In [1]:

```
from nltk import word_tokenize
```

In [2]:

```
def preprocess(d):
    d=d.lower()
    d="eos "+ d
    d=d.replace("."," eos")
    return d
def generate_tokens(d):
    tokens = word_tokenize(d)
    return tokens
def generate_tokens_freq(tokens):
    dct={}
    for i in tokens:
        dct[i]=0
    for i in tokens:
        dct[i]+=1
    return dct
```

In [3]:

```
def generate_ngrams(tokens,k):
    l=[]
    i=0
    while(i<len(tokens)):
        l.append(tokens[i:i+k])
        i=i+1
    l=l[:-1]
    return l
def generate_ngram_freq(bigram):
    dct1={}
    for i in bigram:
        st=" ".join(i)
        dct1[st]=0
    for i in bigram:
        st=" ".join(i)
        dct1[st]+=1
    return dct1
```

In [4]:

```
def find1(s,dct1):
    try:
        return dct1[s]
    except:
        return 0
def print_probability_table(distinct_tokens,dct,dct1):
    n=len(distinct_tokens)
    l=[[]*n for i in range(n)]
    for i in range(n):
        denominator = dct[distinct_tokens[i]]
        for j in range(n):
            numerator = find1(distinct_tokens[i]+" "+distinct_tokens[j],dct1)
            l[i].append(float("{:.3f}".format(numerator/denominator)))
    return l
```

In [ ]:

```
"The cat I saw sat on a mat was sad"
"The cat sat on a mat was sad"
```

Input

In [5]:
```
d=input("Enter corpus = ")
print("\n"+'\033[1m'+"Given Corpus"+'\033[0m')
print(d)
```

```
Enter corpus = The cat sat on a mat was sad

Given Corpus
The cat sat on a mat was sad
```

Preprocess

In [6]:
```
d=preprocess(d)
print("\n"+'\033[1m'+"Preprocessing"+'\033[0m')
print(d)
```

```
Preprocessing
eos the cat sat on a mat was sad
```

Generate Tokens

In [7]:
```
tokens=generate_tokens(d)
print("\n"+'\033[1m'+"Generate Tokens"+'\033[0m')
print(tokens)
```

```
Generate Tokens
['eos', 'the', 'cat', 'sat', 'on', 'a', 'mat', 'was', 'sad']
```

Generate Tokens Frequency

In [8]:
```
distinct_tokens = list(set(sorted(tokens)))
dct=generate_tokens_freq(tokens)
print("\n"+'\033[1m'+"Generate Frequency of Tokens"+'\033[0m')
print(dct)
```

Generate Frequency of Tokens
{'eos': 1, 'the': 1, 'cat': 1, 'sat': 1, 'on': 1, 'a': 1, 'mat': 1, 'was': 1, 'sad': 1}

Generate bigram

In [9]:
```
bigram = generate_ngrams(tokens,2)
print("\n"+'\033[1m'+"Generate bigrams"+'\033[0m')
for i in bigram:
    print("'{}'".format(' '.join(i)), end=", ")
```

Generate bigrams
'eos the', 'the cat', 'cat sat', 'sat on', 'on a', 'a mat', 'mat was', 'was sad',

Generate bigram frequency

In [10]:
```
dct1=generate_ngram_freq(bigram)
print("\n\n"+'\033[1m'+"Generate Frequency of bigrams"+'\033[0m')
print(dct1)
```

Generate Frequency of bigrams
{'eos the': 1, 'the cat': 1, 'cat sat': 1, 'sat on': 1, 'on a': 1, 'a mat': 1, 'mat was': 1, 'was sad': 1}

Probability table

In [11]:
```
probability_table=print_probability_table(distinct_tokens,dct,dct1)
print("\n"+'\033[1m'+"Probability table"+'\033[0m'+"\n")
```

Probability table

In [12]:
```
n=len(distinct_tokens)
print("\t"+'\033[1m', end="")
for i in range(n):
    print(distinct_tokens[i],end="\t")
print('\033[0m'+"\n")

for i in range(n):
    print('\033[1m',distinct_tokens[i],'\033[0m',end="\t")
    for j in range(n):
        print(probability_table[i][j],end="\t")
    print("")
```

	a	mat	sad	the	was	cat	sat	on	eos
a	0.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
mat	0.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0
sad	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
the	0.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0
was	0.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0
cat	0.0	0.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0
sat	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.0	0.0
on	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
eos	0.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0

Testing

In [ ]:

```
"The cat sat on a mat"
"The cat sad on a mat"
```

In [13]:

```
text = input("\nEnter text to check its probability = ")
print("\n"+'\033[1m'+"Given Text"+'\033[0m')
print(text)
```

```
Enter text to check its probability = The cat sad on a mat

Given Text
The cat sad on a mat
```

In [14]:

```
p = preprocess(text)
print("\n"+'\033[1m'+"Preprocessing"+'\033[0m')
print(p)

t=generate_tokens(p)
print("\n"+'\033[1m'+"Generate Tokens"+'\033[0m')
print(t)

n = generate_ngrams(t,2)
print("\n"+'\033[1m'+"Generate bigrams"+'\033[0m')
for i in n:
    print("'{}'".format(' '.join(i)), end=", ")

```

```
Preprocessing
eos the cat sad on a mat

Generate Tokens
['eos', 'the', 'cat', 'sad', 'on', 'a', 'mat']

Generate bigrams
'eos the', 'the cat', 'cat sad', 'sad on', 'on a', 'a mat',
```

In [15]:

```
print("\n\n"+'\033[1m'+"Calculate bigram probability"+'\033[0m')
s=1
dct2={}
for i in n:
    dct2[" ".join(i)]=0

for i in n:
    k=distinct_tokens.index(i[0])
    m=distinct_tokens.index(i[1])

    dct2[" ".join(i)]=probability_table[k][m]

    print("P('{}')\t=  ".format(' '.join(i)),probability_table[k][m])
    s*=probability_table[k][m]
```

Calculate bigram probability

```
P('eos the')    =   1.0
P('the cat')    =   1.0
P('cat sad')    =   0.0
P('sad on')     =   0.0
P('on a')       =   1.0
P('a mat')      =   1.0
```

In [16]:

```
print("\n"+'\033[1m'+ "Calculate Probability of the sentence"+'\033[0m')

print(f"P('{text}') \n= ",end="")

x=dct2.popitem()

for i in dct2:
    print(f"P('{i}')", end=" * ")

print(f"P('{x[0]}')\n= ", end='')

for i in dct2:
    print(dct2[i], end=" * ")

print(x[1],"\n=",s)

print("\n"+'\033[1m'+f"Probability('{text}') = "+"{:.5f}".format(s))
```

Calculate Probability of the sentence
P('The cat sad on a mat')
= P('eos the') * P('the cat') * P('cat sad') * P('sad on') * P('on a') * P('a mat')
= 1.0 * 1.0 * 0.0 * 0.0 * 1.0 * 1.0
= 0.0

Probability('The cat sad on a mat') = 0.00000

In [ ]:

```
1
```