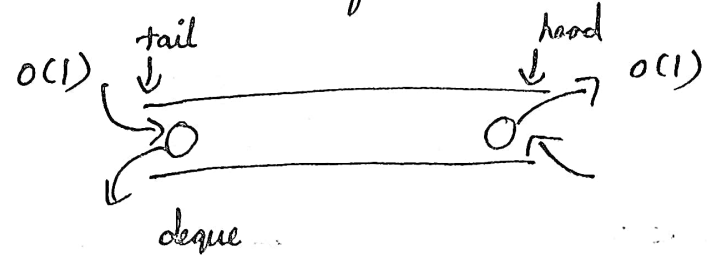
STL Application Idea

Atcoder 158 : D - String Formation

Use deque for this bcoz it has O(1) T.C for inserting & deleting operation at start or from the end.



Given, (|s| upto 10^5, and up to 2×10^5 operations), we must avoid repeated string reversal or appending at beginning.

Approach :-

1. Initialize deque and push each character of string.
2. Use flag = 0 to keep track whether the string is reversed.
3. Loop over q operations :
   - → If t == 1 : reverse the string
   - → If t == 2 :
     - Read f
     - If flipped = 1, flip f (since back & front are swapped in reversed mode)
     - Based on f :
       - If f == 1 → Insert c at front
       - If f == 2 → Insert c at end.
4. After all operations if flipped = 1 → reverse the deque once.

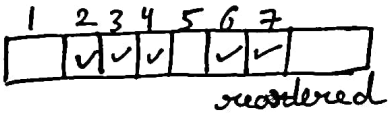
Code :-

```
void solve(){
    string s;
    cin >> s;
    int q;
    cin >> q;
    deque<char> dq;
    for(auto v: s) dq.push_back(v);
```

```
int flipped = 0;
for (int i=0; i<q; i++) {
    int t;
    cin >> t;
    if (t==1) {
        // reverse
        flipped = (1+0) - flipped;
    }
    else {
        int f; char c;
        cin >> f >> c;
        if (flipped) {
            f = (1+2) - f;
        }
        if (f==1) dq.push_front(c);
        else dq.push_back(c);
    }
}
if (flipped) reverse(dq.begin(), dq.end());
for (auto v: dq) {
    cout << v << " ";
}
}
```

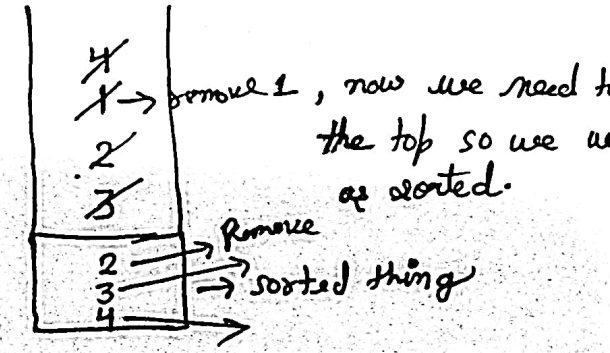
Codeforces : C - Okabe & Boxes

Keep an array reordered



Approach :-

1. If you already reordered once, you don't do it again.
2. Only reorder when a new number comes that breaks the current order.
3. When this happens remove all elements from stack that are not in sorted order.



remove 1, now we need to remove 2 but 4 is at the top so we will remove all and 2 stored in a stack as sorted.

ATcoder 237 → D : LR insertion

Eg : LRRLR

1. start : 0
2. operation 'L' → Incoming : 1, Insert 1 to left of 0 → [1, 0]
3. Operation 'R' → Incoming : 2, Insert 2 to right of 1 → [1, 2, 0]
4. Operation 'R' → Incoming : 3, Insert 3 to right of 2 → [1, 2, 3, 0]
5. operation 'L' → Incoming : 4, Insert 4 to left of 3 → [1, 2, 4, 3, 0]
6. operation 'R' → Incoming : 5, Insert 5 to right of 4 → [1, 2, 5, 4, 3, 0]

Approach : -

1. Use two vectors : one for inserting to the left, one for the right.
2. Start with 0 in the right vector.
3. For each direction in the string :
   → If it's L, insert the no. to the left vector.
   → If it's R, insert to the right vector.
4. At end combine left + right to get final order.

At coder 206 : C - Swappable

Basic Approach :-

check all possible pairs (i,j) where $i<j$ and count those where $A_i \neq A_j$.

T.C : $O(N^2)$

Optimal Approach :-

1. Use a hash map to count the no. of unique value appears in array.
2. For each element in the array
   → Add current count of the no. to ans
   → Inc. count of that no. by 1.

T.C : $O(N)$

Code :-

```
int main ( ){
  int n;
  cin >> n;
  int arr[n];
  for (int i=0; i<n; i++){
      cin >> arr[i];
  }
  long long ans = 0;
  map < int, int > freq;
  for (int j=0; j<n; j++){
    ans += j - freq[arr[j]];
    freq[arr[j]] ++;
  }
  cout << ans << endl;
}
```