马上开始

35tang-C++竞赛系列三阶课程

三五堂

《35tang-C++竞赛系列三阶课程》

学习方法

- 听课要思考
- 例题和练习要搞懂，可以看懂了自己尝试去写，如果作业没有时间先把课堂的例题和练习自己做一下。
- 多练才能熟悉，课后作业最好能做，实在没有时间后面找时间做
- 参考别人的程序是非常好的，比如作业，但是最好不要直接复制完整代码，看懂别人的思路后自己写一遍
- 作业，尤其是原题，一定要自己学会编译调试并验证正确性

greedy

- 从开始状态一步一步推导到最终状态，中间每一次都是选择最优的解决方案，一般用反证法验证是否正确，这也是最难的地方。
- 其实说简单点：就是不需要枚举所有的状态，而是直接指定一个策略，按照这个策略去选择下一步。

看电影

电影节播放n(1=<n<=2*10^5)个电影，每一个电影都有开始播放的时间和结束播放的时间，求你最多可以看几个完整的电影。

输入：第一行一个整数n，表示n个电影；接下来n行，每行空格隔开2个整数，分别表示电影开始和结束的时间（整数不超过10的9次方并且大于等于1）

输出： 一个整数，你最多可以看的完整电影数目。

输入示例：

3

3 5

4 9

5 8

输出示例

2

怎么贪?

3 5
7 9
5 8
5 7

- 先看哪一个？当然是第1个，因为结束的早。结束后再看第4个，因为结束的早，接下来看第2个。
- 怎么才能看尽可能多的电影?
- 尽可能看早结束的，这样就能早一点空出来看后面的可能的其他电影?
- 所以，按照结束时间排序。然后遍历数组，看当前的电影的开始时间是否比上一次的结束时间要晚，如果晚，就看这个电影，并更新上一次的结束时间。

写

贪心代码一般很简单

```
sort(movies+1,movies+n+1,mycmp);//按照结束时间排序
int freetime=0;//空闲时间，也就是上一个看的电影结束时间
for (int i=1;i<=n;i++)
{//由于按照时间排序的，所以最先找到的就是结束时间最近的
    if (movies[i].starttime>=freetime) {
        ans++;
        freetime=movies[i].endtime;
    }
}
cout<<ans<<endl;
```

USACO training Section 1.4 PROB Barn Repair

n个stall（牛棚），有的有牛，给定m个boards（板子，长度不限制）把有牛的盖起来，这些board只能覆盖连续的stall，比如4号和6号stall有牛，如果把4，6用一个board盖起来，长度是3。求覆盖所有有牛的stall的boards总长度最小值。

示例输入如右面，一共50个牛棚，其中18个有牛，有牛的牛棚的编号都给出来了，要求用4个板子覆盖有牛的牛棚。

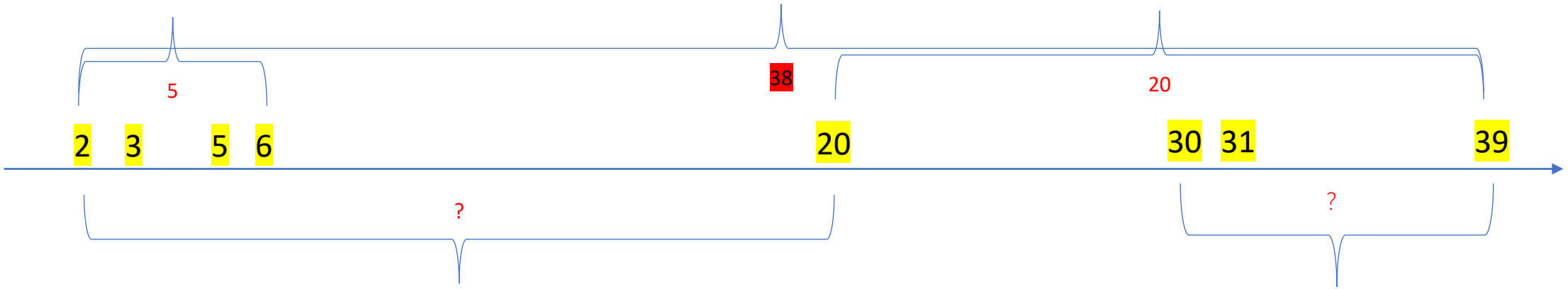
25

[One minimum arrangement is one board covering stalls 3-8, one covering 14-21, one covering 25-31, and one covering 40-43.]

```
4 50 18
3
4
6
8
14
15
16
17
21
25
26
27
30
31
40
41
42
43
```

下面给出8个牛棚有牛，如果有足够多的板子，比如8个板子，显然长度最小，就是8。如果只有1个板子呢？只有2个呢？

- 2 3 5 6 20 30 31 39



算法

- 我们把有牛的stall和stall的间隔都排序，先盖上一个，算出总长度，然后不断的找最大的间隔，找到一个就多用一个board，总长度减去这个间隔，直到board用完。
- 为什么？因为间隔最大的牛棚占用的板子长度最大。

```
fin >> m>> s>>c;
if (m>=c)//如果给的board比有牛的stall还多，那就1个牛棚一个板子，c个牛棚用c个板子
{
		fout << c << endl;
		return 0;
}
int r=0;//答案放在这里面
for(int i=1;i<=c;i++) {//读入有牛的牛棚编号
		fin>>stalls[i];
}
sort(stalls+1,stalls+c+1); //有牛的stall排序
//计算每一个有牛的stall和后面的stall的间距并且排序，c个牛棚有c-1个间隔
for(int i=1;i<=c-1;i++){
		disstalls[i]=stalls[i+1]-stalls[i]-1;
}
sort(disstalls+1,disstalls+c);
//先用一个board全部盖上，然后一块一块的加board，每加一个，把最大当前的最大间距减掉
r=stalls[c]-stalls[1]+1;//注意间隔问题长度要+1
for (int i=c-1;i>=c-1-m+2;i--){//在间隔数组里面从后往前也就是从大到小一共找到m-1个
		r-=disstalls[i];
}
fout << r << endl;
```

NOIP 2008排座椅

- 同学们在教室中坐成了 M 行 N 列（行列从1开始编号），坐在第 i 行第 j 列的同学的位置是 (i,j)，为了方便同学们进出，在教室中设置了 K条横向的通道，L 条纵向的通道。如果一条通道隔开了2 个会交头接耳的同学，那么他们就不会交头接耳了。找出最好的通道划分方案。在该方案下，上课时交头接耳的学生的对数最少。要求输出最优情况下哪些行和哪些列后面有通道。

理解

4 5 1 2 3

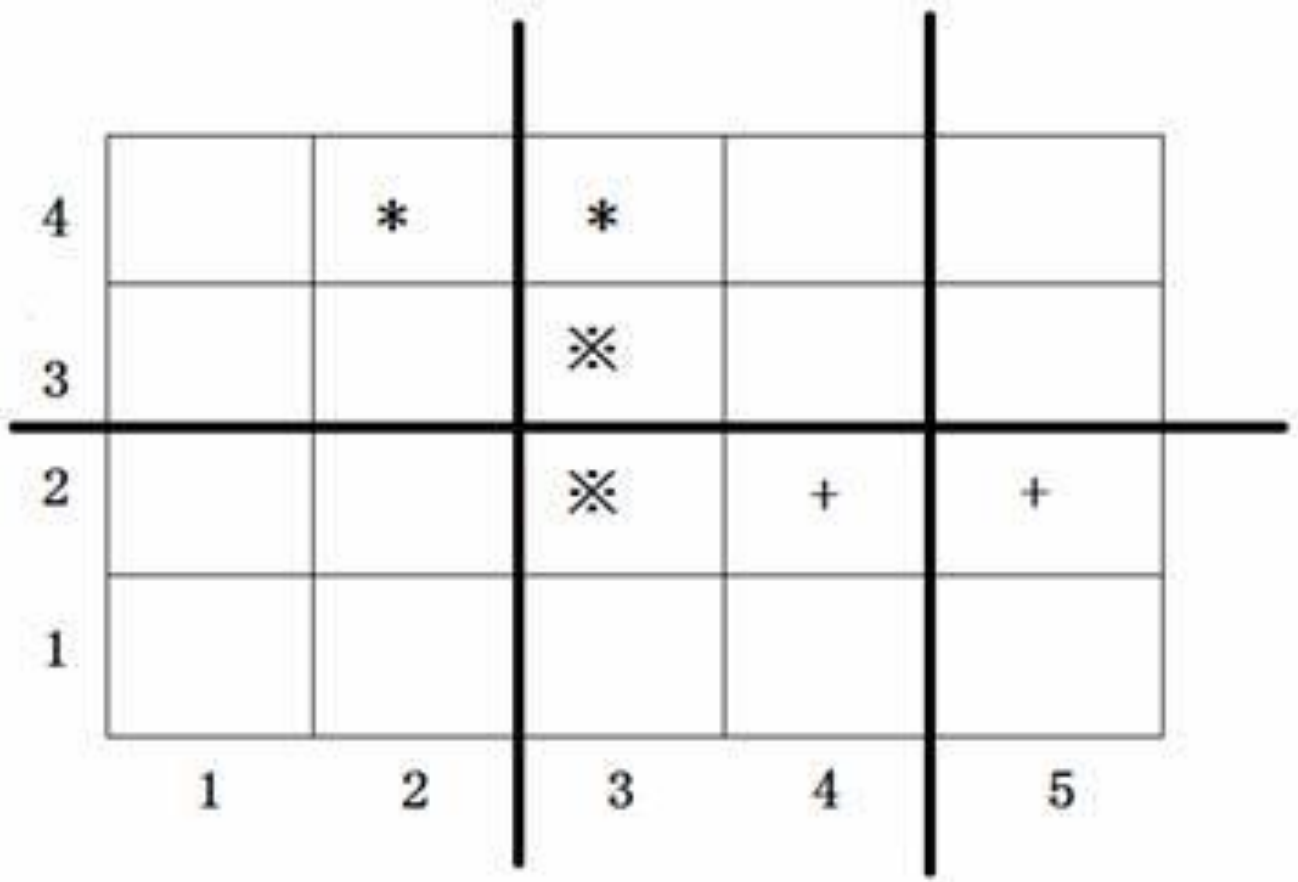
4 2 4 3

2 3 3 3

2 5 2 4

4行5列，用1横2竖隔开

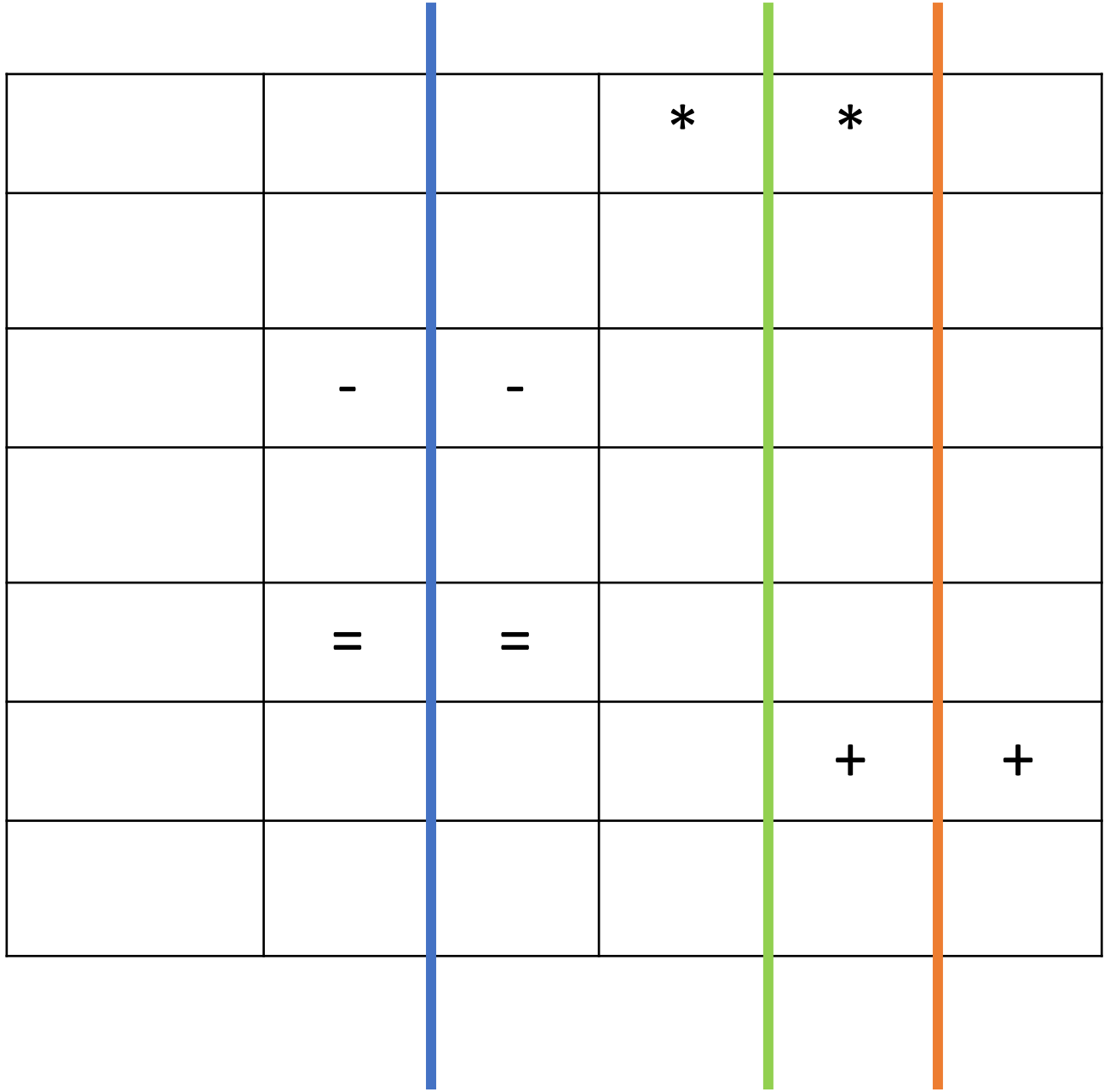
3对交头接耳



最好的也就是说方案不唯一，要最好的一个。这种题目大部分都是贪心

分析

一共4对说话的，如果用一条竖线，怎么隔离最多？如果2条线呢？



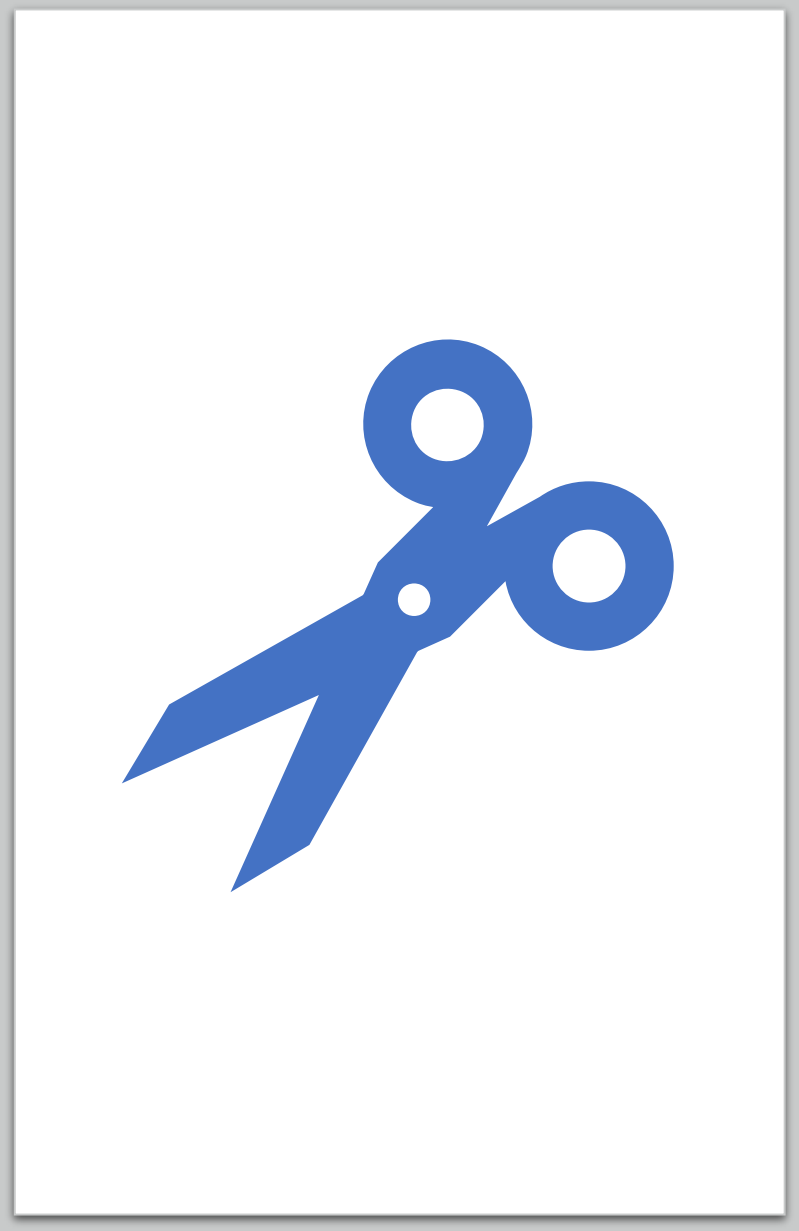
算法: 如果有K个横向通道可以用。

找M-1个位置（M行可能的通道位置M-1个）里面能隔开说话学生最多的K个位置：找到这M-1个位置分别能隔开多少说话的学生，然后找前K个最大的。注意：最后要求输出是哪一行之后隔离，所以flagrow表示的是某一个行之后能隔开多少。

纵向通道一样算法。

首先设计数据结构并读入计算某一个行或者列之后能隔开多少：如果一对说话的学生是4 2 4 3，也就是说4行2列有个人和4行3列的人说话，就是说在2列之后设置纵向通道可以把这对学生隔开，也就是可以flagcol[2]++;

如果所有的数据处理完毕，flagcol[2]中就是在2列之后设置纵向通道一共可以隔离开多少对说话的。



读入预处理

```
int m,n,k,l,d;
 fin >> m>>n>>k>>l>>d;
int flagcol[1001]={0};//每列后面加通道可以隔开的说话学生对数
int flagrow[1001]={0};//每行后面加通道可以隔开的说话学生对数
int r1,r2,c1,c2;
for (int i=1;i<=d;i++)
{
        fin>>r1>>c1>>r2>>c2;//读入说话学生
        if (r1==r2) flagcol[min(c1,c2)]++;//输入行相同，说明需要隔离不同列
        if (c1==c2) flagrow[min(r1,r2)]++;
}
```

怎么找最大的前k个?

- 接下来就简单了，或者对flagrow和flagcol排序，然后取其中最大的k个和l个
- 或者遍历多次，每次找最大的，找到后设置为-1.最后按照顺序输出值为-1的。
- 思考，用第一种方法除了找最大的k和l个，还要做什么处理?

- 需要按照大小排序，并且大小相同的话按照行或者列从小到大排序：因为需要按照行号列号排序，而第二个算法不需要，因为flagrow按照从小到大遍历就好了

遍历多次，每次找最大的，找到后设置为-1.

```
for (int i=1;i<=k;i++)
{
  //找到k个最大的 flagrow,找k次，每一次找到一个就设置为-1
  int maxr=0,maxindex=0;
  for (int j=1;j<=m;j++)
  //遍历当前的flagrow，找最大的，找过的是-1，不会再被找到
  {
    if (flagrow[j]>maxr)
    {
      maxindex=j;
      maxr=flagrow[j];
    }
  }
  flagrow[maxindex]=-1;
}
```

题目要求行尾没有空格

```
bool first=true;
for (int i=1;i<=m;i++)
{
    if (flagrow[i]==-1)
    {
        if (first) {
            cout<<i;
            first=false;

        }
        else  cout<<" "<<i;
    }
}
cout<<endl;
```

NOIP 2010 三国游戏

N个武将两两之间有默契值，小涵和计算机先后选择武将，计算机一方选择武将的原则是尽量破坏对手下一步将形成的最强组合，它采取的具体策略如下：任何时刻，轮到计算机挑选时，它会尝试将对手军队中的每个武将与当前每个自由武将进行一一配对，找出所有配对中默契值最高的那对武将组合，并将该组合中的自由武将选入自己的军队。拥有更高默契值的一对武将组合获胜，问小涵能否获胜，选中的最大武将默契值是多少？如右图，显然，如果小涵可以选到5号和4号武将，得到33，一定是最大的，但是计算机会尽可能的破坏。

武将编号	1	2	3	4	5	6
1		5	28	16	29	27
2	5		23	3	20	1
3	28	23		8	32	26
4	16	3	8		33	11
5	29	20	32	33		12
6	27	1	26	11	12	

一起玩一下：
老师帮助小涵用√，同学们帮助机器用 x ，小涵先选。

武将编号	1	2	3	4	5	6
1		5	28	16	29	50
2	5		23	3	20	1
3	28	23		8	32	26
4	16	3	8		33	11
5	29	20	32	33		12
6	50	1	26	11	12	

分析

- 给出的6个武将的例子，他们形成一个矩阵，矩阵里面的值是默契值，行列是武将编号。
- 最大的武将默契值就是每一行最大的那个数字，但是由于计算机的策略，这个取不到，比如上面最大的是33，武将4，5匹配，如果我们选择了4或者5，计算机一定会选择5或者4。那么，退一步，取第二大的行不行，是可以取到的，因为计算机总是取选最大的配对。比如选择第5号武将，也就是第5行，这个时候计算机为了破坏我们选4号，一定去选择4，所以我们下一步就可以选3号，从而得到第二大32这个数字。
- 程序就很简单了，就是遍历所有行：把所有行中第二大找出来，在里面找出最大的一个。这就是贪心的答案。

武将编号	1	2	3	4	5	6
1		5	28	16	29	27
2	5		23	3	20	1
3	28	23		8	32	26
4	16	3	8		33	11
5	29	20	32	33		12
6	27	1	26	11	12	

注意：二维数组中的一维排序

```
//v[i][j]表示武将i和j之间的默契值，行列从1开始而不是0
int r=0;
 for (int i=1;i<=n;i++)
{
        //找每一行第二大中的最大
        sort(v[i]+1,v[i]+1+n); //排序第i行
        if (v[i][n-1]>r) r=v[i][n-1];//打擂法找第二大中的最大
}
```

贪心的适用性

不是所有问题都可以使用贪心算法。看这个著名的硬币问题，正解是后面的动态规划，但是我们这里看看贪心。

给定硬币面额，求组成一个amount最小需要的硬币数目？

很容易想到的就是贪心，从最大硬币开始尝试，放尽可能多的最大硬币，然后第二大...

