**SAVEETHA SCHOOL OF ENGINEERING**

**SAVEETHA INSTITUTE OF MEDICAL AND TECHNICAL SCIENCES**

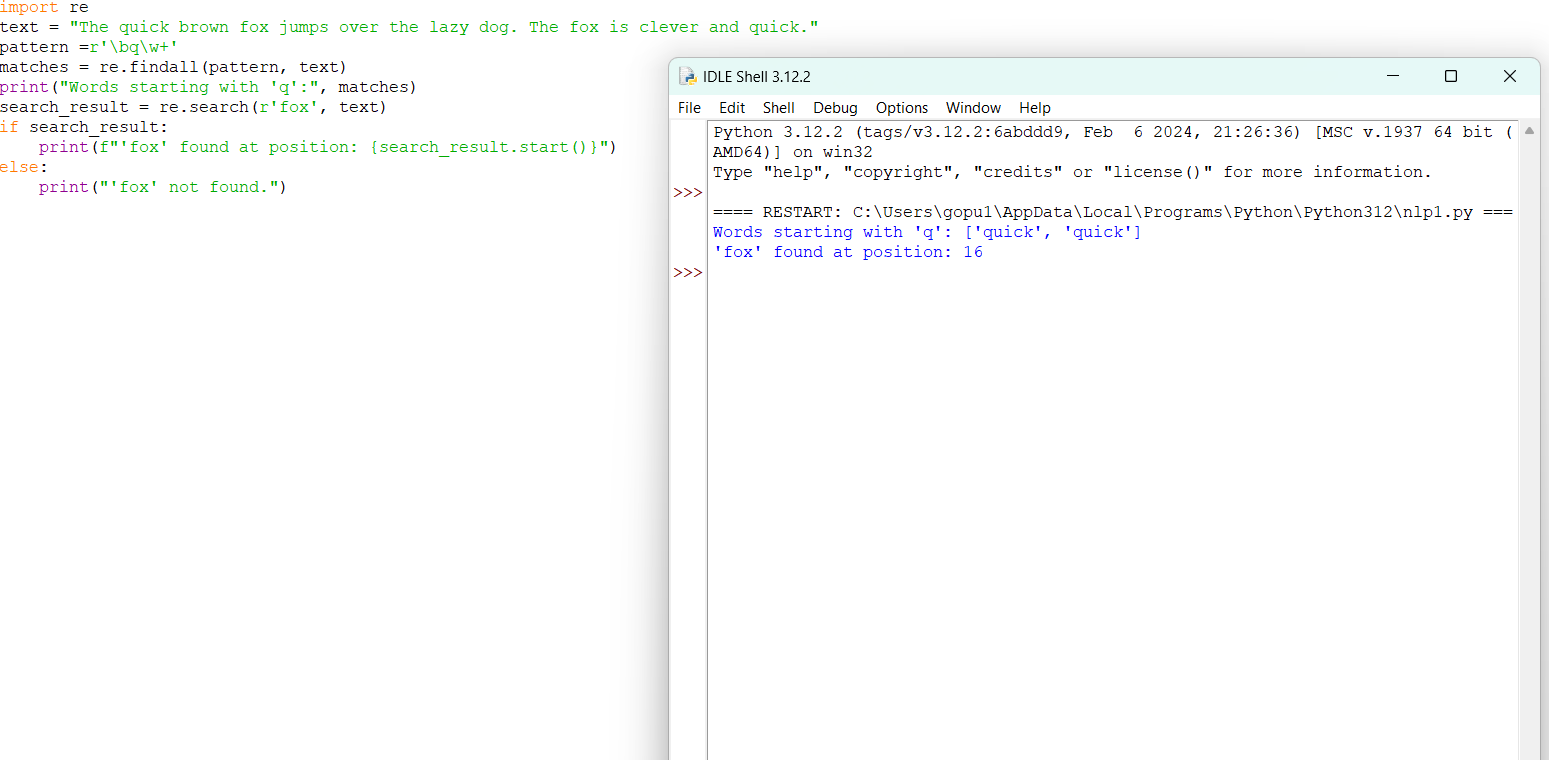
DSA0317 – NATURAL LANGUAGE PROCESSING

Practical lab manual

Submitted by Gopika. M [192224038]

1. Write program demonstrates how to use regular expressions in Python to match and search for patterns in text.

Program:

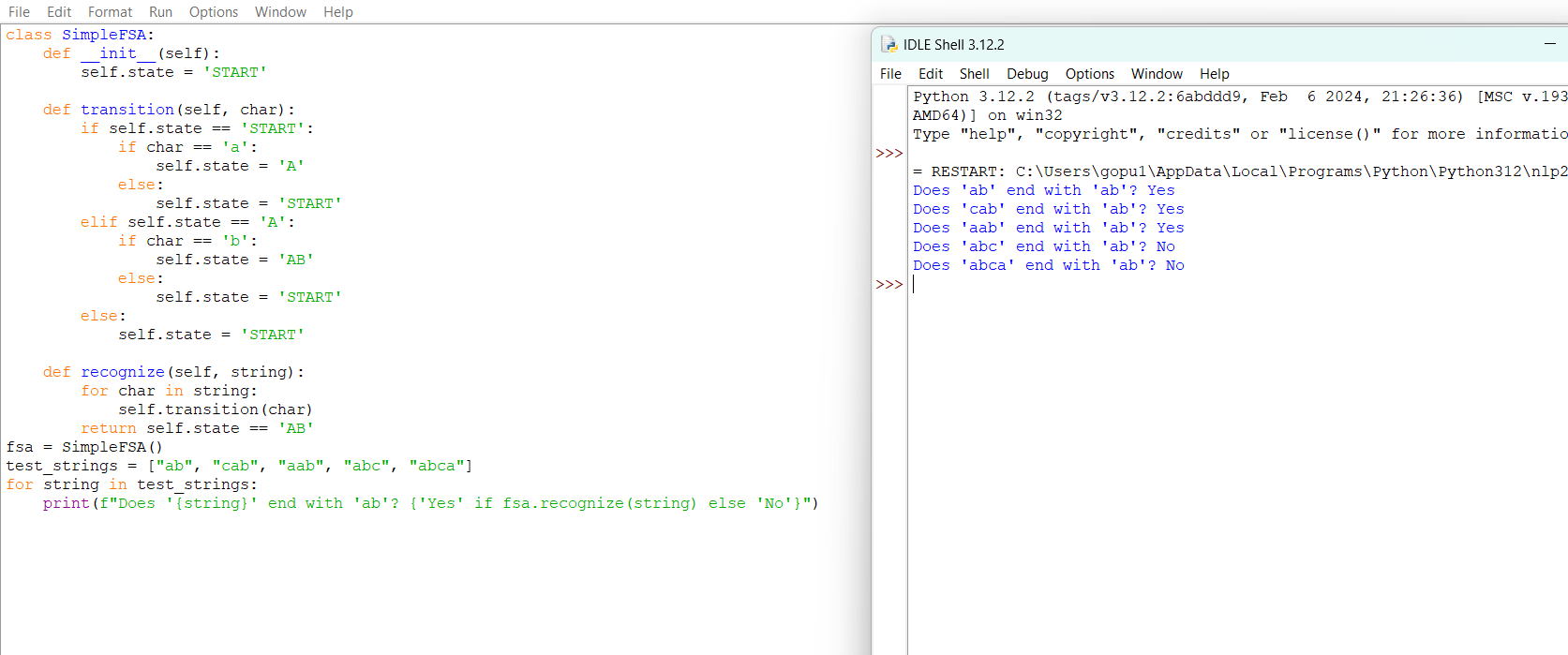


Result:

Regular expression using python program executed successfully.

