PIMPRI CHINCHWAD EDUCATION TRUST's.

PIMPRI CHINCHWAD COLLEGE OF ENGINEERING

(An Autonomous Institute)

Class : SY BTech **Acad. Yr. 2025-26** **Semester : I**

Name of the student: Hariom Shrikrishna Gundale **PRN : 124B1B036**

Department: Computer Engineering **Division : A**

Course Name : Data Structures Laboratory **Course Code:BCE23PC02**

Completion Date : 26/10/2025

October Challenge Assignment

Problem Statement: "Browser History Management"

You are tasked with implementing a simplified browser history management system. This system should allow users to navigate between previously visited pages, and also add new pages to their history. The system needs to efficiently handle both "back" and "forward" navigation, as well as adding new pages, while maintaining a limited history size.

Requirements:

- When a user visits a new page, it should be added to the current history. If there are any "forward" pages available (i.e., pages visited after the current page in the history), they should be cleared, as visiting a new page effectively creates a new branch in the history.
- Users should be able to navigate to the previous page in their history. If there's no previous page, they remain on the current page.
- Users should be able to navigate to a page they previously went "back" from. If there's no page to go forward to, they remain on the current page.
- The browser history should have a maximum capacity. If adding a new page exceeds this capacity, the oldest page in the history should be automatically removed to make space.

Input:

A sequence of operations:

- visit(url): Adds url to the history.
- back(steps): Navigates back by steps pages.
- forward(steps): Navigates forward by steps pages.

Output:

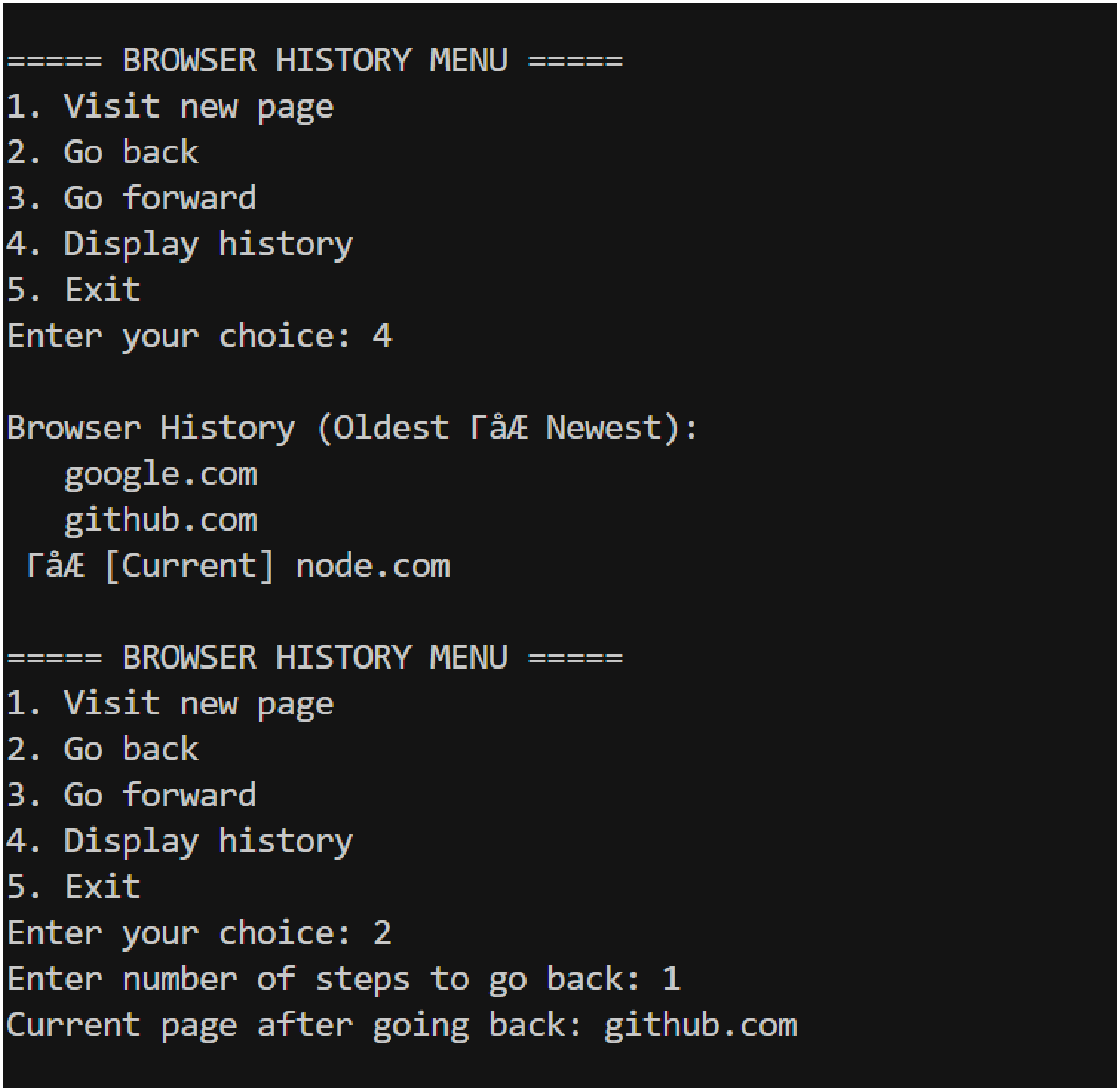
For each visit, back, or forward operation, output the URL of the page the user is currently on after the operation.

