Syntax

Announcements

Natural Language Syntax

English Syntax

```
Programming languages and natural languages both have compositional syntax.
```

English Syntax

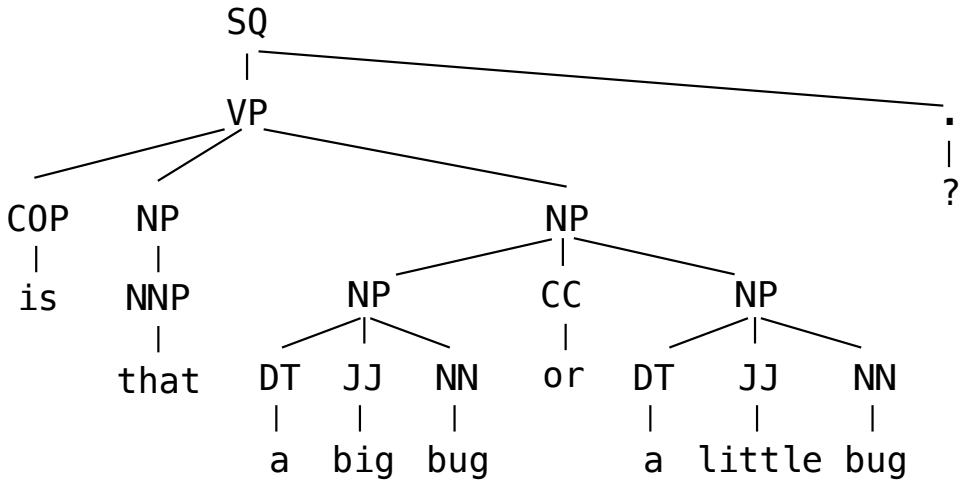
Programming languages and natural languages both have compositional syntax.

Is that a big bug or a little bug?

English Syntax

Programming languages and natural languages both have compositional syntax.

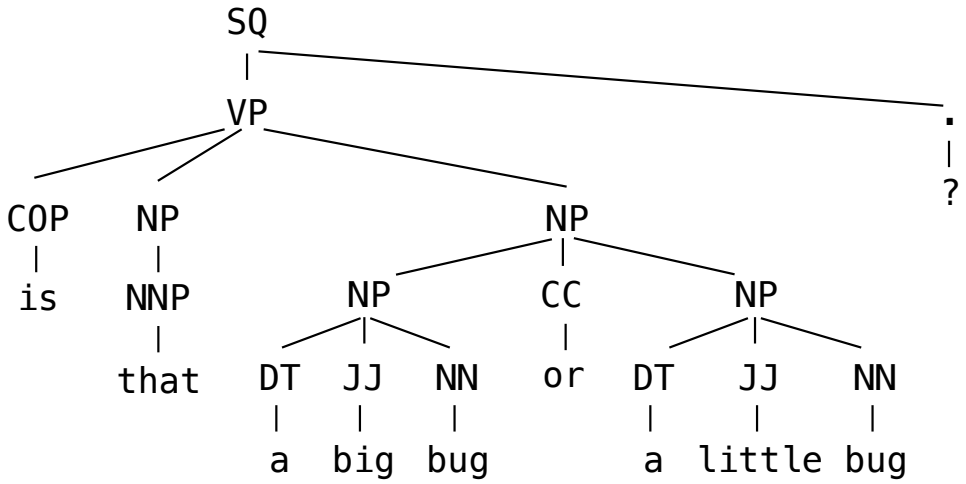
Is that a big bug or a little bug?



English Syntax

Programming languages and natural languages both have compositional syntax.

Is that a big bug or a little bug?

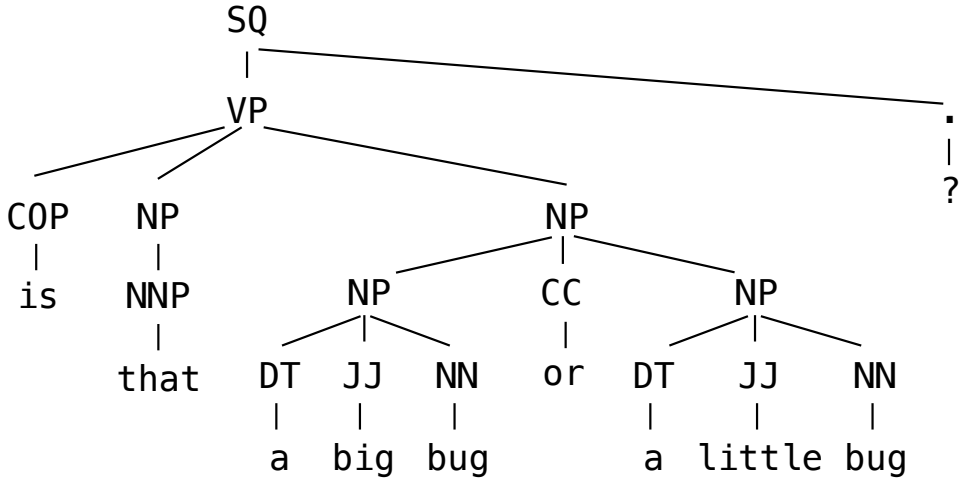


I've never seen such a cute kangaroo.

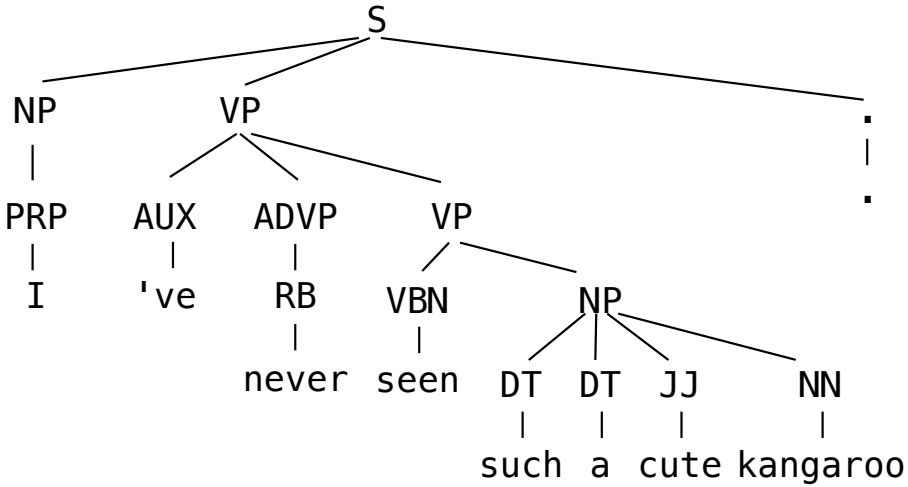
English Syntax

Programming languages and natural languages both have compositional syntax.

Is that a big bug or a little bug?



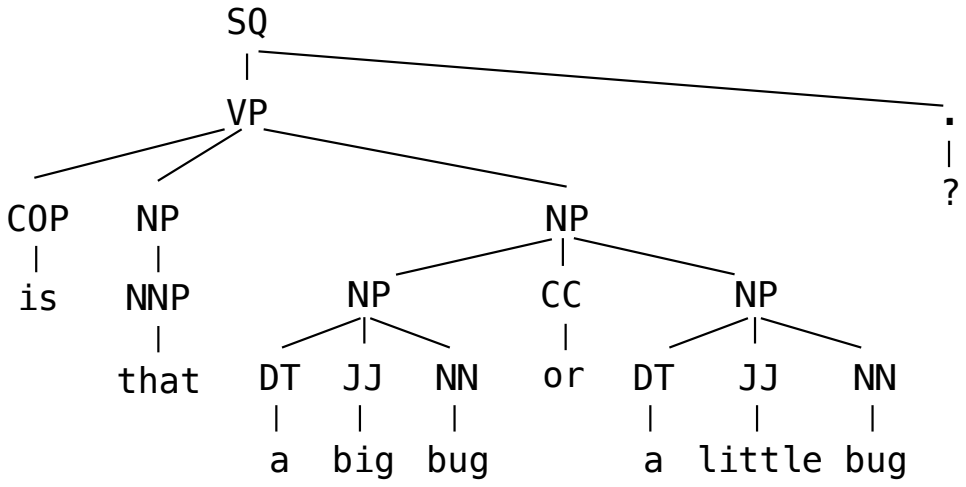
I've never seen such a cute kangaroo.



