## Stemming is the process of reducing a word to its word stem that affixes to suffixes and prefixes or to the roots of words known as a lemma.

```
from nltk.stem import PorterStemmer

stemming = PorterStemmer()

words=["eating","eats","eaten","writing","writes","programming","programs","history","finally","finalized"]


for word in words:
    print(word+"---->"+stemming.stem(word))
```

```
 eating---->eat
 eats---->eat
 eaten---->eaten
 writing---->write
 writes---->write
 programming---->program
 programs---->program
 history---->histori
 finally---->final
 finalized---->final
```

```
# some example of word stem

stemming.stem('congratulations')
```

```
 'congratul'
```

```
stemming.stem('sitting')
```

```
 'sit'
```

```
from nltk.stem import LancasterStemmer

lancaster=LancasterStemmer()

for word in words:
    print(word+"---->"+lancaster.stem(word))
```

```
 eating---->eat
 eats---->eat
 eaten---->eat
 writing---->writ
 writes---->writ
 programming---->program
 programs---->program
 history---->hist
 finally---->fin
 finalized---->fin
```

## RegexpStemmer class

NLTK has RegexpStemmer class with the help of which we can easily implement Regular Expression Stemmer algorithms. It basically takes a single regular expression and removes any prefix or suffix that matches the expression.

```
from nltk.stem import RegexpStemmer

reg_stemmer=RegexpStemmer('ing|s$|e$|able$')

reg_stemmer.stem("eating")
```

```
'eat'
```

```
reg_stemmer.stem("ingplaying")
```

```
'play'
```

## Snowball Stemmer

```
from nltk.stem import SnowballStemmer

snowballstemmer=SnowballStemmer('english',ignore_stopwords=False)

for word in words:
    print(word+"---->"+snowballstemmer.stem(word))
```

```
eating---->eat
eats---->eat
eaten---->eaten
writing---->write
writes---->write
programming---->program
programs---->program
history---->histori
finally---->final
finalized---->final
```

```
# snowball stemmer is better perform than stemming

stemming.stem("fairly"),stemming.stem("sportingly")
```

```
('fairli', 'sportingli')
```

```
snowballstemmer.stem("fairly"),snowballstemmer.stem("sportingly")
```

```
('fair', 'sport')
```

## Wordnet Lemmatizer

Lemmatization technique is like stemming. The output we will get after lemmatization is called 'lemma', which is a root word rather than root stem, the output of stemming. After lemmatization, we will be getting a valid word that means the same thing.

```
import nltk
nltk.download('wordnet')
from nltk.stem import WordNetLemmatizer
lemmatizer = WordNetLemmatizer()
```

```
[nltk_data] Downloading package wordnet to /root/nltk_data...
[nltk_data]   Package wordnet is already up-to-date!
```

```
for word in words:
    print(word+"---->"+lemmatizer.lemmatize(word,pos='a'))
```

```
eating---->eating
eats---->eats
eaten---->eaten
```

```
writing---->writing
writes---->writes
programming---->programming
programs---->programs
history---->history
finally---->finally
finalized---->finalized
```

```
for word in words:
    print(word+"---->"+lemmatizer.lemmatize(word,pos='v'))
```

```
eating---->eat
eats---->eat
eaten---->eat
writing---->write
writes---->write
programming---->program
programs---->program
history---->history
finally---->finally
finalized---->finalize
```

```
'''
POS- Noun-n
verb-v
adjective-a
adverb-r
'''
```

```
'\nPOS- Noun-n\nverb-v\nadjective-a\nadverb-r\n'
```

```
lemmatizer.lemmatize("playing",pos='v')
```

```
'play'
```

```
## for Sentiment Analysis we use stemming
## for Chatbot---lemmatization
```