结构体重载运算符

人员

柳力玮、谢梓轩、田心一、李瑞涵、刘宸熙、姜皓轩、王馨琪、纪博涵 到课, 初锦阳、苑钊、刘子轩 线上

上周作业检查

上周作业链接: https://cppoj.kids123code.com/contest/108

2025-0622 周日15:30 (综合练习)

刷新

#	用户名	姓名	总分	选择分	编程分	时间	A	B	C	D
1	liuliwei	柳力玮	400	0	400	5941	100	100	100	100
2	tianxinyi	田心一	400	0	400	5972	100	100	100	100
3	chujinyang	初锦阳	350	0	350	959	100	100	100	50
4	yuanzhao	苑钊	305	0	305	888	100	100	100	5
5	xiezixuan	谢梓轩	300	0	300	805	100	100	100	
6	hanyuchen	韩昱辰	300	0	300	951	100	100	100	
7	liruihan	李瑞涵	300	0	300	1286	100	100	100	
8	liupai	刘派	300	0	300	1794	100	100	100	
9	wenhaodong	温郝冬	200	0	200	281	100	100		
10	liuzixuan	刘子轩	200	0	200	1108	100	100		
11	zhaomuzhi	赵牧之	200	0	200	2270	100	100		
12	liuchenxi	刘宸熙	100	0	100	114	100			
13	jianghaoxuan	姜皓轩	100	0	100	119	100			

您正在共享屏幕 结束共享

作业

https://cppoj.kids123code.com/contest/135 (课上讲了 A ~ D 题, 课后作业是 E 题, D 题比较复杂一些, 同学们可以选做)

课堂表现

今天的 A 题跟上周 珍珠链 的题其实基本是一模一样的, 但是同学们今天一开始做又都不会做了, 说明之前的内容掌握的不扎实, 所以同学们要加强课后的复习。

课堂内容

“非常男女”计划

跟之前 珍珠链 的题目一样, 把 两种类型 当做 1 和 -1 来进行处理, 找最长的一段, 其实就是找一段区间和是 0 的, 用前缀和数组来处理即可

```cpp
#include <bits/stdc++.h>

using namespace std;

const int maxn = 1e5 + 5;
int w[maxn], p[maxn];
```

```
int main()
{
  int n; cin >> n;
  for (int i = 1; i <= n; ++i) {
    cin >> w[i];
    if (w[i] == 0) w[i] = -1;
    p[i] = p[i-1] + w[i];
  }

  int res = 0;
  map<int, int> mp; mp[0] = 0;
  for (int i = 1; i <= n; ++i) {
    int x = p[i];
    if (mp.count(x)) res = max(res, i-mp[x]);
    else mp[x] = i;
  }
  cout << res << endl;
  return 0;
}
```

移动

set 套 node 维护到过哪些点即可

```
#include<iostream>
#include<set>
using namespace std;
struct node{
    int l,r;
    friend bool operator < (node p,node q)
    {
        if (p.l != q.l) return p.l < q.l;
        return p.r < q.r;
    }
};
int main()
{
    int a;
    set<node> s;
    node x = {0,0},y = {0,0};
    s.insert(y);
    cin >> a;
    for (int i=1;i<=a;i++)
    {
        char c;
        cin >> c;
        if (c == 'L') x.l--;
        if (c == 'R') x.l++;
        if (c == 'U') x.r++;
        if (c == 'D') x.r--;
```

```
        if (s.count(x))
        {
            cout << "Yes";
            return 0;
        }
        else s.insert(x);
    }
    cout << "No";
    return 0;
}
```

重载运算符

```
struct node {
  int x, y, z;
  friend bool operator < (node p, node q) {
    if (p.x != q.x) return p.x < q.x;
    if (p.y != q.y) return p.y < q.y;
    return p.z < q.z;
  }
  friend bool operator > (node p, node q) {
    if (p.x != q.x) return p.x > q.x;
    if (p.y != q.y) return p.y > q.y;
    return p.z > q.z;
  }
};
```

日志分析

插入的时候, 只用在 栈 里面插入 当前值和栈里前一个值 的 max 即可

这样, 每次查询的时候, 输出栈顶就可以了

```
#include <bits/stdc++.h>

using namespace std;

int main()
{
  int n; cin >> n;
  stack<int> s;
  while (n -- ) {
    int op; cin >> op;
    if (op == 0) {
      int x; cin >> x;
      if (!s.empty()) x = max(x, s.top());
      s.push(x);
    } else if (op == 1) {
      if (!s.empty()) s.pop();
```

```
    } else {
      cout << (s.empty() ? 0 : s.top()) << endl;
    }
  }
  return 0;
}
```

验证栈序列

把 a 数组里的数一个一个往栈里放, 每次跟 b 当前的值对应上时, 就把栈里的数删掉, 同时 b 数组对应的位置要后移一个位置

```
#include <bits/stdc++.h>

using namespace std;

const int maxn = 1e5 + 5;
int a[maxn], b[maxn];

void solve() {
    int n; cin >> n;
    for (int i = 1; i <= n; i++) cin >> a[i];
    for (int i = 1; i <= n; i++) cin >> b[i];

    stack<int> stk;
    int j = 1;
    for (int i = 1; i <= n; i++) {
        stk.push(a[i]);
        while (!stk.empty() && stk.top()==b[j]) {
            stk.pop(); j++;
        }
    }

    if (j == n+1) cout << "Yes" << endl;
    else cout << "No" << endl;
}

int main()
{
    int T; cin >> T;
    while (T -- ) solve();
    return 0;
}
```