Information Retrieval- Lab Assignments 2

Aim: Implement a program for retrieval of documents using inverted files.

```
# Define the documents

document1 = "The quick brown fox jumped over the lazy dog."
document2 = "The lazy dog slept in the sun."

# Step 1: Tokenize the documents
# Convert each document to lowercase and split it into words
tokens1 = document1.lower().split()
tokens2 = document2.lower().split()

# Combine the tokens into a list of unique terms
terms = list(set(tokens1 + tokens2))

# Step 2: Build the inverted index
# Create an empty dictionary to store the inverted index
inverted_index = {}

# For each term, find the documents that contain it
for term in terms:
    documents = []
    if term in tokens1:
        documents.append("Document 1")
    if term in tokens2:
        documents.append("Document 2")
    inverted_index[term] = documents

# Step 3: Print the inverted index
for term, documents in inverted_index.items():
    print(term, "->", ", ".join(documents))
```

Explanation of the Code

The first two lines define two sample documents to be used as input to the algorithm.

- **Step 1:** Tokenize the input documents by converting them to lowercase and splitting them into individual words. Then combine the resulting tokens from both documents into a single list of unique terms.
- **Step 2:** Create an empty dictionary to store the inverted index, and then iterate through each term in the list of unique terms. For each term, create an empty list of documents, and then check if the term appears in each input document.

- If the term appears in a document, add the document to the list for that term. Finally, add an entry to the inverted index dictionary for the current term, with the list of documents that contain that term as its value.
- **Step 3:** Iterate through the entries in the inverted index dictionary and print out each term along with the list of documents that contain it.

Output

jumped -> Document 1

fox -> Document 1

lazy -> Document 1, Document 2

the -> Document 1, Document 2

in -> Document 2

dog. -> Document 1

quick -> Document 1

dog -> Document 2

slept -> Document 2

sun. -> Document 2

brown -> Document 1

over -> Document 1

Conclusion: