21MIS1001 06-11-2024

Sabbarish S Wednesday

SWE-4001

System Programming

Lab 10 - Text Editor

**AIM:**

To implement a simple text editor with features like insertion / deletion of a character, word, and sentence.

#include <iostream>

#include <fstream>  // For file operations

#include <conio.h>  // For \_getch() and other console I/O on Windows

#include <string>

#include <vector>

using namespace std;

vector<*string*> buffer; // To store lines of text

int curx = 0, cury = 0; // Cursor position

void displayBuffer() {

    system("cls"); // Clear the screen (for Windows)

    for (int i = 0; i < buffer.size(); ++i) {

        cout << buffer[i] << endl; // Print each line in the buffer

    }

    cout << "> ";

    for (int i = 0; i < curx; i++) cout << " ";

    cout << "^" << endl;  // Display cursor as ^

}

void insertChar(char *ch*) {

    if (curx == buffer[cury].size()) {

        buffer[cury] += *ch*; // Append character at the end

    } else {

        buffer[cury].insert(curx, 1, *ch*); // Insert character at curx position

    }

    curx++;

}

void deleteChar() {

    if (curx > 0) {

        buffer[cury].erase(--curx, 1); // Delete character at curx-1 position

    } else if (cury > 0) { // If at the beginning of the line, merge with previous line

        curx = buffer[cury - 1].size();

        buffer[cury - 1] += buffer[cury];

        buffer.erase(buffer.begin() + cury);

        cury--;

    }

}

void handleInput() {

    char ch;

    while ((ch = \_getch()) != 27) { // ESC to exit

        switch (ch) {

            case 72: // UP arrow

                if (cury > 0) cury--;

                curx = min(curx, (int)buffer[cury].size());

                break;

            case 80: // DOWN arrow

                if (cury < buffer.size() - 1) cury++;

                curx = min(curx, (int)buffer[cury].size());

                break;

            case 75: // LEFT arrow

                if (curx > 0) curx--;

                else if (cury > 0) {

                    cury--;

                    curx = buffer[cury].size();

                }

                break;

            case 77: // RIGHT arrow

                if (curx < buffer[cury].size()) curx++;

                else if (cury < buffer.size() - 1) {

                    cury++;

                    curx = 0;

                }

                break;

            case 8: // BACKSPACE

                deleteChar();

                break;

            case 13: // ENTER key

                buffer.insert(buffer.begin() + cury + 1, buffer[cury].substr(curx));

                buffer[cury] = buffer[cury].substr(0, curx);

                curx = 0;

                cury++;

                break;

            default:

                if (isprint(ch)) insertChar(ch);

                break;

        }

        displayBuffer();

    }

}

void saveToFile(const *string* &*filename*) {

*ofstream* file(*filename*);

    if (file.is\_open()) {

        for (const *string* &line : buffer) {

            file << line << endl;

        }

        file.close();

        cout << "Content saved to " << *filename* << endl;

    } else {

        cerr << "Error: Could not open file " << *filename* << " for writing." << endl;

    }

}

int main() {

*string* filename;

    cout << "Enter filename to save the text: ";

    getline(cin, filename); // Get the filename from the user

    buffer.push\_back(""); // Start with an empty line in the buffer

    displayBuffer();

    handleInput();

    saveToFile(filename); // Save content to file when exiting

    return 0;

}

Output:



