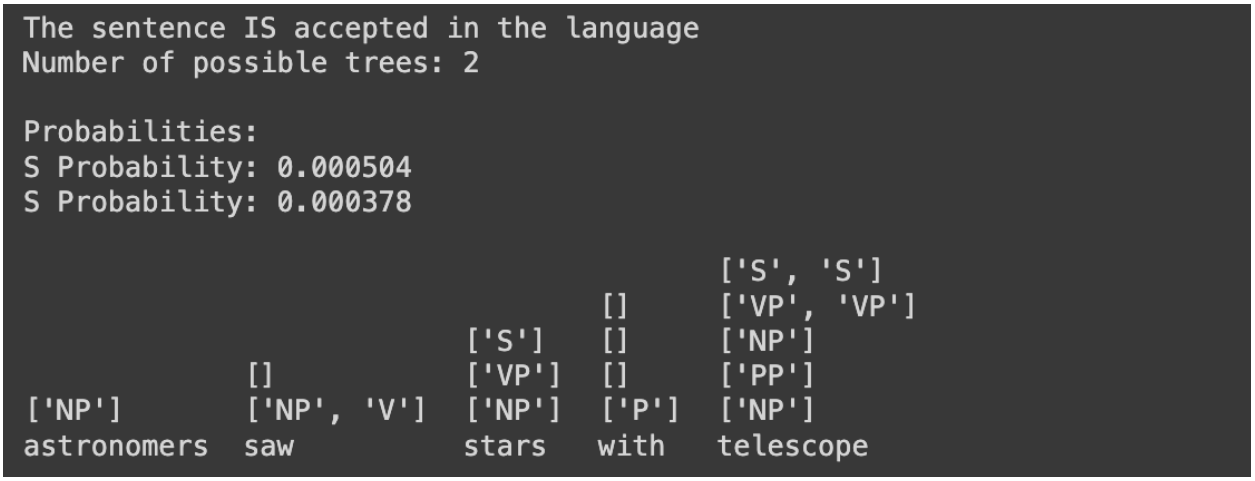
NLP Assignment 4
112003066 – Om Khare – Division 1

Output:



```
The sentence IS accepted in the language
Number of possible trees: 2

Probabilities:
S Probability: 0.000504
S Probability: 0.000378

                                         ['S', 'S']
                                  []     ['VP', 'VP']
                          ['S']   []     ['NP']
             []           ['VP']  []     ['PP']
['NP']       ['NP', 'V']  ['NP']  ['P']  ['NP']
astronomers  saw          stars   with   telescope
```

The provided code represents a Python implementation of a probabilistic context-free grammar (PCFG) parser for natural language processing. In this writeup, I will explain the key components and functionality of the code.

Imports:

- The code imports the necessary libraries and modules:
    - math for mathematical functions.
    - nltk (Natural Language Toolkit) is imported but not used in the code.
    - tabulate for creating formatted tables for displaying the parse table.

Custom Dictionary Class (Dictlist):

- The Dictlist class extends the built-in Python dict class. It allows for dictionary keys to have multiple values in a list.
- It achieves this by overriding the __setitem__ method, creating an empty list for a key if it does not exist and then appends the new value to the list.

ProductionRule Class:

- The ProductionRule class represents a production rule in the PCFG. Each rule has the following attributes:
    - result: The non-terminal symbol on the left-hand side of the rule.
    - p1 and p2: The non-terminal symbols or terminal words on the right-hand side of the rule.
    - probability: The probability associated with this rule.
- It provides properties to access these attributes.

Cell Class:

- The Cell class represents a cell in the parse table. Each cell can store multiple production rules.

- It provides methods to add and retrieve production rules associated with the cell.

Grammar Class:

- The Grammar class represents a probabilistic context-free grammar.
- The class constructor takes a file name as input and initializes the grammar rules from the provided file.
- The grammar rules are stored in a Dictlist.
- The apply_rules method returns the production rules for a given non-terminal symbol (if they exist).
- The parse method is responsible for parsing a sentence. It constructs a parse table and computes the most probable parse.
- The print_probabilities method displays the probabilities associated with the possible parses.
- The print_parse_table method displays the parse table in a tabulated format.
- The code reads a PCFG grammar from the 'pcfg_grammar.txt' file and then parses the sentence 'astronomers saw stars with telescope'.

Execution:

- An instance of the Grammar class is created with the provided grammar file.
- The parse method is called with the sentence as input.
- The code then prints whether the sentence is accepted in the language and the number of possible parse trees.
- Finally, it prints the most probable parse tree and the associated probabilities.