综合练习

人员

谢亚锴、陈洛冉、孙靖轲、于子珈、秦显森、刘闯速、赵熙羽、杨瑾硕、牛同泽、司云心 到课

上周作业检查

上周作业链接: https://cppoj.kids123code.com/contest/1116

王向东老师周日十点半C++综合练习

刷新

#	用户名	姓名	编程分	时间	A	B	C	D
1	xieyakai	谢亚锴	386	9929	100	100	100	86
2	suitianyi	隋天乙	362	2933	100	100	62	100

本周作业

https://cppoj.kids123code.com/contest/1209 (课上讲了 A ~ D 题, 课后作业是 D 题)

课堂表现

今天的题目整体比较难一些, 同学们课上可能比较难全部吸收, 课下要多花点时间复习一下今天的题目。

课堂内容

Go Stone Puzzle (上周作业)

跟 string 变化相关的 bfs

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

using namespace std;

map<string, int> mp;

int main()
{
  int n; string _st, _ed; cin >> n >> _st >> _ed;
  _st += "  ", _ed += "  ";
  queue<string> q; q.push(_st); mp[_st] = 0;
  while (!q.empty()) {
    string u = q.front(); q.pop();
    if (u == _ed) { cout << mp[u] << endl; return 0; }

    int pos = u.find("  ");
    for (int i = 0; i <= n; ++i) {
      if (u[i]==' ' || u[i+1]==' ') continue;
```

```
        string t = u;
        swap(t[i], t[pos]), swap(t[i+1], t[pos+1]);
        if (!mp.count(t)) q.push(t), mp[t] = mp[u]+1;
      }
    }
    cout << -1 << endl;
    return 0;
}
```

Election Quick Report

set 里存放每个候选人编号以及候选人的得票数

每次给一个人加票时, 从 set 里删掉这个人的信息, 并重新插入新的信息即可

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

using namespace std;

const int maxn = 200000 + 5;
int f[maxn];
struct node {
  int id, cnt;
  bool operator < (const node& p) const {
    if (cnt != p.cnt) return cnt > p.cnt;
    return id < p.id;
  }
};

int main()
{
  set<node> s;
  int n, m; cin >> n >> m;
  for (int i = 1; i <= n; ++i) s.insert({i,0});
  while (m -- ) {
    int x; cin >> x;
    int t = f[x]; ++f[x];
    auto it = s.find({x,t});
    s.erase(it); s.insert({x,t+1});
    cout << (*s.begin()).id << endl;
  }
  return 0;
}
```

平铺图案

二维前缀和直接维护即可

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

using namespace std;

const int maxn = 1000 + 5;
char s[maxn][maxn];
int p[maxn][maxn];

int calc(int n, int x, int y) {
  int c1 = x / n, c2 = y / n;
  int sx = x % n, sy = y % n;
  int res = c1 * c2 * p[n][n] + c1 * p[n][sy] + c2 * p[sx][n] + p[sx][sy];
  return res;
}

signed main()
{
  int n, m; cin >> n >> m;
  for (int i = 1; i <= n; ++i) cin >> (s[i]+1);
  for (int i = 1; i <= n; ++i) {
    for (int j = 1; j <= n; ++j) {
      p[i][j] = p[i-1][j] + p[i][j-1] - p[i-1][j-1] + (s[i][j]=='B');
    }
  }

  while (m -- ) {
    int a, b, c, d; cin >> a >> b >> c >> d;
    ++a, ++b, ++c, ++d;
//    cout << "------------------ ";
    cout << calc(n,c,d) - calc(n,a-1,d) - calc(n,c,b-1) + calc(n,a-1,b-1) << endl;
  }
  return 0;
}
```

[蓝桥杯 2025 省 Java B] 数组翻转

找到每个数出现的最长的连续两段的长度

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

using namespace std;

typedef long long LL;
const int maxn = 1e6 + 5;
int w[maxn];
vector<int> vec[maxn];

int main()
{
```

```
  int n; cin >> n;
  for (int i = 1; i <= n; ++i) cin >> w[i];
  int cnt = 1;
  for (int i = 2; i <= n; ++i) {
    if (w[i] == w[i-1]) ++cnt;
    else vec[w[i-1]].push_back(cnt), cnt = 1;
  }
  vec[w[n]].push_back(cnt);

  LL res = 0;
  for (int i = 1; i < maxn; ++i) {
    sort(vec[i].begin(), vec[i].end()), reverse(vec[i].begin(), vec[i].end());
    int len = 0;
    if ((int)vec[i].size() >= 1) len += vec[i][0];
    if ((int)vec[i].size() >= 2) len += vec[i][1];
    res = max(res, (LL)len * i);
  }
  cout << res << endl;
  return 0;
}
```

[蓝桥杯 2023 国 C] 最大区间

单调栈 维护每个数 左边第一个比他小的位置 和 右边第一个比他小的位置

枚举每个数作为最小值即可

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

using namespace std;

typedef long long LL;
const int maxn = 3e5 + 5;
int w[maxn];
int pre[maxn], suf[maxn];

int main()
{
    int n; cin >> n;
    for (int i = 1; i <= n; ++i) cin >> w[i];

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

    while (!stk.empty()) stk.pop();
    for (int i = n; i >= 1; --i) {
        while (!stk.empty() && w[i]<=w[stk.top()]) stk.pop();
```

```
        if (!stk.empty()) suf[i] = stk.top();
        else suf[i] = n+1;
        stk.push(i);
    }

    LL res = 0;
    for (int i = 1; i <= n; ++i) {
        int l = pre[i]+1, r = suf[i]-1;
        res = max(res, (LL)w[i]*(r-l+1));
    }
    cout << res << endl;
    return 0;
}
```