Text Analysis Using nltk

In [2]:
```
from nltk.tokenize import word_tokenize
from nltk.text import Text
```

In [3]:
```
my_string = "Two plus two is four, minus one that's three – quick maths. Every day man's on the block. Smoke trees. See your girl in the park, that girl is an uckers. When the thing went quack quack quack, your men were ducking! Hold tight Asznee, my brother. He's got a pumpy. Hold tight my man, my guy. He's got a frisbee. I trap, trap, trap on the phone. Moving that cornflakes, rice crispies. Hold tight my girl Whitney."
tokens = word_tokenize(my_string)
tokens = [word.lower() for word in tokens]
tokens[:5]
```

Out[3]:
```
['two', 'plus', 'two', 'is', 'four']
```

In [4]:
```
t = Text(tokens)
t
```

Out[4]:
```
<Text: two plus two is four , minus one...>
```

This method of converting raw strings to NLTK `Text` instances can be used when reading text from a file. For instance:

```
f = open('my-file.txt','rU') # Opening a file with the mode 'U' or 'rU' will open a file for reading in universal newline mode. All three line ending conventions will be translated to a "\n"
raw = f.read()
```

In [5]:
```
t.concordance('uckers') # concordance() is a method of the Text class of NLTK. It finds words and displays a context window. Word matching is not case-sensitive.
# concordance() is defined as follows: concordance(self, word, width=79, lines=25). Note default values for optional params.
```

```
Displaying 1 of 1 matches:
 girl in the park , that girl is an uckers . when the thing went quack quack q
```

In [6]:
```
t.collocations() # def collocations(self, num=20, window_size=2). num is the max no. of collocations to print.
```

```
hold tight; quack quack
```

In [7]:
```
t.count('quack')
```

Out[7]:
```
3
```

In [8]:
```
t.index('two')
```

Out[8]:
```
0
```

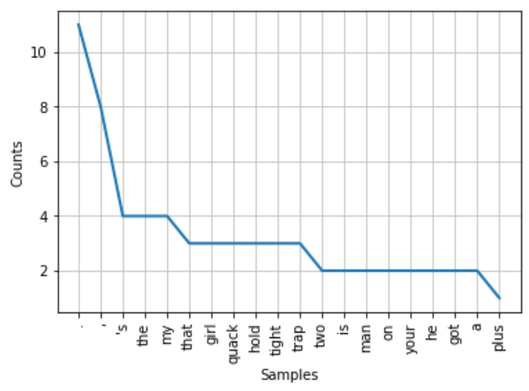
In [9]:
```
t.similar('brother') # similar(self, word, num=20). Distributional similarity: find other words which appear in the same contexts as the specified word; list most similar words first.
```

```
guy
```

In [10]:
```
t.dispersion_plot(['man', 'thing', 'quack']) # Reveals patterns in word positions. Each stripe represents an instance of a word, and each row represents the entire text.
```

```
<Figure size 640x480 with 1 Axes>
```

In [11]:
```
t.plot(20) # plots 20 most common tokens
```



In [12]:
```
t.vocab()
```

Out[12]:
```
FreqDist({'!': 1,
          "'s": 4,
          ',': 8,
          '.': 11,
          'a': 2,
          'an': 1,
          'asznee': 1,
          'block': 1,
          'brother': 1,
          'cornflakes': 1,
          'crispies': 1,
          'day': 1,
          'ducking': 1,
          'every': 1,
          'four': 1,
          'frisbee': 1,
          'girl': 3,
          'got': 2,
          'guy': 1,
          'he': 2,
          'hold': 3,
          'i': 1,
          'in': 1,
          'is': 2,
          'man': 2,
          'maths': 1,
          'men': 1,
          'minus': 1,
          'moving': 1,
          'my': 4,
          'on': 2,
          'one': 1,
          'park': 1,
          'phone': 1,
          'plus': 1,
          'pumpy': 1,
          'quack': 3,
          'quick': 1,
          'rice': 1,
          'see': 1,
          'smoke': 1,
          'that': 3,
          'the': 4,
          'thing': 1,
          'three': 1,
          'tight': 3,
          'trap': 3,
          'trees': 1,
          'two': 2,
          'uckers': 1,
          'went': 1,
          'were': 1,
          'when': 1,
          'whitney': 1,
          'your': 2,
          '–': 1})
```

Another thing that might be useful in analysis is finding common contexts. Our text is too small so we will use a bigger one.

NLTK comes with several interesting **corpora**, which are large collections of text. You can check out what kinds of corpora are found in `nltk.corpus` in Section 1 here.

`reuters` is a corpus of news documents. More specifically, `reuters` is a *corpus reader* for the Reuters corpus which provides us with methods to access the corpus:

In [15]:
```
from nltk.corpus import reuters
text = Text(reuters.words()) # .words() is one method corpus readers provide for reading data from a corpus. We will learn more about these methods in Chapter 2.
text.common_contexts(['August', 'June']) # It seems that .common_contexts() takes 2 words which are used similarly and displays where they are used similarly. It also seems that '_' indicates where the words would be in the text.
```

```
in_1986 and_and by_. begins_1 paid_1986 in_, early_. or_, /_shipment
of_. last_. for_shipment for_to on_3 on_12 on_19 last_when in_.
in_1987 -_.
```

We will further explore the Reuters corpus as well as several others in later chapters.