**Stack and Queue Related Problem**

1. [Reverse Stack Using Recursion](https://www.codingninjas.com/studio/problems/reverse-stack-using-recursion_631875)
2. <https://leetcode.com/problems/design-browser-history/description/>

class BrowserHistory {

public:

stack<string> history;

stack<string> forwd;

BrowserHistory(string homepage) {

history.push(homepage);

}

void visit(string url) {

history.push(url);

// new element visit korle forward ke khali kore dibo

while(!forwd.empty()){

forwd.pop();

}

}

string back(int steps) {

while(steps> 0 and history.size() > 1){

forwd.push(history.top());

history.pop();

steps--;

}

return history.top();

}

string forward(int steps) {

while(steps> 0 and !forwd.empty()){

history.push(forwd.top());

forwd.pop();

steps--;

}

return history.top();

}

};

/\*\*

\* Your BrowserHistory object will be instantiated and called as such:

\* BrowserHistory\* obj = new BrowserHistory(homepage);

\* obj->visit(url);

\* string param\_2 = obj->back(steps);

\* string param\_3 = obj->forward(steps);

\*/

1. Next Greater Element

Problem: Given an array, for each element, find the next element that is greater than the current element. If no greater element exists, output -1 for that element.

Input: [4, 6, 3, 2, 8, 1]

Output: [6, 8, 8, 8, -1, -1]

#include<bits/stdc++.h>

using namespace std;

vector<int> nextGreaterElement(vector<int> &nums)

{

int n = nums.size();

vector<int> result(n, -1);

stack<int> st;

for (int i = 0; i < n; ++i)

{

while (!st.empty() && nums[i] > nums[st.top()])

{

result[st.top()] = nums[i];

st.pop();

}

st.push(i);

}

return result;

}

int main()

{

vector<int> arr = {4, 6, 3, 2, 8, 1};

vector<int> result = nextGreaterElement(arr);

for (int val : result)

{

cout << val << " ";

}

cout << endl;

return 0;

}

1. Problem 1: Ticket Counter Queue

Problem: You're given Q queries that represent commands for a ticket counter queue. The commands are of two types:

Enqueue: Given 0 and the name of a person who stood in line at the ticket counter.

Dequeue: Given only 1, meaning the person in front of the line got the ticket and should be removed from the line. Print the name of that person who got the ticket. If the line is empty, print "Invalid".

Input Format:

The first line contains Q, the number of queries.

The following Q lines contain the commands.