A close-up of a white background

Description automatically generated

**Operating System-2 (Project Documentation)**

**1-Project Description:**

The Word Statistics project aims to provide a user-friendly interface for analyzing word statistics in text files within a selected directory. The application includes a graphical user interface (GUI), methods for word analysis, and utilizes multithreading for efficient processing, so the user can choose the directory by browsing it and can include it’s subdirectories through the (GUI) and the results will appears on the table by the word statistics (number of words per file/directory , longest word , shortest word , number of ‘is’, ‘are’ , ‘you’ ).

**2- What we actually did:**

It took us a while to figure out what to do in this project, it wasn’t easy, but we did it, we found a way of cooperating with one another as a team and we did the project exactly as it was described, we used object oriented programming and Java threads and the concept of the multithreading to implement this project, we did (GUI) represents the main application window , browse button for directory selection , table for displaying word statistics, labels for directory, longest and shortest words, checkbox for including subdirectories,

JnaFileChose, that manages directory selection, TableTextCenterClass, that custom table for centered text and extends Jtable, and we did Word Statistics Method that get the files and the files with subfolders and count the file words, file longest word and multithreading for concurrent processing by doing multiple threads, each thread runs an instance of word statistics, and explanation of word statistics calculation and GUI updates.

**3-Team Members Role:**

GUI: Abdelrahman Mahmoud, Abdalla Samy.

Methods: Mariam Ahmed, Mira Hesham, Warda Khaled, Fatma Elzahra Ebrahim.

Threads: Momen Mahmoud.

Documentation: All of us participate in it.

**4-Code Documentation:**

We have two classes main and word statistics , let’s go through the word statistics class:-

A screenshot of a computer

Description automatically generated

This Java code, within the "Momen" package, imports classes for file handling (BufferedReader, File, FileReader), exception handling (IOException), data structures (ArrayList, List), logging (Level, Logger), and Swing GUI development (DefaultTableModel). These imports facilitate efficient text processing and table creation in a Swing application.

A screenshot of a computer

Description automatically generated

The WordStatistics class is designed to analyze word statistics in text files. It initializes with a main folder, Swing components, and a boolean indicating whether to include subdirectories. It checks if the main folder is a directory, retrieves files accordingly, and sets up the table for displaying results in a Swing GUI.

A screenshot of a computer

Description automatically generated

The getFiles method collects all files from the main folder, adding them to innerFilesResult if they are individual files (not directories).

The getFilesWithSubfolders method recursively explores files within a given folder up to a specified depth level. It adds text files to innerFilesResult and continues into subdirectories, maintaining the depth level.

A screenshot of a computer

Description automatically generated

The getInnerFilesResult method returns an array of files from the innerFilesResult list.

The isTxtFile method checks if a given file has a ".txt" extension.

The countFileWords method reads a file line by line, counts words, and updates a Swing table cell with the word count. It includes a delay for visual effect during concurrent processing.

A screenshot of a computer

Description automatically generated

The findLongestWord method analyzes a file to identify and display the longest word. It iterates through lines, splits them into words, and compares lengths. The method updates a Swing table cell with the longest word dynamically, introducing delays for visual effect. Additionally, it updates a label with the overall longest word in the directory.

A screenshot of a computer

Description automatically generated

A white background with black text

Description automatically generated

The findShortestWord method examines a file to identify and display the shortest word. It iterates through lines, splits them into words, compares lengths, and dynamically updates a Swing table cell. The method also updates a label with the overall shortest word in the directory.

The countWordOccurrences method tallies the occurrences of a specific word in a file, updating a Swing table cell with the count. Both methods introduce delays for visual effects during concurrent processing.

A screenshot of a computer

Description automatically generated

The getShortestWordInDirectory and getLongestWordInDirectory methods provide access to the shortest and longest words found in the analyzed files' directory.

The pop method removes and returns the first file from the list of files for processing in a thread-safe manner.

The rowNumber method increments and returns a synchronized row number, facilitating unique row assignments during concurrent processing in a multithreaded environment.

A screen shot of a computer

Description automatically generated

The run method, part of the Runnable interface, orchestrates the concurrent processing of files. It continuously processes files from the list, assigning each to a thread with a unique row number. The method updates a Swing table with information such as file name, word count, word occurrences, longest and shortest words, introducing delays for visual effects during concurrent execution.

* Let’s go through in the main class:

A screenshot of a computer

Description automatically generated

The Main class represents the main application window. It extends javax. swing.JFrame and includes a GUI created with NetBeans. The code sets up the window components, such as buttons and labels, using the NetBeans-generated code. It also handles events and interactions in GUI. The class utilizes the JnaFileChooser library for directory selection and incorporates the WordStatistics class for word statistics analysis. The code is generated by NetBeans and should not be modified in certain sections to maintain functionality.

A screenshot of a computer

Description automatically generated

The jButton1ActionPerformed method handles the action when the "Browse" button is clicked. It opens a file chooser dialog to select a directory, updates GUI labels, initializes a WordStatistics object for word analysis, and starts multiple threads for concurrent file processing.

The jCheckBox1ActionPerformed method is an event handler for the checkbox. Currently, it doesn't contain specific handling code, and it's marked with a comment indicating that additional functionality can be added as needed.

A screenshot of a computer

Description automatically generated

The main method initializes the GUI application, sets the look and feel (Nimbus) for the user interface, and then creates an instance of the Main class, making the application visible to the user. The code also includes the declaration of various GUI components, such as buttons, checkboxes, labels, and a table.

**Now this is the status of our GUI it’s view and without checking the include subdirectories button and with:**

**A screenshot of a computer

Description automatically generated**

**A screenshot of a computer

Description automatically generated**

**A screenshot of a computer

Description automatically generated**