对顶堆

人员

袁晨峻、李锦澍、陈欣妙、杨俊彦、刘奕辰 到课

上周作业检查

上周作业链接: https://cppoj.kids123code.com/contest/2683

#	用户名	姓名	编程分	时间	A	B	C	D
1	lijinshu	李锦澍	300	7923	100	100	100	
2	yuanchenjun	袁晨峻	223	538	100	100	23	
3	yangjunyan	杨俊彦	200	651	100	100		
4	chenxinmiao	陈欣妙	200	1022	100	100		

本周作业

https://cppoj.kids123code.com/contest/2900 (课上讲了 A ~ C 题, 课后作业是 C 题)

课堂表现

今天的题目整体难度不是很大, 同学们课上做题表现都比较好。

课堂内容

Grid Filling (上周作业)

p[i][j][k]: (1,1) ~ (i,j) 中, 有多少个 k

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

using namespace std;

const int maxn = 300 + 5;
int w[maxn][maxn], p[maxn][maxn][maxn], f[maxn];

int get_sum(int x1, int y1, int x2, int y2, int c) {
  return p[x2][y2][c] - p[x1-1][y2][c] - p[x2][y1-1][c] + p[x1-1][y1-1][c];
}

int main()
{
  int n, m, N, hh, ww; cin >> n >> m >> N >> hh >> ww;
  for (int i = 1; i <= n; ++i) {
```

```
    for (int j = 1; j <= m; ++j) {
      cin >> w[i][j], f[w[i][j]]++;
      for (int k = 1; k <= N; ++k) {
        p[i][j][k] = p[i-1][j][k] + p[i][j-1][k] - p[i-1][j-1][k] + (w[i][j]==k);
      }
    }
  }

  int res = 0;
  for (int i = 1; i <= N; ++i) {
    if (f[i]) ++res;
  }

  for (int i = 1, i2 = hh; i2 <= n; ++i, ++i2) {
    for (int j = 1, j2 = ww; j2 <= m; ++j, ++j2) {
      int cnt = 0;
      for (int k = 1; k <= N; ++k) {
        if (get_sum(i,j,i2,j2,k)==f[k] && f[k]) ++cnt;
      }
      cout << res - cnt << " ";
    }
    cout << endl;
  }
  return 0;
}
```

中位数

对顶堆, 用两个 优先队列 维护

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

using namespace std;

int main()
{
  priority_queue<int,vector<int>,less<int>> q1;
  priority_queue<int,vector<int>,greater<int>> q2;

  int n; cin >> n;
  for (int i = 1; i <= n; ++i) {
    int x; cin >> x;
    if (q1.empty() || x<=q1.top()) q1.push(x);
    else q2.push(x);

    while (q1.size() < q2.size()) {
      int u = q2.top(); q2.pop(); q1.push(u);
    }
    while (q2.size()+1 < q1.size()) {
      int u = q1.top(); q1.pop(); q2.push(u);
    }
```

```
    if (i&1) cout << q1.top() << endl;
  }
  return 0;
}
```

Least Elements

两个 multiset 维护, 用类似于对顶堆的思路做即可

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

using namespace std;

const int maxn = 2e5 + 5;
int w[maxn];

signed main()
{
  int n, m, K, sum = 0; cin >> n >> m >> K;
  for (int i = 1; i <= n; ++i) cin >> w[i];

  multiset<int> s1, s2;
  for (int i = 1; i <= n; ++i) {
    if (s1.empty() || w[i]<=*s1.rbegin()) s1.insert(w[i]), sum += w[i];
    else s2.insert(w[i]);

    if (i >= m) {
      while ((int)s1.size() < K) {
        int u = *s2.begin(); s2.erase(s2.find(u)); s1.insert(u); sum += u;
      }
      while ((int)s1.size() > K) {
        int u = *s1.rbegin(); s1.erase(s1.find(u)); sum -= u; s2.insert(u);
      }

      cout << sum << " ";

      int x = w[i-m+1];
      if (s2.count(x)) s2.erase(s2.find(x));
      else s1.erase(s1.find(x)), sum -= x;
    }
  }
  return 0;
}
```

Dist Max 2

把所有点按照 x 排序, 维护后缀最大 y 和后缀最小 y

然后 二分 套 二分 做即可

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

using namespace std;

const int maxn = 2e5 + 5;
struct node {
  int x, y;
  bool operator < (const node& p) const {
    if (x != p.x) return x < p.x;
    return y < p.y;
  }
} w[maxn];
int suf_minY[maxn], suf_maxY[maxn];
int n;

bool check(int mid) {
  for (int i = 1; i <= n; ++i) {
    int pos = lower_bound(w+1, w+n+1, node{w[i].x+mid, (int)-1e9-10}) - w;
    if (pos <= n) {
      int d1 = abs(w[i].y - suf_minY[pos]), d2 = abs(w[i].y - suf_maxY[pos]);
      if (max(d1,d2) >= mid)
        return true;
    }
  }
  return false;
}

int main()
{
  cin >> n;
  for (int i = 1; i <= n; ++i) cin >> w[i].x >> w[i].y;
  sort(w+1, w+n+1);
  suf_minY[n] = suf_maxY[n] = w[n].y;
  for (int i = n-1; i >= 1; --i) {
    suf_minY[i] = min(suf_minY[i+1], w[i].y);
    suf_maxY[i] = max(suf_maxY[i+1], w[i].y);
  }

  int l = 0, r = 1e9;
  while (l <= r) {
    int mid = (l + r) / 2;
    if (check(mid)) l = mid+1;
    else r = mid-1;
  }
  cout << r << endl;
  return 0;
}
```