

```
[1]  import spacy

[9]  from spacy import displacy

[10] nlp = spacy.load ('en_core_web_sm')

[15] text = 'The government raised interest rates.'

[16] doc = nlp(text)

[17] for token in doc:
       print(token.text,
             token.dep_,
             token.head.text,
             token.pos_,
            [child for child in token.children])
```

```
The det government DET []
government nsubj raised NOUN [The]
raised ROOT raised VERB [government, rates, .]
interest compound rates NOUN []
rates dobj raised NOUN [interest]
. punct raised PUNCT []
```

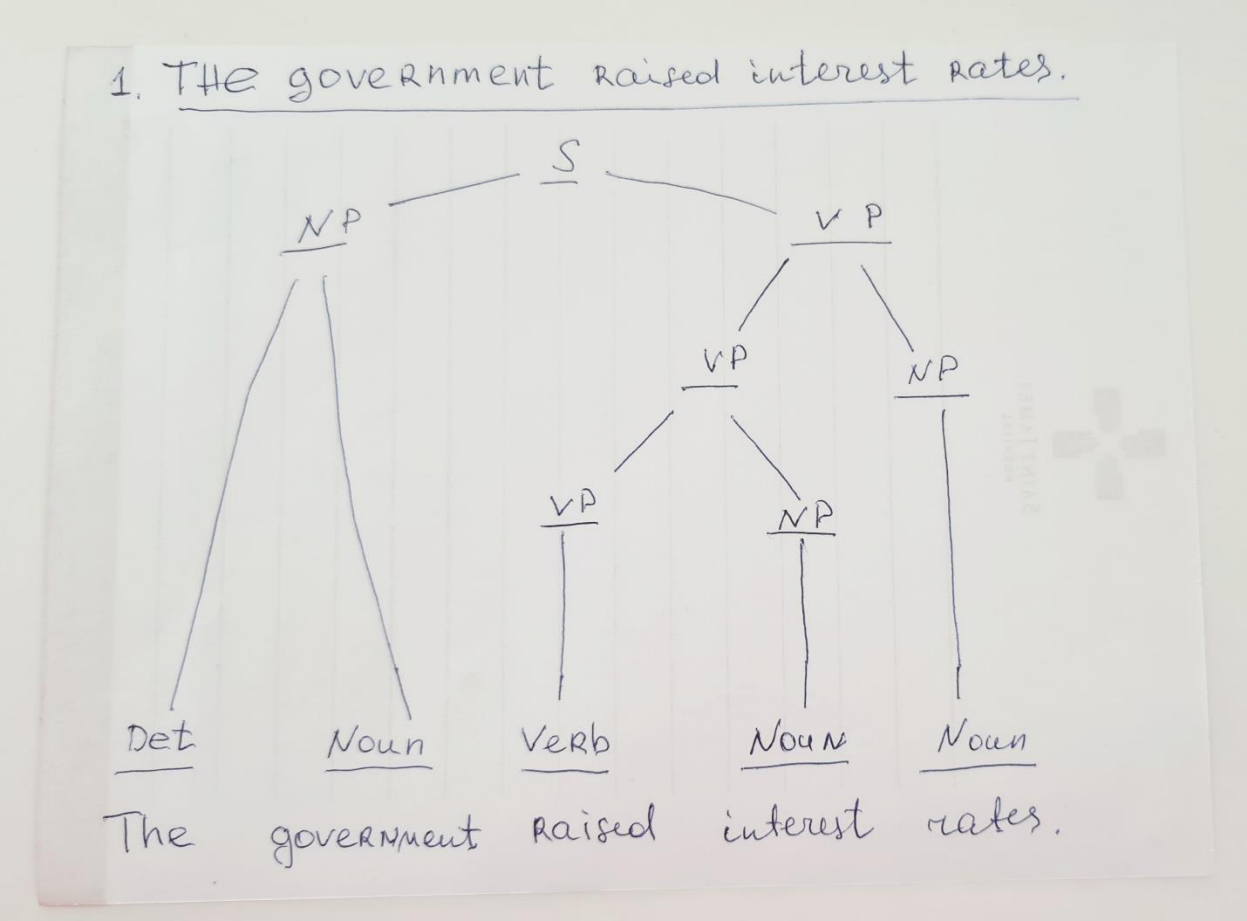
```
displacy.render(doc, style='dep', jupyter=True)
```



Screenshot 1: Code snippet parsing the string 'The government raised interest rates.' and its output.

