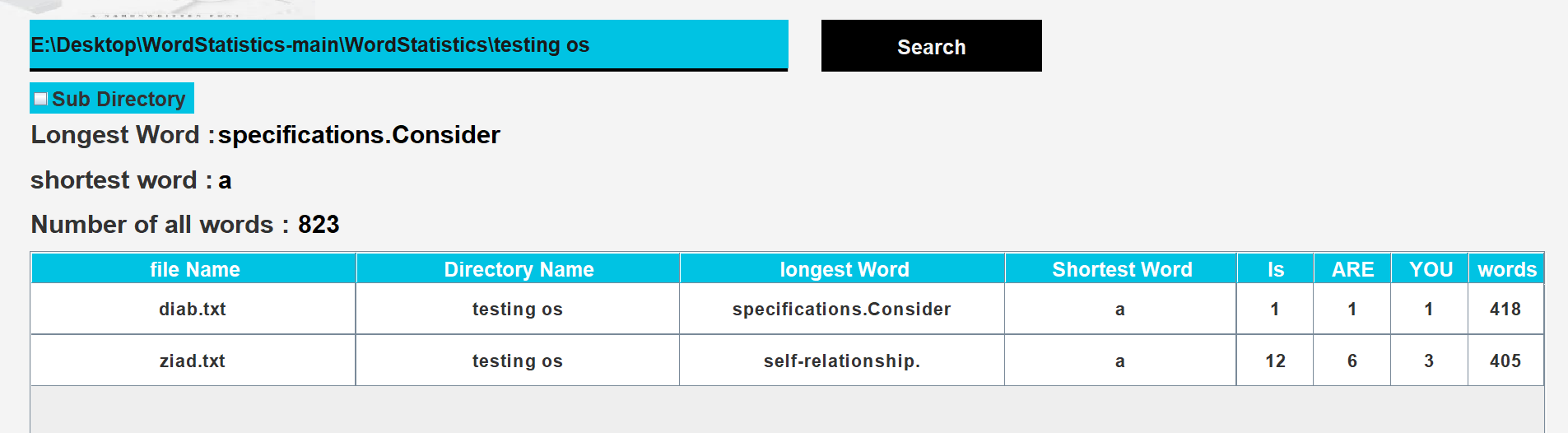
**Project description:**

our project read the text of files in specific folder and return the number of words,longest word , shortest, and number of repeat “is”,”are”,and “you”.

There is an option to select sub directories , by default its not executed , and program read the text files only that siblings to the sub directories When choose it , the program can read the files into the sub directories.

**The inputs are given as:** the path to the choosen directory

**Example on the output:**



**What we have actually did:**

The idea is that the program is divided on 4 threads which are running in parallel so that they are running at the same time and read all the files in the directory and the sub directories

Team members roles:

Read\_runnable class: Diab mosa , Khaled sameh

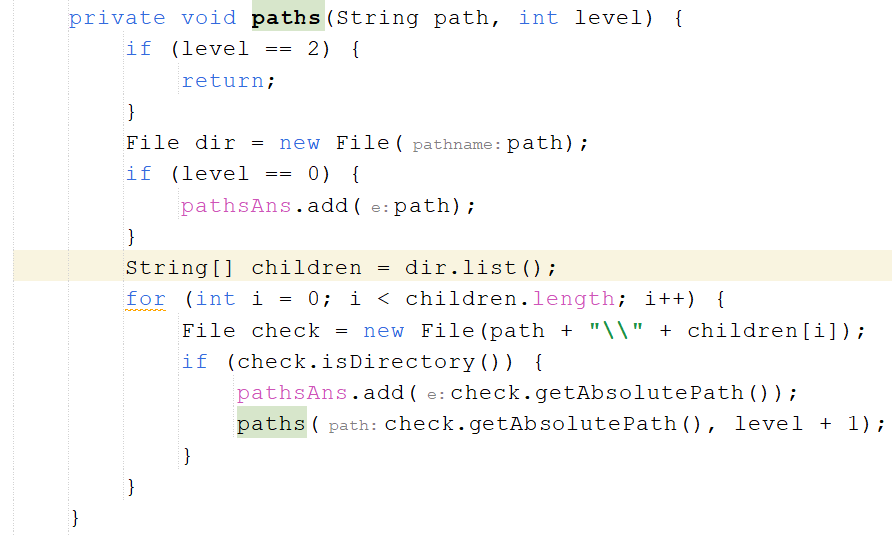
Read\_dir class: Ziad Nabil , Diab mosa

Testing the code: ziad amr,ziad yasser

Documentation: Ziad nabil

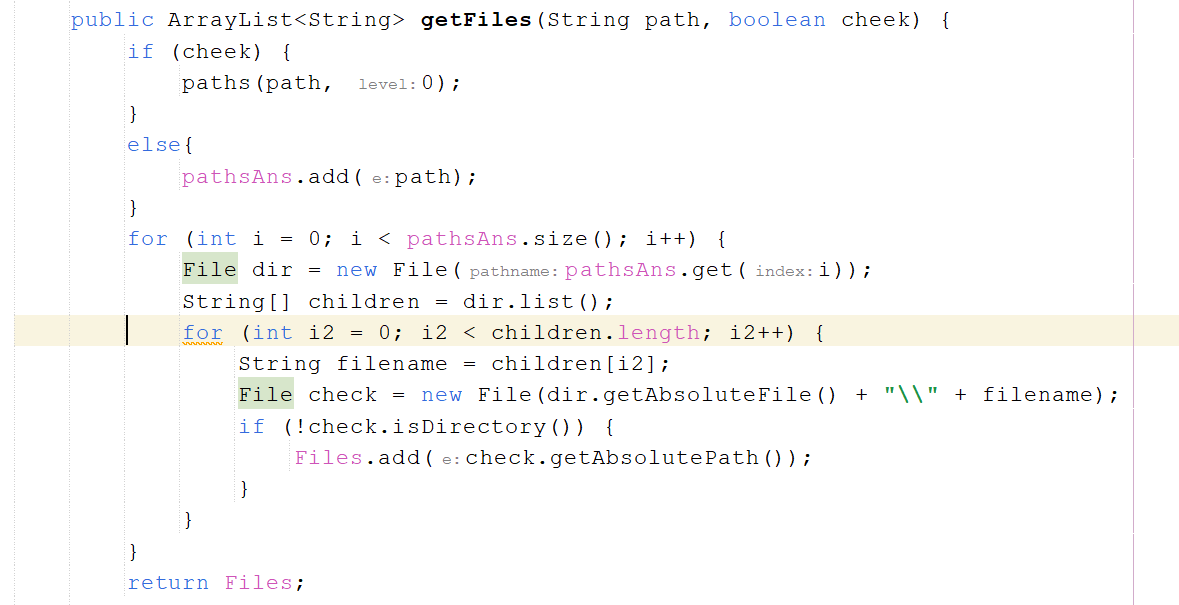
Gui: Ziad yasser , Ziad amr

So,we made some functions to handle this

1. paths 

==>This function traverses directories to collect paths within a specific level limit

1. **getFiles()**



==>This getFiles method in the Read\_dir class is designed to retrieve files from a given directory path

How it works?

1-Initialization: This method accepts a path (directory path) and a cheek (boolean) parameter.

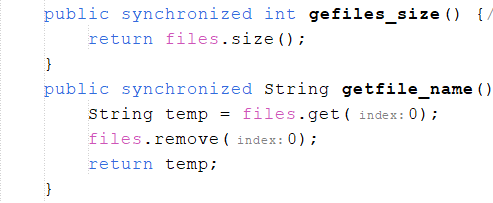
2-Paths Collection: If cheek is true, it initiates the process of collecting paths by calling the paths method with the provided directory path and a level of 0. Otherwise, if cheek is false, it directly adds the provided path to the pathsAns list.

3-File Retrieval: It iterates through the collected paths (pathsAns) and retrieves files within each directory. For each path, it gets the list of children (files and directories) within that directory.

4-Filtering Files: For each child in the directory, it checks if it's a file (not a directory) and adds the absolute path of the file to the Files list.

