In [5]:

```python
def readData():
    data = ['This is a dog','This is a cat','I love my cat','This is my name ']
    dat=[]
    for i in range(len(data)):
        for word in data[i].split():
            dat.append(word)
    print(dat)
    return dat
```

In [2]:

```python
def createBigram(data):
    listOfBigrams = []
    bigramCounts = {}
    unigramCounts = {}
    for i in range(len(data)-1):
        if i < len(data) - 1 and data[i+1].islower():
            listOfBigrams.append((data[i], data[i + 1]))
            if (data[i], data[i+1]) in bigramCounts:
                bigramCounts[(data[i], data[i + 1])] += 1
            else:
                bigramCounts[(data[i], data[i + 1])] = 1
        if data[i] in unigramCounts:
            unigramCounts[data[i]] += 1
        else:
            unigramCounts[data[i]] = 1
    return listOfBigrams, unigramCounts, bigramCounts
```

In [3]:

```python
def calcBigramProb(listOfBigrams, unigramCounts, bigramCounts):
    listOfProb = {}
    for bigram in listOfBigrams:
        word1 = bigram[0]
        word2 = bigram[1]
        listOfProb[bigram] = (bigramCounts.get(bigram))/(unigramCounts.get(word1))
    return listOfProb
```

In [6]:

```python
if __name__ == '__main__':
    data = readData()
    listOfBigrams, unigramCounts, bigramCounts = createBigram(data)
    print("\n All the possible Bigrams are ")
    print(listOfBigrams)
    print("\n Bigrams along with their frequency ")
    print(bigramCounts)
    print("\n Unigrams along with their frequency ")
    print(unigramCounts)
    bigramProb = calcBigramProb(listOfBigrams, unigramCounts, bigramCounts)
    print("\n Bigrams along with their probability ")
    print(bigramProb)
    inputList="This is my cat"
    splt=inputList.split()
    outputProb1 = 1
    bilist=[]
    bigrm=[]
    for i in range(len(splt) - 1):
        if i < len(splt) - 1:
            bilist.append((splt[i], splt[i + 1]))
    print("\n The bigrams in given sentence are ")
    print(bilist)
    for i in range(len(bilist)):
        if bilist[i] in bigramProb:
            outputProb1 *= bigramProb[bilist[i]]
        else:
            outputProb1 *= 0
    print('\n' + 'Probablility of sentence \"This is my cat\" = ' + str(outputProb1
```

```
['This', 'is', 'a', 'dog', 'This', 'is', 'a', 'cat', 'I', 'love', 'my', 'cat', 'Th
is', 'is', 'my', 'name']

 All the possible Bigrams are 
[('This', 'is'), ('is', 'a'), ('a', 'dog'), ('This', 'is'), ('is', 'a'), ('a', 'ca
t'), ('I', 'love'), ('love', 'my'), ('my', 'cat'), ('This', 'is'), ('is', 'my'),
('my', 'name')]

 Bigrams along with their frequency 
{('This', 'is'): 3, ('is', 'a'): 2, ('a', 'dog'): 1, ('a', 'cat'): 1, ('I', 'lov
e'): 1, ('love', 'my'): 1, ('my', 'cat'): 1, ('is', 'my'): 1, ('my', 'name'): 1}

 Unigrams along with their frequency 
{'This': 3, 'is': 3, 'a': 2, 'dog': 1, 'cat': 2, 'I': 1, 'love': 1, 'my': 2}

 Bigrams along with their probability 
{('This', 'is'): 1.0, ('is', 'a'): 0.6666666666666666, ('a', 'dog'): 0.5, ('a', 'c
at'): 0.5, ('I', 'love'): 1.0, ('love', 'my'): 1.0, ('my', 'cat'): 0.5, ('is', 'm
y'): 0.3333333333333333, ('my', 'name'): 0.5}

 The bigrams in given sentence are 
[('This', 'is'), ('is', 'my'), ('my', 'cat')]

Probablility of sentence "This is my cat" = 0.16666666666666666
```

In [ ]: