



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:** Samruddhi Ramswami Bansode.**PRN : 124B1B019****Department:** Computer Engineering**Division : A****Course Name :** Data Structures**Course Code:** BCE23PC02**Completion Date :22/10/2025**

Practice 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> dq;
    int current;
    int cap;

public:
    BrowserHistory(string homepage, int cap) {
        dq.push_back(homepage);
        current = 0;
        cap = cap;
    }

    string visit(string url) {
        // remove all forward pages
        while (dq.size() > current + 1)
            dq.pop_back();

        // add new page
        dq.push_back(url);
        current++;

        // limit capacity
        if (dq.size() > cap) {
            dq.pop_front();
            current--;
        }

        return dq[current];
    }
}
```

```
string back(int steps) {
    if (current - steps < 0)
        current = 0;
    else
        current -= steps;
    return dq[current];
}

string forward(int steps) {
    if (current + steps >= dq.size())
        current = dq.size() - 1;
    else
        current += steps;
    return dq[current];
}

string getCurrentPage() {
    return dq[current];
}

void showHistory() {
    cout << "\nHistory: ";
    for (int i = 0; i < dq.size(); i++) {
        if (i == current)
            cout << "[" << dq[i] << "]" ";
        else
            cout << dq[i] << " ";
    }
    cout << endl;
}

};

int main() {
    int cap;
    cout << "Enter maximum history size: ";
    cin >> cap;

    BrowserHistory browser("home.com", cap);

    int choice, steps;
```

```
string url;
cout << "\n==== Browser Menu =====\n";
cout << "1. Visit new page\n";
cout << "2. Go back\n";
cout << "3. Go forward\n";
cout << "4. Show current page\n";
cout << "5. Show full history\n";
cout << "6. Exit\n";
do {
    cout << "Enter your choice: ";
    cin >> choice;

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

    case 2:
        cout << "Enter steps to go back: ";
        cin >> steps;
        cout << "Current Page: " << browser.back(steps) << endl;
        browser.showHistory();
        break;

    case 3:
        cout << "Enter steps to go forward: ";
        cin >> steps;
        cout << "Current Page: " << browser.forward(steps) << endl;
        browser.showHistory();
        break;

    case 4:
        cout << "Current Page: " << browser.getCurrentPage() << endl;
        break;

    case 5:
        browser.showHistory();
```

```
        break;

    case 6:
        cout << "Exiting Browser..." << endl;
        break;

    default:
        cout << "Invalid choice!" << endl;
    }

    } while (choice != 6);

    return 0;
}
```

Screen Shot of Output :

```
PS C:\C++ DSA> g++ p_assignment.cpp
PS C:\C++ DSA> ./a.exe
Enter maximum history size: 5

===== Browser Menu =====
1. Visit new page
2. Go back
3. Go forward
4. Show current page
5. Show full history
6. Exit
Enter your choice: 1
Enter URL: www.google
Current Page: www.google

History: home.com [www.google]
Enter your choice: 1
Enter URL: www.leetcode
Current Page: www.leetcode

History: home.com www.google [www.leetcode]
Enter your choice: 1
Enter URL: www.classroom
Current Page: www.classroom

History: home.com www.google www.leetcode [www.classroom]
Enter your choice: 1
Enter URL: www.youtube
Current Page: www.youtube

History: home.com www.google www.leetcode www.classroom [www.youtube]
```

```
Enter your choice: 2
Enter steps to go back: 2
Current Page: www.leetcode

History: home.com www.google [www.leetcode] www.classroom www.youtube
Enter your choice: 3
Enter steps to go forward: 1
Current Page: www.classroom

History: home.com www.google www.leetcode [www.classroom] www.youtube
Enter your choice: 4
Current Page: www.classroom
Enter your choice: 5

History: home.com www.google www.leetcode [www.classroom] www.youtube
Enter your choice: 6
Exiting Browser...
PS C:\C++ DSA>
```

Conclusion:

This assignment demonstrates the implementation of a Browser History Management System using a Doubly Ended Queue (Deque). It efficiently handles visiting new pages, going back, and moving forward while maintaining a limited history. It helped me understand the practical application of data structures in real-world scenarios like web browsers.