

C++

day1 SH-B260101 集训-讲解

T1 人员分配

给你 $2*n$ 对数字,你要选出其中 n 对
数字取其左数, n 对数字取其右数,
使得这 $2*n$ 个数字之和最大化

输入

2

50 80

60 80

60 60

30 40

输出

250

莱奥编程

动态
规划

DP第一步
状态设计

尝试

自然定义法
抄原题大法

仿照题目问题
设计状态含义

$g[2*n][n]$ 表示考虑前 $2n$ 人,
 n 人在足球队的最大满意程度

$g[i][j]$ 表示只考虑前 i 人,
 j 人在足球队的最大满意程度

输入
2
50 80
60 80
60 60
30 40

$c=4$ $n=2$

$b[1]=50, f[1]=80$

$b[2]=60, f[2]=80$

$b[3]=60, f[3]=60$

$b[4]=30, f[4]=40$

$g[i][j]$ 表示只考虑前 i 人,
 j 人在足球队的最大满意程度

$g[i][j]$	$j=0$	$j=1$	$j=2$
$i=0$	0	0	0
$i=1$			
$i=2$			
$i=3$			
$i=4$			

$$g[4][1] = \max(f[3][0] + f[4], f[3][1] + b[4])$$

01
决策

决策
足球队

决策
篮球队

方法1

$g[i][j]$ 表示只考虑前 i 人， j 人在足球队的最大满意程度

```
9   for(int i=1;i<=2*n;i++)
10       cin>>b[i]>>f[i];
11   for(int i=1;i<=2*n;i++){
12       for(int j=0;j<=min(i,n);j++) {
13           if(j<i) g[i][j]=max(g[i][j],g[i-1][j]+b[i]);
14           if(j>0) g[i][j]=max(g[i][j],g[i-1][j-1]+f[i]);
15       }
16   }
17   cout<<g[2*n][n]<<endl;
```

方法2

i号成员有两个属性 f_i, b_i

简化问题

考虑只有两人
如何决定？

贪心 算法

每次挑选足球意愿比篮球意愿强最多的人加入足球队

剩下的人加入篮球队

讨论题：为什么该贪心法正确？

莱奥编程

贪心法

```
4  const int N=1009;
5  struct ren{
6      ll b,f;
7  };
8  ren a[2*N];
9  bool cmp(const ren &x,const ren &y){
10     return 
11 }

15     int n;
16     cin>>n;
17     for(int i=1;i<=2*n;i++) cin>>a[i].b>>a[i].f;
18     sort(a+1,a+2*n+1,cmp);
19     int ans=0;
20     for(int i=1;i<=n;i++) ans+=a[i].b;
21     for(int i=n+1;i<=2*n;i++) ans+=a[i].f;
22     cout<<ans<<endl;
```

净收益排序

T2 高频词汇

莱奥编程

请同学写出题目大意
已知什么求什么

给出若干个单词，求出现频率最高的前K个
典型TopK问题

思考该如何处理输入？

方案一：单词数量未知，考虑while-cin结构
但K的值是在第二行给出，考虑字符串转数字

方案二：把第一行读进一个字符串S
然后遍历S，提取单词

菜鸟编程

map做频率统计

TopK问题

求出现频率最高的K个字符串

常用方法：

map用于统计次数很方便

莱奥编程

map映射

每个	字符串	对应一个	出现次数
----	-----	------	------

```
map<string,int> cnt;
```

```
string s;  
while(cin>>s) cnt[s]++;
```

cnt[s]对应字符串s出现的次数

while搭配cin处理
不能提前预知
输入个数的情况

DevC++里运行
输入结束时
按Ctrl Z再换行

处理输入

(while-cin)+字符串转数字

```
11  int k = 0;
12  string s;
13  while (cin >> s) {
14      if ('0' <= s[0] && s[0] <= '9') {
15          for (int i = 0; i < s.size(); ++i) {
16              k = k * 10 + 
17          }
18          break;
19      }
20      
21  }
```

遍历map

迭代器iterator

key-value(first-second)

一对键值(pair)

```
23 vector<string> v;  
24  
25 map<string, int>::iterator it;  
26 for(it=cnt.begin(); it!=cnt.end(); ++it) {  
27     v.push_back( );  
28 }  
29  
30 sort(  cmp);  
31  
32 for (int i = 0; i < k; ++i) {  
33     cout << v[i] << endl;  
34 }
```

排序

sort自定义排序规则

```
3 map<string, int> cnt;  
4 bool cmp(string x, string y) {  
5     return  || cnt[x] == cnt[y] && ;  
6 }
```

把map里的内容重新捆绑成结构体
然后对结构体数组排序也是可以的

T3飞盘

莱奥编程

请同学简述题意
突出核心要点

已知1到n的一个排列，找到排列中两端大于其中点的区间，对其长度求和

$1 \leq n \leq 300000$ $hi \leq 1e9$

莱奥编程

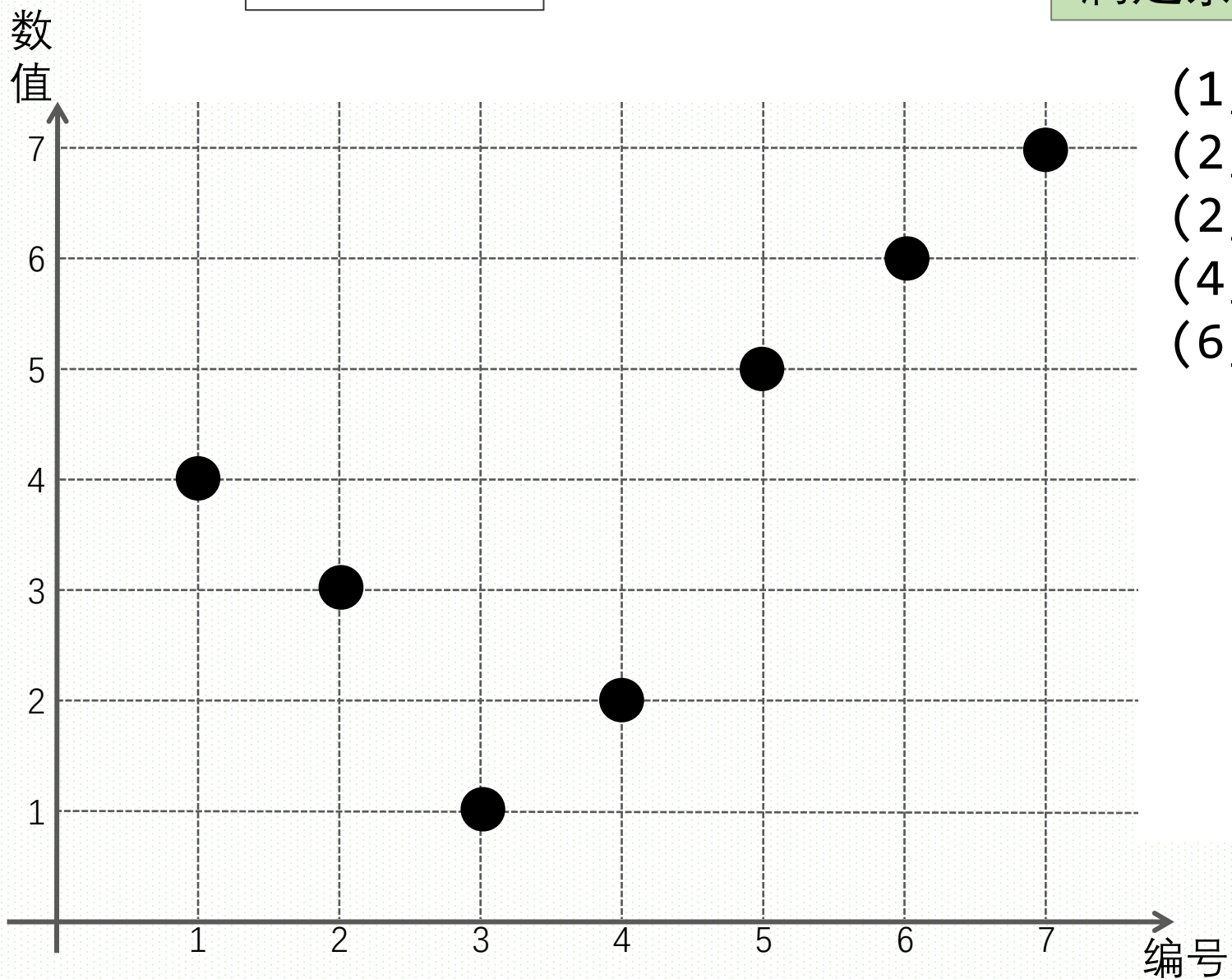
手算样例

输入
7

4 3 1 2 5 6 7

输出
24

满足条件的区间:



(1, 2) (1, 5)
(2, 3) (2, 4)
(2, 5) (3, 4)
(4, 5) (5, 6)
(6, 7)