1. Implement a basic finite state automaton that recognizes a specific language or pattern. In this example, we'll create a simple automaton to match strings ending with 'ab' using python.

Program:

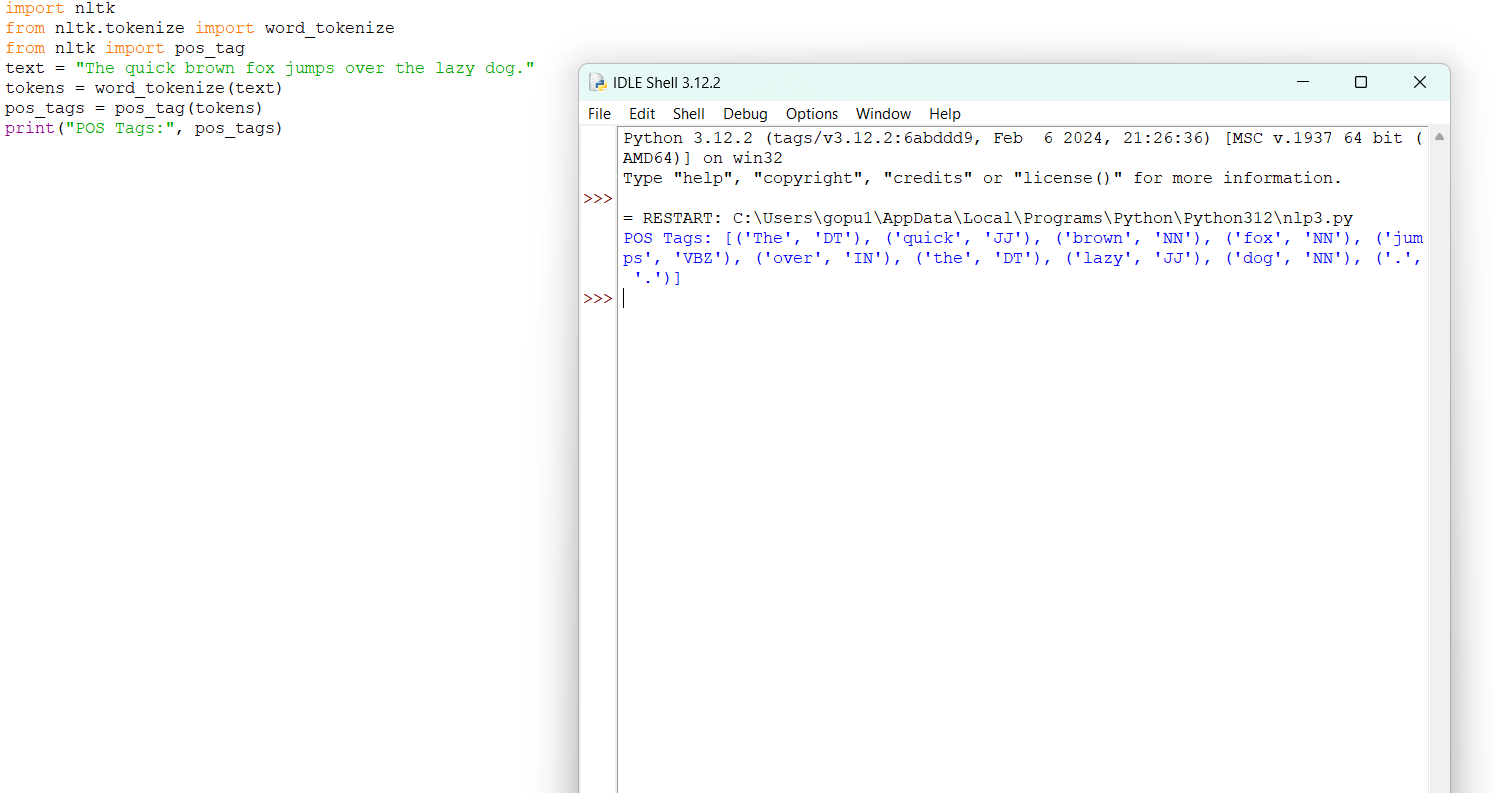


Result:

FSA program using python program executed successfully.

3.Write program demonstrates how to perform morphological analysis using the NLTK library in Python.

Program:

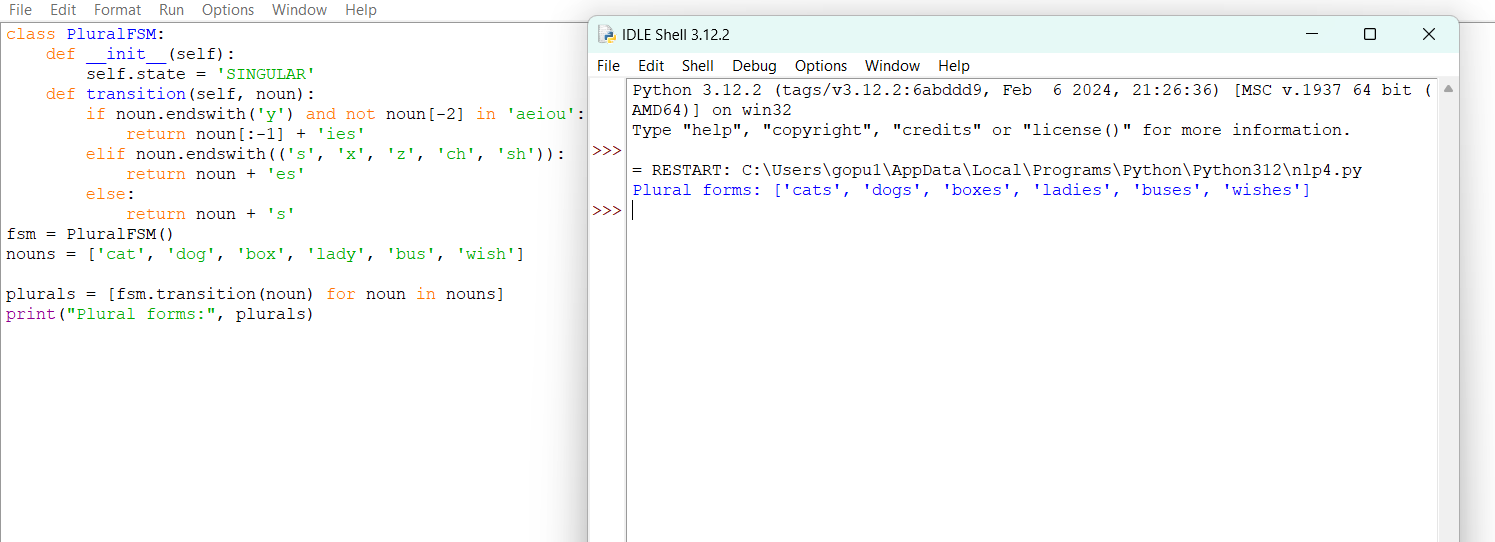


Result:

Morphological analysis using python program executed successfully.

4.Implement a finite-state machine for morphological parsing. In this example, we'll create a simple machine to generate plural forms of English nouns using python.

Program:

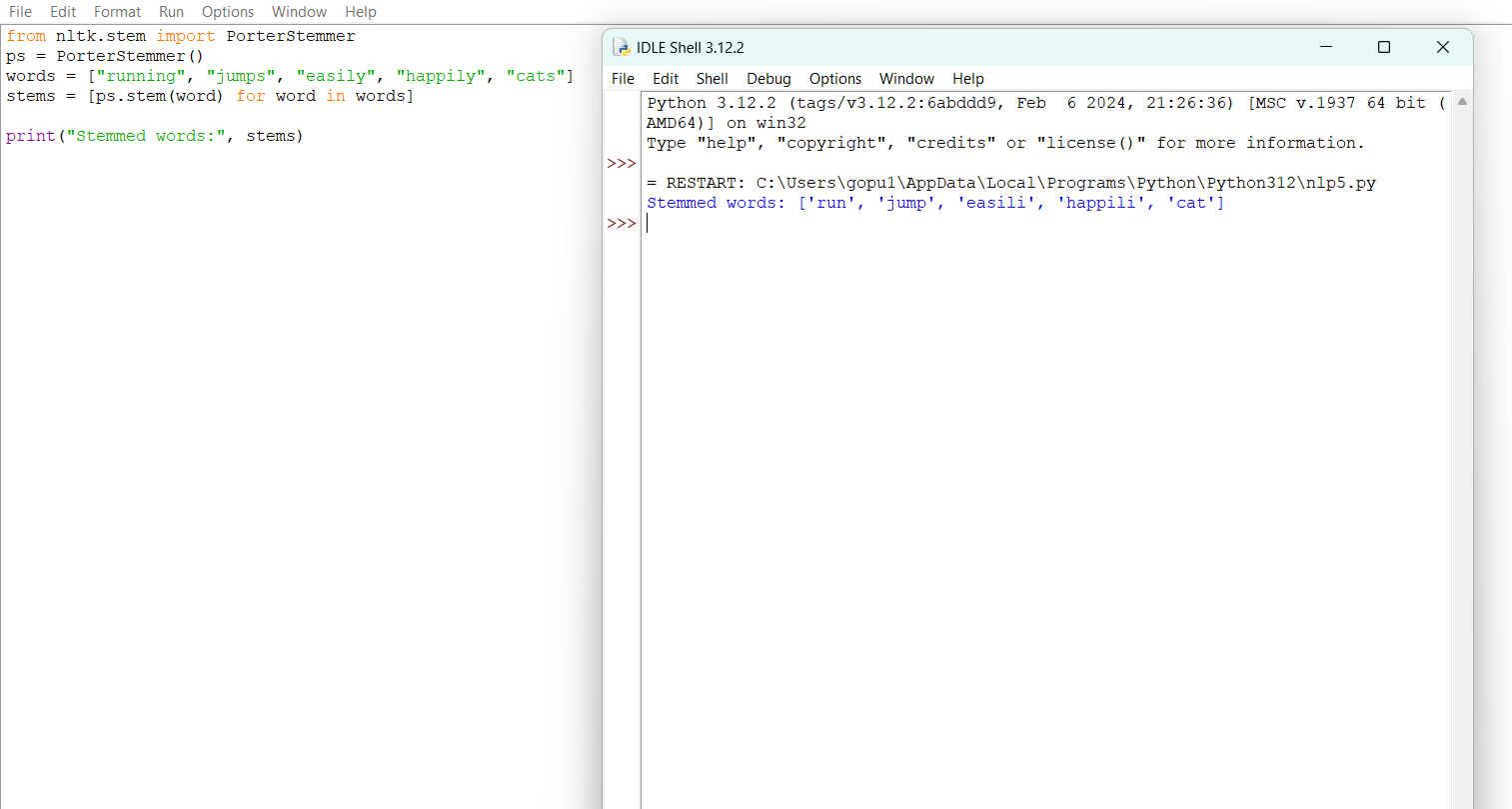


Result:

Morphological parser using python program executed successfully.

5.Use the Porter Stemmer algorithm to perform word stemming on a list of words using python libraries.

Program:

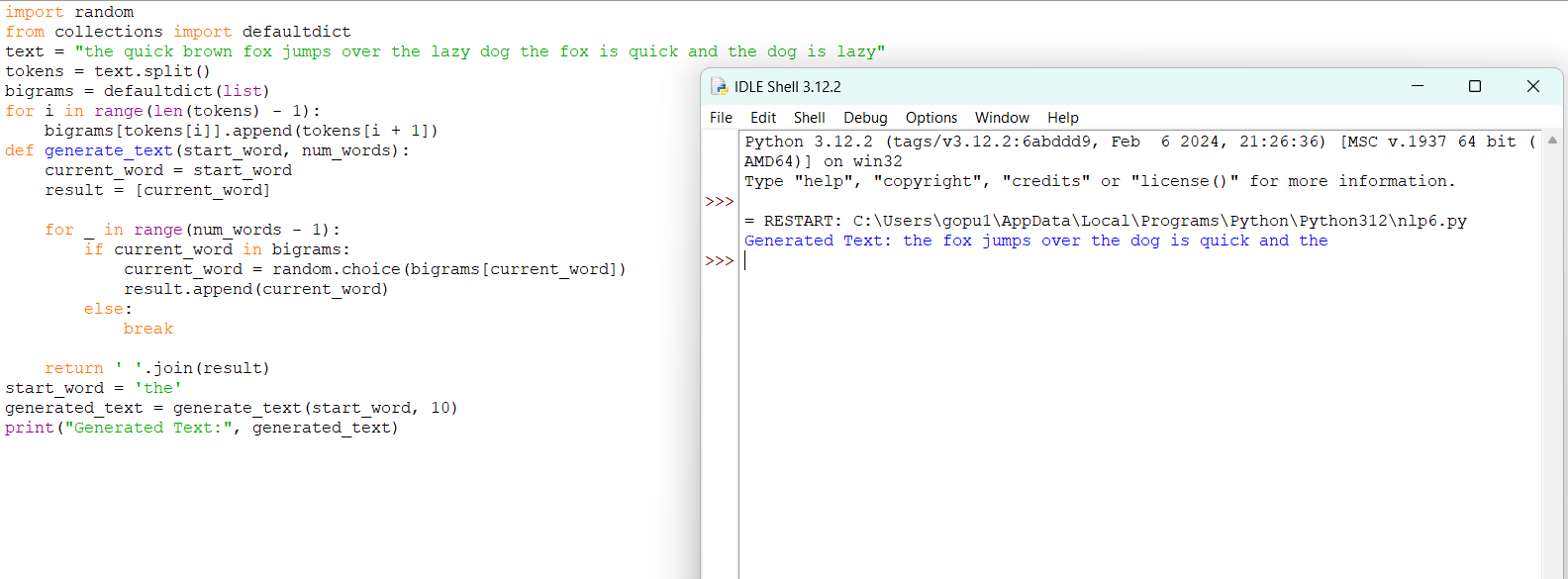


Result:

Porter stemmer algorithm using python program executed successfully.

6.Implement a basic N-gram model for text generation. For example, generate text using a bigram model using python.

Program:

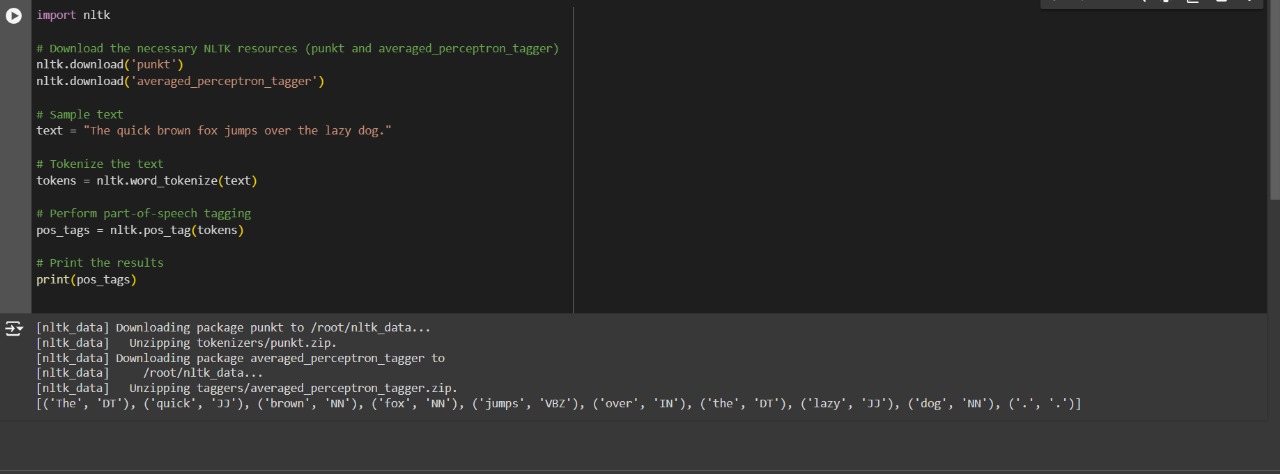


Result:

N-gram model using python program executed successfully.

7.Write program using the NLTK library to perform part-of-speech tagging on a text.

Program:

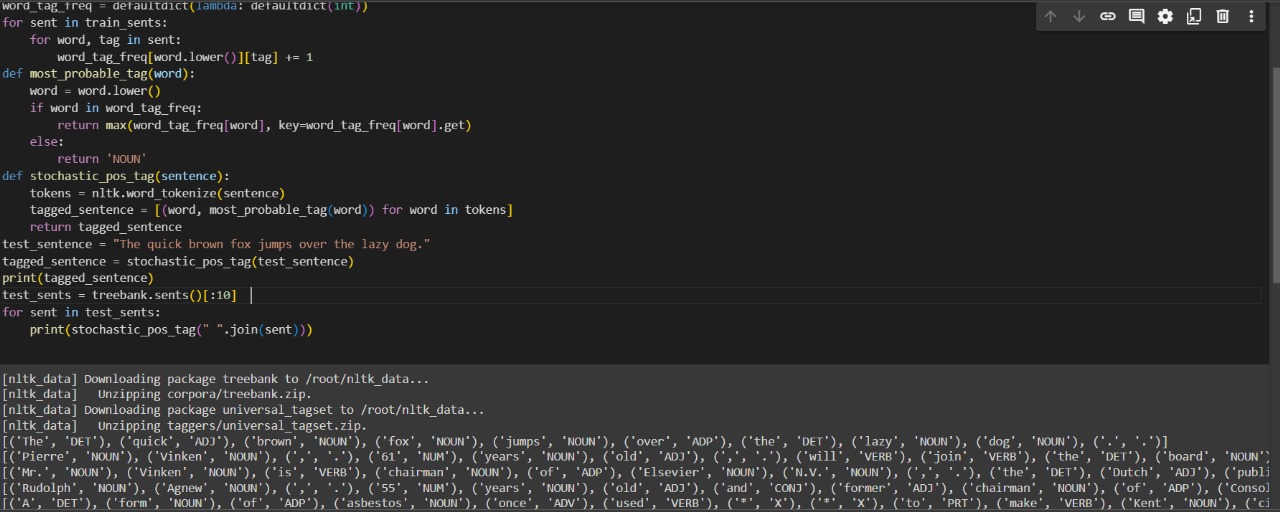


Result:

POS tagging using python program executed successfully.

8.Implement a simple stochastic part-of-speech tagging algorithm using a basic probabilistic model to assign POS tags using python.

