LAB 2. Remove Stopwords from a given random sentence using the NLTK library.

What are Stop words?

A stop word is a commonly used word (such as “the”, “a”, “an”, or “in”) that a search engine has been programmed to ignore, both when indexing entries for searching and when retrieving them as the result of a search query.

We would not want these words to take up space in our database or take up valuable processing time. For this, we can remove them easily, by storing a list of words that you consider to stop words. NLTK (Natural Language Toolkit) in Python has a list of stopwords stored in 16 different languages. You can find them in the nltk_data directory. `/home/PratimaPython/nltk_data/corpora/stopwords` is the directory address. (Do not forget to change your home directory name)

Stop word removal using NLTK

Need to remove the Stopwords

The necessity of removing stopwords in NLP is contingent upon the specific task at hand. For text classification tasks, where the objective is to categorize text into distinct groups, excluding stopwords is common practice. This is done to channel more attention towards words that truly convey the essence of the text. As illustrated earlier, certain words like “there,” “book,” and “table” contribute significantly to the text’s meaning, unlike less informative words such as “is” and “on.”

Conversely, for tasks like machine translation and text summarization, the removal of stopwords is not recommended. In these scenarios, every word plays a pivotal role in preserving the original meaning of the content.

Types of Stopwords

Stopwords are frequently occurring words in a language that are frequently omitted from natural language processing (NLP) tasks due to their low significance for deciphering textual meaning. The particular list of stopwords can change based on the language being studied and the context. The following is a broad list of stopword categories:

- **Common Stopwords:** These are the most frequently occurring words in a language and are often removed during text preprocessing. Examples include “the,” “is,” “in,” “for,” “where,” “when,” “to,” “at,” etc.
- **Custom Stopwords:** Depending on the specific task or domain, additional words may be considered as stopwords. These could be domain-specific terms that don’t contribute much to the overall meaning. For example, in a medical context, words like “patient” or “treatment” might be considered as custom stopwords.
- **Numerical Stopwords:** Numbers and numeric characters may be treated as stopwords in certain cases, especially when the analysis is focused on the meaning of the text rather than specific numerical values.
- **Single-Character Stopwords:** Single characters, such as “a,” “I,” “s,” or “x,” may be considered stopwords, particularly in cases where they don’t convey much meaning on their own.

- **Contextual Stopwords:** Words that are stopwords in one context but meaningful in another may be considered as contextual stopwords. For instance, the word “will” might be a stopword in the

1. Imports and Setup:

In [1]:
```python
import nltk
from nltk.corpus import stopwords

nltk.download('stopwords', quiet=True)
```

Out[1]:
```
True
```

In [12]:
```python
print(stopwords.words('english'))
```

```
['i', 'me', 'my', 'myself', 'we', 'our', 'ours', 'ourselves', 'you', "you'r
e", "you've", "you'll", "you'd", 'your', 'yours', 'yourself', 'yourselves',
'he', 'him', 'his', 'himself', 'she', "she's", 'her', 'hers', 'herself', 'i
t', "it's", 'its', 'itself', 'they', 'them', 'their', 'theirs', 'themselve
s', 'what', 'which', 'who', 'whom', 'this', 'that', "that'll", 'these', 'th
ose', 'am', 'is', 'are', 'was', 'were', 'be', 'been', 'being', 'have', 'ha
s', 'had', 'having', 'do', 'does', 'did', 'doing', 'a', 'an', 'the', 'and',
'but', 'if', 'or', 'because', 'as', 'until', 'while', 'of', 'at', 'by', 'fo
r', 'with', 'about', 'against', 'between', 'into', 'through', 'during', 'be
fore', 'after', 'above', 'below', 'to', 'from', 'up', 'down', 'in', 'out',
'on', 'off', 'over', 'under', 'again', 'further', 'then', 'once', 'here',
'there', 'when', 'where', 'why', 'how', 'all', 'any', 'both', 'each', 'fe
w', 'more', 'most', 'other', 'some', 'such', 'no', 'nor', 'not', 'only', 'o
wn', 'same', 'so', 'than', 'too', 'very', 's', 't', 'can', 'will', 'just',
'don', "don't", 'should', "should've", 'now', 'd', 'll', 'm', 'o', 're', 'v
e', 'y', 'ain', 'aren', "aren't", 'couldn', "couldn't", 'didn', "didn't",
'doesn', "doesn't", 'hadn', "hadn't", 'hasn', "hasn't", 'haven', "haven't",
'isn', "isn't", 'ma', 'mightn', "mightn't", 'mustn', "mustn't", 'needn', "n
eedn't", 'shan', "shan't", 'shouldn', "shouldn't", 'wasn', "wasn't", 'were
n', "weren't", 'won', "won't", 'wouldn', "wouldn't"]
```

