

# 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:

Section 24.2 Grammar			885
$S$	$\rightarrow NP VP$	[0.90]	I + feel a breeze
	$S Conj S$	[0.10]	I feel a breeze + and + It stinks
$NP$	$\rightarrow Pronoun$	[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
$VP$	$\rightarrow Verb$	[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
$Adjs$	$\rightarrow Adjective$	[0.80]	smelly
	$Adjective Adjs$	[0.20]	smelly + dead
$PP$	$\rightarrow Prep NP$	[1.00]	to + the east
$RelClause$	$\rightarrow RelPro VP$	[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  $\mathcal{E}_0$

## 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)
```

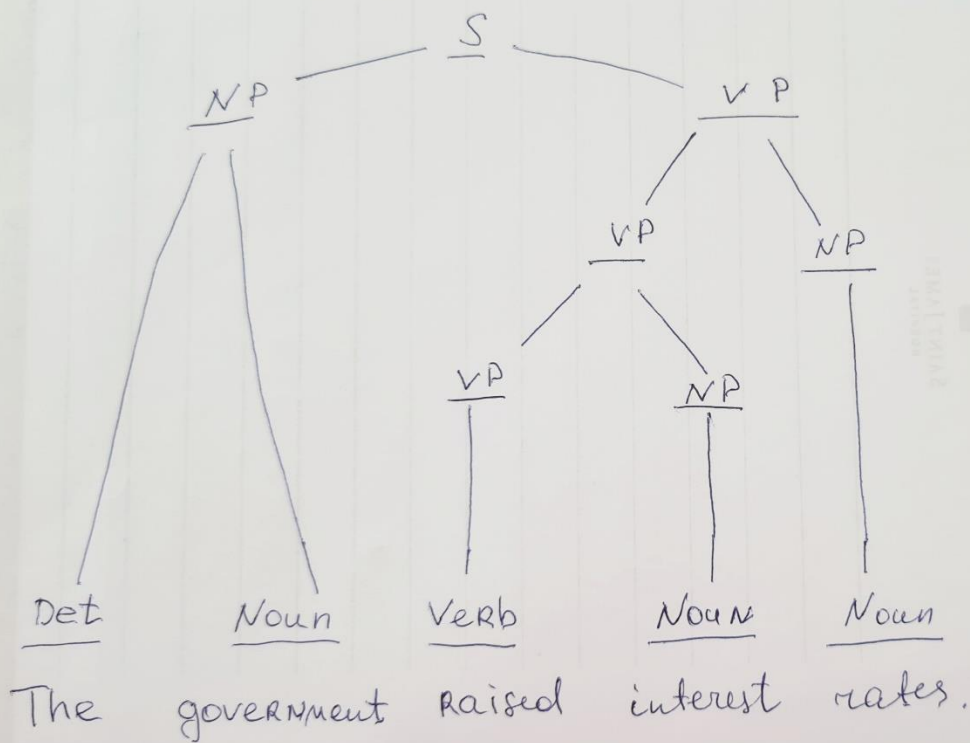
The diagram illustrates the dependency parse for the sentence "The government raised interest rates.". It shows the following tokens and their parts of speech: "The" (DET), "government" (NOUN), "raised" (VERB), "interest" (NOUN), and "rates." (NOUN). The dependencies are represented by curved arrows with labels: "det" from "The" to "government", "nsubj" from "government" to "raised", "dojb" from "raised" to "rates.", and "compound" from "interest" to "rates.".

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

List of items	Rule
<b>S</b>	
<b>NP VP</b>	<b>S -&gt; NP VP</b>
<b>NP VP NP</b>	<b>VP -&gt; VP NP</b>
<b>NP VP NP NP</b>	<b>VP -&gt; VP NP</b>
<b>NP Verb Noun Noun</b>	<b>NP -&gt; Noun</b>
<b>NP Verb Noun rates</b>	<b>Noun -&gt; rates</b>
<b>NP Verb interest rates</b>	<b>Noun -&gt; interest</b>
<b>NP raised interest rates</b>	<b>Verb -&gt; raised</b>
<b>Det Noun raised interest rates</b>	<b>NP -&gt; Det Noun</b>
<b>Det government raised interest rates</b>	<b>Noun -&gt; government</b>
<b>The government raised interest rates</b>	<b>Det -&gt; The</b>

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

1. The government raised interest rates.



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)
```

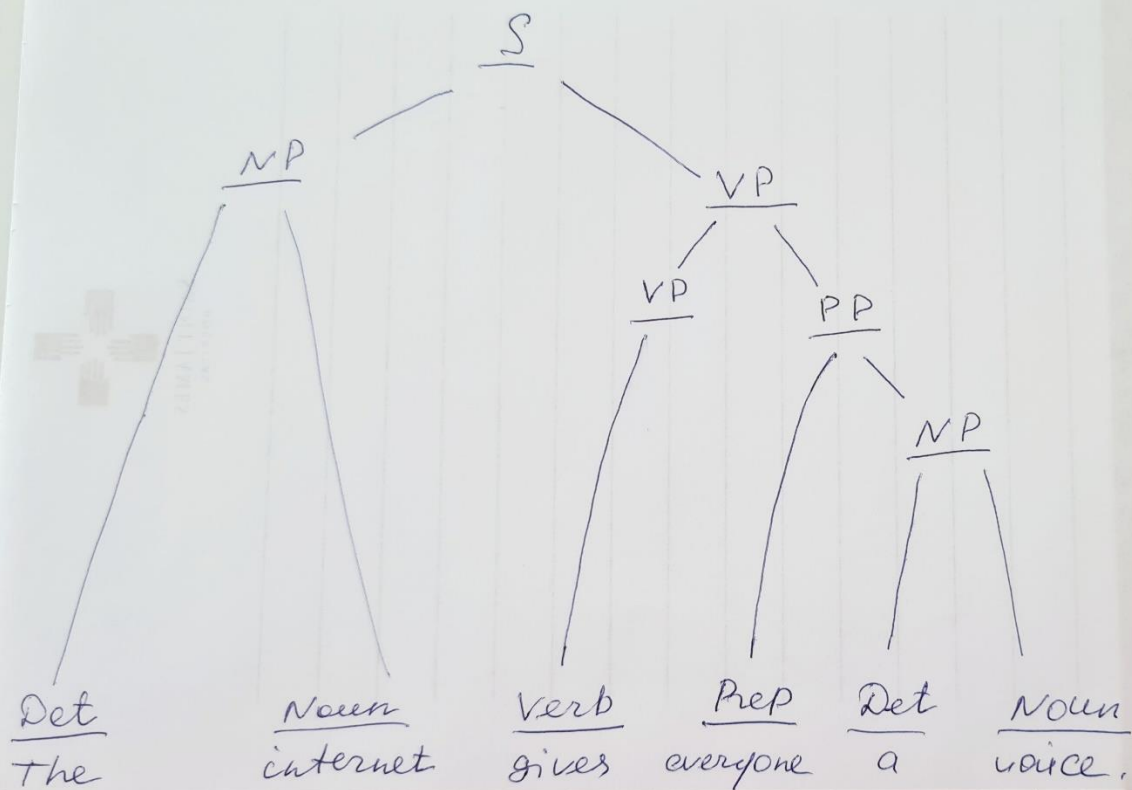
The diagram illustrates the dependency parse for the sentence "The internet gives everyone a voice.". The tokens are arranged horizontally: "The", "internet", "gives", "everyone", "a", "voice.". Below each token is its part of speech (POS) tag: "DET", "NOUN", "VERB", "PRON", "DET", "NOUN". Curved arrows (arcs) connect the tokens based on their grammatical dependencies, with labels above the arcs: "det" connects "The" to "internet"; "nsubj" connects "internet" to "gives"; "dative" connects "gives" to "everyone"; "dobj" connects "gives" to "voice."; and another "det" connects "a" to "voice.". The "dobj" arc is the longest, spanning from "gives" to "voice.".

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

List of items	Rule
<b>S</b>	
<b>NP VP</b>	<b>S -&gt; NP VP</b>
<b>NP VP PP</b>	<b>VP -&gt; VP PP</b>
<b>NP Verb Prep NP</b>	<b>PP -&gt; Prep NP</b>
<b>NP Verb Prep Det Noun</b>	<b>NP -&gt; Det Noun</b>
<b>NP Verb Prep Det voice</b>	<b>Noun -&gt; voice</b>
<b>NP Verb Prep a voice</b>	<b>Det -&gt; a</b>
<b>NP Verb everyone a voice</b>	<b>Prep -&gt; everyone</b>
<b>NP gives everyone a voice</b>	<b>Verb -&gt; gives</b>
<b>Det Noun gives everyone a voice</b>	<b>NP -&gt; Det Noun</b>
<b>Det internet gives everyone a voice</b>	<b>Noun -&gt; internet</b>
<b>The internet gives everyone a voice</b>	<b>Det -&gt; The</b>