1. Returning Files: Finally, it returns the list of absolute file paths (Files).

**Threads functions:**



**Getfiles\_size():**

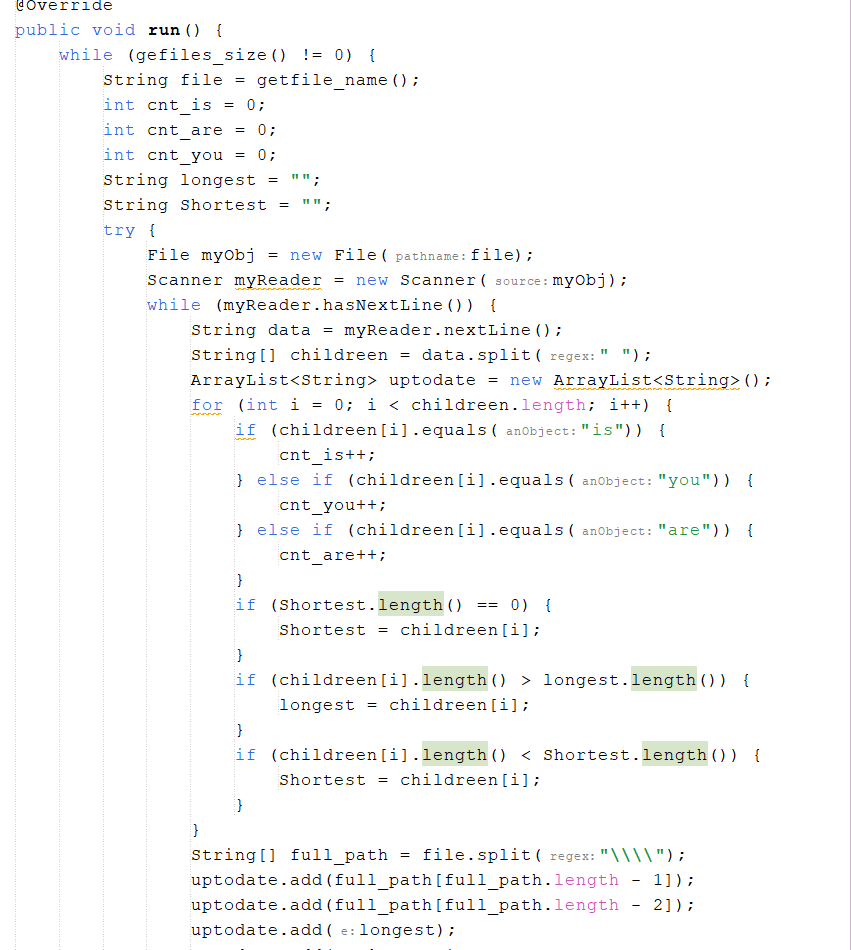
returns the size of the files collection.

**Getfile\_name():**

retrieves the first element from the files collection, removes it from the collection, and returns it.

===>> These methods ensure thread safety by synchronizing access to the files collection, making sure that only one thread can access these methods at a time to prevent concurrency issues.

\*\*\*One of the most important functions that we did is the run() functions which is responsible for processing files



That process can be made through:

**1**-Looping through Files:

It checks if there are files left to process (gefiles\_size() != 0) and retrieves the filename using getfile\_name().

**2**-Processing File Content:

**.** Opens a file (File myObj = new File(file)) and reads it line by line (while (myReader.hasNextLine())).

**.** Splits each line into words (String[] childreen = data.split(" ")).

**.** Counts occurrences of specific words (is, are, you) and finds the longest and shortest words within each line.

**.** Creates an ArrayList<String> called uptodate to store file-related data like filename, directory name, longest word, shortest word, word counts, and occurrences of specific words.

**.** Updates the uptodate list and sends this information to Start.synch\_uptodate(uptodate) (which seems to synchronize this data elsewhere in the program).

1. **Threading:**

Utilizes Thread.sleep(slp) to introduce a delay between processing lines of a file. This can be useful in controlling the processing speed.

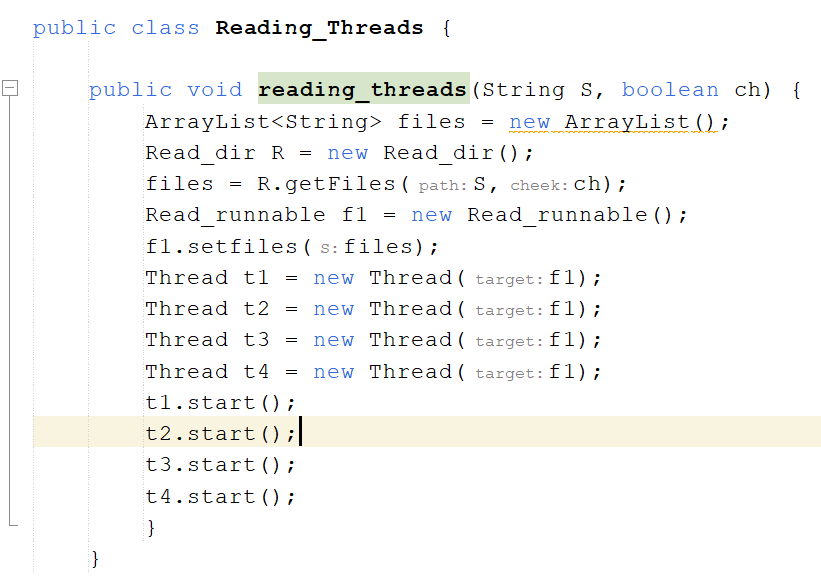
**4-Exception Handling:**

Properly handles file-related exceptions (FileNotFoundException).

**The reading threads functions:**

reading\_threads():

orchestrate the multi-threaded reading of files.



1**-reading\_threads Method:**

Takes a directory path (S) and a boolean value (ch) indicating whether to include subdirectories.

Initializes an ArrayList<String> called files.

Creates an instance of Read\_dir called R and uses it to obtain a list of files by invoking the getFiles method with the provided directory path and the boolean flag indicating subdirectories inclusion.

Initializes a Read\_runnable instance f1 and sets the obtained files using the setfiles method.

