Name : Devkumar Biswas

Class: BE(AI&DS)

Div: B

Subject : IR(CL-II Lab)

Roll no. : BEAD21267

**Assignment No. - 2**

**Problem Statement** : Implement a program for retrieval of documents using inverted files.

Code:-

from collections import defaultdict

import re

def tokenize(text):

# Convert to lowercase and tokenize the text by words, keeping only alphanumeric tokens

tokens = re.findall(r'\b\w+\b', text.lower())

return tokens

def build\_inverted\_index(documents):

inverted\_index = defaultdict(list)

for doc\_id, text in enumerate(documents):

tokens = tokenize(text)

for token in tokens:

if doc\_id not in inverted\_index[token]:

inverted\_index[token].append(doc\_id) # Add doc\_id if not already present

return inverted\_index

def retrieve\_documents(query, inverted\_index):

query\_tokens = tokenize(query)

doc\_sets = []

for token in query\_tokens:

if token in inverted\_index:

doc\_sets.append(set(inverted\_index[token]))

else:

return [] # If any token is not found, return an empty list (no documents found)

# Intersect all sets of document IDs to get the final matching documents

result\_docs = set.intersection(\*doc\_sets) if doc\_sets else set()

return list(result\_docs)

# List of example documents

documents = [

"Natural language processing with Python.",

"Deep learning models for AI and NLP tasks.",

"Python programming for machine learning.",

"AI models in healthcare and NLP."

]

# Build the inverted index

inverted\_index = build\_inverted\_index(documents)

print("Inverted Index:", dict(inverted\_index))

# Query the inverted index

query = "NLP and Python"

matching\_docs = retrieve\_documents(query, inverted\_index)

# Display the matching documents

print("Documents matching query:", matching\_docs)

for doc\_id in matching\_docs:

print(f"Doc {doc\_id}: {documents[doc\_id]}")

OUTPUT:-



