Aditya Ganesh Khedekar (245974)
Sarvesh Kulkarni (245891)
Siddhi Belgamwar (246167)
Amita Rohidas Kashid (248642)

Part A)

Analyzer Configuration:

- An instance of Analyzer named BiWordAnalyzer is created. This analyzer is specialized for bi-word tokenization.
- The createComponents method is overridden within BiWordAnalyzer. It configures a chain of tokenization filters:
  1. StandardTokenizer: Splits text into words and removes punctuation.
  2. StandardFilter: Applies additional text processing, potentially removing some tokens or modifying them.
  3. FixedShingleFilter: Generates bi-word tokens by combining adjacent words.

Indexing Preparation:

- A Lucene RAMDirectory named directory is instantiated to store the indexed content in memory.
- An IndexWriterConfig is set up with the BiWordAnalyzer to define how the content should be indexed.
- An IndexWriter named indexWriter is initialized with the directory and IndexWriterConfig.

Document Creation and Indexing:

- A new Document named document is created to hold the text content.
- Text content is assigned to the text variable.
- A FieldType is defined to specify how the field will be indexed, storing various attributes such as term vectors, positions, offsets, etc.
- A field named "Main" is added to the document with the specified FieldType, containing the text.

Indexing the Document:

- The indexWriter adds the document to the index.
- After adding the document, the indexWriter is closed, finalizing the indexing process.

Retrieving and Printing BiWords:

- To retrieve the bi-word tokens, a TokenStream named tokenStream is obtained by invoking BiWordAnalyzer.tokenStream("Main", text).
- Offset and term attributes (OffsetAttribute and CharTermAttribute) are acquired from the TokenStream to extract necessary information about the tokens.
- The try block begins:
  1. The tokenStream is reset to the start of the stream.
  2. While there are tokens available (tokenStream.incrementToken() returns true), the term (bi-word token) is extracted using charTermAttribute.toString() and printed out.
  3. Finally, the tokenStream is closed within the finally block to ensure proper resource handling.

Part B)
Indexing a New Document:

- A new RAMDirectory named directory2 is created to store the indexed content in memory.
- An IndexWriterConfig named config2 is initialized with the previously defined BiWordAnalyzer.
- An IndexWriter named indexWriter2 is created using directory2 and config2.
- A new Document named document2 is created to hold text content.
- The text2 variable is set to "York University in Bhandup, UK".
- A FieldType named fieldType2 is defined to specify indexing options for the "Main" field, including term vectors, positions, and offsets.
- A field named "Main" is added to document2 with the specified fieldType2.

Indexing the Document:

- indexWriter2 adds document2 to the index.
- indexWriter2 is closed, finalizing the indexing process.

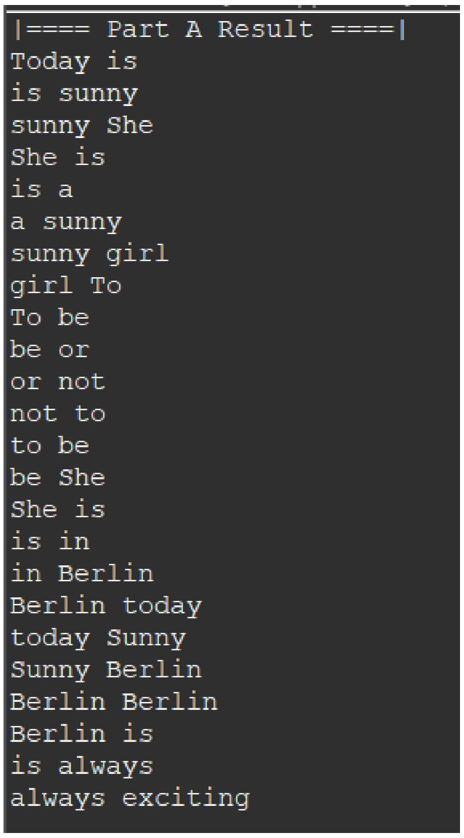
Performing a Search:

- A new String named inputquery is created, containing the text "New York University".
- DirectoryReader named directoryReader is initialized by opening directory2 to read the indexed content.
- An IndexSearcher named indexSearcher is created using directoryReader to perform searches.
- A QueryParser named parser is initialized to parse queries for the "Main" field using the BiWordAnalyzer.
- The inputquery string is parsed into a Lucene Query object using the parser.
- indexSearcher searches for matching documents based on the generated Query, limiting results to 1000 (indexSearcher.search(query, 1000)), and stores the results in topDocs.
- The retrieved search results are stored in an array of ScoreDoc named hits.

Printing Search Results:

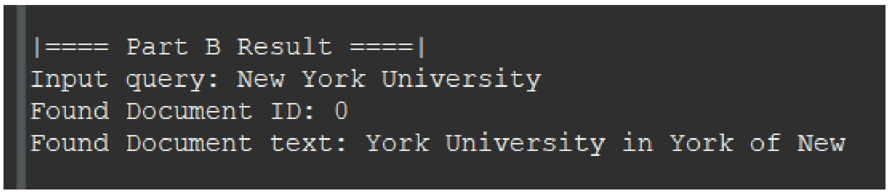
- The code block inside the if(hits.length > 0) condition executes if any matching documents are found.
- It prints the input query and information about the found documents:
    - Prints the input query: "Input query: New York University".
    - For each document found (hits[i].doc):
        - Displays the ID of the found document: "Found Document ID: <document ID>".
        - Retrieves and prints the text content of the found document from the "Main" field: "Found Document text: <document text>".

Expected output:
Part A:



```
|==== Part A Result ====|
Today is
is sunny
sunny She
She is
is a
a sunny
sunny girl
girl To
To be
be or
or not
not to
to be
be She
She is
is in
in Berlin
Berlin today
today Sunny
Sunny Berlin
Berlin Berlin
Berlin is
is always
always exciting
```

Part B:



```
|==== Part B Result ====|
Input query: New York University
Found Document ID: 0
Found Document text: York University in York of New
```