List of items	Rule
S	
NP VP	**S -> NP VP**
NP VP NP	**VP -> VP NP**
NP VP NP NP	**VP -> VP NP**
NP Verb Noun Noun	**NP -> Noun**
NP Verb Noun rates	**Noun ->** rates
NP Verb interest rates	**Noun ->** interest
NP raised interest rates	**Verb ->** raised
Det Noun raised interest rates	**NP -> Det Noun**
Det government raised interest rates	**Noun ->** government
The government raised interest rates	**Det ->** The

Table 1: Parsing the string 'The government raised interest rates.' as a sentence, according to the grammar E0.



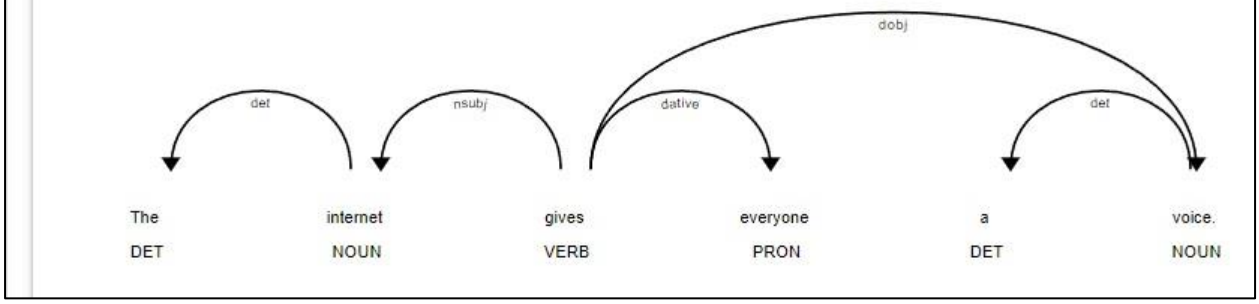
Picture 1: Parsing tree of the string 'The government raised interest rates.'.

2. The internet gives everyone a voice.

```
[1] import spacy

[9] from spacy import displacy

[10] nlp = spacy.load ('en_core_web_sm')

[19] text = 'The internet gives everyone a voice.'

[20] doc = nlp(text)

[21] for token in doc:
       print(token.text,
             token.dep_,
             token.head.text,
             token.pos_,
             [child for child in token.children])
```

```
The det internet DET []
internet nsubj gives NOUN [The]
gives ROOT gives VERB [internet, everyone, voice, .]
everyone dative gives PRON []
a det voice DET []
voice dobj gives NOUN [a]
. punct gives PUNCT []
```

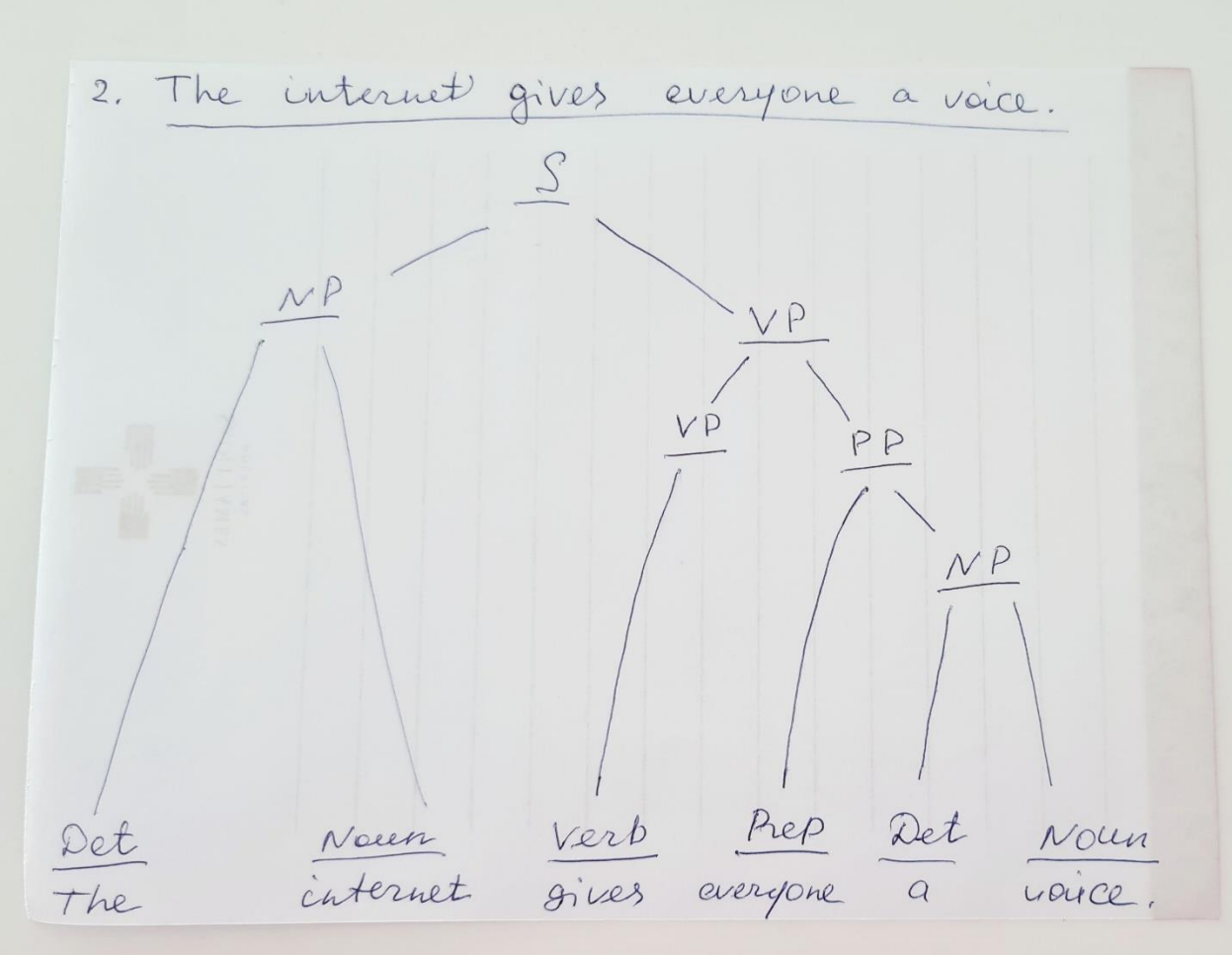
```
displacy.render(doc, style='dep', jupyter=True)
```



Screenshot 2: Code snippet parsing the string 'The internet gives everyone a voice.' and its output.

List of items	Rule
S	
NP VP	**S -> NP VP**
NP VP PP	**VP -> VP PP**
NP Verb Prep NP	**PP -> Prep NP**
NP Verb Prep Det Noun	**NP -> Det Noun**
NP Verb Prep Det voice	**Noun ->** voice
NP Verb Prep a voice	**Det ->** a
NP Verb everyone a voice	**Prep ->** everyone
NP gives everyone a voice	**Verb ->** gives
Det Noun gives everyone a voice	**NP -> Det Noun**
Det internet gives everyone a voice	**Noun ->** internet
The internet gives everyone a voice	**Det ->** The

Table 2: Parsing the string 'The internet gives everyone a voice.' as a sentence, according to the grammar E0



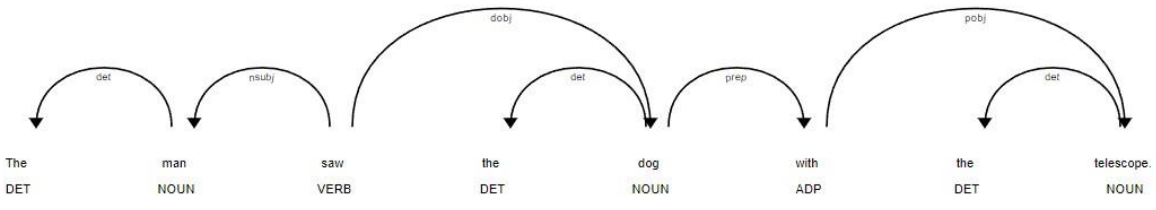
Picture 2: Parsing tree of the string 'The internet gives everyone a voice.'.

3. The man saw the dog with the telescope.

```
[1] import spacy

[9] from spacy import displacy

[10] nlp = spacy.load ('en_core_web_sm')

[11] text = 'The man saw the dog with the telescope.'

[12] doc = nlp(text)

[13] for token in doc:
       print(token.text,
             token.dep_,
             token.head.text,
             token.pos_,
             [child for child in token.children])
```

```
The det man DET []
man nsubj saw NOUN [The]
saw ROOT saw VERB [man, dog, .]
the det dog DET []
dog dobj saw NOUN [the, with]
with prep dog ADP [telescope]
the det telescope DET []
telescope pobj with NOUN [the]
. punct saw PUNCT []
```

```
displacy.render(doc, style='dep', jupyter=True)
```

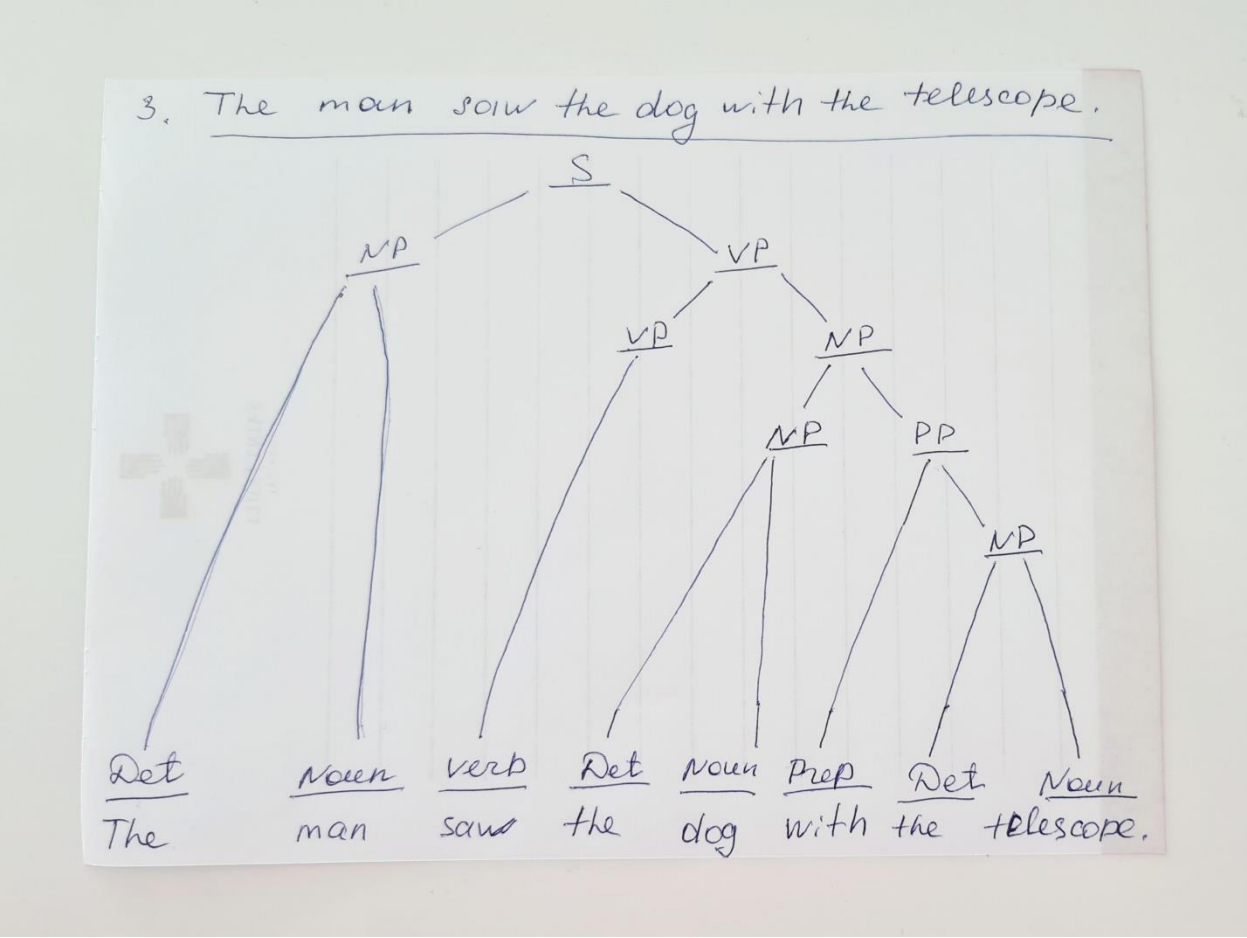


Screenshot 3: Code snippet parsing the string 'The man saw the dog with the telescope.' and its output.

List of items	Rule
S	
NP VP	**S -> NP VP**
NP VP NP	**VP -> VP NP**
NP Verb NP	**VP -> Verb**
NP Verb NP PP	**NP -> NP PP**
NP Verb NP Prep NP	**PP -> Prep NP**
NP Verb NP Prep Det Noun	**NP -> Det Noun**
NP Verb NP Prep Det telescope	**Noun ->** telescope
NP Verb NP Prep the telescope	**Det ->** the
NP Verb NP with the telescope	**Prep ->** with
NP Verb Det Noun with the telescope	**NP -> Det Noun**
NP Verb Det dog with the telescope	**Noun ->** dog
NP Verb the dog with the telescope	**Det ->** the
NP saw the dog with the telescope	**Verb ->** saw

Det Noun raised interest rates	**NP -> Det Noun**
Det man saw the dog with the telescope	**Noun ->** man
The man saw the dog with the telescope	**Det ->** The

Table 3: Parsing the string 'The man saw the dog with the telescope.' as a sentence, according to the grammar E0



Picture 3: Parsing tree of the string 'The man saw the dog with the telescope.'.

References:

Russell, S. J. & Norvig, P. (2021) *Artificial intelligence : a modern approach*. Fourth edition / contributing writers, Ming-Wei Chang [and eight others]. Upper Saddle River: Pearson.

Zimmerman, V. (2019) Towards Data Science. Getting to Grips with Parse Trees Available from: https://towardsdatascience.com/getting-to-grips-with-parse-trees-6e19e7cd3c3c Accessed [28.06 2023]