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

2. The internet gives everyone a voice.



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)
```

The dependency parse tree for the sentence "The man saw the dog with the telescope." is shown below. The tokens are arranged horizontally, and arcs connect them based on their grammatical dependencies.

Token	Part of Speech
The	DET
man	NOUN
saw	VERB
the	DET
dog	NOUN
with	ADP
the	DET
telescope.	NOUN

Dependency arcs (labeled with the dependency type):

- The (DET) to man (NOUN) via `det`
- man (NOUN) to saw (VERB) via `nsubj`
- saw (VERB) to the (DET) via `dobj`
- the (DET) to dog (NOUN) via `det`
- dog (NOUN) to with (ADP) via `prep`
- with (ADP) to the (DET) via `pobj`
- the (DET) to telescope. (NOUN) via `det`

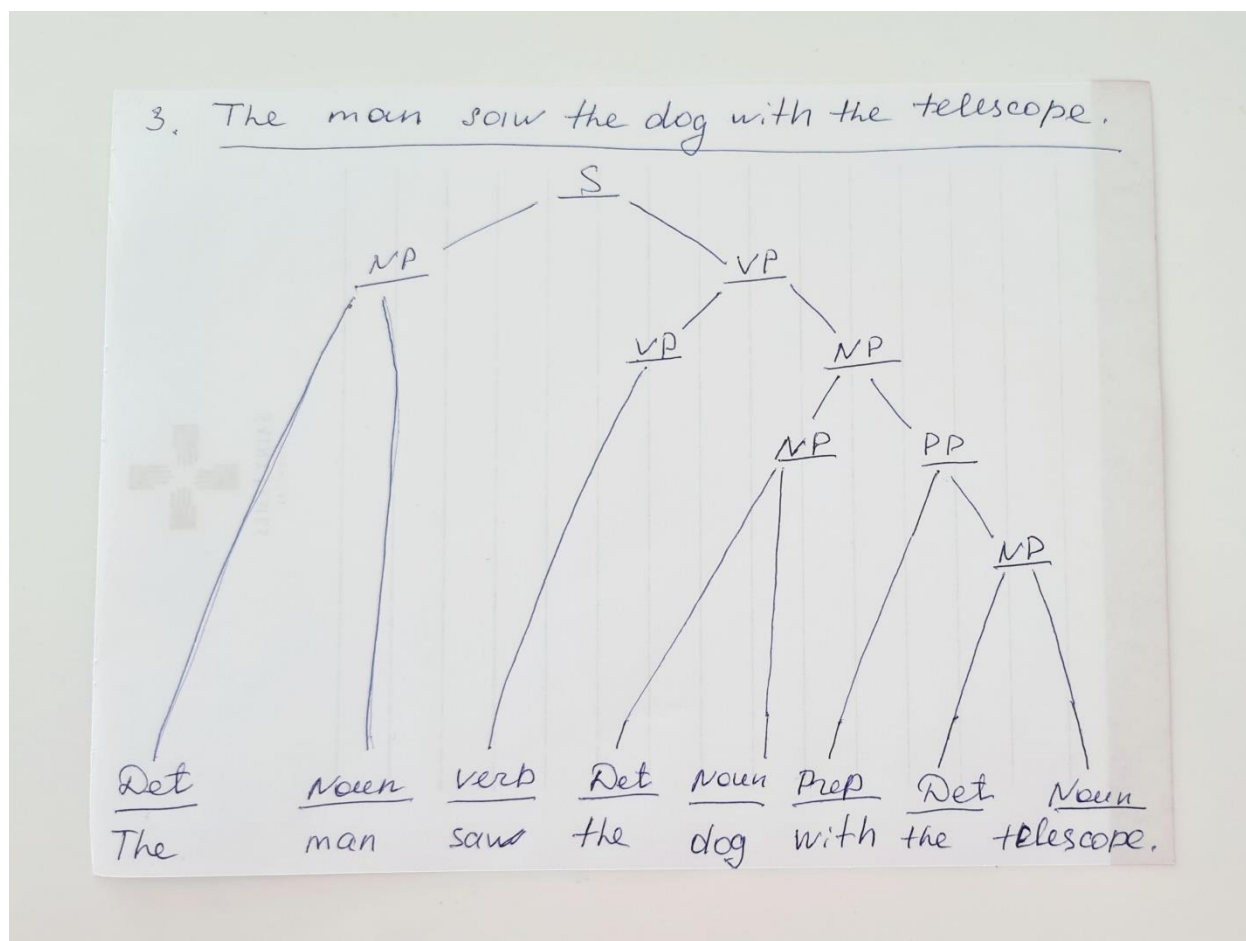
Screenshot 3: Code snippet parsing the string 'The man saw the dog with the telescope.' and its output.



List of items	Rule
<b>S</b>	
<b>NP VP</b>	<b>S -&gt; NP VP</b>
<b>NP VP NP</b>	<b>VP -&gt; VP NP</b>
<b>NP Verb NP</b>	<b>VP -&gt; Verb</b>
<b>NP Verb NP PP</b>	<b>NP -&gt; NP PP</b>
<b>NP Verb NP Prep NP</b>	<b>PP -&gt; Prep NP</b>
<b>NP Verb NP Prep Det Noun</b>	<b>NP -&gt; Det Noun</b>
<b>NP Verb NP Prep Det</b> telescope	<b>Noun -&gt; telescope</b>
<b>NP Verb NP Prep</b> the telescope	<b>Det -&gt; the</b>
<b>NP Verb NP</b> with the telescope	<b>Prep -&gt; with</b>
<b>NP Verb Det Noun</b> with the telescope	<b>NP -&gt; Det Noun</b>
<b>NP Verb Det</b> dog with the telescope	<b>Noun -&gt; dog</b>
<b>NP Verb</b> the dog with the telescope	<b>Det -&gt; the</b>
<b>NP</b> saw the dog with the telescope	<b>Verb -&gt; saw</b>

<b>Det Noun</b> raised interest rates	<b>NP -&gt; Det Noun</b>
<b>Det</b> man saw the dog with the telescope	<b>Noun -&gt; man</b>
The man saw the dog with the telescope	<b>Det -&gt; The</b>

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]