复杂度 $O(N^2)$

暴力枚举

枚举区间左端点 i

枚举区间右端点 j

判断区间内元素最大值
小于 $h[i]$ 和 $h[j]$

累加 $j-i+1$

莱奥编程

复杂度 $O(N^2)$

暴力枚举

30分

```
7 void input(){
8     cin>>n;
9     for(ll i=1;i<=n;i++) cin>>h[i];
10 }
11 void solveBF(){
12     ans=0;
13     for (ll i=1;i<=n;++i) {
14         ll mx=-1;
15         for (ll j=i+1;j<=n;++j) {
16             if(mx<min(h[i],h[j])) ans+=j-i+1;
17             mx=max(mx,h[j]);
18         }
19     }
20     cout<<ans<<endl;
21 }
```

问题转换

对于所有可行区间 (i, j) :

如果 $h[i] < h[j]$: j 为 i 右侧最早出现比 $h[i]$ 大的位置

如果 $h[i] > h[j]$: i 为 j 左侧最早出现比 $h[j]$ 大的位置

已知1到 n 的一个排列，找到排列中两端大于其中点的区间，对其长度求和

原问题



给定1到 n 的一个排列
对每一位询问：
它右侧第一个比它大的是几号位

右侧更大值最近问题

另一个左侧最大值最近问题怎么办？

反转数组，再做一遍！

右侧更大值最近问题

离线
询问

可以看完问题
换顺序回答

挑选最简单的
问题是哪个

1

最小

2

最大

3

最右侧

三种
思路
都可
行

莱奥编程

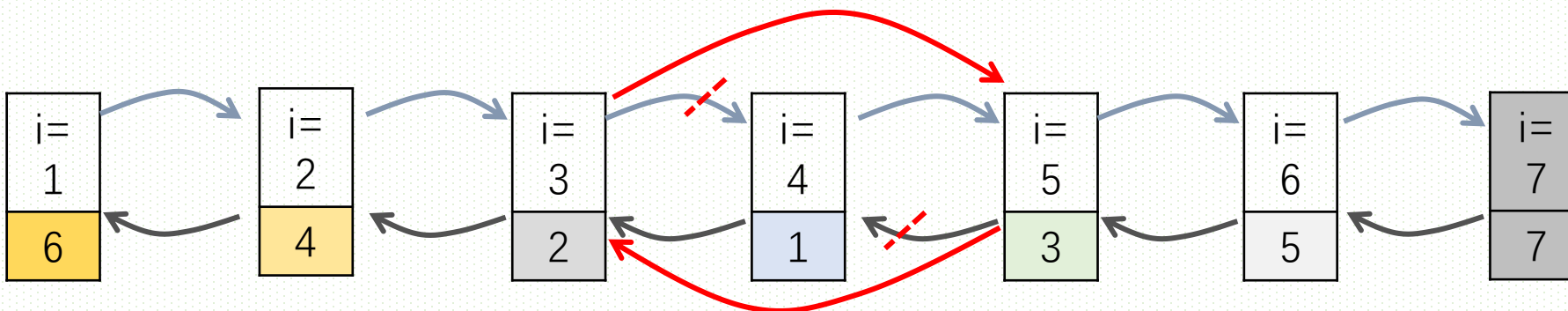
1、最小

依次挑选当前最小数值
回答问题

离线问询+双链表删除

双链表处理删除

	i=1	i=2	i=3	i=4	i=5	i=6
得分	6	4	2	1	3	5



如何删除最小数据

1. 定位其在哪个位置

2. 左右邻居互相指认

双链表删除

	i=1	i=2	i=3	i=4	i=5	i=6
高度	6	4	2	1	3	5

高度为1的哨兵，它的答案是此时它贴身右边邻居位置

高度为1的哨兵确定后，高度为2的哨兵能否确定呢？

将高度为1的哨兵删除后分析高度为2的哨兵

它的答案就是此时它贴身右边邻居位置

将高度为2的哨兵删除后分析高度为3的哨兵

它的答案就是此时它贴身右边邻居位置

莱奥编程


```
23 void solve(){
24     for (ll i=1;i<=n;++i) p[h[i]]=i;
25     for (ll i=1;i<=n;++i) {
26         pre[i]=i-1;
27         nxt[i]=i+1;
28     }
29     for (ll i=1;i<=n;++i) {
30         if(nxt[p[i]]!=n+1) ans+=nxt[p[i]]-p[i]+1;
31         nxt[pre[p[i]]]=nxt[p[i]];
32         pre[nxt[p[i]]]=pre[p[i]];
33     }
34 }
```