Program:

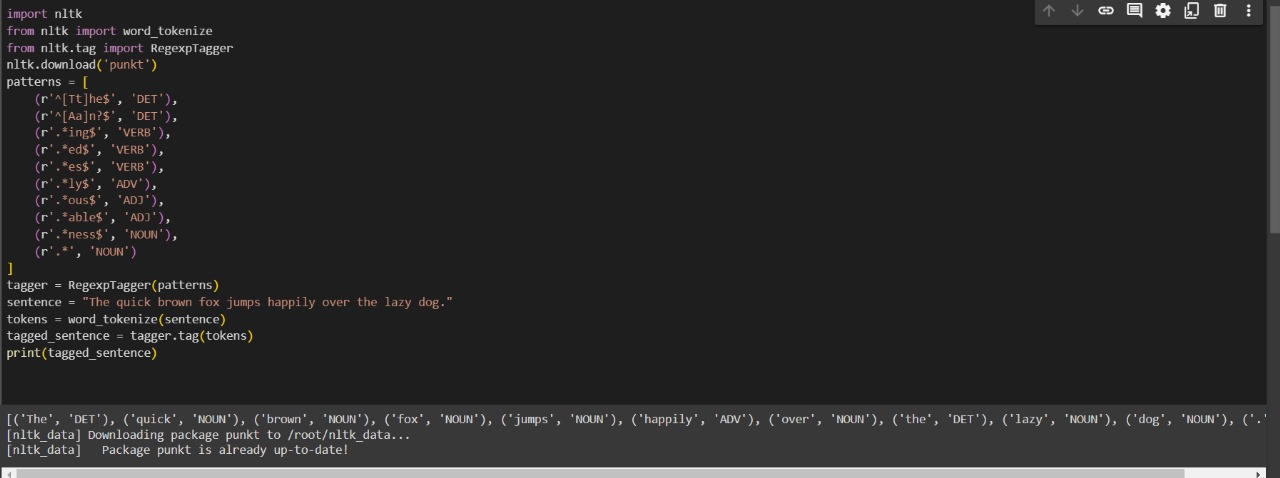


Result:

Stochastic POS tagging using python program executed successfully.

9.Implement a rule-based part-of-speech tagging system using regular expressions using python.

Program:

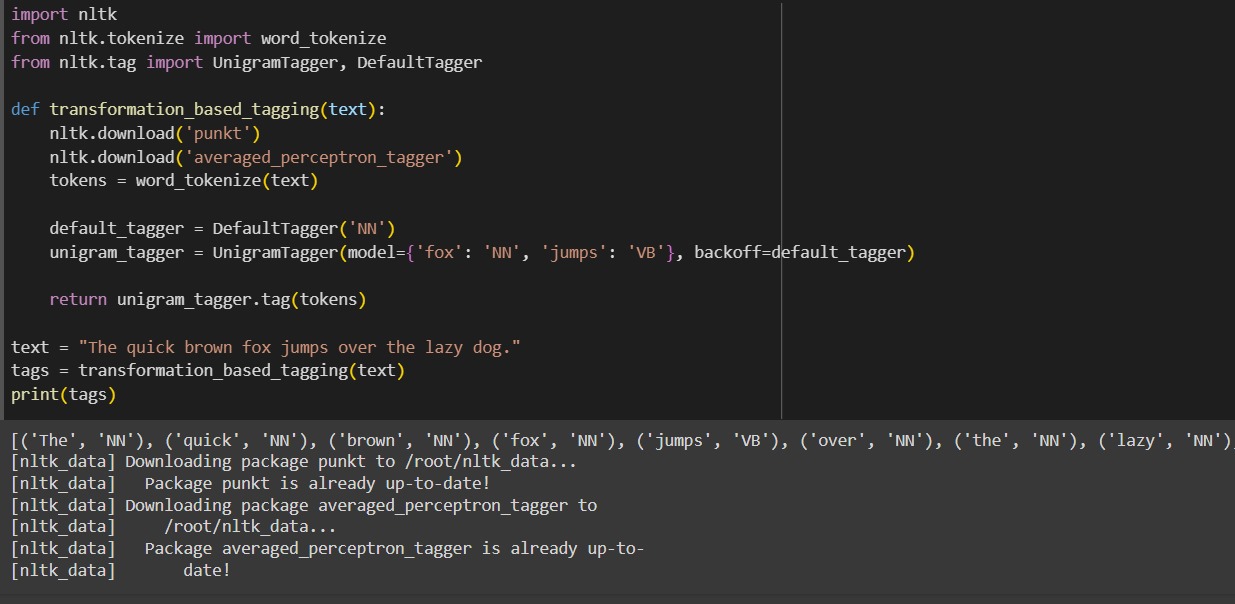


Result:

Rule based POS tagging using python program executed successfully.

10.Implement transformation-based tagging using a set of transformation rules, apply a simple rule to tag words using python.

Program:

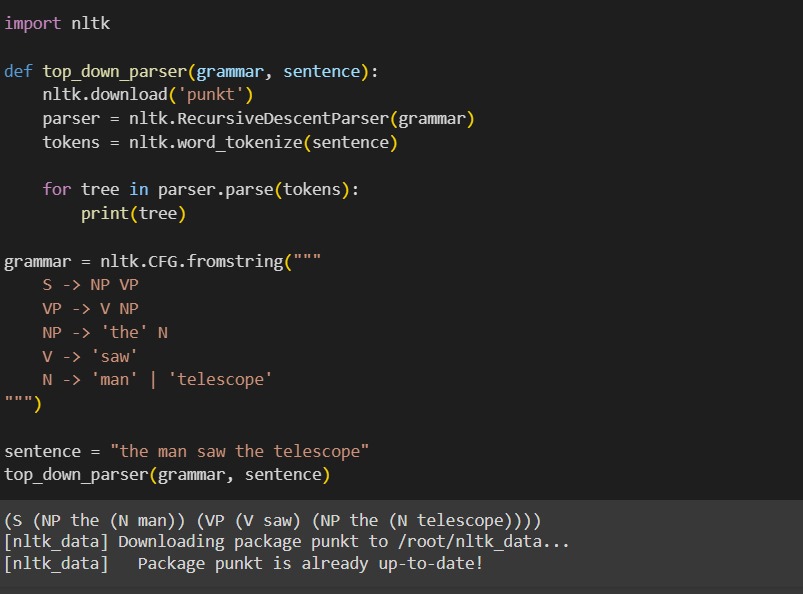


Result:

transformation-based tagging using python program executed successfully.

11.Implement a simple top-down parser for context-free grammars using python.

Program:



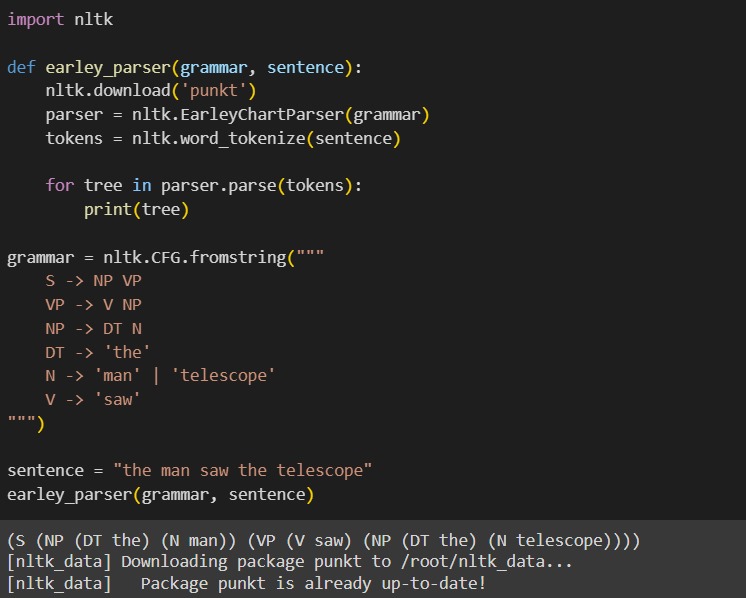
Result:

simple top-down parser for context-free grammars using python

program executed successfully.

