Summer Internship Training

(June 2025 - July 2025)

on

Fundamental of Data Structure using C++

Submitted by

Name of Student 1 : Vikas Yadav	**Registration Number : 12308095**
Name of Student 2 : Navneet Kumar	**Registration Number: 12320763**
Name of Student 3 : Riya	**Registration Number : 12316877**
Name of Student 4 : Samridhi Raturi	**Registration Number : 12319946**

Under the Guidance of

Dr. Om Prakash Yadav
Associate Professor

School of Computer Science and Engineering
Lovely Professional University

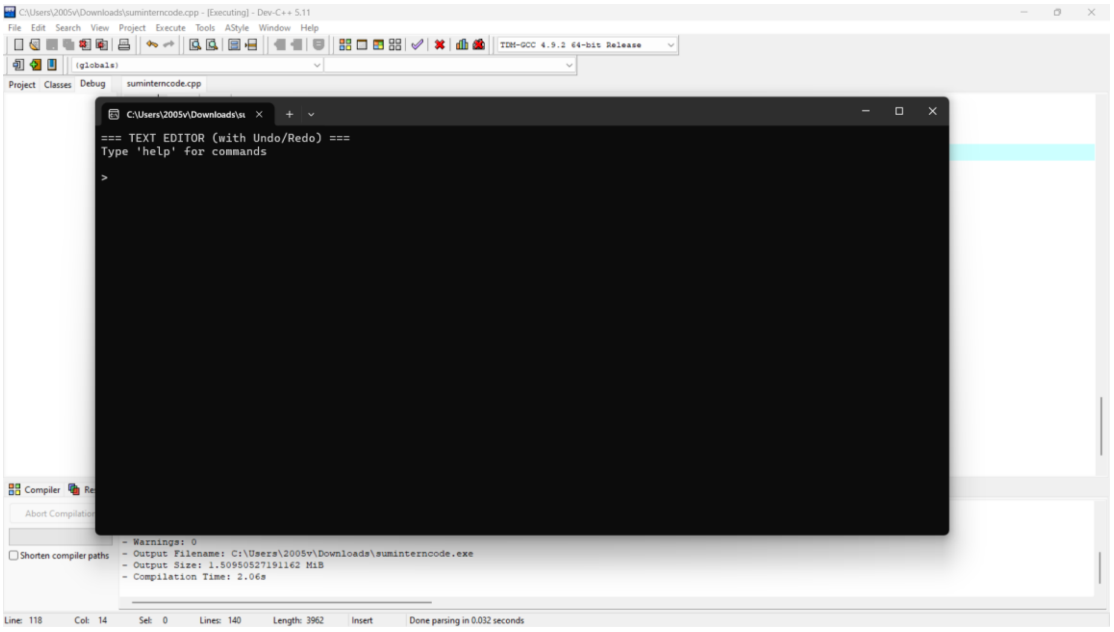
Jalandhar - Delhi G.T. Road, Phagwara, Punjab (India) - 144411

LinkedIn Link : https://www.linkedin.com/in/samridhiraturi18/s

Project Code : (with white background only)

```
#include <iostream>
#include <string>
#include <vector>

#define MAX_HISTORY_SIZE 1000

using namespace std;  // ? Added this

class TextAction {
private:
    string content;

public:
    TextAction(const string& text) : content(text) {}
    string getText() const { return content; }
};

class TextEditor {
private:
    string currentText;
    vector<TextAction> undoStack;
    vector<TextAction> redoStack;

    void saveState() {
        if (undoStack.size() >= MAX_HISTORY_SIZE) {
            undoStack.erase(undoStack.begin());
        }
        undoStack.push_back(TextAction(currentText));
        redoStack.clear(); // Clear redo stack when a new action is performed
    }

public:
    void type(const string& text) {
        if (text.empty()) {
            cout << "Error: Cannot type empty text\n";
            return;
        }
        saveState();
        currentText += text;
        cout << "Added text: " << text << "\n";
    }

    void undo() {
        if (undoStack.empty()) {
            cout << "Nothing to undo!\n";
            return;
        }
        redoStack.push_back(TextAction(currentText));
        currentText = undoStack.back().getText();
        undoStack.pop_back();
        cout << "Undo successful. Current text: " << currentText << "\n";
    }

    void redo() {
        if (redoStack.empty()) {
            cout << "Nothing to redo!\n";
            return;
        }
        undoStack.push_back(TextAction(currentText));
        currentText = redoStack.back().getText();
        redoStack.pop_back();
        cout << "Redo successful. Current text: " << currentText << "\n";
    }
    void display() const {
        cout << "\nCurrent text content:\n" << currentText << "\n";
    }

    void clear() {
        saveState();
        currentText.clear();
        cout << "Editor cleared.\n";
    }
};

void showHelp() {
    cout << "\nTEXT EDITOR COMMANDS:\n";
    cout << "type <text>  - Add text to editor\n";
    cout << "undo         - Undo last action\n";
    cout << "redo         - Redo last undone action\n";
    cout << "display      - Show current text\n";
    cout << "clear        - Clear all text\n";
    cout << "help         - Show this help\n";
    cout << "exit         - Quit editor\n";
}

int main() {
    TextEditor editor;
    string input;
    string command, text;

    cout << "=== TEXT EDITOR (Pure C++ with Undo/Redo) ===\n";
    cout << "Type 'help' for commands\n";

    while (true) {
        cout << "\n> ";
        getline(cin, input);
        // Separate command and text
        size_t spacePos = input.find(' ');
        if (spacePos != string::npos) {
            command = input.substr(0, spacePos);
            text = input.substr(spacePos + 1);
        } else {
            command = input;
            text.clear();
        }

        if (command == "type") {
            editor.type(text);
        } else if (command == "undo") {
            editor.undo();
        } else if (command == "redo") {
            editor.redo();
        } else if (command == "display") {
            editor.display();
        } else if (command == "clear") {
            editor.clear();
        } else if (command == "help") {
            showHelp();
        } else if (command == "exit") {
            cout << "Exiting editor. Goodbye!\n";
            break;
        } else {
            cout << "Invalid command. Type 'help' for options.\n";
        }
    }

    return 0;
}
```

Screenshot : 3 to 5 screenshot during execution of code

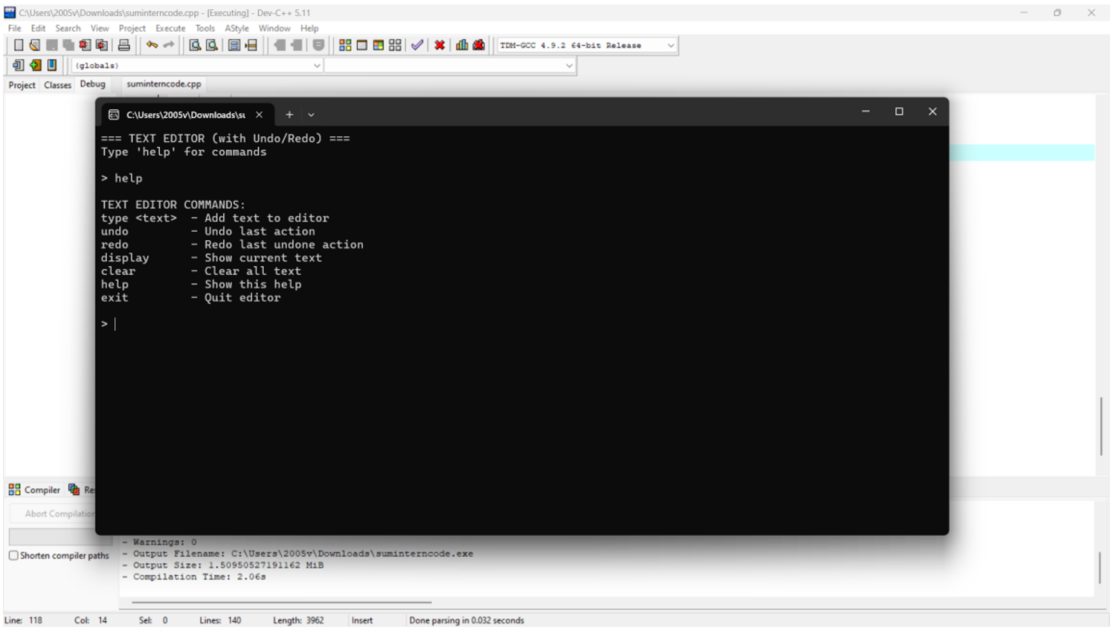


```
=== TEXT EDITOR (with Undo/Redo) ===
Type 'help' for commands

>
```

```
- Warnings: 0
- Output Filename: C:\Users\2005v\Downloads\suminterncode.exe
- Output Size: 1.50950527191162 MiB
- Compilation Time: 2.06s
```

Line: 118 Col: 14 Sel: 0 Lines: 140 Length: 3962 Insert Done parsing in 0.032 seconds

(1)

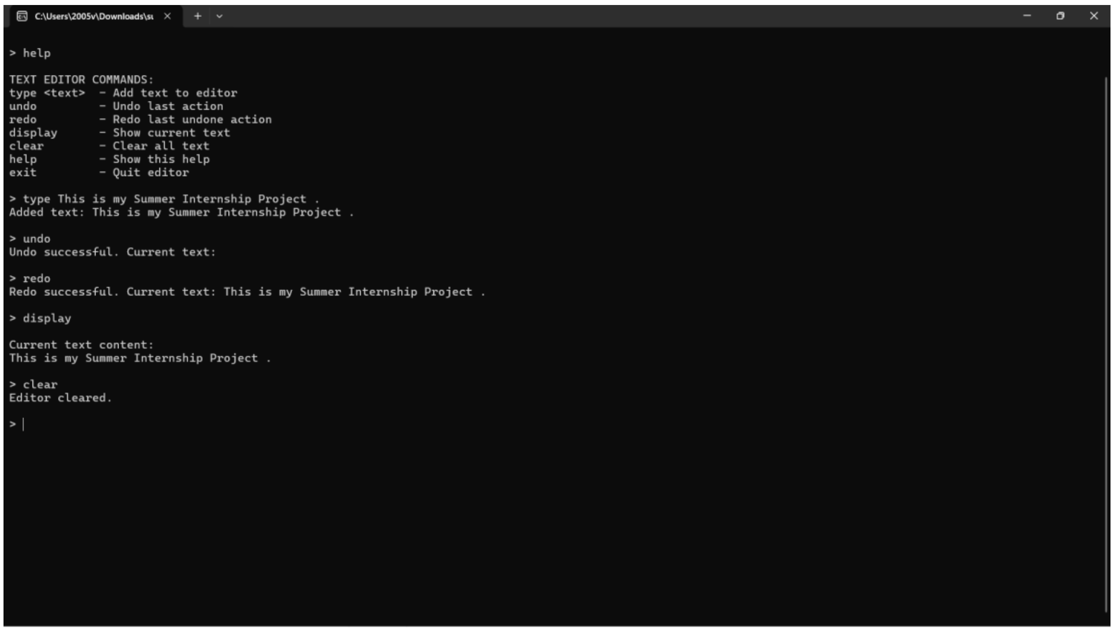


```
=== TEXT EDITOR (with Undo/Redo) ===
Type 'help' for commands

> help

TEXT EDITOR COMMANDS:
type <text>  - Add text to editor
undo         - Undo last action
redo         - Redo last undone action
display      - Show current text
clear        - Clear all text
help         - Show this help
exit         - Quit editor

> |
```

```
- Warnings: 0
- Output Filename: C:\Users\2005v\Downloads\suminterncode.exe
- Output Size: 1.50950527191162 MiB
- Compilation Time: 2.06s
```

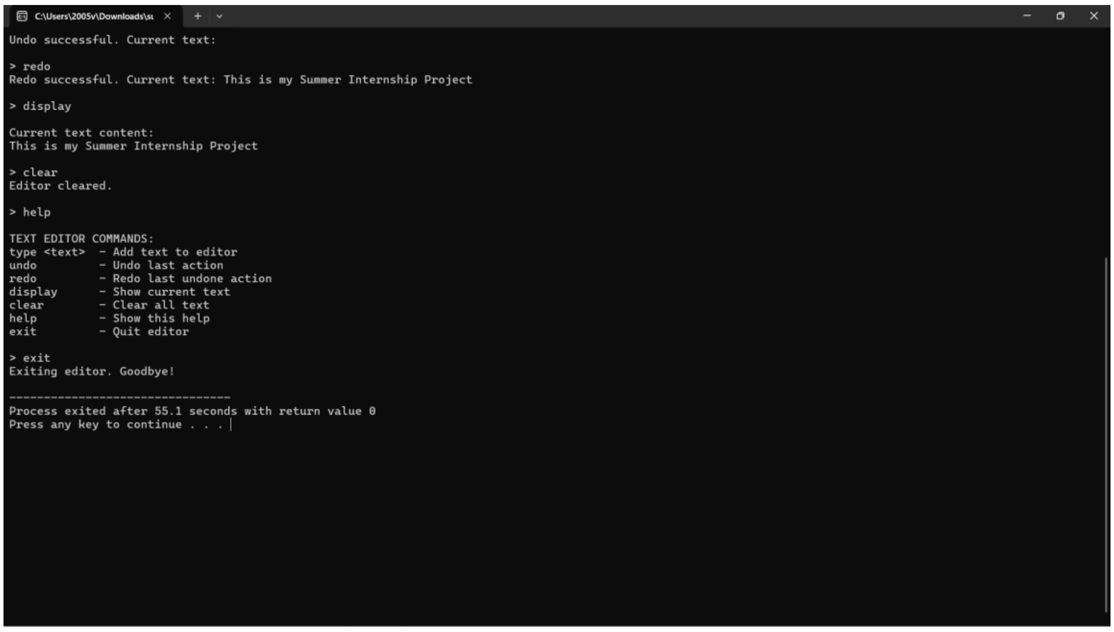
Line: 118 Col: 14 Sel: 0 Lines: 140 Length: 3962 Insert Done parsing in 0.032 seconds

(2)



```
> help

TEXT EDITOR COMMANDS:
type <text>  - Add text to editor
undo         - Undo last action
redo         - Redo last undone action
display      - Show current text
clear        - Clear all text
help         - Show this help
exit         - Quit editor

> type This is my Summer Internship Project .
Added text: This is my Summer Internship Project .

> undo
Undo successful. Current text:

> redo
Redo successful. Current text: This is my Summer Internship Project .

> display

Current text content:
This is my Summer Internship Project .

> clear
Editor cleared.

> |
```

(3)



```
Undo successful. Current text:

> redo
Redo successful. Current text: This is my Summer Internship Project

> display

Current text content:
This is my Summer Internship Project

> clear
Editor cleared.

> help

TEXT EDITOR COMMANDS:
type <text>  - Add text to editor
undo         - Undo last action
redo         - Redo last undone action
display      - Show current text
clear        - Clear all text
help         - Show this help
exit         - Quit editor

> exit
Exiting editor. Goodbye!

--------------------------------
Process exited after 55.1 seconds with return value 0
Press any key to continue . . . |
```

(4)