依次挑选当前最大数值
回答问题

```
35 void solveSet(){
36     for (ll i=1;i<=n;++i) p[h[i]]=i;
37     set<ll> s;
38     s.insert(n+1);
39     for (ll v=n;v>=1;--v) {
40         ll t=*s.lower_bound(p[v]);
41         if(t!=n+1) ans+=t-p[v]+1;
42         s.insert(p[v]);
43     }
44 }
```

依次挑选当前最右位置
回答问题

```
46 void solveMStack(){
47     id[1]=n;
48     ll j=1;
49     for(ll i=n-1;i>=1;--i){
50         while(j>=1&&h[i]>h[id[j]])
51             --j;
52         if(j>=1) ans+=id[j]-i+1;
53         id[++j]=i;
54     }
55 }
```

```
59 | input();
60 | if(n<=5000) solveBF();
61 | else {
62 |     solve();
63 |     // solveSet();
64 |     // solveMStack();
65 |     reverse(h+1,h+1+n);
66 |     solve();
67 |     // solveSet();
68 |     // solveMStack();
69 |     cout<<ans<<endl;
70 | }
71 | return 0;
```



等等

一定要反转数组分两次处理吗？

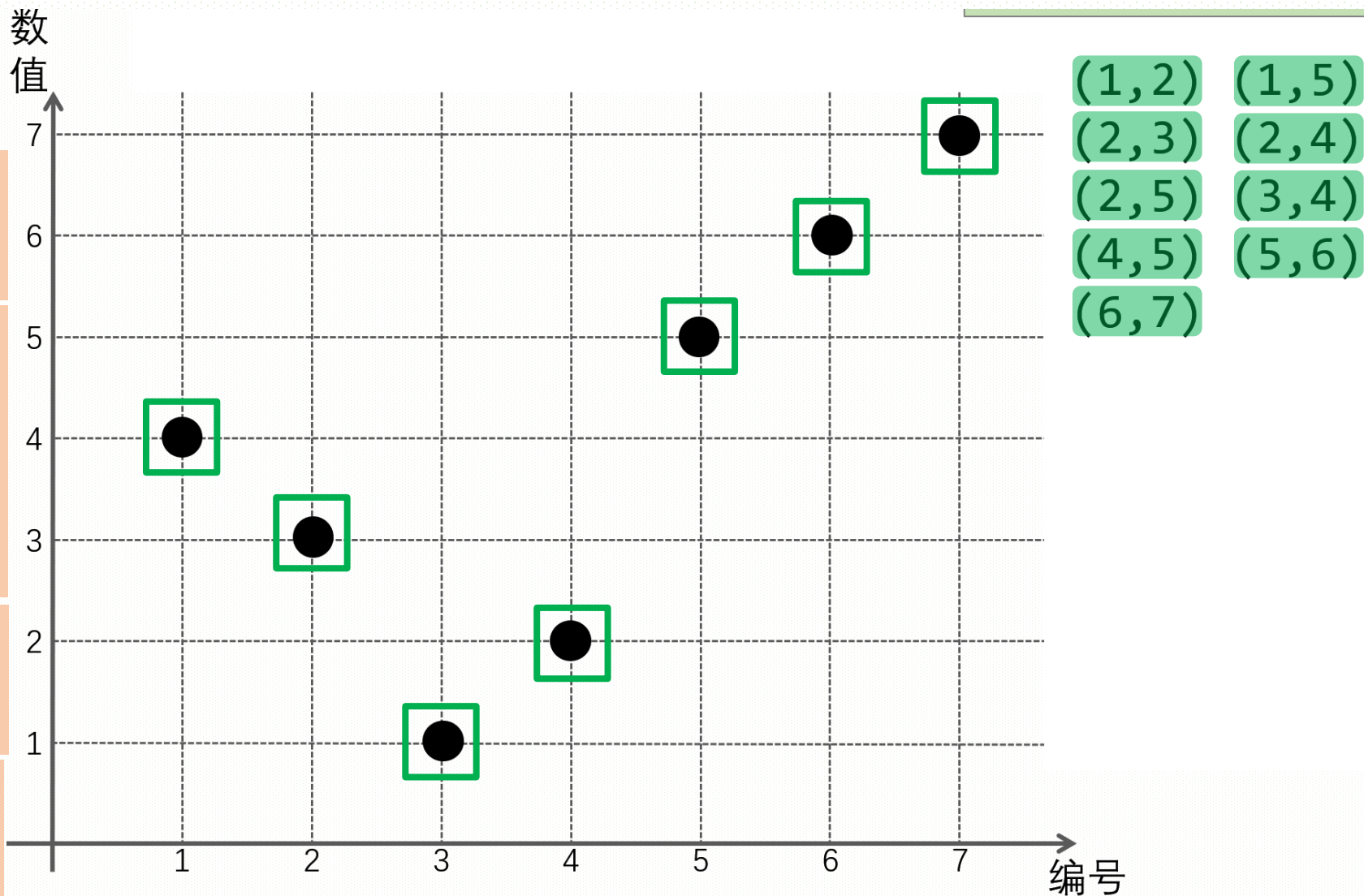
尝试从左侧第一个数，找右侧可以配对的区间右端点

用栈记录有可能成为后续配对区间的左端点

枚举每一个新的位置时，从栈顶开始检测，比当期位置低，和第一个比自己位置高的元素，都可以配对。

同时淘汰掉栈内比当前位置低的元素。新位置的数据一定进栈。

栈内存储的数据具有单调性（下降序列）



单调栈

```
11 stk[N]; //数组模拟栈
int top = 0;
void solveMStack() { //单调栈
    for (int i = 1; i <= n; i++) { //依次枚举每一个h[i]
        while (top > 0 && ) { //栈顶元素高度小于当前h[i]
            ans += () ; //h[i]为stk[top]右侧第一个更大值 / 左低右高
            top--; //栈顶出栈
        }

        if (top > 0) {
            ans += () ; //stk[top]为h[i]左侧的第一个更大值 / 左高右低
        }

        stk[++top] = i; //新位置进栈
    }
}
```

T4

莱奥编程

请同学写出题目大意
已知什么求什么

从每个集合选1个数，求选出数字组成的可重集合的极差最小值。

部分分策略讨论

测试点编号	特殊性质
1	$k=2$
2	所有人的身高均不超过 2
3	所有 c_i 均为 1
4	所有 c_i 均为 2
5~10	无

分类处理：

1、确保得分

2、可以启发正解的思路

数据分流

```
71 int main(){
72     freopen("parade.in", "r", stdin);
