Deklarasi Library dan Namespace

```
#include <iostream>
#include <stack>
#include <string>

using namespace std;
```

- `#include <iostream>`: Digunakan untuk input*output standar dalam C++.
- `#include <stack>`: Digunakan untuk menggunakan struktur data stack yang tersedia di C++ STL (Standard Template Library).
- `#include <string>`: Digunakan untuk tipe data string.
- `using namespace std;`: Digunakan untuk mempermudah penggunaan objek dan fungsi yang terdapat dalam namespace `std`.

Fungsi `clearScreen()`

```
void clearScreen() {
    cout << "\033[2J\033[1;1H";
}
```

Fungsi ini menggunakan ANSI escape sequence untuk membersihkan layar konsol dan menggeser kursor ke posisi (1,1). Escape sequence `\033[2J` membersihkan layar dan `\033[1;1H` menggeser kursor ke baris 1, kolom 1.

Fungsi `typeText`

```
void typeText(string& currentText, stack<string>& undoStack, stack<string>&
redoStack) {
    cout << "Masukkan teks yang ingin ditambahkan: ";
    string newText;
    getline(cin, newText);

    undoStack.push(currentText); // Menyimpan teks saat ini dalam undoStack
    if (!currentText.empty()) {
        currentText += ", ";
    }
    currentText += newText;

    redoStack = stack<string>(); // Mengosongkan redoStack
}
```

- Fungsi ini meminta pengguna untuk memasukkan teks baru (`newText`) dan menambahkannya ke dalam `currentText`.
- `undoStack.push(currentText)`: Menyimpan `currentText` saat ini di `undoStack` agar nantinya dapat di*“undo”.

- Jika `currentText` tidak kosong, fungsi ini menambahkan koma dan spasi sebelum menambahkan `newText` untuk memisahkan antara teks yang sudah ada dengan teks baru.
- `redoStack = stack<string>();`: Mengosongkan `redoStack` karena setiap operasi baru yang dilakukan akan menghapus operasi redo yang sebelumnya.

Fungsi performUndo

```
void performUndo(string& currentText, stack<string>& undoStack,
stack<string>& redoStack) {
    if (!undoStack.empty()) {
        redoStack.push(currentText); // Menyimpan teks saat ini di redoStack
        currentText = undoStack.top(); // Mengembalikan teks sebelumnya dari
undoStack
        undoStack.pop();
        cout << "Undo berhasil dilakukan." << endl;
    } else {
        cout << "Tidak ada operasi undo yang dapat dilakukan." << endl;
    }
}
```

- Fungsi ini melakukan operasi undo dengan mengambil teks sebelumnya dari `undoStack` dan memindahkannya ke `currentText`.
- `redoStack.push(currentText)`: Menyimpan `currentText` saat ini di `redoStack` untuk memungkinkan operasi redo nantinya.
- Jika `undoStack` kosong, fungsi akan memberikan pesan bahwa tidak ada operasi undo yang dapat dilakukan.

Fungsi performRedo

```
void performRedo(string& currentText, stack<string>& undoStack,
stack<string>& redoStack) {
    if (!redoStack.empty()) {
        undoStack.push(currentText); // Menyimpan teks saat ini di undoStack
        currentText = redoStack.top(); // Mengembalikan teks yang dibatalkan
dari redoStack
        redoStack.pop();
        cout << "Redo berhasil dilakukan." << endl;
    } else {
        cout << "Tidak ada operasi redo yang dapat dilakukan." << endl;
    }
}
```

- Fungsi ini melakukan operasi redo dengan mengambil teks yang telah di*“undo” sebelumnya dari `redoStack` dan memindahkannya kembali ke `currentText`.
- `undoStack.push(currentText)`: Menyimpan `currentText` saat ini di `undoStack` untuk memungkinkan operasi undo nantinya.
- Jika `redoStack` kosong, fungsi akan memberikan pesan bahwa tidak ada operasi redo yang dapat dilakukan.

Fungsi checkEmptyUndo dan checkEmptyRedo

```
void checkEmptyUndo(stack<string>& undoStack) {
    cout << "IsEmptyUndo: " << (undoStack.empty() ? "true" : "false") <<
endl;
}

void checkEmptyRedo(stack<string>& redoStack) {
    cout << "IsEmptyRedo: " << (redoStack.empty() ? "true" : "false") <<
endl;
}
```

- `checkEmptyUndo`: Memeriksa apakah `undoStack` kosong dan mencetak hasilnya. Jika ada data yang di undo maka nilai yg keluar adalah `False`, begitupun sebaliknya maka nilai yang keluar adalah `True`.
- `checkEmptyRedo`: Memeriksa apakah `redoStack` kosong dan mencetak hasilnya. Jika ada data yang di redo maka nilai yang keluar adalah `Flse`, begitupun sebaliknya maka nilai yang keluar adalah `True`.

Fungsi main()

```
int main() {
    // Deklarasi variabel dan stack
    stack<string> undoStack;
    stack<string> redoStack;
    string currentText;

    // Loop utama untuk menampilkan menu dan memproses input pengguna
    while (true) {
        clearScreen(); // Membersihkan layar konsol
        cout << "Teks saat ini: " << currentText << endl;
        cout << "1. Ketik teks" << endl;
        cout << "2. Undo" << endl;
        cout << "3. Redo" << endl;
        cout << "4. IsEmptyUndo" << endl;
        cout << "5. IsEmptyRedo" << endl;
        cout << "6. Keluar" << endl;
        cout << "Masukkan pilihan anda: ";

        int choice;
        cin >> choice;
        cin.ignore();

        switch (choice) {
            case 1:
                typeText(currentText, undoStack, redoStack);
                break;
            case 2:
                performUndo(currentText, undoStack, redoStack);
                break;
            case 3:
```

```
                performRedo(currentText, undoStack, redoStack);
                break;
            case 4:
                checkEmptyUndo(undoStack);
                break;
            case 5:
                checkEmptyRedo(redoStack);
                break;
            case 6:
                cout << "Terima kasih telah menggunakan program ini. Sampai
jumpa!" << endl;
                return 0;
            default:
                cout << "Pilihan tidak valid. Silakan coba lagi." << endl;
                break;
        }

        cout << "\nTekan Enter untuk melanjutkan...";
        cin.get();
    }

    return 0;
}
```

- `main()`: Berisi loop utama yang menampilkan menu dan memproses input pengguna. Setiap pilihan menu akan memanggil fungsi yang sesuai untuk melakukan operasi tambah teks, undo, redo, memeriksa kekosongan undoStack atau redoStack, atau keluar dari program.
- Pengguna diminta untuk menekan Enter setelah setiap operasi agar program dapat melanjutkan ke langkah berikutnya dalam loop.