```
import requests
from bs4 import BeautifulSoup
import nltk
from nltk.tokenize import
sent_tokenize, word_tokenize
from nltk.corpus import stopwords
from collections import
Counter
import matplotlib.pyplot as plt
from wordcloud import WordCloud

# Download required
NLTK resources
nltk.download('punkt')
nltk.download('stopwords')

def getContent(url):

response = requests.get(url)
    text_content = ''

    if response.status_code == 200:

soup = BeautifulSoup(response.content, 'html.parser')
        paragraphs = soup.findAll('p')

     for paragraph in paragraphs:
            text_content += ' ' + paragraph.text
    else:

     print(f'Failed to retrieve the webpage. Status code: {response.status_code}')


return text_content.strip()

def processContent(text):
    sentence_list = sent_tokenize(text)

   stop_words = set(stopwords.words('english'))
    filtered_words = []

    for sentence
in sentence_list:
        words = word_tokenize(sentence)
        filtered_words.extend([word
for word in words if word.lower() not in stop_words and word.isalpha()])

    return
filtered_words

def wordFrequency(filtered_words):
    word_counts = Counter(filtered_words)

 return word_counts

def barGraphFrequency(word_counts):
    plt.figure(figsize=(10, 6))

plt.bar(list(word_counts.keys()), list(word_counts.values()))
    plt.xlabel('Words')

plt.ylabel('Frequency')
    plt.title('Word Frequencies')
    plt.xticks(rotation=45)

plt.show()

def barWordcloudFrequency(word_counts) :
    wordcloud = WordCloud(width=800,
height=400, background_color='white').generate_from_frequencies(word_counts)

plt.figure(figsize=(10, 6))
    plt.imshow(wordcloud, interpolation='bilinear')
```

```
plt.axis('off')
    plt.title('Word Cloud')
    plt.show()
    plt.figure(figsize=(10, 6))

plt.bar(list(word_counts.keys()), list(word_counts.values()))
    plt.xlabel('Words')

plt.ylabel('Frequency')
    plt.title('Word Frequencies')
    plt.xticks(rotation=45)

plt.show()


url = input("Please enter the URL: ")
text =
getContent(url)
filtered_words = processContent(text)
print(filtered_words)
word_counts =
wordFrequency(filtered_words)
for word, count in word_counts.most_common():
    print(word,
": ", count)

top_N = 10
top_word_counts = word_counts.most_common(top_N)

words,
counts = zip(*top_word_counts)

barGraphFrequency(Counter(dict(zip(words,
counts))))
barWordcloudFrequency(word_counts)
```