Source Code:

```
#include <iostream>
#include <deque>
#include <string>
using namespace std;

class BrowserHistory {
private:
    deque<string> history;
    int current;
    int capacity;

public:
    BrowserHistory(int cap) {
        capacity = cap;
        current = -1;
    }


    void visit(const string& url) {

        while ((int)history.size() > current + 1)
            history.pop_back();


        history.push_back(url);
        current++;


        if ((int)history.size() > capacity) {
            history.pop_front();
            current--;
        }

        cout << "Visited: " << history[current] << endl;
    }
    void back(int steps) {
        if (current == -1) {
            cout << "No pages visited yet!" << endl;
            return;
        }

        current = max(0, current - steps);
        cout << "Current page after going back: " << history[current] << endl;
    }

    void forward(int steps) {
        if (current == -1) {
            cout << "No pages visited yet!" << endl;
            return;
        }

        current = min((int)history.size() - 1, current + steps);
        cout << "Current page after going forward: " << history[current] << endl;
    }

    void display() {
        if (history.empty()) {
            cout << "History is empty!\n";
            return;
        }

        cout << "\nBrowser History (Oldest -> Newest):\n";
        for (int i = 0; i < (int)history.size(); i++) {
            if (i == current)
                cout << " -> [Current] " << history[i] << endl;
            else
                cout << "    " << history[i] << endl;
        }
    }
};

int main() {
```

```
int capacity, choice, steps;
string url;

cout << "Enter maximum browser history capacity: ";
cin >> capacity;

BrowserHistory browser(capacity);

while (true) {
    cout << "\n===== BROWSER HISTORY MENU =====\n";
    cout << "1. Visit new page\n";
    cout << "2. Go back\n";
    cout << "3. Go forward\n";
    cout << "4. Display history\n";
    cout << "5. Exit\n";
    cout << "Enter your choice: ";
    cin >> choice;

    switch (choice) {
    case 1:
        cout << "Enter URL to visit: ";
        cin >> url;
        browser.visit(url);
        break;

    case 2:
        cout << "Enter number of steps to go back: ";
        cin >> steps;
        browser.back(steps);
        break;

    case 3:
        cout << "Enter number of steps to go forward: ";
        cin >> steps;
        browser.forward(steps);
        break;

    case 4:
            browser.display();
            break;

        case 5:
            cout << "Exiting program...\n";
            return 0;

        default:
            cout << "Invalid choice! Try again.\n";
        }
    }

    return 0;
}
```

Screen Shot of Output :



```
===== BROWSER HISTORY MENU =====
1. Visit new page
2. Go back
3. Go forward
4. Display history
5. Exit
Enter your choice: 4

Browser History (Oldest ΓåÆ Newest):
   google.com
   github.com
 ΓåÆ [Current] node.com

===== BROWSER HISTORY MENU =====
1. Visit new page
2. Go back
3. Go forward
4. Display history
5. Exit
Enter your choice: 2
Enter number of steps to go back: 1
Current page after going back: github.com
```

Conclusion:

Thus, we have successfully implemented the C++ Program for Browser Management Sysytem Using Double Ended Queue