73     freopen("parade.out", "w", stdout);
74     input();
75     if(nClass==2)
76         solveK2();
77     else if(isH12())
78         solveH12();
79     else if(isC1())
80         solveC1();
81     else
82         solve();
83     return 0;
84 }
```

思考：输入数据如何管理？

```
15 void input(){
16     cin>>nClass;
17     int h;
18     for(int i=1;i<=nClass;++i){
19         cin>>c[i];
20         for(int j=1;j<=c[i];++j){
21             cin>>h;
22             students[++nStu]=(Student){h,i};
23         }
24     }
25     sort(students+1,students+1+nStu,cmp);
26 }
```

```
7 struct Student{  
8     int h,id;  
9 } students[N];  
10 bool cmp(const Student&x,const Student&y){  
11     return x.h<y.h||x.h==y.h&& x.id<y.id;  
12 }  
13 int nStu;  
14 int nClass;
```

特判k=2

```
27 void solveK2(){
28     int ans=2e9;
29     for(int i=2;i<=nStu;++i){
30         if(students[i].id==students[i-1].id)continue;
31         ans=min(ans,students[i].h-students[i-1].h);
32     }
33     cout<<ans<<endl;
34 }
```

特判：所有人的身高均不超过2

```
35 bool isH12(){
36     for(int i=1;i<=nStu;++i)
37         if(students[i].h>2) return 0;
38     return 1;
39 }
40 void solveH12(){
41     int nUnqStu=unique(students+1,students+1+nStu,cmp)-students-1;
42     int cH1=0,cH2=0;
43     for(int i=1;i<=nUnqStu;++i)
44         if(students[i].h==1) ++cH1;
45         else ++cH2;
46     if(cH1==nClass||cH2==nClass) cout<<0<<endl;
47     else cout<<1<<endl;
48 }
```

特判：所有ci均为1

```
49 bool isC1(){
50     for(int i=1;i<=nClass;++i)
51         if(c[i]>1) return 0;
52     return 1;
53 }
54 void solveC1(){
55     int ans=students[nClass].h-students[1].h;
56     cout<<ans<<endl;
57 }
```

