1. 贪心算法：

求解最优化问题的方法，考虑在当前状态下局部最优的策略，使得全局的结果达到最优的状态。因为，全局的最优是由中间的每一步最优组成的，那么可以用局部的最优一步步推得全部的最优。

一个问题的最优解可以有它的子问题的最优解构造出来。这其实用的就是迭代的思维，将一个问题分解为许多子问题求解。

二．区间贪心

区间不相交问题：给出N个开区间(x,y)，从中选择尽可能多的开区间，使得这些开区间两两之间没有交集。

三．思路：

局部的最优，将区间按照左端点或者右端点排序，然后从左端点最大的区间开始，向左查找是否有区间的右端点不大于该区间的左端点，这样一直循环，直到结束，就可以找出所有的不相交区间。

四．代码：

#include<iostream>

#include<algorithm>

#include<vector>

using namespace std;

class Line

{

public:

int x;

int y;

};

vector<Line> lines;

bool cmp(Line l1, Line l2)

{

if (l1.x != l2.x)

return l1.x > l2.x;

else

return l1.y < l2.y;

}

int main()

{

int n;

cin >> n;

for (int i = 0; i < n; ++i)

{

Line l1 = Line();

cin >> l1.x >> l1.y;

lines.push\_back(l1);

}

sort(lines.begin(), lines.end(), cmp);

int ans = 1; //不相交区间个数

int lastx = lines[0].x; //上一个被选中区间的左端点

cout << lines[0].x << "-"<< lines[0].y << endl;

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

{

if (lines[i].y <= lastx) //该区间左端点在lastx左边

{

lastx = lines[i].x;

ans++; //选中

cout << lines[i].x <<"-"<< lines[i].y << endl;

}

}

cout << ans << endl;

return 0;

}