**Program Execution -**

The execution of program starts from the main method of Problem1A file. Once the file is run outputs are displayed on the screen and the cleaned and transformed data is stored in Collection named **news** present in the **ReuterDb** i.e the name of the database I have created in mongoDb.

A screenshot of a computer

Description automatically generated

Fig. 1: output on console after Program1A execution

Below is the snapshot of the documents that are stored in the database after the execution.

A screenshot of a computer

Description automatically generated

Fig. 2: 1000 documents inserted

**Test Execution –**

The functional test cases are written in the Problem1ATest.java file -

Execution of the test file prints on the console if the test passes or fails.

**A screenshot of a computer

Description automatically generated**

Fig. 3: Successfully passed test cases

**Flowchart –**

Below is the flowchart representing all the steps taken to read data from reut2-009.sgm file and clean and transform the content and finally store it in document database.(MongoDb)

A diagram of a flowchart

Description automatically generated

Fig. 4: Flowchart of ReuterDb clean and transform

**Algorithm: Reuters Data cleaning/transformation program**

1. Begin:
2. Try:
   1. Initialize empty lists for titles and bodies.
   2. Read the content of the Reuters data file into a string variable using a FileReader.
   3. Parse articles from the file content using a ReuterArticleParser:
      1. Initialize empty lists for parsed titles and bodies.
      2. Define a regular expression pattern to match <REUTERS> tags with <TITLE> and <BODY> tags.
      3. Initialize matcher with the pattern and file content.
      4. While matcher finds a match:
         1. Extract title from the group corresponding to <TITLE> tag.
         2. Extract body from the group corresponding to <BODY> tag.
         3. Add the title to the parsed titles list.
         4. Add the body to the parsed bodies list.
   4. Clean the parsed titles using a TextCleaner.
   5. Iterate over each cleaned title and its corresponding body:
      1. For each cleaned title and its corresponding body:
      2. Create a new NewsArticle object with the cleaned title and body.
      3. Convert the NewsArticle object to a MongoDB Document.
      4. Insert the document into MongoDB using a MongoDBConnector.
3. Catch any exceptions:
   1. Print the stack trace.
4. End.

**Code Walkthrough –**

Let's break down the role of each file:

**MongoNewsConnector.java**: This class is responsible for interacting with MongoDB. It establishes a connection to the MongoDB database, inserts documents into a specified collection, and extracts the database name from the connection string.

**NewsArticle.java**: This class represents a news article. It contains fields for the title and body of the article, as well as methods to get and set these fields, and a method to convert the article object to a MongoDB document.

**ReuterArticleParser.java**: This class is responsible for parsing Reuters articles from the file content. It defines a method to parse articles, using regular expressions to extract titles and bodies from <TITLE> and <BODY> tags.

**ReutReader.java**: This is main class of the application. It reads the contents of the Reuters data file, parses articles using the ReuterArticleParser, cleans the parsed text using the TextCleaner, and inserts the cleaned data into a MongoDB database using the MongoNewsConnector

**TextCleaner.java**: This class is responsible for cleaning text. It defines a method to remove special characters from the text, making it suitable for insertion into a MongoDB database.

**Prerequisites –**

Before code execution it is required to add the VM Options **-Djdk.tls.client.protocols=TLSv1.2**  
This is to make mongo compatible with java 11. (Note - the issue doesn't exist in version > jdk 13.0.5)

A screenshot of a computer

Description automatically generated

Fig. 5: Run configuration to use VM option

**References –**

* 1. Draw.IO. Flowchart[Online]. Available: <https://app.diagrams.net/>