1.Write a python program to perform tokenization by word and sentence using nltk.

```
import nltk
nltk.download('punkt_tab')
from nltk.tokenize import sent_tokenize
def tokenize_sentences(text):
    sentences = sent_tokenize(text)
    return sentences
text = """NLTK is a leading platform for building Python programs to work with human language data.
It provides easy-to-use interfaces to over 50 corpora and lexical resources such as WordNet.
It also includes a suite of text processing libraries for classification, tokenization, stemming,
tagging, parsing, and semantic reasoning."""
sentences = tokenize_sentences(text)
for i, sentence in enumerate(sentences):
    print(f"Sentence {i+1}: {sentence}")
```

WORD TOKENIZATION

```
import nltk
nltk.download('punkt') # Download the necessary tokenization models
from nltk.tokenize import word_tokenize
def tokenize_words(text):
 words = word_tokenize(text)
 return words
# Example text
text = "NLTK is a leading platform for building Python programs to work with human language data."
# Tokenize words
words = tokenize_words(text)
# Print tokenized words
print(words)
```

2.Write a python program to eliminate stopwords using nltk.

```
# Stopwords
import nltk
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
# Download NLTK stopwords and tokenizer models
nltk.download('stopwords')
nltk.download('punkt')
def remove_stopwords(text):
 # Tokenize the text into words
 words = word_tokenize(text)
 # Get English stopwords
 english_stopwords = set(stopwords.words('english'))
 # Remove stopwords from the tokenized words
 filtered_words = [word for word in words if word.lower() not in english_stopwords]
 # Join the filtered words back into a single string
 filtered_text = ' '.join(filtered_words)
 return filtered_text
# Example text
text = "NLTK is a leading platform for building Python programs to work with human language data."
# Remove stopwords
filtered_text = remove_stopwords(text)
# Print filtered text
print(filtered_text)
```

3.Write a python program to perform stemming using nltk.

```
# Stemming
import nltk
from nltk.stem import PorterStemmer
from nltk.tokenize import word_tokenize
```

```
# Download NLTK tokenizer and stemmer models
nltk.download('punkt')
def stem_text(text):
 # Initialize the Porter Stemmer
 porter_stemmer = PorterStemmer()
 # Tokenize the text into words
 words = word_tokenize(text)
 # Apply stemming to each word
 stemmed_words = [porter_stemmer.stem(word) for word in words]
 # Join the stemmed words back into a single string
 stemmed_text = ' '.join(stemmed_words)
 return stemmed_text
# Example text
text = "NLTK is a leading platform for building Python programs to work with human language data."
# Perform stemming
stemmed_text = stem_text(text)
# Print stemmed text
print(stemmed_text)
```

(or)

Write a Python program for the following preprocessing of text in NLP:

- **Tokenization**
- **Filtration**
- **Script Validation**
- **Stop Word Removal**
- **Stemming the given the code**

```
import re
import nltk
from nltk.tokenize import word_tokenize
from nltk.corpus import stopwords
```

```
from nltk.stem import PorterStemmer
# Download necessary NLTK data files
nltk.download('punkt')
nltk.download('stopwords')
text = """Natural Language Processing is a fascinating field of AI.
It involves processing human languages, like English!"""
# 1. Tokenization
def tokenize(text):
    return word_tokenize(text)
# 2. Filtration (Remove special characters)
def filter_text(tokens):
    filtered_tokens = [re.sub(r'[^\w\s]', '', token) for token in tokens]
    filtered_tokens = [token for token in filtered_tokens if token.isalpha()]
    return filtered_tokens
# 3. Script Validation (English only)
def validate_script(tokens):
    return [token for token in tokens if re.match(r'^[a-zA-Z]+$', token)]
# 4. Stop Word Removal
def remove_stopwords(tokens):
    stop_words = set(stopwords.words('english'))
    return [token for token in tokens if token.lower() not in stop_words]
# 5. Stemming
def stem_tokens(tokens):
    stemmer = PorterStemmer()
    return [stemmer.stem(token) for token in tokens]
# Pipeline for text preprocessing
def preprocess_text(text):
    print(f"Original Text:\n{text}\n")
    tokens = tokenize(text)
    print(f"Tokens:\n{tokens}\n")
```

```
    filtered_tokens = filter_text(tokens)
    print(f"Filtered Tokens:\n{filtered_tokens}\n")
    validated_tokens = validate_script(filtered_tokens)
    print(f"Validated Tokens (English Only):\n{validated_tokens}\n")
    tokens_no_stopwords = remove_stopwords(validated_tokens)
    print(f"Tokens after Stop Word Removal:\n{tokens_no_stopwords}\n")
    stemmed_tokens = stem_tokens(tokens_no_stopwords)
    print(f"Stemmed Tokens:\n{stemmed_tokens}\n")
    return stemmed_tokens
# Run the preprocessing pipeline
preprocessed_tokens = preprocess_text(text)
```