2

November 3, 2023

Implement a program for retrieval of documents using inverted files

[2]:
```python
import re
import collections

# Sample documents
documents = {
    1: "This is the first document. It contains some words.",
    2: "This is the second document. It also contains words.",
    3: "The third document is different from the first two.",
    4: "Inverted index is essential for document retrieval.",
}

# Function to preprocess and tokenize text
def preprocess(text):
    text = text.lower()
    tokens = re.findall(r'\w+', text)
    return tokens

# Create an inverted index
def build_inverted_index(documents):
    inverted_index = collections.defaultdict(list)
    for doc_id, document in documents.items():
        tokens = preprocess(document)
        for token in tokens:
            inverted_index[token].append(doc_id)
    return inverted_index

# Function to perform document retrieval
def retrieve_documents(query, inverted_index):
    query_tokens = preprocess(query)
    result = set()

    # Retrieve documents containing each query token
    for token in query_tokens:
        if token in inverted_index:
            if not result:
                result = set(inverted_index[token])
```

```
            else:
                result = result.intersection(inverted_index[token])

    return result

# Build the inverted index
inverted_index = build_inverted_index(documents)

# Example queries
query1 = input("Enter query: ")


# Retrieve documents for the queries
result1 = retrieve_documents(query1, inverted_index)


# Display the results
print("Query:", query1)
print("Matching Documents:", result1)
```

```
Enter query: It
Query: It
Matching Documents: {1, 2}
```

[ ]:

[ ]: