

# 信奥算法

# digit

请同学简述题意  
突出核心要点

把两个整数 $a + b$ 的结果从低位开始每隔三位补一个逗号

# 请阅读[数据规模和约定] 识别部分得分点

50%数据,  $|a|, |b| \leq 500$

100%数据,  $|a|, |b| \leq 1000000$

对于50%数据, 只有当 $a+b = |1000|$ 时才有可能出现逗号

# 算法1： 分段求解

```
24 cin>>a>>b;  
25 if (abs(a)<=500 && abs(b)<=500) {  
26     solve1();  
27 } else {  
28     solve2();  
29 }
```

先拿50分再说

```
4 void solve1() {  
5     if (a+b == 1000){  
6         cout<<"1,000"<<endl;  
7     }else if (a+b == -1000){  
8         cout<<"-1,000"<<endl;  
9     }else {  
10        cout<<a+b<<endl;  
11    }  
12 }
```

# 手算找规律

通过手算多组数据寻找灵感

123

123

-999

-999

-1000

-1,000

-1234567

-1,234,567

从右往左数，第三个字符c修改为逗号。

有两个特殊情况需排除：

- 1.待修改字符刚好是0号位置
- 2.待修改字符虽然是1号位置，但0号是'-'

# 算法2： 模拟+字符处理

```
13 void solve2() {  
14     string s = to_string(a + b);  
15     for(int i=s.size()-3; i>=0; i-=3)  
16         if(i>=1 && s[i-1]!='-')  
17             s.insert(i, ",");  
18     cout<<s<<endl;  
19 }
```

如何手动实现to\_string()

```
4 string toString(int n){  
5     if (n==0) return "0";  
6     bool isF=false;  
7     if(n<0) n=-n,isF=true;  
8     string s;  
9     while (n != 0) {  
10         s = s + char(n % 10 + '0');  
11         n /= 10;  
12     }  
13     reverse(s.begin(), s.end());  
14     if (isF) s.insert(0, "-");  
15     return s;  
16 }
```

注意0需要特判

3065



# 电影节

## 请同学简述题意 突出核心要点

在数轴上共 $n$ 个区间， $i$ 号区间左端点 $st[i]$ ，右端点 $ed[i]$ ，对每个区间询问：其右端点右侧第一个遇到几号区间的左端点？

# 请同学分析部分分策略

## 【数据规模】

10%数据,  $n \leq 3$

50%数据,  $n \leq 10000$

100%数据,  $n \leq 200000$

# 暴力枚举决策

枚举询问区间的编号*i*  
枚举答案区间的编号*j*

判断*j*号区间的左端点  
是否在*i*号区间右端点右侧

尝试更新*i*号区间的答案

# 暴力枚举决策

```
3 const int INF=2e9;  
4 const int N=200009;  
5 int n,st[N],ed[N];
```

st是start缩写  
ed是end缩写

st[i]表示i号区间左端点  
ed[i]表示i号区间右端点

# 暴力枚举决策

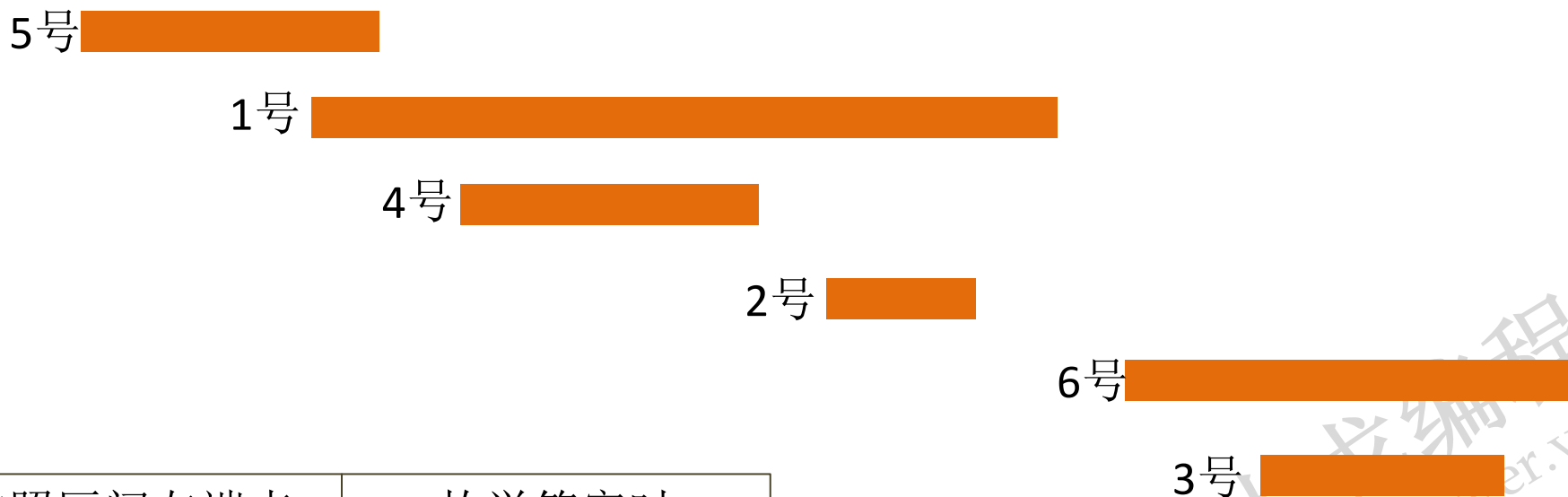
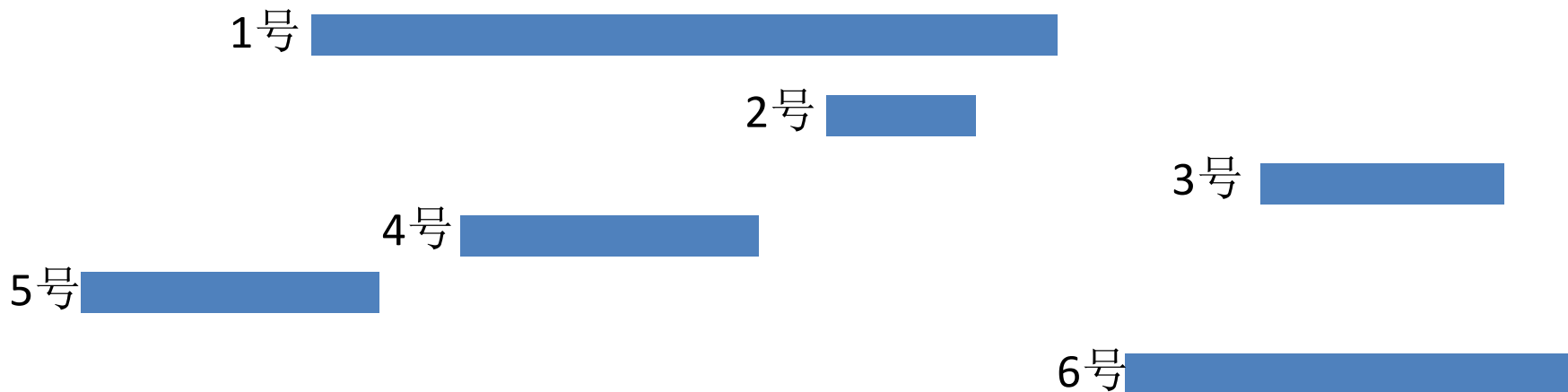
```
9  cin>>n;
10 for(int i=1;i<=n;i++)cin>>st[i];
11 for(int i=1;i<=n;i++)cin>>ed[i];
12 for(int i=1;i<=n;i++){
13     int ans=n+1;
14     int pos=INF;
15     for(int j=1;j<=n;j++){
16         if(
17             if(st[j]<pos){
18                 pos=st[j];
19                 ans=j;
20             }
21             else if(
22                 ans=j;
23         }
24     cout<<ans<<" ";
25 }
```

目前的暴力枚举  
无法提前跳出循环

如何加速？

排序！

区间排序的规则如何选择  
按照左端点排序？  
还是按照右端点排序？



按照区间左端点  
从小到大排序

枚举答案时  
可能提前跳出循环



# 暴力+加速

```
3  const int INF=2e9;
4  const int N=200009;
5  int n;
6  struct Movie{int st,ed,id;} d[N];
7  bool cmp(const Movie&a,const Movie&b){
8      if(a.st<b.st)return 1;
9      if(a.st>b.st)return 0;
10     if(a.id<b.id)return 1;
11     return 0;
12 }
```

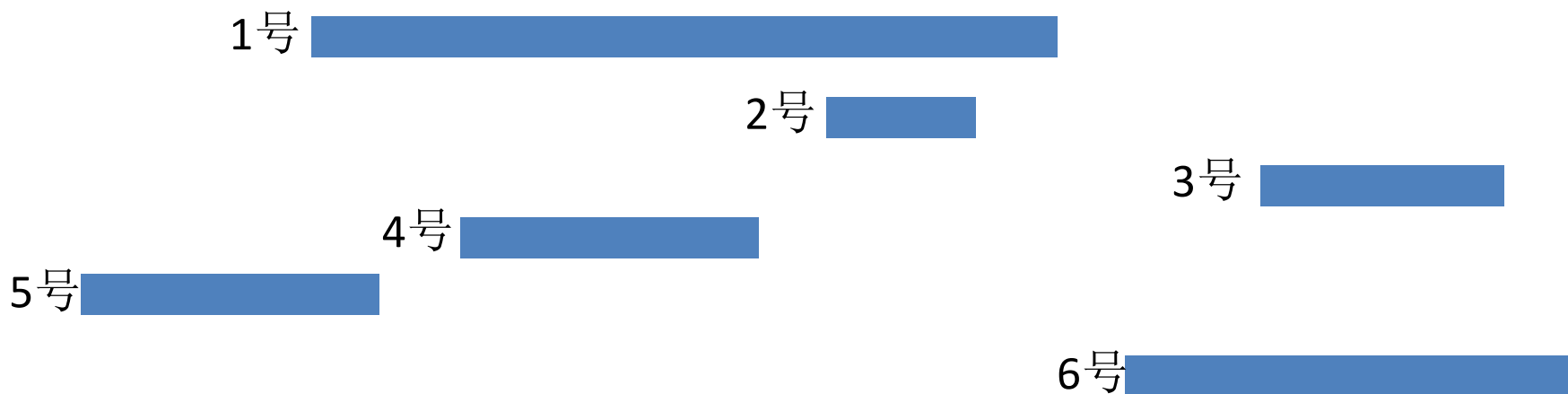
```
17 cin>>n;  
18 for(int i=1;i<=n;i++)cin>>d[i].st;  
19 for(int i=1;i<=n;i++)cin>>d[i].ed;  
20 for(int i=1;i<=n;i++)d[i].id=i;  
21 sort(d+1,d+1+n,cmp);
```

```
22 d[n+1]=(Movie){INF,INF,};
23 for(int i=1;i<=n;i++)
24     for(int j=1;j<=n+1;j++)
25         if(d[i].ed<d[j].st){
26             
27             break;
28         }
29 for(int i=1;i<=n;i++)
30     cout<<ans[i]<<" ";
31 cout<<endl;
```

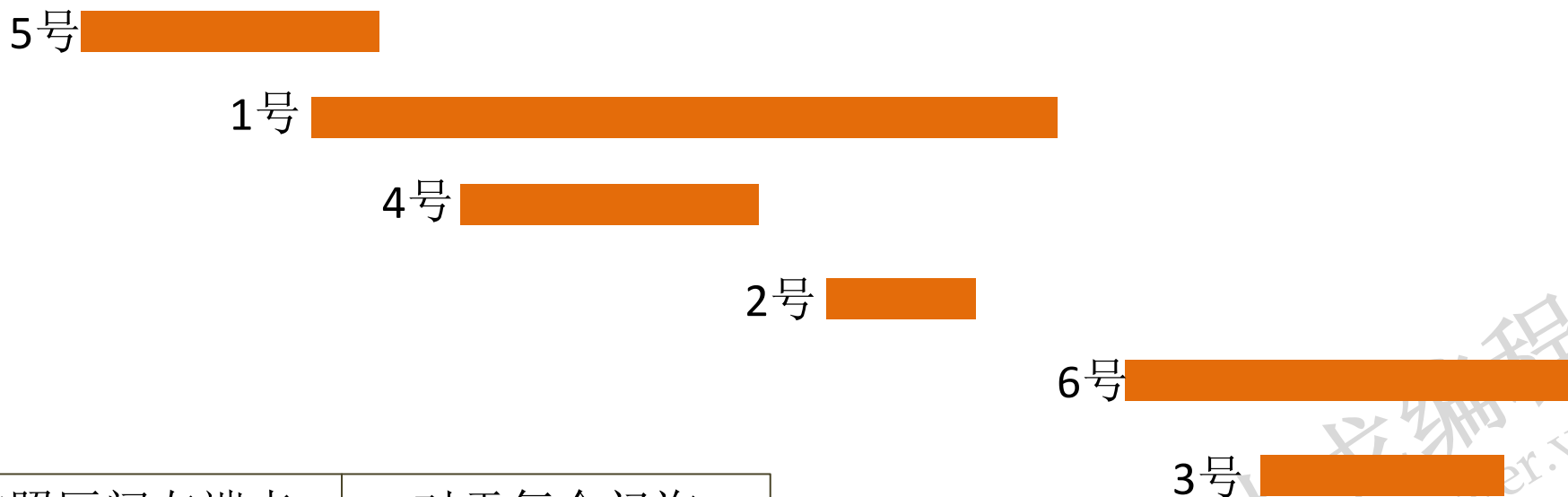
## 正解1

区间右端点代表询问  
区间左端点代表答案

区间排序后枚举 $n$ 个询问  
二分法定位答案位置



最小化ans, 使得: 排序后第ans部电影开始时间在i号电影结束后



按照区间左端点  
从小到大排序

对于每个问询  
二分定位答案

```
3  const int N=200009;
4  struct Info{int t,id;} st[N],ed[N];
5  bool cmp(const Info&a,const Info&b){
6      if(a.t<b.t)return 1;
7      if(a.t>b.t)return 0;
8      if(a.id<b.id)return 1;
9      return 0;
10 }
```

```
15  cin>>n;
16  for(int i=1;i<=n;++i){
17      cin>>st[i].t;
18      st[i].id=i;
19  }
20  for(int i=1;i<=n;++i){
21      cin>>ed[i].t;
22      ed[i].id=i;
23  }
24  sort(st+1,st+1+n,cmp);
```

最小化ans,使得:排序后第ans部电影开始时间在i号电影结束后

```
24  sort(st+1,st+1+n,cmp);
25  for(int i=1;i<=n;++i){
26      int l=1;
27      int r=n;
28      int ans=n+1;
29      while(l<=r){
30          int mid=l+(r-l)/2;
31          if(ed[i].t<st[mid].t)
32              
33          else
34              
35      }
36      if(ans==n+1)cout<<n+1<<" ";
37      else cout<<<<" ";
38  }
```



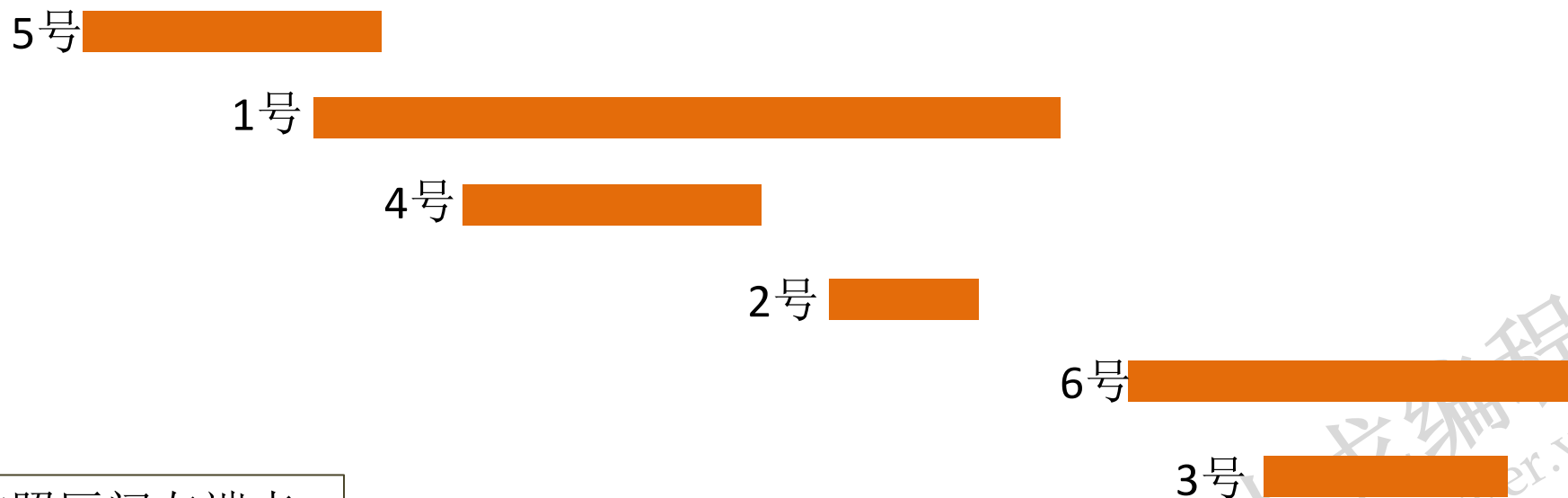
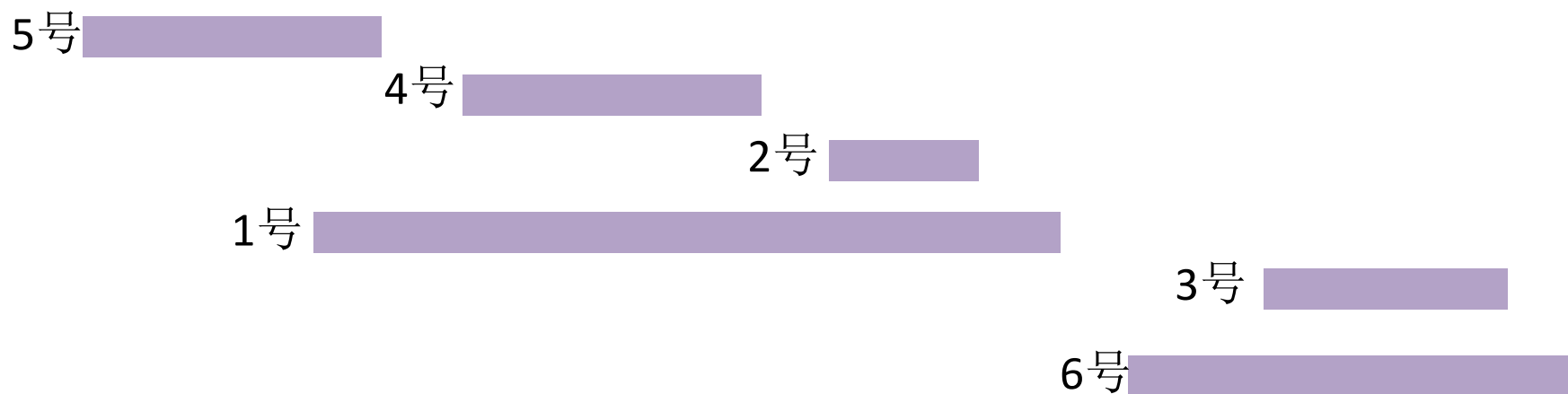
## 正解2

区间右端点代表询问  
区间左端点代表答案

区间右端询问点从小到大排序  
区间左端答案点从小到大排序

双游标算法  
主要的游标枚举询问  
辅助的游标枚举答案

按照区间右端点  
从小到大排序



按照区间左端点  
从小到大排序

```
3  const int N=200009;
4  struct Info{int t,id;} st[N],ed[N];
5  bool cmp(const Info&a,const Info&b){
6      if(a.t<b.t)return 1;
7      if(a.t>b.t)return 0;
8      if(a.id<b.id)return 1;
9      return 0;
10 }
```

```
16 cin>>n;
17 for(int i=1;i<=n;++i){
18     cin>>st[i].t;
19     st[i].id=i;
20 }
21 for(int i=1;i<=n;++i){
22     cin>>ed[i].t;
23     ed[i].id=i;
24 }
25 sort(st+1,st+1+n,cmp);
26 sort(ed+1,ed+1+n,cmp);
```

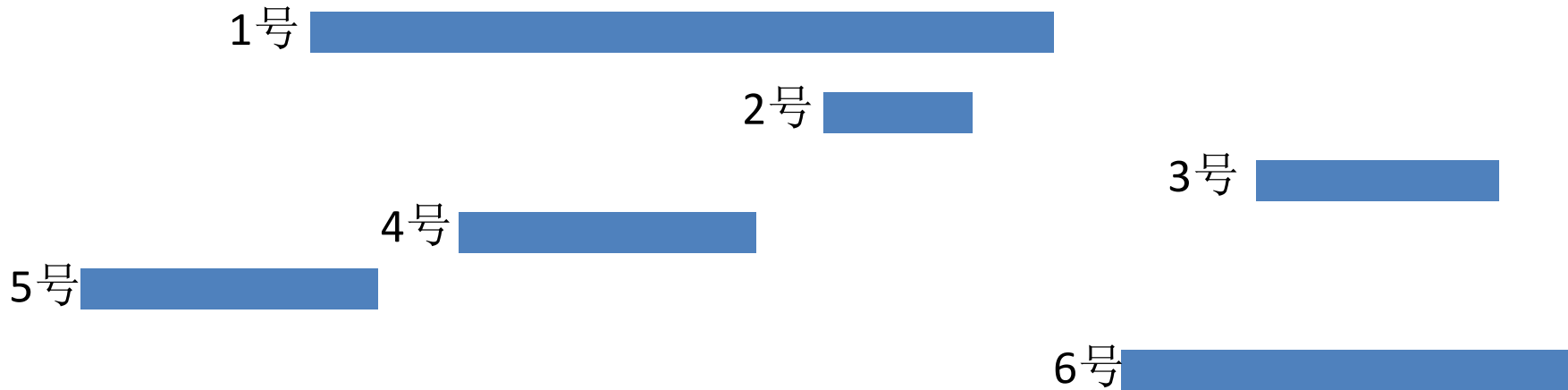
```
25 sort(st+1,st+1+n,cmp);
26 sort(ed+1,ed+1+n,cmp);
27 int j=1;
28 for(int i=1;i<=n;++i){
29     while()
30         ++j;
31     if(j==n+1) ans[ed[i].id]=n+1;
32     else ans[ed[i].id]=st[j].id;
33 }
```

正解3

离线询问

挑选最容易回答的问题先回答

按照从右往左  
查看每个端点



5

1

5

4

4

2

2

1

6

3

3

6

所有区间端点  
从小到大排序

```
4 const int N=200009;
5 struct Point{
6     int id,x;
7     bool L;
8 } p[N*2];
9 bool cmp(const Point&a,const Point&b){
10     if(a.x>b.x)return 1;
11     if(a.x<b.x)return 0;
12     if(a.L<b.L)return 1;
13     if(a.L>b.L)return 0;
14     if(a.id>b.id)return 1;
15     return 0;
16 }
```



```
21  cin>>n;
22  for(int i=1;i<=n;i++){
23      cin>>p[i].x;
24      p[i].id=i;
25      p[i].L=1;
26  }
27  for(int i=1;i<=n;i++){
28      cin>>p[i+n].x;
29      p[i+n].id=i;
30      p[i+n].L=0;
31  }
32  int nP=2*n;
33  sort(p+1,p+1+nP,cmp);
```

```
32  int nP=2*n;  
33  sort(p+1,p+1+nP,cmp);  
34  int nxt=n+1;  
35  for(int i=1;i<=nP;i++)  
36      if(p[i].L)   
37      else 
```

# 解法汇总

纯暴力枚举

区间排序+暴力+提前跳出循环

区间排序+枚举询问+二分定位答案

区间排序+双游标枚举询问和答案

离线询问+逆向求解

1119

# 炼金术

请同学写出题目大意  
已知什么求什么

给出编号**1**到**N**的原数组 $a_i$   
再给出**K**个配方  
各配方可通过减少一个或某几个**1**单位 $a_{p_i}$   
制造出**1**单位的 $a_q$  ( $q$ 大于 $p_i$ )  
求一系列转化后 $a_N$ 的最大值

请同学阅读[数据规模和约定]  
识别部分得分点

【数据规模与约定】

$1 \leq i \leq N \leq 100$ 。  $0 \leq a_i \leq 10^4$ 。  $1 \leq K < N$ 。

测试点 2 中，对于  $1 \leq i < N$ ，一单位金属  $i$  可以被转化为一单位金属  $i+1$ 。

测试点 3-4 中，每个配方均将一单位的一种金属转化为另一种金属。

测试点 5-11 没有额外限制。

对于每种金属，最多知道一种制造该金属的配方。

### 输入样例

3

2 0 0

2

3 2 1 2

2 1 1

### 输出样例

1

用配方2:

将1个 $a_1$  -> 1个 $a_2$

a序列:

2 0 0 -> 1 1 0

再用配方1:

将1个 $a_1$ 和1个 $a_2$  -> 1个 $a_3$

a序列:

1 1 0 -> 0 0 1

### 输入样例

5

2 0 0 1 0

3

5 2 3 4

2 1 1

3 1 2

### 输出样例

1

### 转化步骤

- 1: 将1个 $a_1$ 转化为1个 $a_2$
- 2: 再将1个 $a_2$ 转化为1个 $a_3$
- 3: 最后将1个 $a_3$ 和1个 $a_4$   
转化为1个 $a_5$



## 二分答案

### 枚举答案：

猜想 $n$ 号的某个数值,检验通过转化能否达标  
考虑到随 $n$ 号期望值增大,达标可能性单调降低  
可二分枚举答案

$ans$ 保底是 $a_n$

$l=a_n+1, r=\sum a_i$

# 算法分析

定义: 为达目标 $i$ 号需求量是 $b_i$   
 $b_n$ 初始化为目标 $x$ , 其余 $b_i$ 初始化为0

从后往前检查 $i$ 号

若需求 $b_i$ 超过存量 $a_i$ ,

且没有可生成 $i$ 的配方, 则失败

若 $b_i$ 超出 $a_i$ 的量,

超过了前 $i-1$ 号的存量总和, 也失败  
(哪怕所有前序值都1对1地转成 $i$ 号也不够)

若存量 $a_i$ 足够满足需求 $b_i$ , 则跳过 $i$

否则, 拿出可生成 $i$ 的唯一配方  
给每个原料 $j$ 的需求量 $b_j$ 都加上 $b_i - a_i$   
把希望寄托在更前面的点上

$s[i]$ 前 $i$ 项之和

$to[i]$ 是生成 $i$ 号的配方序列

```
4 int n,a[N],b[N],s[N];
5 vector<int> to[N];
6 bool OK(int x){
7     for(int i=1;i<=n;i++) b[i]=(i==n?x:0);
8     for(int i=n;i>=1;i--){
9         if(b[i]>a[i]&& [ ] ) return 0;
10        if(b[i]<=a[i]) continue;
11        if(b[i]-a[i]>s[i-1]) return 0;
12        for(int j=0;j<to[i].size();j++)
13            [ ]
14    }
15    return 1;
16 }
```

请逐行翻译代码  
并分析整个程序时间复杂度

3054



# 混乱



请同学简述题意  
突出核心要点

字符串长度为 $n$ ,都是a到z的字母组成,  
最小化: 子串内字母种类/子串长度

# [数据规模]启示录

保证有20%数据，  
 $n \leq 2000$

提示 $O(n^2)$ 的  
暴力算法

保证有20%数据，  
字符串只包含a和b

提示简化问题  
启发思路

保证有100%数据，  
 $n \leq 100000$

最难部分

# 暴力枚举决策

枚举左端点a  
枚举右端点b

统计编号[a,b]区间内  
共有多少个不同的字母能否全部清零

计算子串内字母种类/子串长度  
更新最优解



$O(n^2)$

20分

# 暴力枚举决策

```
10 cin>>n>>s;
11 for(int i=1;i<=n;++i) d[i]=s[i-1]-'a';
12 int fenzi=1,fenmu=1;
13 for(int a=1;a<=n;++a){
14     for(int i=0;i<26;++i)ok[i]=0;
15     int cDiff=0;
16     int len=0;
17     for(int b=a;b<=n;++b){
18         ++len;
19         if(
20             ok[d[b]]=1;
21             ++cDiff;
22         }
23         if(fenzi*len>fenmu*cDiff||
24             fenzi*len==fenmu*cDiff&&len>fenmu)
25             fenzi=cDiff,fenmu=len;
26     }
27 }
28 cout<<fenzi<<"/"<<fenmu<<endl;
```

ok[i]表示i是否出现过

cDiff表示不同字母个数

len表示子串长度

fenzi/fenmu  
>cDiff/len

# 简化问题启发思路

字符串只包含a

字符串只包含a和b

字符串只包含a,b,c

总结经验，再推广到一般情况

字符串  
只包含a

aaaaaaaa

请同学算出答案

1/7

1/字符串总长度

字符串  
只含ab

aabbaaaa

请同学写出答案

1/4,2/8里选2/8为答案

含1种

含2种

1/只含1种字符的子串最长长度

2/字符串总长度

2种情况  
取最优

字符串  
只含abc

aabbaca

请同学写出答案

1/2,2/5,3/7里选2/5为答案

含1种  
含2种  
含3种

1/只含1种字符的子串最长长度
2/只含2种字符的子串最长长度
3/字符串总长度

3种情况  
取最优

# 简化问题 推广到原问题

字符串只包含a

字符串只包含a和b

字符串只包含a,b,c

总结经验，再推广到一般情况

请同学描述满分算法

枚举分子 $x$ ，即子串包含字符种类数

算出恰包含 $x$ 种字符的子串最长长度 $len$

# 简化问题 推广到原问题

字符串只包含a

字符串只包含a和b

字符串只包含a,b,c

总结经验，再推广到一般情况

请同学描述满分算法

枚举分子 $x$ ，即子串包含字符种类数

算出恰包含 $x$ 种字符的子串最长长度 $len$



蠕动区间

尝试用 $x/len$ 更新最优解

将字符串平移从1号开始

```
35 string s;  
36 cin>>n>>s;  
37 for(int i=1;i<=n;++i) d[i]=s[i-1]-'a';  
38 int nDiff=countDiff();  
39 int fenzi=1,fenmu=1;  
40 for(int x=1;x<=nDiff;++x){  
41     int len=solve(x);  
42     if(fenzi*len>fenmu*x||  
43         )  
44         fenzi=x,fenmu=len;  
45 }  
46 cout<<fenzi<<"/"<<fenmu<<endl;
```



```
3  const int N=100009;
4  int n,d[N];
5  int countDiff(){
6      set<int> s;
7      for(int i=1;i<=n;++i)
8          
9      return 
10 }
```

统计字符串内  
共有几种字符

```
11 int solve(int C){
12     int l=1,r=1;
13     int cDiff=0;
14     int ret=1;
15     int cnt[30]={0};
16     while(1){
17         while(r<=n&&cDiff<=C){
18             if(cnt[d[r]]==0)
19                 ++cDiff;
20             ++cnt[d[r]];
21             ++r;
22         }
23         if(cDiff<=C)ret=max(ret,r-1);
24         else ret=max(ret,r-1-1);
25         if(r>n)return ret;
26
27
28
29
30     }
31 }
```

求出恰包含C种字符的子串最长长度

当前子串编号[l,r-1]，若l==r表示空串

cDiff表示当前不同字符种类数

cnt[i]表示当前子串i出现几次

2109