特判：所有 c_i 均为2

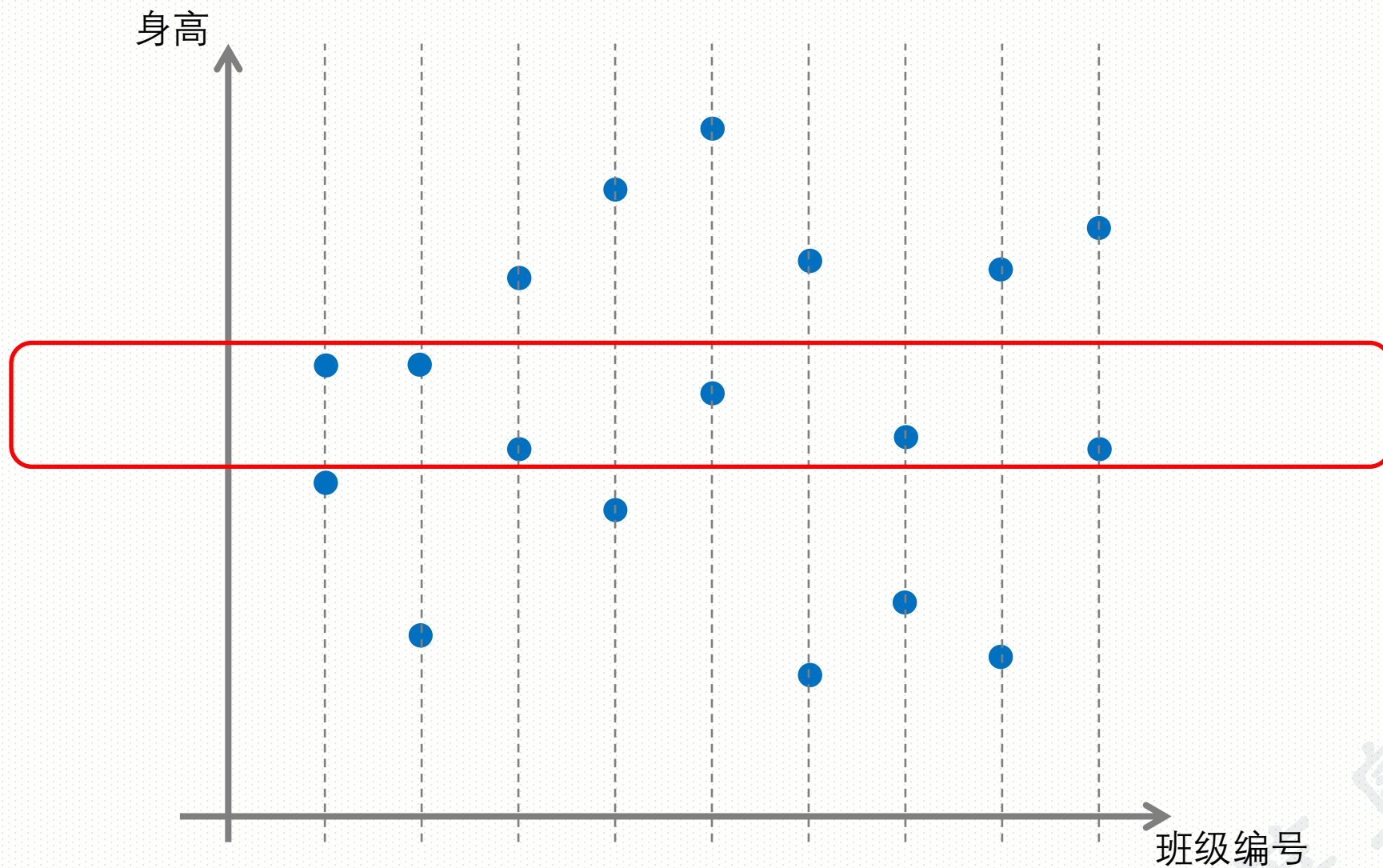
正解分析

每人都有两个属性：身高，班级编号

二维平面点可视化

思考：有什么启发？

$$c_i=2$$

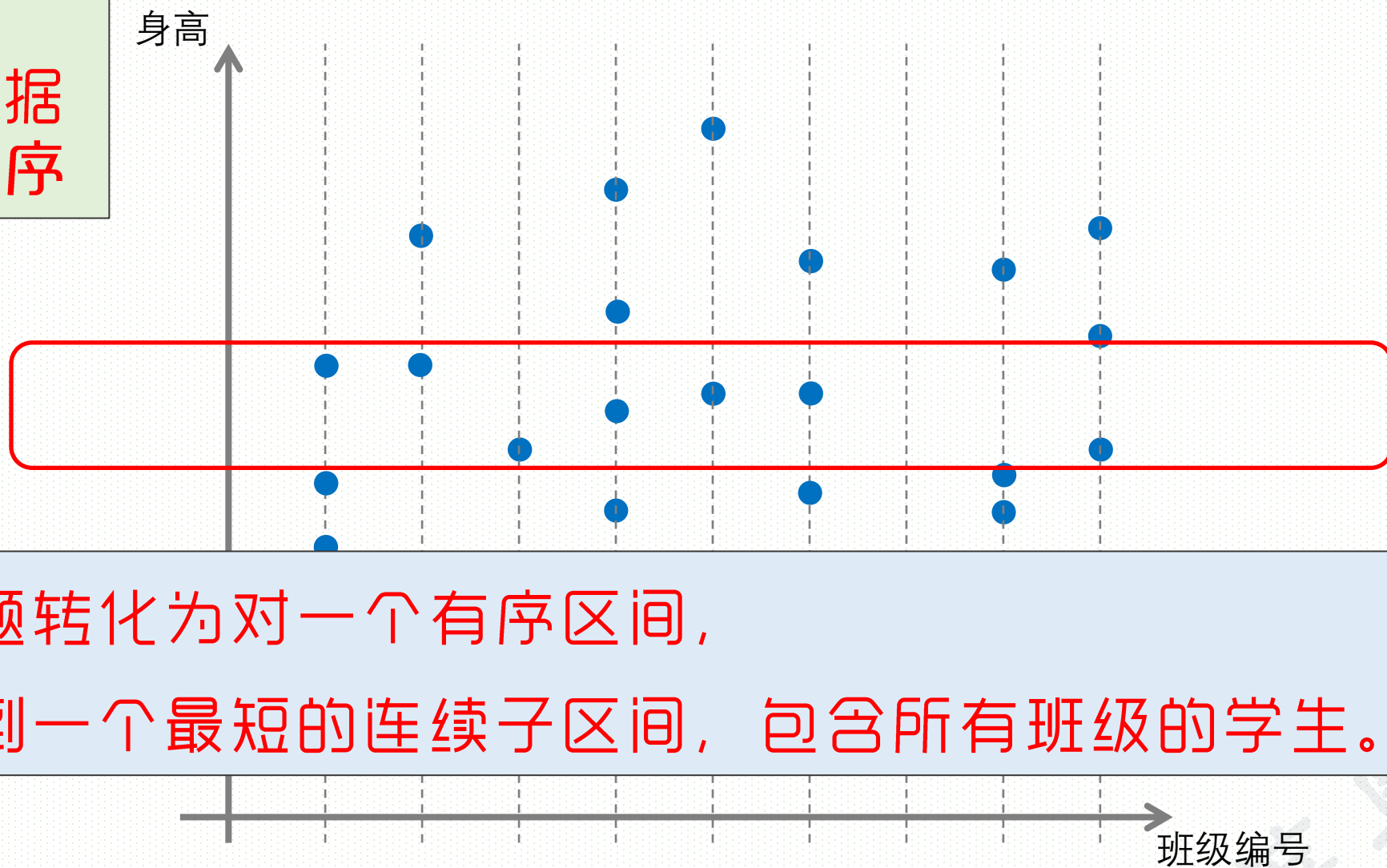


米奥编程

$$c_i > 2$$

最终选择的元素，集中在某一个区间

输入时，
已经对数据
进行了排序



问题转化为对一个有序区间，
找到一个最短的连续子区间，包含所有班级的学生。

蠕动区间/双游标

对于所有学生排序：先看身高再看班级

主要游标 i ：表示 i 号学生作为选出的最高人

辅助游标 j ：表示 j 号学生作为选出的最矮人

确保从 j 号到 i 号的学生包含所有班级

蠕动区间/双游标

从矮到高枚举i号学生作为选出的最高人

当j号学生所在班级被选中不止一人

j号学生不用选，将j加1

如果从j号到i号的学生包含所有班级

尝试用i号身高减j号身高更新答案

```
58 void solve(){
59     int ans=2e9;
60     int cUnqClass=0;
61     int j=1;
62     for(int i=1;i<=nStu;++i){
63         cUnqClass+=(  
64         while(  
65             --cnt[  
66             if(  
67             ans=min(ans,  
68     }
69     cout<<ans<<endl;
70 }
```

记录已有班级的数量

收缩左边界
移除冗余的学生

检测是否包含全部班级

比较记录当前区间内的极差值

T5车站分级

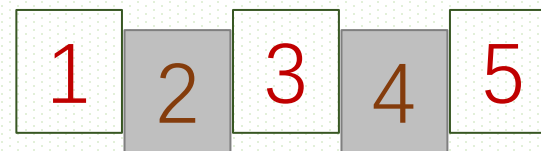
莱奥编程

车站分级

车站编号	1		2		3		4		5		6		7		8		9
车站级别 车次	3		1		2		1		3		2		1		1		3
1	始	→	→	→	停	→	→	→	停	→	终						
2					始	→	→	→	停	→	终						
3	始	→	→	→	→	→	→	→	停	→	→	→	→	→	→	→	终
4							始	→	停	→	停	→	停	→	停	→	终
5					始	→	→	→	停	→	→	→	→	→	→	→	终