2. User Input:

In [2]:
```python
sentence = input("Enter a sentence to remove stop words from: ")
print("\nYour input sentence:", sentence)
```

```
Enter a sentence to remove stop words from: hello my name is tilak, the of
my is tilaks parajuli is good boy

Your input sentence: hello my name is tilak, the of my is tilaks parajuli i
s good boy
```

3. Stop Word Removal Function:

In [3]:
```python
def remove_stop_words(sentence):

    try:
        sentence_lower = sentence.lower()

        words = nltk.word_tokenize(sentence_lower)

        stop_words = set(stopwords.words('english'))

        filtered_words = [word for word in words if word not in stop_words]

        filtered_sentence = " ".join(filtered_words)

        return filtered_sentence

    except LookupError:
        return "Error: Could not load stopwords from NLTK. Please try again
```

In [4]:
```python
filtered_sentence = remove_stop_words(sentence)
```

4. Output and Visualization:

In [5]:
```python
# Display the original and filtered sentences with headings and colors
print("\nOriginal Sentence (with stop words):")
print(f"\t{sentence}")

print("\nFiltered Sentence (without stop words):")
print(f"\t{filtered_sentence}\n")
```

```
Original Sentence (with stop words):
        hello my name is tilak, the of my is tilaks parajuli is good boy

Filtered Sentence (without stop words):
        hello name tilak , tilaks parajuli good boy
```

5. Visualization

In [6]:
```python
words_before = len(sentence.split())
words_after = len(filtered_sentence.split())
```

In [7]:
```python
words_before
```

Out[7]:
```
14
```

In [8]:
```python
words_after
```

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

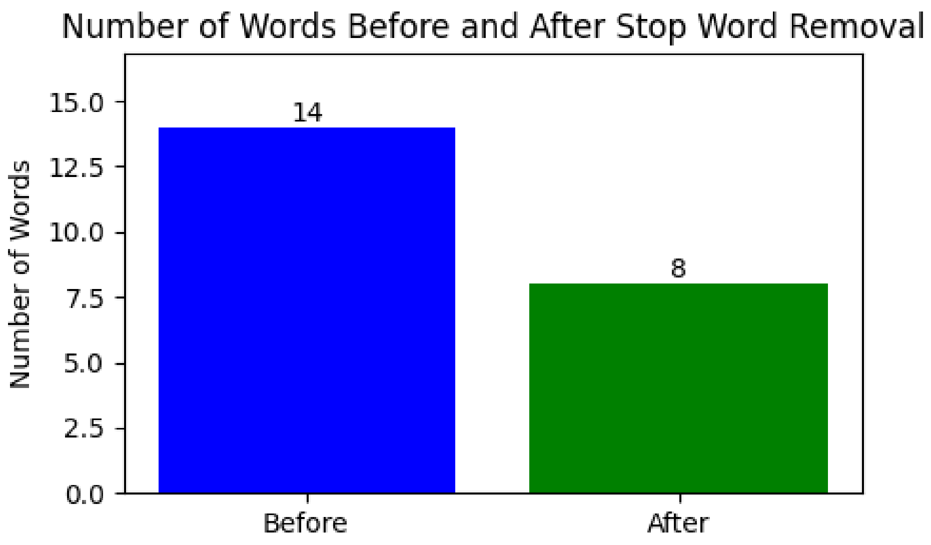
In [10]:
```python
import matplotlib.pyplot as plt

# Plot words before and after stop word removal
plt.figure(figsize=(5, 3))

# Plot words before
plt.bar(['Before'], [words_before], color='blue')
plt.text('Before', words_before, str(words_before), ha='center', va='bottom'

# Plot words after
plt.bar(['After'], [words_after], color='green')
plt.text('After', words_after, str(words_after), ha='center', va='bottom')

plt.title('Number of Words Before and After Stop Word Removal')
plt.ylabel('Number of Words')
plt.ylim(0, max(words_before, words_after) * 1.2)
plt.show()
```


Number of Words Before and After Stop Word Removal
Number of Words
Before
After