如果给定序列是{1, 5, 10, 25} 这个是ok的。

可是如果给定{1, 5, 8, 10} 比如组成amount是13，显然是不行的，比如组成amount是13，按照贪心，会选择硬币10，1，1，1，但是实际上可以是8，5。

小tips

- 二阶期末练习：NOIP真题寻宝。输出要求：
  - 一个整数，表示打开宝箱的密钥，这个数可能会很大，请输出对20123取模的结果即可。
  - 比如我们结果是ans，那么就要输出ans%20123。
  - 而且如果ans在程序中不停的累加，每一次操作最后最好都能%20123，因为你不知道什么时候就太大了。

Tips-最大最小值

忘记头文件怎么办？
万能头文件
有的时侯本地编译通过，提交linux测试不同的编译器版本可能就会因为缺少头文件编译不通过

```
#include <climits>
#include <iostream>   //#include <bits/stdc++.h>
using namespace std;
int main(){
        int a=INT_MIN;
        cout<<"INT_MIN "<<a<<endl;
        int b=0x80000000;
        cout<<"0x80000000 "<<b<<endl;
        int c=-0x7fffffff;
        cout<<"-0x7fffffff "<<c<<endl;
        int d=INT_MAX;
        cout<<"INT_MAX "<<d<<endl;
        int e=0x7fffffff;
        cout<<"0x7fffffff "<<e<<endl;
        long long f =LLONG_MAX;
        cout<<"LLONG_MAX "<<(long long)(f)<<endl;
        long long g =0x7fffffffffffffff;
        cout<<"0x7fffffffffffffff "<<(long long)(g)<<endl;
        long long h =LLONG_MIN;
        cout<<"LLONG_MIN "<<(long long)(h)<<endl;
        long long i =0x8000000000000000;
        cout<<"0x8000000000000000 "<<(long long)(i)<<endl;
         return 0;
}
```

下节课例题预习

- 尝试理解题目，思考可能的算法，不需要做
- NOIP 2010 接水问题
- USACO 2018 December Contest, Bronze The Bucket List

作业1 apple.cpp

- 有三箱苹果。第一个箱子里有n1个苹果，第二箱有n2个，第三箱有n3个。吃掉一些苹果后要求：

  (1)每个箱子都不能空。

  (2)剩下苹果数量应该是递增的。也就是说，第一个箱子的苹果数少于第二个箱子，第二箱的苹果少于第三箱。

- 请你输出满足上述条件的情况下吃掉的最少的苹果数量。如果无法完成就输出-1。
- 输入
- 第一行，一个正整数n，表示数据的组数，n<=5

  接下来n行，每行三个正整数，依次代表n1，n2，n3，其中1<=n1,n2,n3<=30000
- 输出
- n行，每行一个整数，即题目要求的答案。

示例输入1：

1

15 40 22

示例输出1：

19

示例输入2：

2

1 3 1

1 1234 30000

示例输出2：

-1

0

挑战作业

- 2. USACO training Section 1.4 PROB Mixing Milk

https://www.luogu.com.cn/problem/P1208

  - 题目大意：需要购买一定量牛奶，有若干农名不同牛奶和价格，最少花多少钱（某一个农名的牛奶可以不卖光）。
  - 提示：贪心：从最便宜的开始买
- 3. NOIP 2007纪念品分组
  - 题目大意：不超过3万个纪念品都有价格，现在两两分组，要求每组的两个的价格和不超过W，问最少的分组数目？
  - 提示：贪心：每一次怎么选才能保证选出来的一组是“合适”的。

由易到难，思维体系训练
实战结合，创新协作培养
兴趣导向，未来职业引领

https://www.35tang.com

https://www.三五堂.com

扫码关注公众号

添加辅导老师