**4-1会场安排问题**

问题描述:假设在足够多的会场里安排一批活动，并希望使用尽可能少的会场。设计一个有效的贪心算法进行安排。（这个问题实际上是著名的图着色问题。若将每一个活动作为图的一个顶点，不相容活动间用边相连。使相邻顶点有不同颜色的最小着色数，相当于要找的最小会场数）。  
算法设计:对于给定的k个待安排的活动，计算使用最少会场的时间表。  
数据输入:由文件input.txt给出输入数据。第1行有一个正整数k,表示有k个待安排的活动。接下来的k行中，每行有两个正整数，分别表示k个待安排的活动的开始时间和结束时间。时间以0点开始的分钟计。  
结果输出:将计算的最少会场数输出到文件output.txto  
输入文件示例,  
输出文件示例  
Input.txt output.txt  
5 3  
1 23  
12 28  
25 35  
27 80  
36 50  
算法描述:  
Stept1:输入各个活动的开始时间(s)和结束时间(e)，初始化各活动的会场号。  
Step2:按活动的开始时间和活动时间排序，s最早(第一优先级)和持续时间最短(第二优先级)的活动排在最前。  
Step3:执行贪心算法，即s最早和持续时间最短的优先安排会场，并记录会场号，其余活动的s大于或等于已安排活动的e的安排在同一会场，若某活动的s小于安排活动的e,则安排在另一会场，记录会场号，依次循环，直到所有活动均被安排。  
Step4:统计会场号数,输出。  
程序代码：  
#include<iostream>

using namespace std;

#define M 50 //最大活动数

struct Active

{

int s;//开始时间

int e;//结束时间

int no;//预安排会场号

}a[M];

//两元素交换位置

void swap(Active&a,Active&b)

{

Active t;

t=a;

a=b;

b=t;

}

void main()

{

int k;

int i,j;

cout<<"输入待安排活动数:"<<endl;

cin>>k;

cout<<"输入待安排活动的开始时间和结束时间:"<<endl;

//输入活动时间

for(i=1;i<=k;i++)

{

cin>>a[i].s>>a[i].e;

a[i].no=0;

}

//活动时间排序

for(i=1;i<=k;i++)

{

for(j=i;j<=k;j++)

{

if(a[i].s>a[j].s)

swap(a[i],a[j]);

if(a[i].s==a[j].s)

{

if(a[i].e>a[j].e)

swap(a[i],a[j]);

}

}

}

int sum=1;//使用的会场数初始化

int n;

a[1].no=sum;

for(i=2;i<=k;i++)

{

for(n=1;n<i;n++)

{

if(a[n].no!=0&&a[n].e<=a[i].s)

{

a[i].no=a[n].no;

a[n].no=0;//已经安排过的活动就不再比较

break;

}

}

if(n==i)

{

sum+=1;

a[i].no=sum;

}

}

cout<<"输出最少会场数:\n"<<sum<<endl;

system("pause");

}