输入样例
6 2
3 1 3 5
2 3 6

输出样例
3



图的识别
点的含义
边的含义

i号节点对应
i号车站

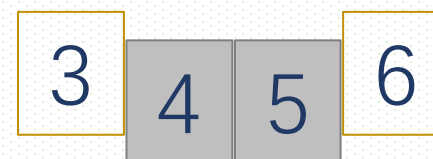
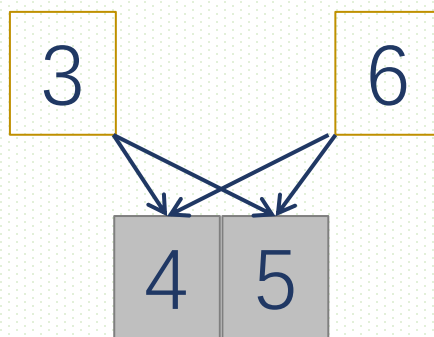
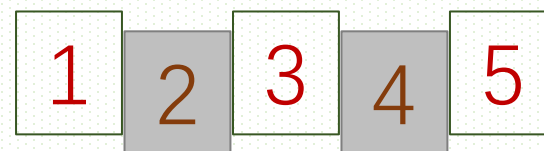
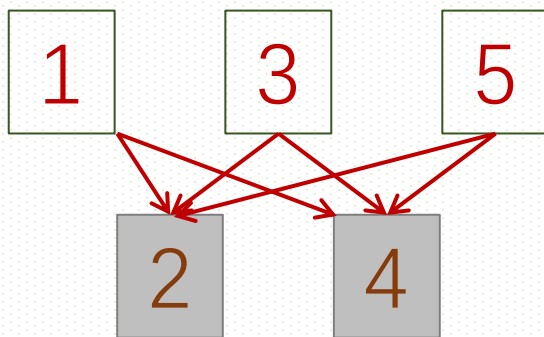
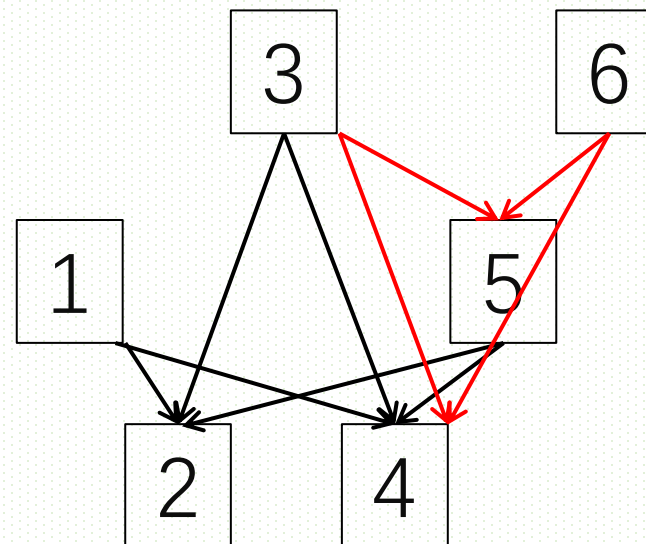
(i,j)有向边代表
i的级别高于j



DAG图

DAG最长路径

需要分3个等级



```

5 int F(int x){
6     if(ok[x])return f[x];
7     ok[x]=1; f[x]=1;
8     for(int i=1;i<=n;i++)if(d[x][i])
9         f[x]=max(f[x],F(i)+1);
10    return f[x];
11 }

```

建图存图
占据大量篇幅

```

15 cin>>n>>m;
16 for(int i=1;i<=m;i++){
17     cin>>s;
18     for(int j=0;j<s;j++)cin>>stop[j];
19     int c=0;
20     for(int j=0;j<s-1;j++)
21         for(int k=stop[j]+1;k<=stop[j+1]-1;k++)
22             nonstop[c++]=k;
23     for(int j=0;j<s;j++)
24         for(int k=0;k<c;k++)
25             d[stop[j]][nonstop[k]]=1;
26 }
27 int ans=0;
28 for(int i=1;i<=n;i++)
29     ans=max(ans,F(i));
30 cout<<ans<<endl;

```

SH-B260101 订正题号

T1 3306 人员分配

T2 3133 高频词汇

T3 3120 飞盘

T4 3467 仪仗队

T5 495 车站分级