Data Structures & Algorithms – Theory

Project Report

Name	Student ID#
Huzefa Saifuddin	22K-5125
Ruhaan Ahmed	22K-6014
Ijlal Iqbal	22K-5034

Contents

Report on the Text Editor in C++

Introduction

The code under review is a simple text editor written in C++. The text editor provides basic file operations such as creating, reading, updating, and deleting files. It also supports undo and redo operations, copy, paste, and cut operations.

Code Structure and Functionality

The code is structured around a main `menu` function that displays a menu of operations and performs the selected operation. The operations are performed by calling other functions such as `undo`, `redo`, `copyFile`, `pasteFile`, and `cutFile`.

Undo and Redo Operations

The `undo` function undoes the last operation performed on the file by popping the last operation from the `undoStack` and pushing it to the `redoStack`. The `redo` function does the opposite: it pops the last operation from the `redoStack` and pushes it to the `undoStack`. These operations are implemented using stack data structures, which are a type of abstract data type that follows a LIFO (Last In, First Out) principle.

Copy, Paste, and Cut Operations

The `copyFile`, `pasteFile`, and `cutFile` functions perform the copy, paste, and cut operations, respectively. The `copyFile` function reads the content of a file and pushes it to the `copyStack`. The `pasteFile` function pops the content from the `copyStack` and appends it to the file. The `cutFile` function removes all content from the file. These operations are implemented using file I/O operations, which allow the program to read from and write to files.

Menu Function

The `menu` function is the main function of the program. It displays the menu of operations and performs the selected operation. It uses a switch statement to select the operation based on the user's input. The `menu` function also handles the display of messages to the user, such as *"File Created Successfully"*, *"File Updated Successfully"*, *"File Emptied Successfully", "File Deleted Successfully"*, and *"File Not Found"*. These messages are displayed based on the value of the `msg` parameter passed to the `menu` function.

Code Presentation

The code is well-structured and easy to understand. It follows good programming practices such as using meaningful variable names, using comments to explain what the code does, and organizing the code into functions that each perform a specific task. The code also uses the `iostream` library for input and output operations, the `fstream` library for file operations, and the `stack` library for stack operations.

Conclusion

The text editor in C++ provides a simple yet functional interface for basic file operations. It demonstrates the use of stack data structures for undo and redo operations, and file I/O operations for copy, paste, and cut operations. The code is well-structured and easy to understand, making it a good example of a text editor in C++.