

AARYAN BAIRAGI

BE-47004

ISR LAB 3:

```
import java.io.*;
import java.util.*;

public class InvertedFileIndex {

    static Map<String, Set<Integer>> invertedIndex = new HashMap<>();

    // Indexing function: reads file and updates inverted index
    public static void indexFile(String fileName, int fileNumber) throws IOException {
        BufferedReader br = new BufferedReader(new FileReader(fileName));
        String line;

        while ((line = br.readLine()) != null) {
            String[] words = line.toLowerCase().split("[ ,;:\\\"()?!]+");
            for (String word : words) {
                if (word.trim().length() == 0) continue;
                invertedIndex.computeIfAbsent(word, k -> new HashSet<>()).add(fileNumber);
            }
        }
        br.close();
    }

    // Display the entire inverted index
    public static void displayIndex() {
        System.out.println(" Inverted Index:");
        for (Map.Entry<String, Set<Integer>> entry : invertedIndex.entrySet()) {
            System.out.print(entry.getKey() + " -> ");
            for (int doc : entry.getValue()) {

```

```

        System.out.print("File" + doc + " ");
    }

    System.out.println();
}

}

// Retrieve files that contain a given word

public static void retrieveDocuments(String query) {

    query = query.toLowerCase();

    if (invertedIndex.containsKey(query)) {

        System.out.println(" The word '" + query + "' is found in: ");

        for (int fileNum : invertedIndex.get(query)) {

            System.out.println(" - File" + fileNum);

        }
    } else {

        System.out.println(" The word '" + query + "' is NOT found in any document.");
    }
}

// Main method

public static void main(String[] args) throws IOException {

    BufferedReader br = new BufferedReader(new InputStreamReader(System.in));

    System.out.print(" Enter number of documents to index: ");

    int n = Integer.parseInt(br.readLine());

    for (int i = 1; i <= n; i++) {

        System.out.print("Enter file name for File" + i + ": ");

        String fileName = br.readLine();

        indexFile(fileName, i);
    }
}

```

```

displayIndex();

// Query phase

String query;

do {

    System.out.print(" Enter a word to search (or type 'exit' to quit): ");

    query = br.readLine();

    if (!query.equalsIgnoreCase("exit")) {

        retrieveDocuments(query);

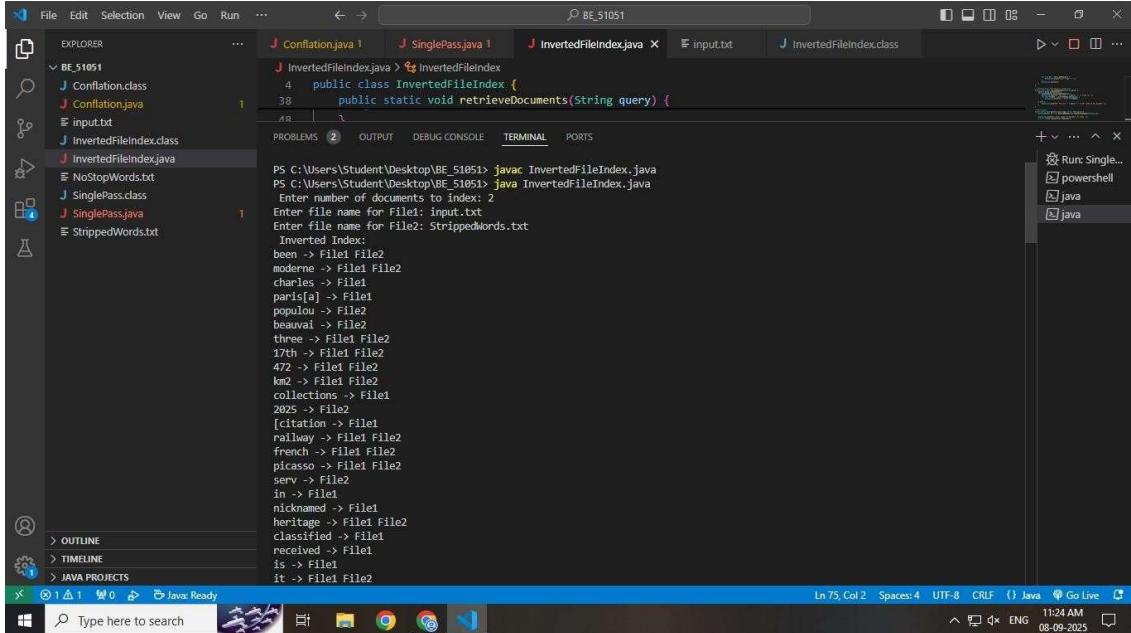
    }

} while (!query.equalsIgnoreCase("exit"));

}

```

OUTPUT:



The screenshot shows a Java development environment with multiple tabs open. The tabs include Conflation.java, SinglePass.java, InvertedfileIndex.java, input.txt, and InvertedfileIndex.class. The InvertedfileIndex.java tab is active, displaying code for an inverted index. The code includes a main method that reads a document count from the command line, reads file names from standard input, and prints an inverted index mapping words to file pairs. The output window shows the execution of the Java compiler (javac) and the application (java InvertedfileIndex). The application prompts for the number of documents (2), file names (input.txt and StrippedWords.txt), and then lists the inverted index entries.

```

PS C:\Users\Student\Desktop\BE_51051> javac InvertedfileIndex.java
PS C:\Users\Student\Desktop\BE_51051> java InvertedfileIndex.java
Enter number of documents to index: 2
Enter file name for file1: input.txt
Enter file name for file2: StrippedWords.txt
Inverted Index:
been -> File1 File2
moderne -> File1
charles -> File1
paris[a] -> File1
populou -> File2
beauvai -> File2
three -> File1 File2
17th -> File1 File2
472 -> File1 File2
km2 -> File1 File2
collections -> File1
2025 -> File2
[citation -> File1
railway -> File1 File2
french -> File1 File2
picasso -> File1 File2
serv -> File2
in -> File1
nicknamed -> File1
heritage -> File1 File2
classified -> File1
received -> File1
is -> File1
it -> File1 File2

```