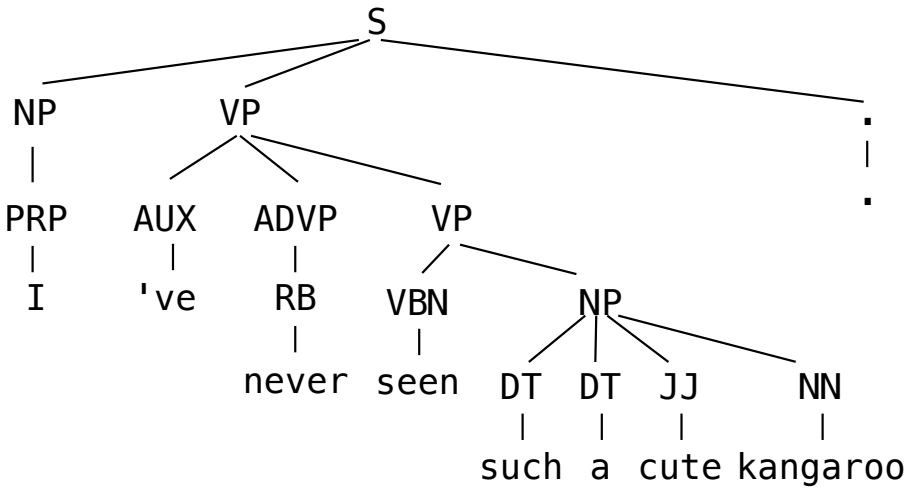
English Syntax

Programming languages and natural languages both have compositional syntax.

Is that a big bug or a little bug?



I've never seen such a cute kangaroo.



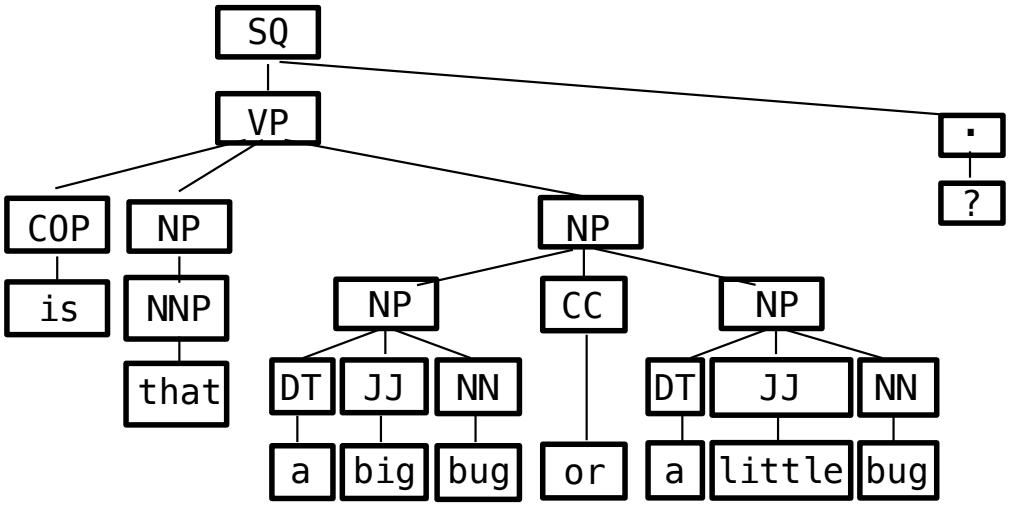
Utterances from the Suppes subject in the "Child Language Data Exchange System (CHILDES)" project

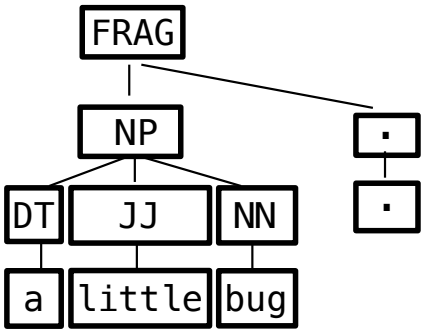
MacWhinney, B. (2000). The CHILDES Project: Tools for analyzing talk. Third Edition. Mahwah, NJ: Lawrence Erlbaum Associates.

Representing Syntax

Representing English Syntax

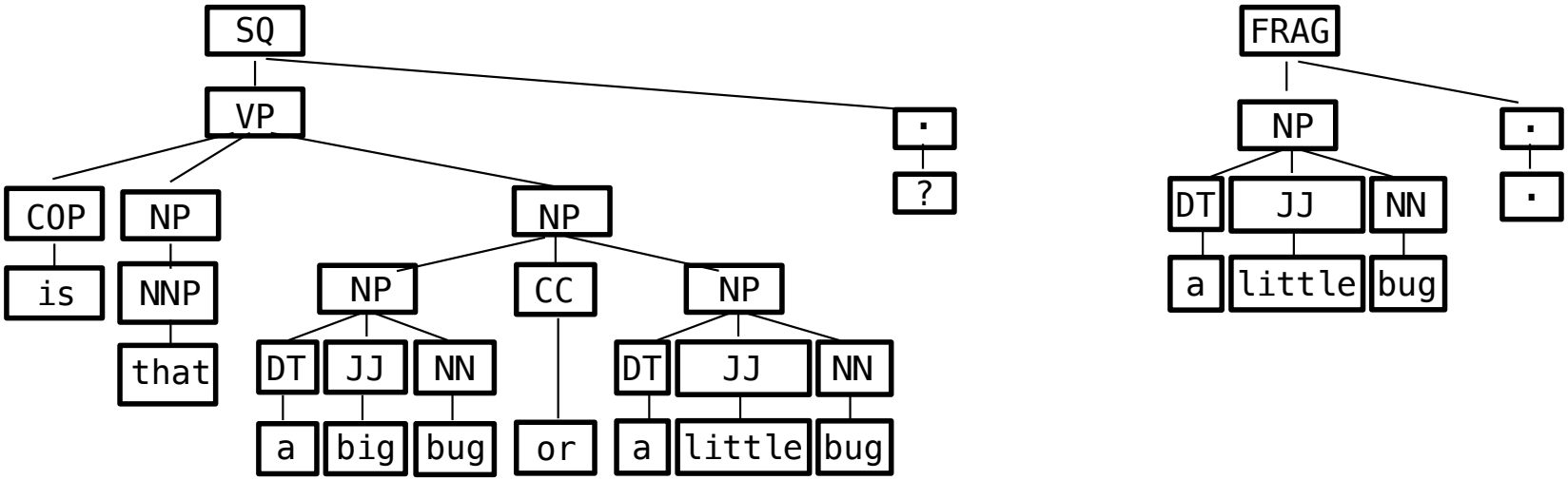
The tree data abstraction can represent the structure of a sentence.





Representing English Syntax

The tree data abstraction can represent the structure of a sentence.



(Demo)

Reading Data

Files, Strings, and Lists

Files, Strings, and Lists

Some files are plain text and can be read into Python as either:

Files, Strings, and Lists

Some files are plain text and can be read into Python as either:

- One string containing the whole contents of the file: `open('/some/file.txt').read()`

Files, Strings, and Lists

Some files are plain text and can be read into Python as either:

- One string containing the whole contents of the file: `open('/some/file.txt').read()`
- A list of strings, each containing one line: `open('/some/file.txt').readlines()`

Files, Strings, and Lists

Some files are plain text and can be read into Python as either:

- One string containing the whole contents of the file: `open('/some/file.txt').read()`
- A list of strings, each containing one line: `open('/some/file.txt').readlines()`

Useful string methods for processing the contents of a file:

Files, Strings, and Lists

Some files are plain text and can be read into Python as either:

- One string containing the whole contents of the file: **open('/some/file.txt').read()**
- A list of strings, each containing one line: **open('/some/file.txt').readlines()**

Useful string methods for processing the contents of a file:

.strip() returns a string without whitespace (spaces, tabs, etc.) on the ends

Files, Strings, and Lists

Some files are plain text and can be read into Python as either:

- One string containing the whole contents of the file: **open('/some/file.txt').read()**
- A list of strings, each containing one line: **open('/some/file.txt').readlines()**

Useful string methods for processing the contents of a file:

.strip() returns a string without whitespace (spaces, tabs, etc.) on the ends

```
>>> ' hello '.strip()
'hello'
```

Files, Strings, and Lists

Some files are plain text and can be read into Python as either:

- One string containing the whole contents of the file: **open('/some/file.txt').read()**
- A list of strings, each containing one line: **open('/some/file.txt').readlines()**

Useful string methods for processing the contents of a file:

.strip() returns a string without whitespace (spaces, tabs, etc.) on the ends

```
>>> ' hello '.strip()
'hello'
```

.split() returns a list of strings that were separated by whitespace

Files, Strings, and Lists

Some files are plain text and can be read into Python as either:

- One string containing the whole contents of the file: **open('/some/file.txt').read()**
- A list of strings, each containing one line: **open('/some/file.txt').readlines()**

Useful string methods for processing the contents of a file:

.strip() returns a string without whitespace (spaces, tabs, etc.) on the ends

```
>>> ' hello '.strip()
'hello'
```

.split() returns a list of strings that were separated by whitespace

```
>>> 'hi  there'.split()
['hi', 'there']
```

Files, Strings, and Lists

Some files are plain text and can be read into Python as either:

- One string containing the whole contents of the file: **open('/some/file.txt').read()**
- A list of strings, each containing one line: **open('/some/file.txt').readlines()**

Useful string methods for processing the contents of a file:

.strip() returns a string without whitespace (spaces, tabs, etc.) on the ends

```
>>> ' hello '.strip()
'hello'
```

.split() returns a list of strings that were separated by whitespace

```
>>> 'hi  there'.split()
['hi', 'there']
```

.replace(a, b) returns a string with all instances of string **a** replaced by string **b**

Files, Strings, and Lists

Some files are plain text and can be read into Python as either:

- One string containing the whole contents of the file: **open('/some/file.txt').read()**
- A list of strings, each containing one line: **open('/some/file.txt').readlines()**

Useful string methods for processing the contents of a file:

.strip() returns a string without whitespace (spaces, tabs, etc.) on the ends

```
>>> ' hello '.strip()
'hello'
```

.split() returns a list of strings that were separated by whitespace

```
>>> 'hi  there'.split()
['hi', 'there']
```

.replace(a, b) returns a string with all instances of string **a** replaced by string **b**

```
>>> '2+2'.replace('+', ' + ')
'2 + 2'
```

Files, Strings, and Lists

Some files are plain text and can be read into Python as either:

- One string containing the whole contents of the file: **open('/some/file.txt').read()**
- A list of strings, each containing one line: **open('/some/file.txt').readlines()**

Useful string methods for processing the contents of a file:

.strip() returns a string without whitespace (spaces, tabs, etc.) on the ends

```
>>> ' hello '.strip()
'hello'
```

.split() returns a list of strings that were separated by whitespace

```
>>> 'hi  there'.split()
['hi', 'there']
```

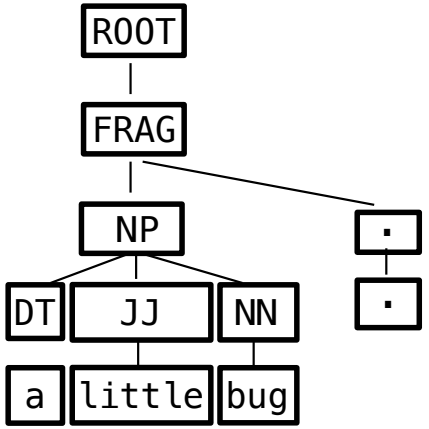
.replace(a, b) returns a string with all instances of string **a** replaced by string **b**

```
>>> '2+2'.replace('+', ' + ')
'2 + 2'
```

(Demo)

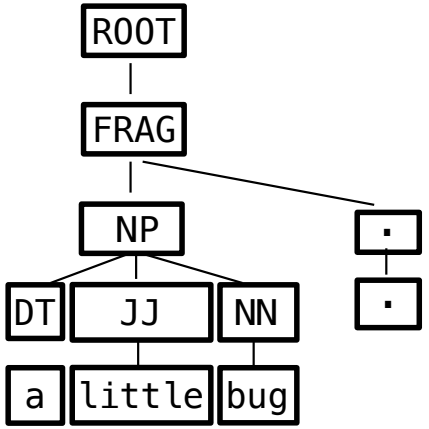
Tree Representation

A Tree Represented as a List of Tokens



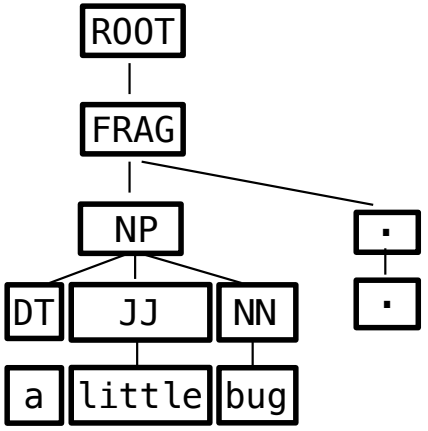
A Tree Represented as a List of Tokens

```
['(', 'ROOT', '(', 'FRAG', '(', 'NP', '(', 'DT', 'a', ')',
                                   '(', 'JJ', 'little', ')',
                                   '(', 'NN', 'bug', ')', ')',
                          '(', '.', '.', ')',                  ')', ')']
```



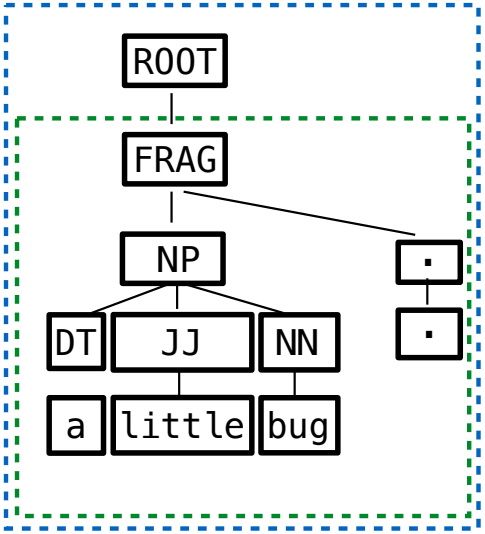
A Tree Represented as a List of Tokens

```
['(', 'ROOT', '(', 'FRAG', '(', 'NP', '(', 'DT', 'a', ')',
                                    '(', 'JJ', 'little', ')',
                                    '(', 'NN', 'bug', ')', ')',
                              '(', '.', '.', ')',                  ')', ')']
```



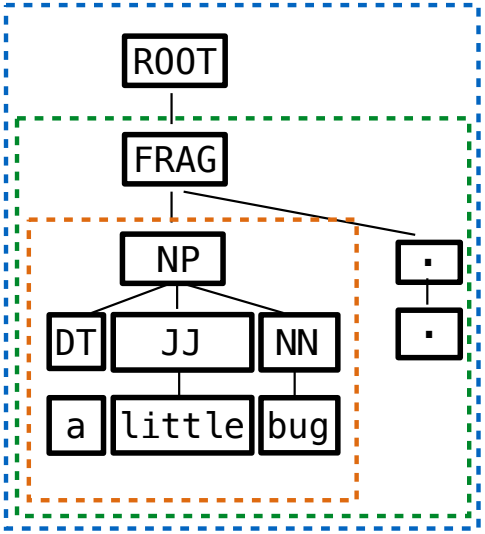
A Tree Represented as a List of Tokens

```
['(', 'ROOT', '(', 'FRAG', '(', 'NP', '(', 'DT', 'a', ')',
                                '(', 'JJ', 'little', ')',
                                '(', 'NN', 'bug', ')', ')',
                          '(', '.', '.', ')',                  ')', ')']
```



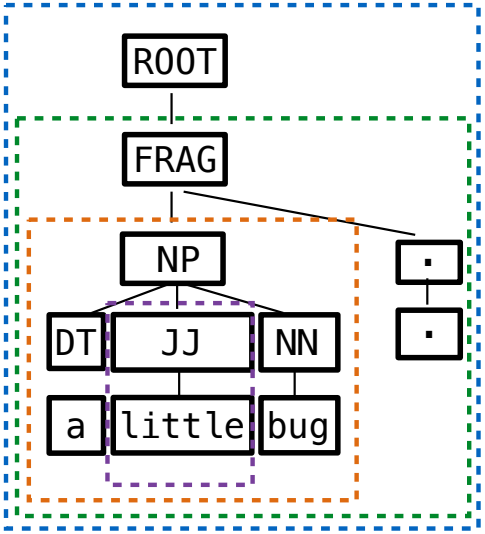
A Tree Represented as a List of Tokens

```
['(', 'ROOT', '(', 'FRAG', '(', 'NP', '(', 'DT', 'a', ')',
                                '(', 'JJ', 'little', ')',
                                '(', 'NN', 'bug', ')', ')',
                '(', '.', '.', ')',                    ')', ')']
```



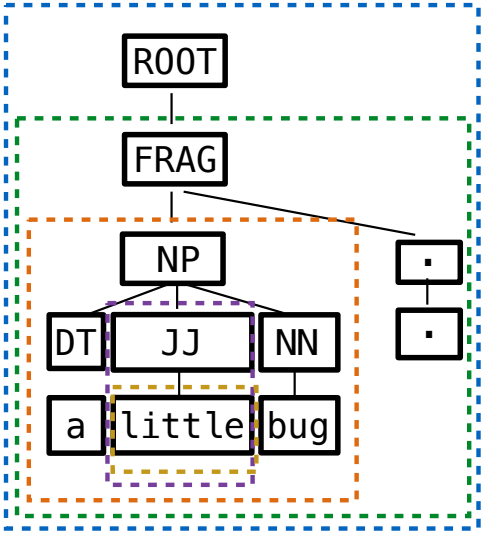
A Tree Represented as a List of Tokens

```
['(', 'ROOT', '(', 'FRAG', '(', 'NP', '(', 'DT', 'a', ')',
                                '(', 'JJ', 'little', ')',
                                '(', 'NN', 'bug', ')', ')',
                '(', '.', '.', ')',                    ')', ')']
```



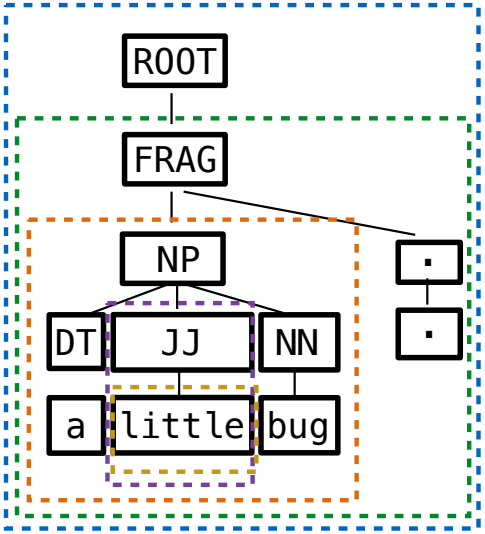
A Tree Represented as a List of Tokens

```
['(', 'ROOT', '(', 'FRAG', '(', 'NP', '(', 'DT', 'a', ')',
                                '(', 'JJ', 'little', ')',
                                '(', 'NN', 'bug', ')', ')',
                '(', '.', '.', ')',                    ')', ')']
```



A Tree Represented as a List of Tokens

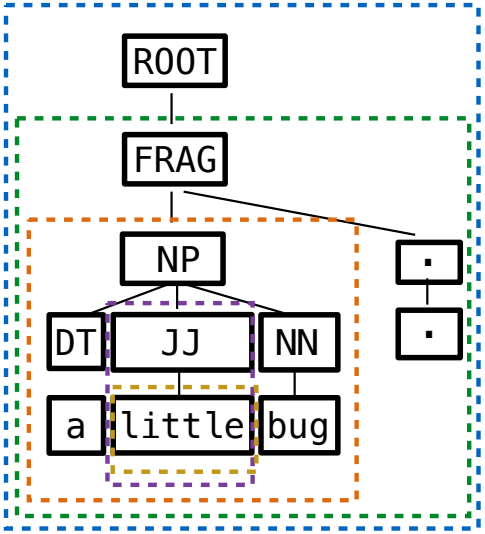
```
['(', 'ROOT', '(', 'FRAG', '(', 'NP', '(', 'DT', 'a', ')',
                                '(', 'JJ', 'little', ')',
                                '(', 'NN', 'bug', ')', ')',
                  '(', '.', '.', ')',                    ')', ')']
```




```
def tree(label, branches=[]):
    if not branches:
        return [label]
    else:
        return ['(', label] + sum(branches, start=[]) + [')']
```

A Tree Represented as a List of Tokens

```
['(', 'ROOT', '(', 'FRAG', '(', 'NP', '(', 'DT', 'a', ')',
                                '(', 'JJ', 'little', ')',
                                '(', 'NN', 'bug', ')', ')',
                  '(', '.', '.', ')',                    ')', ')']
```




```
def tree(label, branches=[]):
    if not branches:
        return [label]
    else:
        return ['(', label] + sum(branches, start=[]) + [')']
```

(Demo)

Finding Branches

```
['(', 'NP', '(', 'DT', 'a', ')', '(', 'JJ', 'little', ')',  '(', 'NN', 'bug', ')', ')']
```

Finding Branches

```
['(', 'NP', '(', 'DT', 'a', ')', '(', 'JJ', 'little', ')',  '(', 'NN', 'bug', ')', ')']
```

```
current_branch: [                                                                    ]
```

Finding Branches

```
['(', 'NP', '(', 'DT', 'a', ')', '(', 'JJ', 'little', ')',  '(', 'NN', 'bug', ')', ')']


current_branch: [                                                                  ]

all_branches:   [                                                                  ]
```

Finding Branches

```
['(', 'NP', '(', 'DT', 'a', ')', '(', 'JJ', 'little', ')',  '(', 'NN', 'bug', ')', ')']


current_branch: [ '('                                                              ]

all_branches:   [                                                                  ]
```

Finding Branches

```
['(', 'NP', '(', 'DT', 'a', ')', '(', 'JJ', 'little', ')',  '(', 'NN', 'bug', ')', ')']


current_branch: [  '(', 'DT'                                                    ]
all_branches:   [                                                               ]
```

Finding Branches

```
['(', 'NP', '(', 'DT', 'a', ')', '(', 'JJ', 'little', ')',  '(', 'NN', 'bug', ')', ')']


current_branch: [  '(', 'DT', 'a'                                                    ]
all_branches:   [                                                                    ]
```

Finding Branches

```
['(', 'NP', '(', 'DT', 'a', ')', '(', 'JJ', 'little', ')',  '(', 'NN', 'bug', ')', ')']


current_branch: [ '(', 'DT', 'a', ')'                                                ]
all_branches:   [                                                                    ]
```

Finding Branches

```
['(', 'NP', '(', 'DT', 'a', ')', '(', 'JJ', 'little', ')',  '(', 'NN', 'bug', ')', ')']


current_branch: [  '(', 'DT', 'a', ')'                                                ]
all_branches:   [ ['(', 'DT', 'a', ')']                                               ]
```

Finding Branches

```
['(', 'NP', '(', 'DT', 'a', ')', '(', 'JJ', 'little', ')',  '(', 'NN', 'bug', ')', ')']


current_branch: [                                                                  ]
all_branches:   [['(', 'DT', 'a', ')']                                            ]
```

