EXERCISE-3: Bigram Frequencies of Jane Austen Novels

In [ ]:
```
NAME:Swetha.K
ROLL NO:235229143
```

In [1]:
```
import nltk
```

In [2]:
```
nltk.data.path
```

Out[2]:
```
['C:\\Users\\1mscds43/nltk_data',
 'C:\\ProgramData\\Anaconda3\\nltk_data',
 'C:\\ProgramData\\Anaconda3\\share\\nltk_data',
 'C:\\ProgramData\\Anaconda3\\lib\\nltk_data',
 'C:\\Users\\1mscds43\\AppData\\Roaming\\nltk_data',
 'C:\\nltk_data',
 'D:\\nltk_data',
 'E:\\nltk_data']
```

In [3]:
```
nltk.data.path.append('C:\\ProgramData\\Anaconda3\\nltk_data')
nltk.download('punkt')
```

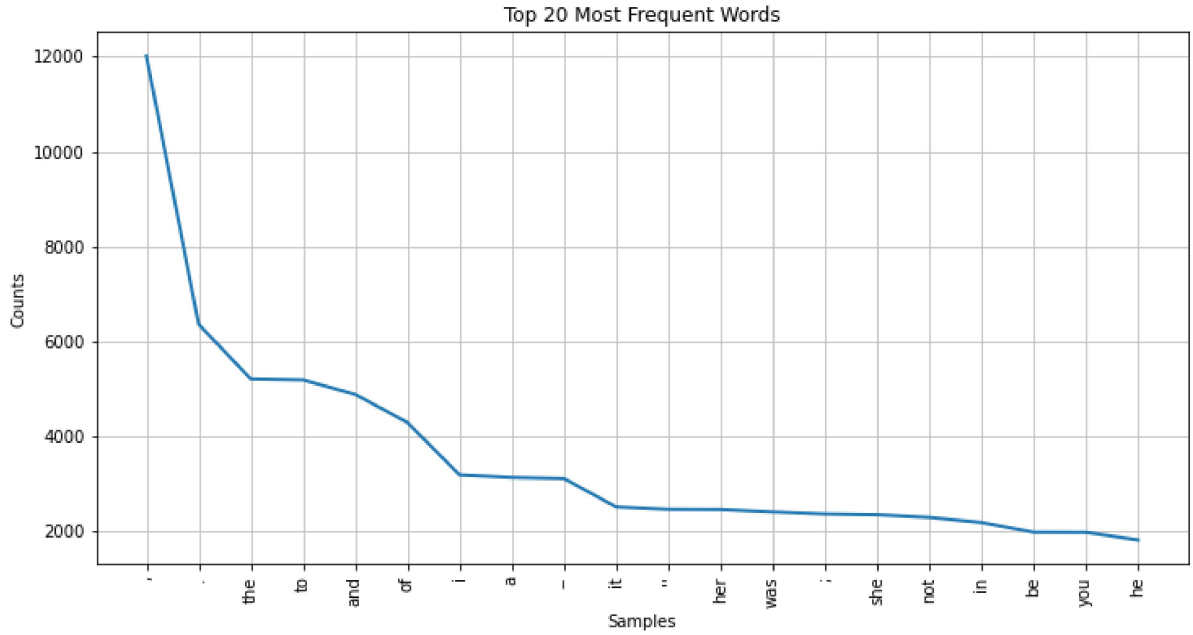
```
[nltk_data] Downloading package punkt to
[nltk_data]     C:\Users\1mscds43\AppData\Roaming\nltk_data...
[nltk_data]   Package punkt is already up-to-date!
```

Out[3]:
```
True
```

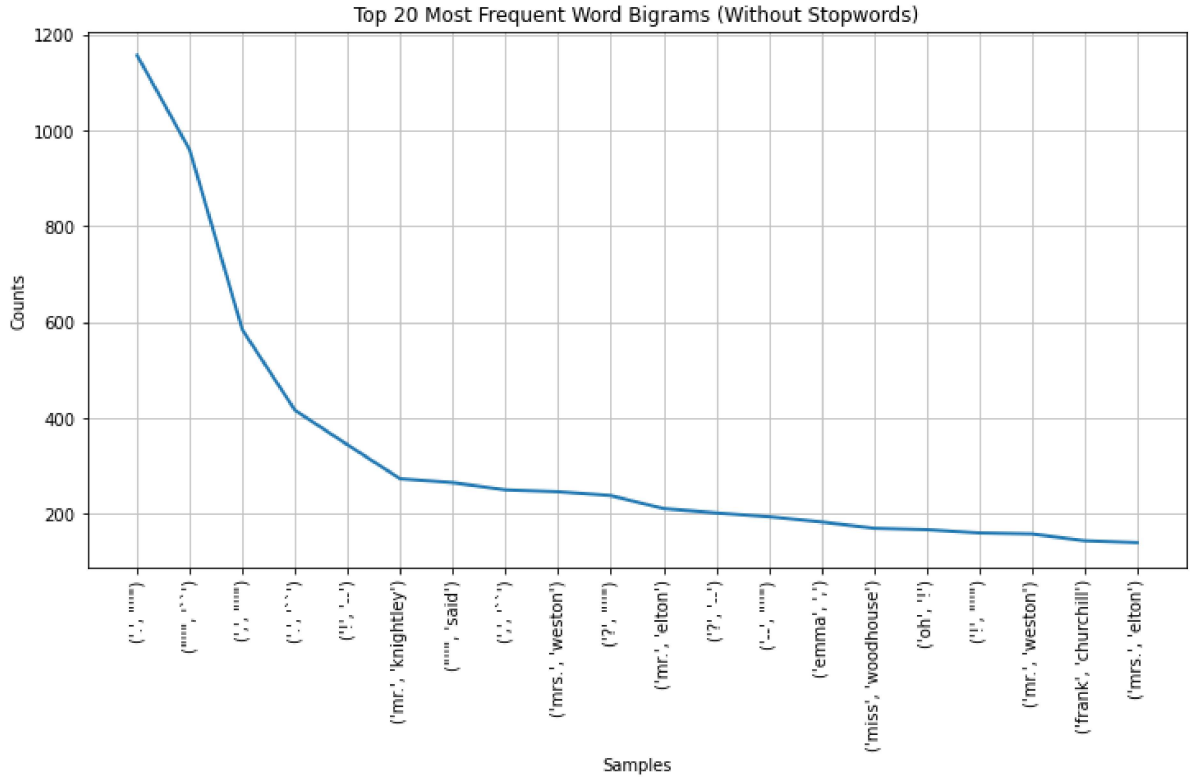
In [4]:
```
import nltk
import matplotlib.pyplot as plt
import pickle
from nltk import FreqDist
from nltk.corpus import stopwords
from nltk.util import bigrams
from nltk.tokenize import word_tokenize

# A. Imports necessary modules
nltk.download('punkt')
nltk.download('stopwords')

# B. Opens the text files and reads in the content as text strings
with open('austen-emma.txt', 'r', encoding='utf-8') as file:
    text = file.read()

# C. Builds the following objects for Austen
# 1. a_toks: word tokens, all in lowercase
a_toks = [word.lower() for word in word_tokenize(text)]

# 2. a_tokfd: word frequency distribution
a_tokfd = FreqDist(a_toks)

# 3. a_bigrams: word bigrams, cast as a list
a_bigrams = list(bigrams(a_toks))

# 4. a_bigramfd: bigram frequency distribution
a_bigramfd = FreqDist(a_bigrams)

# 5. a_bigramcfd: bigram (w1, w2) conditional frequency distribution
a_bigramcfd = nltk.ConditionalFreqDist((w1, w2) for w1, w2 in a_bigrams)

# D. Pickles the bigram CFDs
with open('austen_bigramcfd.pkl', 'wb') as file:
    pickle.dump(a_bigramcfd, file, protocol=pickle.HIGHEST_PROTOCOL)

# E. Answers the following questions
# 1. How many word tokens and types are there?
print(f"Total Word Tokens: {len(a_toks)}")
print(f"Total Word Types: {len(set(a_toks))}")

# 2. What are the top 20 most frequent words and their counts?
print("Top 20 Most Frequent Words:")
print(a_tokfd.most_common(20))

# Plotting the top 20 most frequent words
plt.figure(figsize=(12, 6))
a_tokfd.plot(20, title='Top 20 Most Frequent Words')
plt.show()

# 3. What are the top 20 most frequent word bigrams and their counts? Omitting
filtered_bigrams = [bigram for bigram in a_bigrams if bigram[0] not in stopword
filtered_bigramfd = FreqDist(filtered_bigrams)
print("Top 20 Most Frequent Word Bigrams (Without Stopwords):")
print(filtered_bigramfd.most_common(20))

# 4. What are the top 20 most frequent word bigrams and their counts? Omitting
```

```
print("Top 20 Most Frequent Word Bigrams (With Stopwords):")
print(a_bigramfd.most_common(20))

# 5. What are the top 20 most frequent word bigrams and their counts? Draw char
plt.figure(figsize=(12, 6))
filtered_bigramfd.plot(20, title='Top 20 Most Frequent Word Bigrams (Without St
plt.show()

# 6. How many times does the word 'so' occur? What is their relative frequency
so_count = a_tokfd['so']
relative_frequency = so_count / len(a_toks)
print(f"The word 'so' occurs {so_count} times with a relative frequency of {rel

# 7. What are the top 20 'so-initial' bigrams and their counts?
so_initial_bigrams = [bigram for bigram in a_bigrams if bigram[0] == 'so']
so_initial_bigramfd = FreqDist(so_initial_bigrams)
print("Top 20 'so-initial' Bigrams:")
print(so_initial_bigramfd.most_common(20))

# 8. Given the word 'so' as the current word, what is the probability of gettir
so_much_probability = a_bigramcfd['so']['much'] / a_tokfd['so']
print(f"Probability of getting 'much' after 'so': {so_much_probability:.4%}")

# 9. Given the word 'so' as the current word, what is the probability of gettir
so_will_probability = a_bigramcfd['so']['will'] / a_tokfd['so']
print(f"Probability of getting 'will' after 'so': {so_will_probability:.4%}")
```

```
[nltk_data] Downloading package punkt to
[nltk_data]     C:\Users\1mscds43\AppData\Roaming\nltk_data...
[nltk_data]   Package punkt is already up-to-date!
[nltk_data] Downloading package stopwords to
[nltk_data]     C:\Users\1mscds43\AppData\Roaming\nltk_data...
[nltk_data]   Package stopwords is already up-to-date!

Total Word Tokens: 191785
Total Word Types: 7944
Top 20 Most Frequent Words:
[(',', 12016), ('.', 6355), ('the', 5201), ('to', 5181), ('and', 4877), ('o
f', 4284), ('i', 3177), ('a', 3124), ('--', 3100), ('it', 2503), ("'", 245
2), ('her', 2448), ('was', 2396), (';', 2353), ('she', 2336), ('not', 2281),
('in', 2173), ('be', 1970), ('you', 1967), ('he', 1806)]
```




```
Top 20 Most Frequent Word Bigrams (Without Stopwords):
[(('.', "''"), 1157), (("''", '``'), 959), ((',', "''"), 584), (('.', '``'),
416), (('!', '--'), 344), (('mr.', 'knightley'), 273), (("''", 'said'), 265),
((',', '``'), 250), (('mrs.', 'weston'), 246), (('?', "''"), 238), (('mr.',
'elton'), 211), (('?', '--'), 202), (('--', "''"), 194), (('emma', ','), 18
3), (('miss', 'woodhouse'), 170), (('oh', '!'), 167), (('!', "''"), 160),
(('mr.', 'weston'), 158), (('frank', 'churchill'), 144), (('mrs.', 'elton'),
140)]
Top 20 Most Frequent Word Bigrams (With Stopwords):
[((',', 'and'), 1882), (('.', "''"), 1157), (("''", '``'), 959), ((';', 'an
d'), 867), (('to', 'be'), 605), ((',', "''"), 584), (('.', 'i'), 570), ((',',
'i'), 569), (('of', 'the'), 559), (('in', 'the'), 445), (('it', 'was'), 442),
((';', 'but'), 427), (('.', '``'), 416), (('.', 'she'), 413), (('i', 'am'), 3
94), ((',', 'that'), 360), (('!', '--'), 344), (('--', 'and'), 334), (('she',
'had'), 332), (('she', 'was'), 328)]
```




```
The word 'so' occurs 968 times with a relative frequency of 0.5047%
Top 20 'so-initial' Bigrams:
[(('so', 'much'), 98), (('so', 'very'), 86), (('so', ','), 34), (('so', 'wel
l'), 31), (('so', 'many'), 29), (('so', 'long'), 27), (('so', '.'), 21), (('s
o', 'little'), 20), (('so', 'far'), 19), (('so', 'i'), 18), (('so', 'kind'),
14), (('so', ';'), 13), (('so', 'good'), 12), (('so', 'often'), 10), (('so',
'soon'), 9), (('so', 'great'), 8), (('so', 'it'), 8), (('so', 'you'), 8),
(('so', 'she'), 8), (('so', 'fond'), 7)]
Probability of getting 'much' after 'so': 10.1240%
Probability of getting 'will' after 'so': 0.1033%
```

In [ ]: