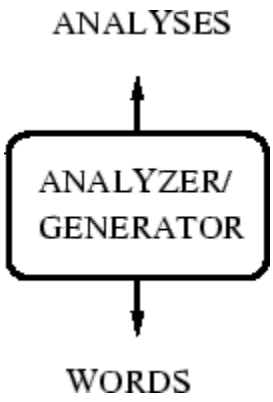
EXPERIMENT 5 :: EXPLAIN ABOUT WORD GENERATION

INTRODUCTION ::

A word can be simple or complex. For example, the word 'cat' is simple because one cannot further decompose the word into smaller part. On the other hand, the word 'cats' is complex, because the word is made up of two parts: root 'cat' and plural suffix '-s'



PROGRAM ::

```
# ---------- word Generation  ----------#

import nltk
import random
nltk.download('punkt')
nltk.download('reuters')

from nltk.corpus import reuters

def build_word_model(corpus,order=2):
    word_model = {}
    for sc in corpus:
        words = nltk.word_tokenize(sc.lower())
```

```
        for i in range(len(words)-order):
            cur_state,next = tuple(words[i:i+order]),words[i + order]
            if cur_state not in word_model:
                word_model[cur_state] = []
                word_model[cur_state].append(next)
            return word_model

def generate_words(word_model,seed,length=10):
    cur_state = tuple(seed)
    generated_words = list(cur_state)

    for _ in range(length):
        if cur_state in word_model:
            next = random.choice(word_model[cur_state])
            generated_words.append(next)
            cur_state = tuple(generated_words[-len(seed):])
        else:
            break
        return "".join(generated_words)

if __name__=="__main__":
    corpus = reuters.sents()
    word_model = build_word_model(["".join(sent) for sent in corpus])

    seed = ["Natural","Processing","Language","Artificial","Deep","Learning","Welcome"]
    generated_text = generate_words(word_model,seed,length=20)
    print("Generated Text:")
    print(generated_text)
```

```
print("\n")



# ---------- another process word Generation  ----------#

import random
corpus = ["Natural","Processing","Language","Artificial","Deep","Learning","Welcome"]
r_words = random.choice(corpus)
print(r_words)
```

OUTPUT ::

```
Generated Text:
None


Learning
```