12.Implement an Earley parser for context-free grammars using a simple python program

Program:

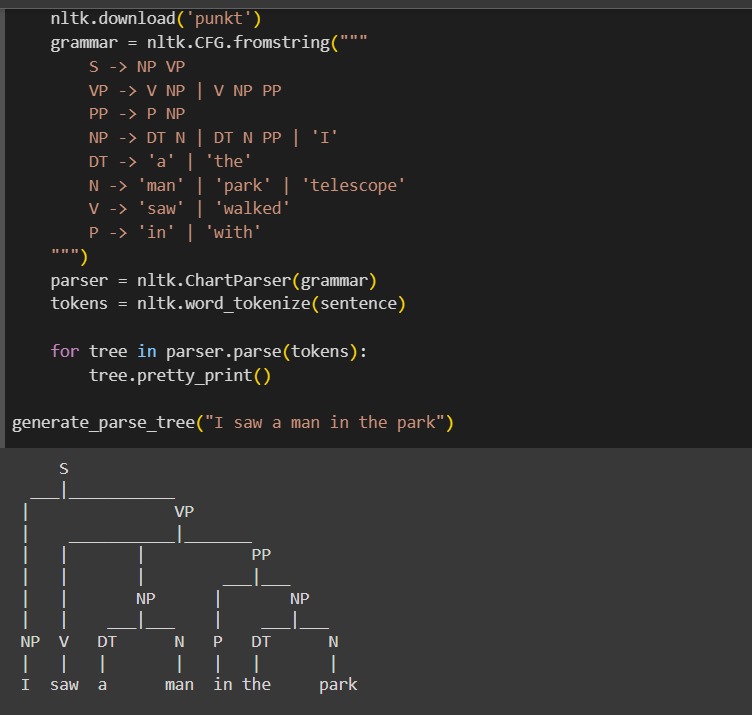


Result:

Earley parser for context-free grammars using python program executed successfully.

13.Generate a parse tree for a given sentence using a context-free grammar using python program.

Program:

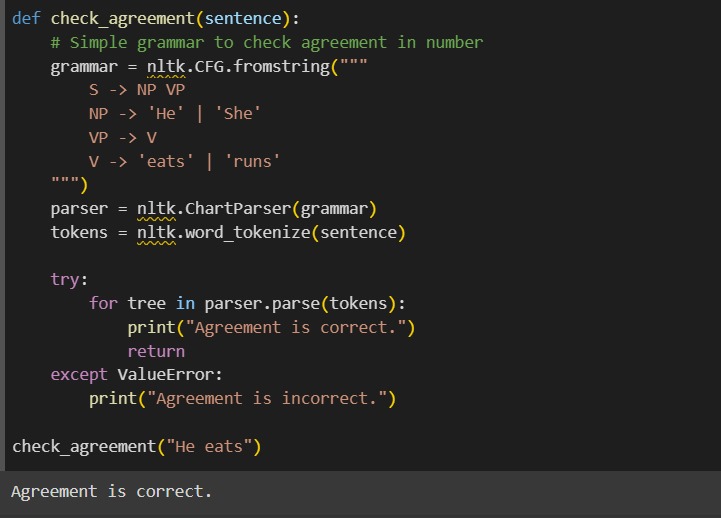


Result:

parser for context-free grammars using python program executed successfully.

14.Create a program in python to check for agreement in sentences based on a context-free grammar's rules.

Program:

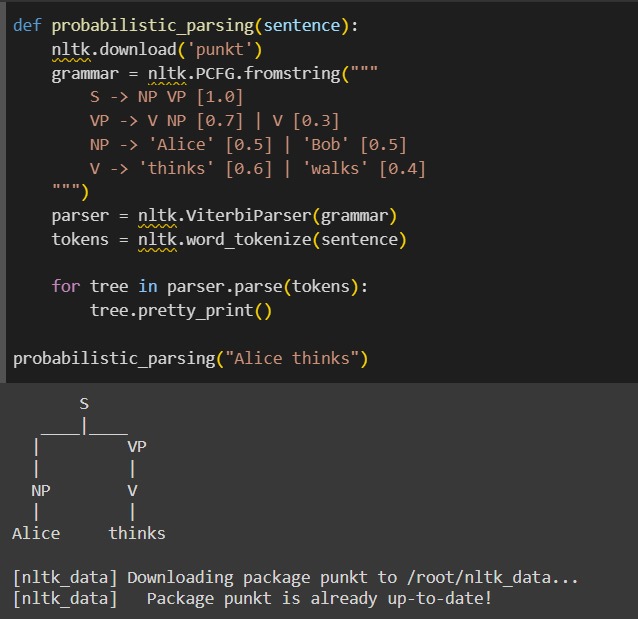


Result:

Agreement in sentence based context-free grammars using python program executed successfully.

15.Implement probabilistic context-free grammar parsing for a sentence using python.

Program:



Result:

probabilistic context-free grammars using python program executed successfully.

16.Implement a Python program using the SpaCy library to perform Named Entity Recognition (NER) on a given text.

Program:



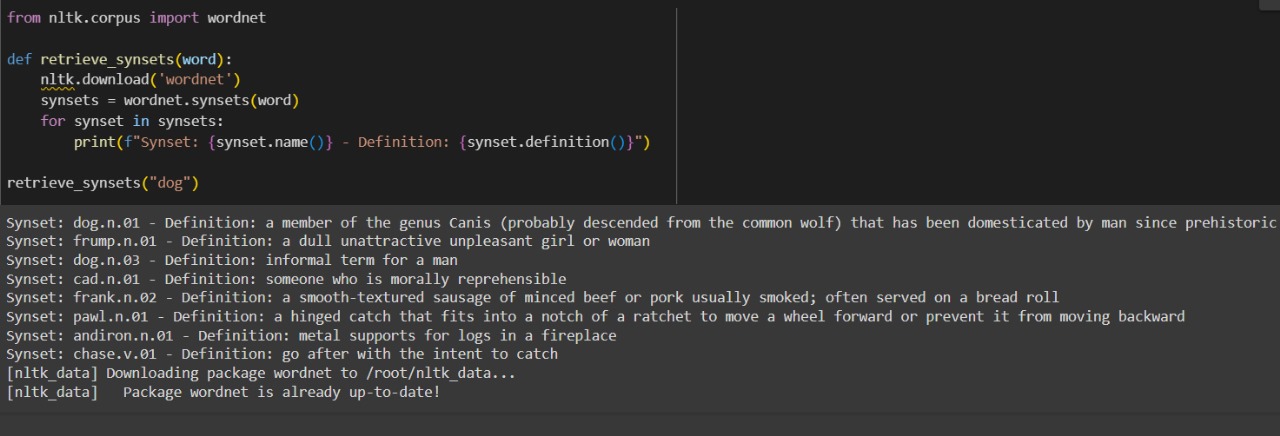
Result:

To perform Named Entity Recognition (NER) on a given text.

using python program executed successfully.

17.Write program demonstrates how to access WordNet, a lexical database, to retrieve synsets and explore word meanings in python.

Program:

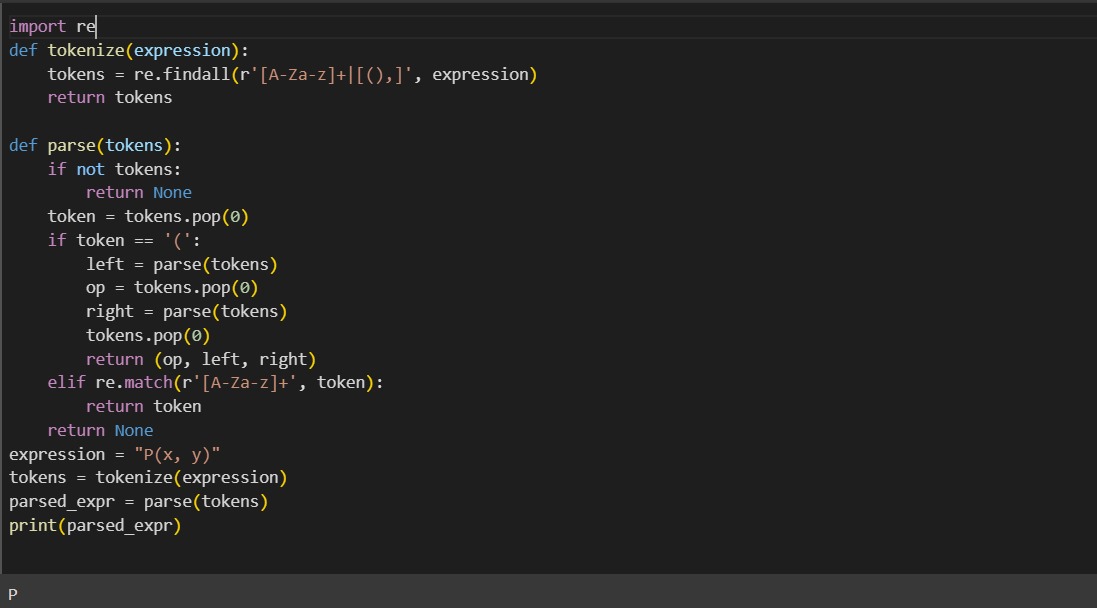


Result:

To access WordNet, a lexical database, to retrieve synsets and explore word meanings python program executed successfully.

18.Implement a simple FOPC parser for basic logical expressions using python program.

Program:

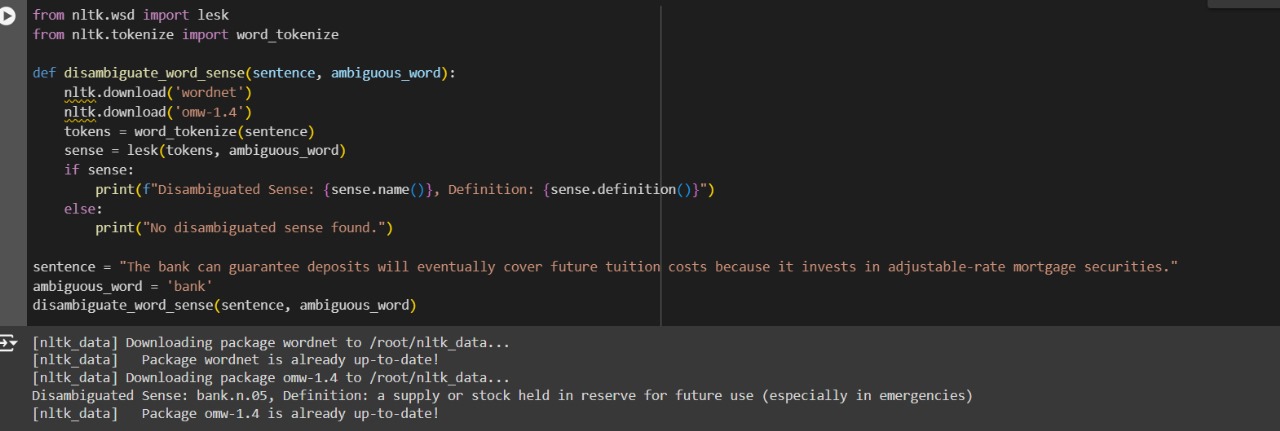


Result:

A simple FOPC parser for basic logical expressions using python program executed successfully.

19.Create a program for word sense disambiguation using the Lesk algorithm using python.

Program:

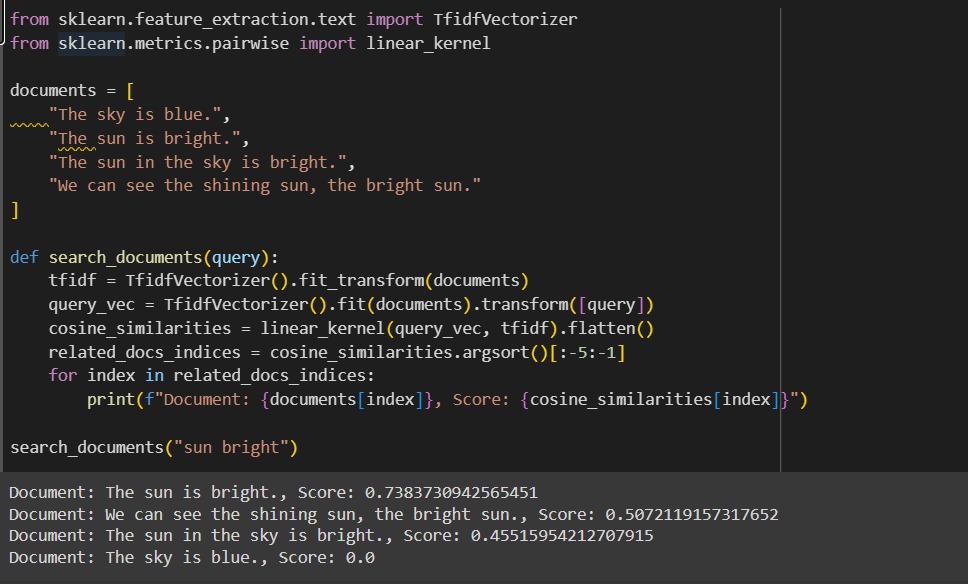


Result:

Word sense disambiguation using the Lesk algorithm using python program executed successfully.

20.Implement a basic information retrieval system using TF-IDF (Term Frequency-Inverse Document Frequency) for document ranking using python.

Program:

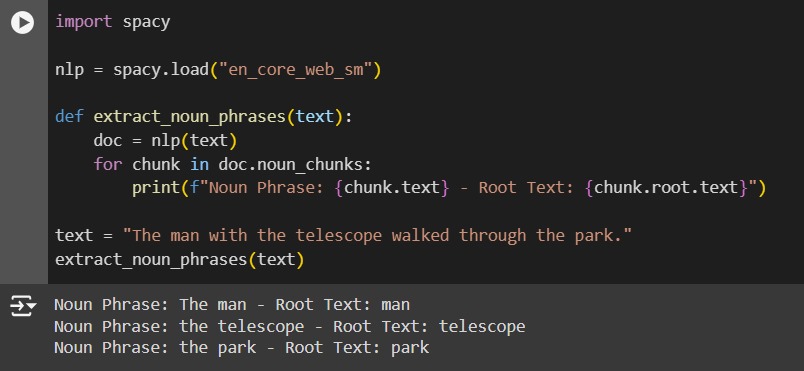


Result:

A basic information retrieval system using TF-IDF using python program executed successfully.

21.Create a python program that performs syntax-driven semantic analysis by extracting noun phrases and their meanings from a sentence.

Program:

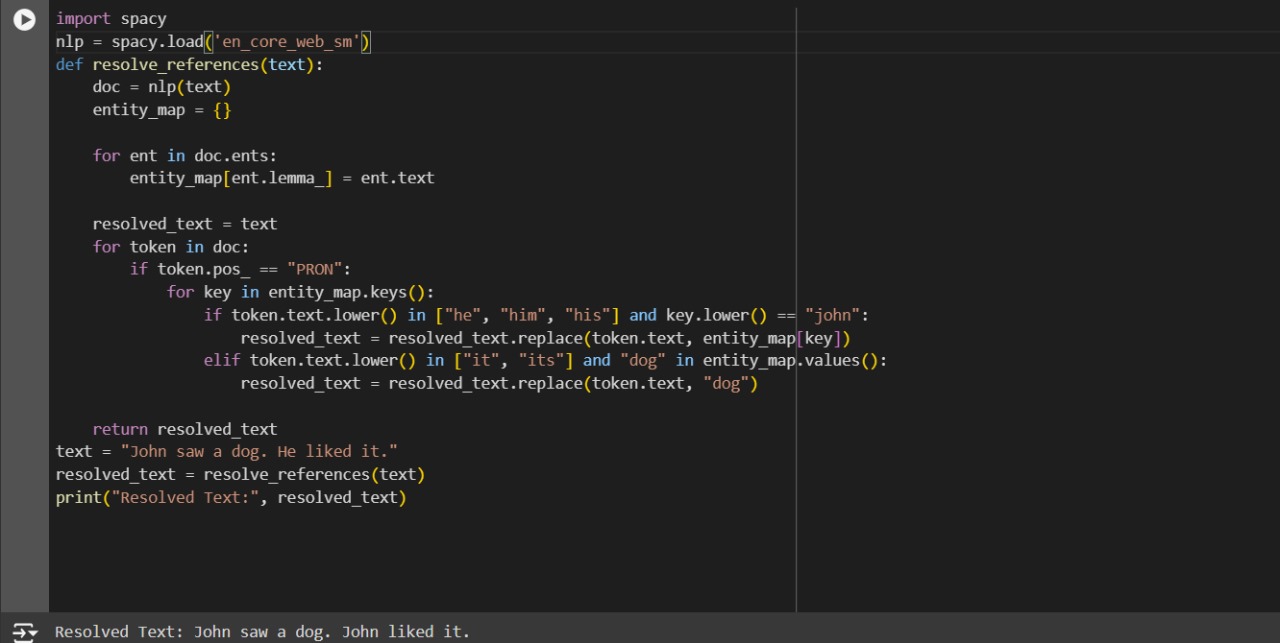


Result:

syntax-driven semantic analysis using python program executed successfully.

22.Create a python program that performs reference resolution within a text.

Program:

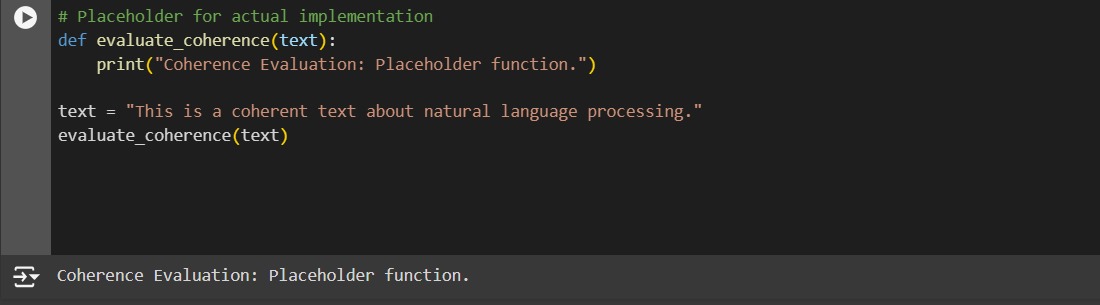


Result:

Reference resolution within a text using python program executed successfully.

23.Develop a python program that evaluates the coherence of a given text

Program:

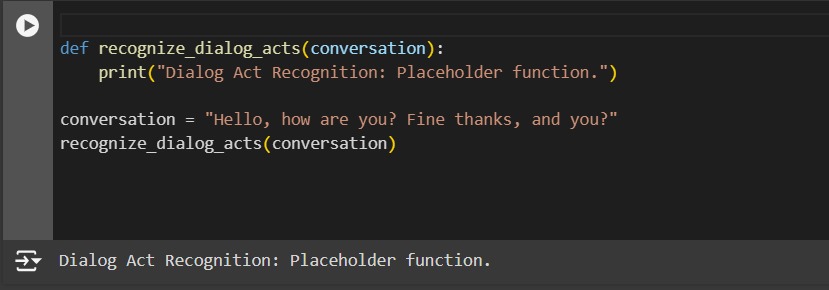


Result:

Evaluates the coherence of a given text using python program executed successfully.

24.Create a python program that recognizes dialog acts in a given dialog or conversation.

Program:



Result:

Recognizes dialog acts in a given dialog or conversation using python program executed successfully.