Finding Branches

```
['(', 'NP', '(', 'DT', 'a', ')', '(', 'JJ', 'little', ')',  '(', 'NN', 'bug', ')', ')']


current_branch: [                        '('                                    ]
all_branches:   [['(', 'DT', 'a', ')']                                         ]
```

Finding Branches

```
['(', 'NP', '(', 'DT', 'a', ')', '(', 'JJ', 'little', ')',  '(', 'NN', 'bug', ')', ')']


current_branch: [                        '(', 'JJ'                                    ]
all_branches:   [['(', 'DT', 'a', ')']                                               ]
```

Finding Branches

```
['(', 'NP', '(', 'DT', 'a', ')', '(', 'JJ', 'little', ')',  '(', 'NN', 'bug', ')', ')']


current_branch: [                        '(', 'JJ', 'little'                                ]
all_branches:   [['(', 'DT', 'a', ')']                                                     ]
```

Finding Branches

```
['(', 'NP', '(', 'DT', 'a', ')', '(', 'JJ', 'little', ')',  '(', 'NN', 'bug', ')', ')']


current_branch: [                        '(', 'JJ', 'little', ')'                          ]
all_branches:   [['(', 'DT', 'a', ')']                                                    ]
```

Finding Branches

```
['(', 'NP', '(', 'DT', 'a', ')', '(', 'JJ', 'little', ')',  '(', 'NN', 'bug', ')', ')']


current_branch: [                        '(', 'JJ', 'little', ')'                          ]
all_branches:   [['(', 'DT', 'a', ')'], ['(', 'JJ', 'little', ')']                        ]
```

Finding Branches

```
['(', 'NP', '(', 'DT', 'a', ')', '(', 'JJ', 'little', ')',  '(', 'NN', 'bug', ')', ')']


current_branch: [                                                                    ]
all_branches:   [['(', 'DT', 'a', ')'], ['(', 'JJ', 'little', ')']                  ]
```

Finding Branches

```
['(', 'NP', '(', 'DT', 'a', ')', '(', 'JJ', 'little', ')',  '(', 'NN', 'bug', ')', ')']


current_branch: [                                                '('                    ]
all_branches:   [['(', 'DT', 'a', ')'], ['(', 'JJ', 'little', ')']                     ]
```

Finding Branches

```
['(', 'NP', '(', 'DT', 'a', ')', '(', 'JJ', 'little', ')',  '(', 'NN', 'bug', ')', ')']


current_branch: [                                                '(', 'NN'              ]
all_branches:   [['(', 'DT', 'a', ')'], ['(', 'JJ', 'little', ')']                     ]
```

Finding Branches

```
['(', 'NP', '(', 'DT', 'a', ')', '(', 'JJ', 'little', ')',  '(', 'NN', 'bug', ')', ')']


current_branch: [                                                '(', 'NN', 'bug'      ]
all_branches:   [['(', 'DT', 'a', ')'], ['(', 'JJ', 'little', ')']                    ]
```

Finding Branches

```
['(', 'NP', '(', 'DT', 'a', ')', '(', 'JJ', 'little', ')',  '(', 'NN', 'bug', ')', ')']


current_branch: [                                                '(', 'NN', 'bug', ')'  ]
all_branches:   [['(', 'DT', 'a', ')'], ['(', 'JJ', 'little', ')']                      ]
```

Finding Branches

```
['(', 'NP', '(', 'DT', 'a', ')', '(', 'JJ', 'little', ')',  '(', 'NN', 'bug', ')', ')']


current_branch: [                                                  '(', 'NN', 'bug', ')'  ]
all_branches:   [['(', 'DT', 'a', ')'], ['(', 'JJ', 'little', ')'], ['(', 'NN', 'bug', ')']]
```

Finding Branches

```
['(', 'NP', '(', 'DT', 'a', ')', '(', 'JJ', 'little', ')',  '(', 'NN', 'bug', ')', ')']


current_branch: [                                                                  ]
all_branches:   [['(', 'DT', 'a', ')'], ['(', 'JJ', 'little', ')'], ['(', 'NN', 'bug', ')']]
```

Finding Branches

```
['(', 'NP', '(', 'DT', 'a', ')', '(', 'JJ', 'little', ')',  '(', 'NN', 'bug', ')', ')']


current_branch: [                                                                  ]
all_branches:   [['(', 'DT', 'a', ')'], ['(', 'JJ', 'little', ')'], ['(', 'NN', 'bug', ')']]
```

(Demo)

Manipulating Language

(Demo)