

Terna Engineering College
Computer Engineering Department
Program: Sem VIII

Course: Natural Language Processing

Experiment No. 8

A.1 Aim: Implement morphological parser to accept and reject given string.

PART B
(PART B: TO BE COMPLETED BY STUDENTS)

Roll No. 50	Name: AMEY THAKUR
Class: BE COMPS B	Batch: B3
Date of Experiment: 01/04/2022	Date of Submission: 01/04/2022
Grade:	

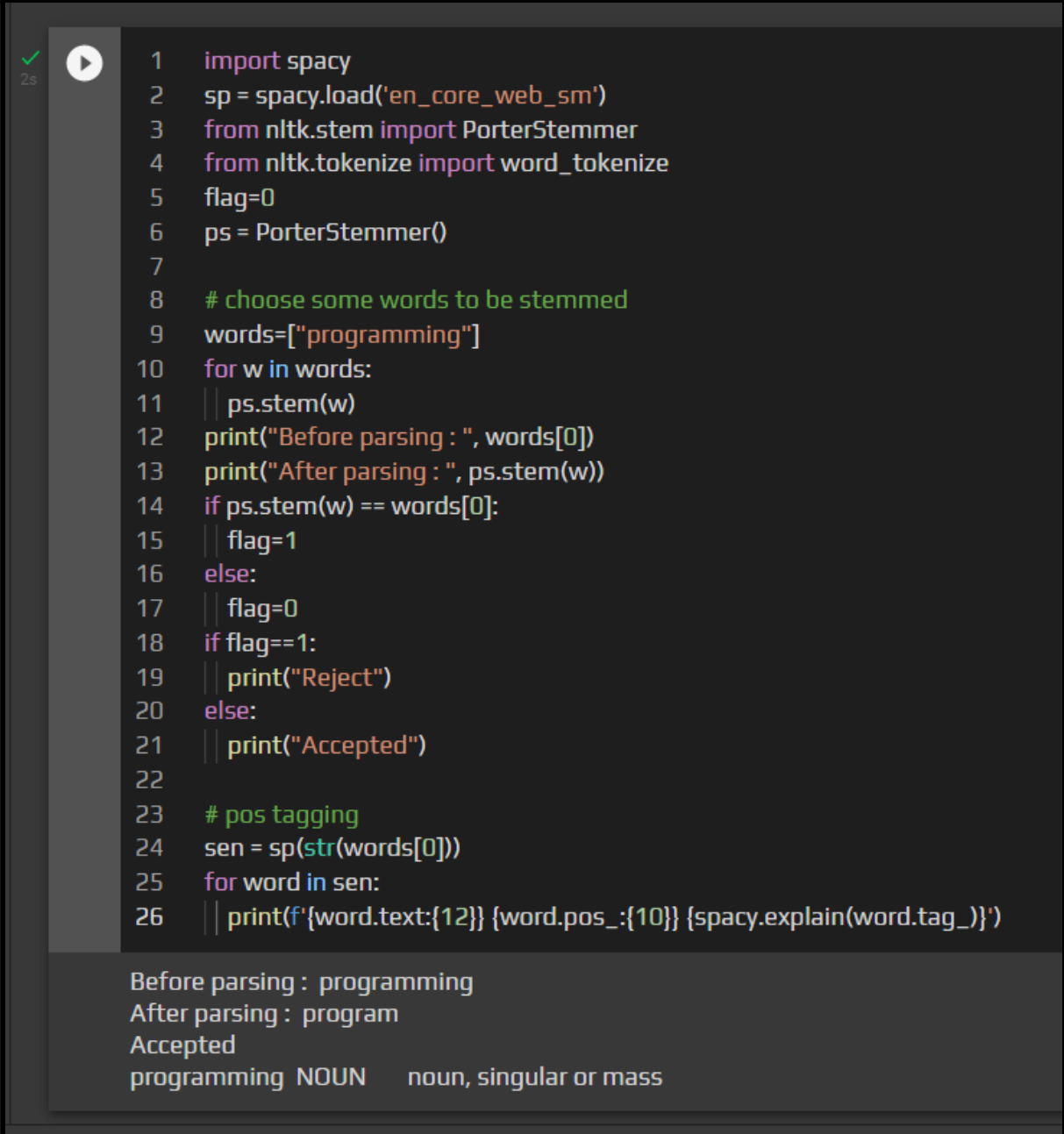
B.1 Software Code written by a student:

```
import spacy
sp = spacy.load('en_core_web_sm')
from nltk.stem import PorterStemmer
from nltk.tokenize import word_tokenize
flag=0
ps = PorterStemmer()

# choose some words to be stemmed
words=["programming"]
for w in words:
    ps.stem(w)
print("Before parsing : ", words[0])
print("After parsing : ", ps.stem(w))
if ps.stem(w) == words[0]:
    flag=1
else:
    flag=0
if flag==1:
    print("Reject")
else:
    print("Accepted")

# pos tagging
sen = sp(str(words[0]))
for word in sen:
    print(f'{word.text:{12}} {word.pos_{10}} {spacy.explain(word.tag_)})')
```

B.2 Input and Output:



```
1 import spacy
2 sp = spacy.load('en_core_web_sm')
3 from nltk.stem import PorterStemmer
4 from nltk.tokenize import word_tokenize
5 flag=0
6 ps = PorterStemmer()
7
8 # choose some words to be stemmed
9 words=["programming"]
10 for w in words:
11     ps.stem(w)
12     print("Before parsing : ", words[0])
13     print("After parsing : ", ps.stem(w))
14     if ps.stem(w) == words[0]:
15         flag=1
16     else:
17         flag=0
18     if flag==1:
19         print("Reject")
20     else:
21         print("Accepted")
22
23 # pos tagging
24 sen = sp(str(words[0]))
25 for word in sen:
26     print(f'{word.text:{12}} {word.pos_:{10}} {spacy.explain(word.tag_)}')
```

Before parsing : programming
After parsing : program
Accepted
programming NOUN noun, singular or mass

B.3 Observations and learning:

- The most sophisticated methods for lemmatization involve complete morphological parsing of the word. Morphology is the study of the way words are built up from smaller meaning-bearing units called morphemes. Two broad classes of morphemes can be distinguished: stems—the central morpheme of the word, supplying the main meaning—and affixes—adding “additional” meanings of various kinds.

- The goal of morphological parsing is to find out what morphemes a given word is built from. For example, a morphological parser should be able to tell us that the word cats is the plural form of the noun stem cat and that the word mice is the plural form of the noun stem mouse.

B.4 Conclusion:

Hence, we understood how to implement a morphological parser to accept and reject given strings using python.

B.5 Question of Curiosity:

Q1: Explain the top-down parser used in NLP applications.

ANS:

- Top-down Parsing is a parsing technique that first looks at the highest level of the parse tree and works down the parse tree by using the rules of grammar.
- Top-down parsing attempts to find the leftmost derivations for an input string.
- In this parsing technique, we start parsing from the top (start symbol of parse tree) to down (the leaf node of parse tree) in a top-down manner.
- This parsing technique uses Left Most Derivation.
- Its main decision is to select what production rule to use in order to construct the string.