Task 2
Creating a plagiarism checker is a complex task that typically involves natural language processing and text similarity techniques. Here's a simplified outline of how you can build a basic plagiarism checker in Python using libraries such as NLTK (Natural Language Toolkit) and scikit-learn:

1. **Preprocessing**:
   - Load the source text (the document to be checked for plagiarism) and the target texts (potential sources of plagiarism).
   - Tokenize the text: Break the text into words or sentences.
   - Remove stopwords and punctuation.
   - Convert text to lowercase.

2. **Text Similarity Measure**:
   - Choose a text similarity metric to compare the source text to the target texts. Common metrics include:
     - Cosine similarity
     - Jaccard similarity
     - Levenshtein distance
     - Similarity based on TF-IDF (Term Frequency-Inverse Document Frequency)

3. **Feature Extraction**:
   - Compute the chosen similarity metric for each target text compared to the source text.

4. **Threshold Setting**:
   - Define a threshold value above which the source text is considered to be plagiarized. The threshold depends on the similarity metric and the specific use case.

5. **Plagiarism Detection**:
   - Compare the computed similarity scores with the threshold to identify potential cases of plagiarism.

Here's a simplified Python code example using NLTK and cosine similarity to check for plagiarism between a source text and a set of target texts: