


CS153

A Mars rover, likely a Curiosity rover, is shown on a rocky, sandy surface under a blue sky. The rover is equipped with various scientific instruments and cameras. A long shadow is cast by the rover onto the ground. The entire image has a blue color overlay.

算法
建模

课件下载链接:

<https://pan.baidu.com/s/1htbqTfA>

密码: imfv

作业网站:

<http://120.132.18.213:8080/thrall-web/main#home>

区间模型2

不重叠区间最少分组数

有 n 个区间分成若干组，求最少分组数量，
能保证每组内区间互相不重叠

教室安排

共有 n 门课要安排教室，每门课 i 都有一个开始时间 s_i 和结束时间 t_i 。如果两门课不重叠（结束时间和开始时间也不能重叠）就可以安排在同一个教室。至少需要几间教室？
 $0 \leq n \leq 100$ ，其他输入都小于10000

样例输入

```
4
1 2
5 7
1 4
3 6
```

样例输出

```
2
```

样例输入

```
3
1 3
3 4
1 4
```

样例输出

```
3
```

世界杯

世界杯小组赛如火如荼地开展，今天共有 n 场比赛，每场比赛 i 都有一个开始时间 t_i （以分钟为单位），如果两场比赛的开始时间相差大于等于200分钟，那么这两场球赛可以安排在一个足球场进行。否则，由于时间间隔太短容易引起骚乱，必须将两场球赛分开到不同球场。输入第一行为 n ，第二行为 n 个正整数分别代表每场比赛的开赛时间。输出至少需要几个足球场。

$0 \leq n \leq 16$ ，其他输入都小于10000

样例输入

5
30 10 20 200 210

样例输入

2
30 30

样例输出

4

样例输出

2

每门课：抽象成一个区间

样例输入

5
1 2
5 7
1 4
3 6

每门课对应一个区间
区间左端点为 开课时间
区间右端点为 介绍时间

样例输出

2



每场比赛：抽象成一个区间

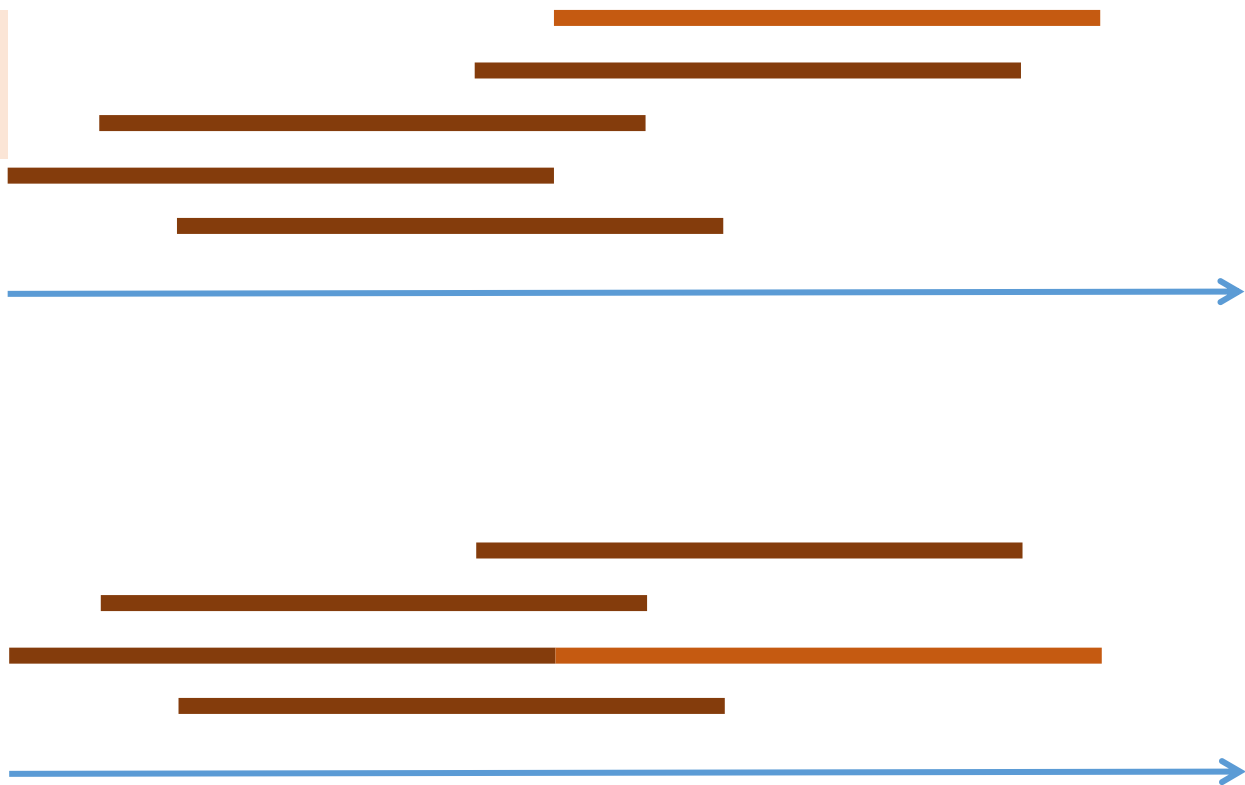
样例输入

5
30 10 20 200 210

每场比赛对应一个区间
区间左端点为 开赛时间
区间右端点为 开赛时间+200分钟

样例输出

4



现场算法讨论

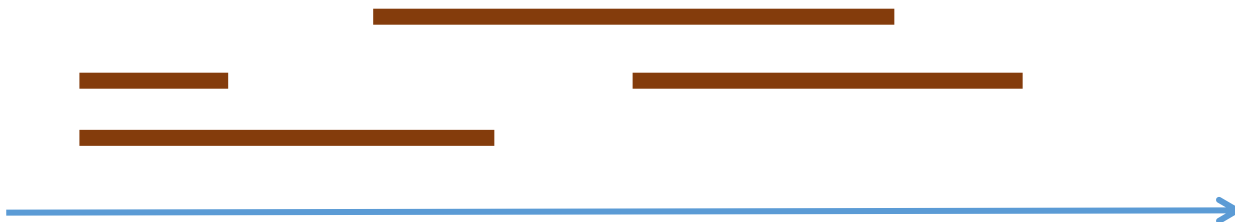
以下算法是否正确？
如果正确，请证明
如果不正确，请找出反例

教室安排 算法1

初始化教室数量为零

逐个开放教室直到所有课程都被安排完：

对于每个教室，**尽量排最多门课**



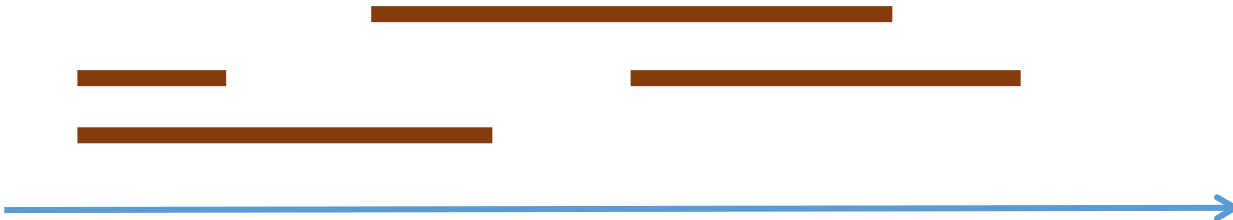
教室安排 算法2

初始化教室数量为零

按照**结束时间**先后
依次查看每堂课：

如果所有目前教室都无法上这节课
就新增一间教室来上这节课

否则就在可以使用的教室里选**最早空闲**
的教室上这节课



教室安排 算法3

初始化教室数量为零

按照**结束时间**先后
依次查看每堂课：

如果所有目前教室都无法上这堂课
就新增一间教室来上这堂课

否则就在可以使用的教室里选**最晚空闲**
的教室上这堂课

教室安排 算法4

初始化教室数量为零

按照**开始时间先后**
依次查看每堂课：

如果所有目前教室都无法上这堂课
就新增一间教室来上这堂课

否则就在可以使用的教室里**选任意一间**
的教室上这堂课

教室安排： 算法3

```
5 struct ke{int s,t;};
6 bool cmp(const ke& a,const ke& b){
7     return a.t<b.t||a.t==b.t&&a.s<b.s;
8 }
9 ke d[N];
```

定义ke类型:课的开始和结束时间

定义两课间比较顺序

定义d数组包含N门课程

```
11 int n,i,j,k,x[N],cnt=0;
12 cin>>n;
13 for(i=0;i<n;i++) cin>>d[i].s>>d[i].t;
14 sort(d,d+n,cmp);
15 d[n]=-1;
16 for(i=0;i<n;i++){
17     for(j=0,k=n;j<cnt;j++)
18         if(d[i].s>x[j]&&x[j]>x[k]) k=j;
19     if(k<n) x[k]=d[i].t;
20     else x[cnt++]=d[i].t;
21 }
22 cout<<cnt<<endl;
```

x[i]为教室i里目前课程结束时间

cnt为教室总数

按照课程结束时间从小到大排序

设置临时教室

按照结束时间 安排每门课程

查看已有教室,找出最晚空闲的教室k

如果找到空教室,就在教室k上课i

否则增加新空教室,上课i

教室安排： 算法4

```
5 struct ke{int s,t};
```

定义ke类型:课的开始和结束时间

```
6 bool cmp(const ke& a,const ke& b){
```

```
7     return a.s<b.s||a.s==b.s&&a.t<b.t;
```

```
8 }
```

定义两课间比较顺序

```
9 ke d[N];
```

定义d数组包含N门课程

```
11 int n,i,j,x[N],cnt=0;
```

```
12 cin>>n;
```

```
13 for(i=0;i<n;i++) cin>>d[i].s>>d[i].t;
```

```
14 sort(d,d+n,cmp);
```

```
15 for(i=0;i<n;i++){
```

```
16     for(j=0;j<cnt;j++)
```

```
17         if(d[i].s>x[j]) break;
```

```
18     if(j<cnt) x[j]=d[i].t;
```

```
19     else x[cnt++]=d[i].t;
```

```
20 }
```

```
21 cout<<cnt<<endl;
```

x[i]为教室i里目前课程结束时间

cnt为教室总数

按照课程开始时间从小到大排序

按照开始时间 安排每门课程

查看已有教室,找出任意空闲教室j

如果找到空教室,就在教室j上课i

否则增加新空教室,上课i

世界杯:算法4

定义**game**类型:
开赛和结束时间

定义**cmp**比较规则:
开赛时间越早越靠前

定义一个数组
game d[N];
存放所有比赛信息

定义整数数组**x[N]**:
x[i]记录*i*号足球场里
目前比赛结束时间



数据结构

按照**开赛时间先后**循环查看每场比赛*i*:

循环查看目前所有已有的球场,寻找空闲球场

如果找到任意空闲球场*j*,就在球场*j*举办比赛*i*

否则新增一个球场举办比赛*i*



算法步骤

```
1 #include<iostream>
2 #include<algorithm>
3 #define N 20
4 using namespace std;
5 struct game{int s,t;};
6 bool cmp(const game& a,const game& b){
7     return a.s<b.s||a.s==b.s&&a.t<b.t;
8 }
9 game d[N];
10 int main(){
11     int n,i,j,x[N],cnt=0;
12     cin>>n;
13     for(i=0;i<n;i++) {
14         cin>>d[i].s;
15         d[i].t=d[i].s+200;
16     }
17     sort(d,d+n,cmp);
18     for(i=0;i<n;i++){
19         for(j=0;j<cnt;j++)
20             if(d[i].s>=x[j]) break;
21         if(j<cnt) x[j]=d[i].t;
22         else x[cnt++]=d[i].t;
23     }
24     cout<<cnt<<endl;
25     return 0;
26 }
```


自编题

仿照课堂例题，请自编一道编程题

要求以“**最大不重叠区间数**”的算法为核心求解步骤。

鼓励加入各类算法元素，构成原问题的变种形式。

参考资料

周小博，浅谈信息学竞赛中的区间问题
2008年信息学国家集训队论文

区间模型2

综合练习