Creates four threads (t1, t2, t3, t4) and assigns the same Read\_runnable instance f1 to all of them.

Starts these threads concurrently.

2-**Thread Concurrency:**

It appears that the intention is to concurrently process the files using four threads (t1, t2, t3, t4), all sharing the same Read\_runnable instance f1 and thus the same list of files (files).

3**-Error Handling:**

There's no explicit error handling within this method. If exceptions occur during file reading or thread execution, they might not be handled within this method.

4-**Logging:**

There's a logger import, but it's not used within this method.

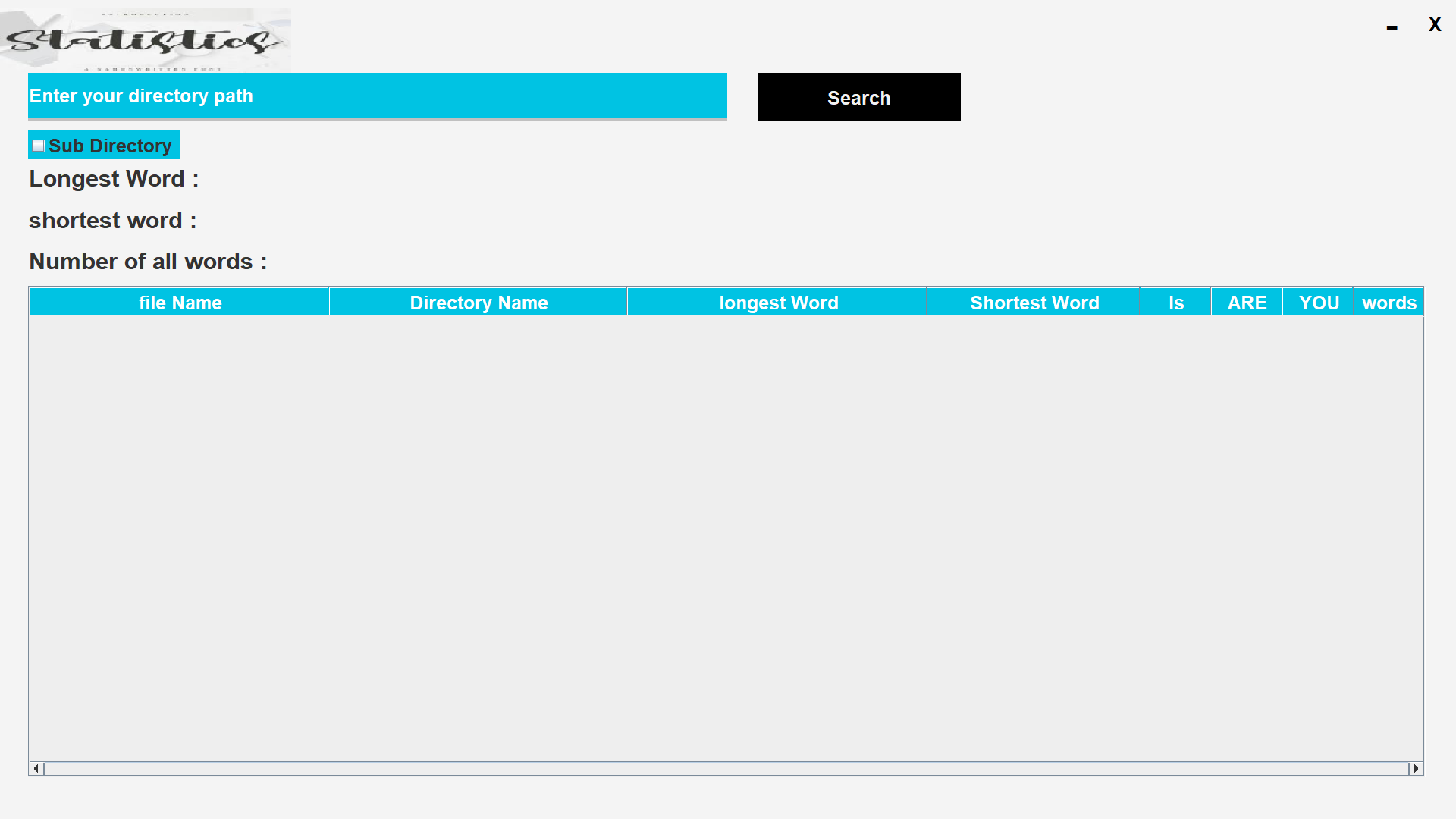
**5-Thread Synchronization:**

There doesn't seem to be explicit synchronization or coordination of these threads' execution.

6-**Resource Management:**

It creates multiple threads for file reading, which could be beneficial for handling multiple files concurrently.

**The last thing is our Graphical user interface(GUI):**



===>If we didn’t put any directory **OR** a wrong path to the directory in the directory path there will be a message error

