实验舱
青少年编程
走近科学 走进名校

蛟龙四班

简单贪心

Mas

贪心

贪心算法（*greedy algorithm*）,是用计算机来模拟一个“贪心”的人做出决策的过程

并不是所有的时候贪心法都能获得最优解,所以一般使用贪心法的时候,都要确保自己能证明其正确性

适用范围

贪心算法在有最优子结构的问题中尤为有效

最优子结构是指问题能够分解成子问题来解决,子问题的最优解能递推到最终问题的最优解

证明方法

- 反证法

  如果交换方案中任意两个元素/相邻的两个元素后,答案不会变得更好,那么可以推定目前的解已经是最优解了

- 归纳法

  先验证出边界情况(如$n=1$)的最优解 ,然后再证明：对于每个$F(n)$, 都可以由$F(n)$推导出结果$F(n+1)$

#1103 旅行者的背包(部分背包问题)

题目描述

一个旅行者有一个最多能装 M 公斤的背包，现在有 n 种物品，它们的重量分别是 w_i ，它们的价值分别是 v_i 元/公斤，求旅行者能获得最大总价值。

尽可能的选择单位价值大的装入背包

输入

第 1 行：两个整数， M 背包容量（$M \le 1000$）和 N 物品数量（$N \le 30$）；
第 2 至 $N+1$ 行：每行两个整数 w_i, v_i ,表示每个物品的重量和价值。

输出

一个数，表示最大总价值。

样例输入

```
100 3
40 2
50 3
70 3
```

样例输出

```
300
```

#375 最大整数

题面描述

设有 n（$n \le 1000$）个正整数（每个在 $long\ long$ 范围内），将它们连接成一排，组成一个最大的多位整数。

输入第一行 1 个整数 n

第二行为 n 个正整数，分别用空格分隔输出一行，一个数，表示连接成的最大整数。

输入样例1

```
3
13 312 343
```

输出样例1

```
34331213
```

样例输入2

```
4
7 13 4 246
```

样例输出2

```
7424613
```

要让数字最大,高位需要尽可能大

使用$string$存储各个数值

按照字典序排序即可

#614、数列分段

题目描述

对于给定的一个长度为 N 的**正整数**数列 A_i，现要将其分成**连续的**若干段，并且**每段和**不超过 M（可以等于 M），问最少能将其分成多少段使得满足要求。

输入格式

第一行包含两个正整数 N, M，表示了数列 A_i 的长度与每段和的最大值；
第二行包含 N 个空格隔开的非负整数 A_i。

输出格式

输出文件仅包含一个正整数，输出最少划分的段数。

样例输入

```
5 6
4 2 4 5 1
```

样例输出

```
3
```

数据范围与提示

对于 20% 的数据，有 $N \le 10$；
对于 40% 的数据，有 $N \le 1000$；
对于 100% 的数据，有 $N \le 10^5, M \le 10^9$，M 大于所有数的最大值，A_i 之和不超过 10^9。

尽可能累加,当不能累加时分段

#996 删数问题

题目描述

键盘输入一个高精度的正整数 $n(n \le 10^{250})$ ，

去掉任意 s 个数字后剩下的数字按原左右次序将组成一个新的正整数。

编程对给定的 n 和 s ，寻找一种方案，使得剩下的数最小。

输入

第一行，一个不超过 250 位的整数。
第二行，一个整数 s 。

输出

删除 s 个整数后，保持原顺序的最小整数，前缀 " 0 " 不输出。

样例输入

```
178543
4
```

样例输出

```
13
```

删除最大的s个数位?

1003?

#996 删数问题

- 若数字序列是非递减的　,如：1344,123那么只需要删除最后的S位数字
- 若数字序列是有增有减的,如：17283,434434那么找到第一个下标i,满足i上的数字大于$i+1$上的数字,删除i上的数字
- 对于无意义的前导0应当删除(如1001)

#999 排队接水

问题描述

有 n 个人在一个水龙头前排队接水，假如每个人接水的时间为 T_i ，请编程找出这 n 个人排队的一种顺序，使得 n 个人的平均等待时间最小。

输入格式

输入，第一行为 n ；
第二行分别表示第 1 个人到第 n 个人每人的接水时间 T_1， T_2， $\ldots$， T_n ，每个数据之间有 1 个空格。

输出格式

输出两行，第一行为一种排队顺序，即 1 到 n 的一种排列；(如果多种排队方式，请保证先来的同学排在前面)
第二行为这种排列方案下的平均等待时间(输出结果精确到小数点后两位)。

输入样例

```
10
56 12 1 99 1000 234 33 55 99 812
```

输出样例

```
3 2 7 8 1 4 9 6 10 5
291.90
```

数据规模

对于全部的数据 $n \leq 1000, t \leq 10^6$

设一个等待时间序列为$T_1, T_2, \cdots, T_n$

不难发现总等待时间为

$$\sum_{i=1}^{n} T_i \times (n-i)$$

要使平均等待时间最小(总等待时间最小), 让序列单调非降即可

对序列升序排序即可

注意需要使用稳定的排序算法

#908 合并果子

实验舱 青少年编程 走近科学 走进名校

题目描述

在一个果园里，多多已经将所有的果子打了下来，而且按果子的不同种类分成了不同的堆。多多决定把所有的果子合成一堆。

每一次合并，多多可以把两堆果子合并到一起，消耗的体力等于两堆果子的重量之和。
可以看出，所有的果子经过 $n-1$ 次合并之后，就只剩下一堆了。多多在合并果子时总共消耗的体力等于每次合并所耗体力之和。

因为还要花大力气把这些果子搬回家，所以多多在合并果子时要尽可能地节省体力。假定每个果子重量都为 1 ，并且已知果子的种类数和每种果子的数目，你的任务是设计出合并的次序方案，使多多耗费的体力最少，并输出这个最小的体力耗费值。

例如有 3 种果子，数目依次为 1， 2， 9 。可以先将 1、 2 堆合并，新堆数目为 3 ，耗费体力为 3 。接着，将新堆与原先的第三堆合并，又得到新的堆，数目为 12 ，耗费体力为 12 。所以多多总共耗费体力 $3+12=15$ 。可以证明 15 为最小的体力耗费值。

输入

包括两行，第一行是一个整数 n （$1 \leq n \leq 30000$），表示果子的种类数。第二行包含 n 个整数，用空格分隔，第 i 个整数 a_i （$1 \leq a_i \leq 20000$） 是第 i 种果子的数目。

输出

包括一行，这一行只包含一个整数，也就是最小的体力耗费值。输入数据保证这个值小于 2^{31} 。

样例输入

```
3
1 2 9
```

样例输出

```
15
```

【数据规模】

对于 30% 的数据，保证有 $n \leq 100$；
对于 50% 的数据，保证有 $n \leq 5000$；
对于全部的数据，保证有 $n \leq 30000$。

#908 合并果子

不难发现对于任意n堆合并的过程都是类似与右图的树形结构

其中绿色节点为果子重量,红色节点为合并代价,线段代表一次搬运过程

总代价可看作各叶节点权值乘上它到根的距离

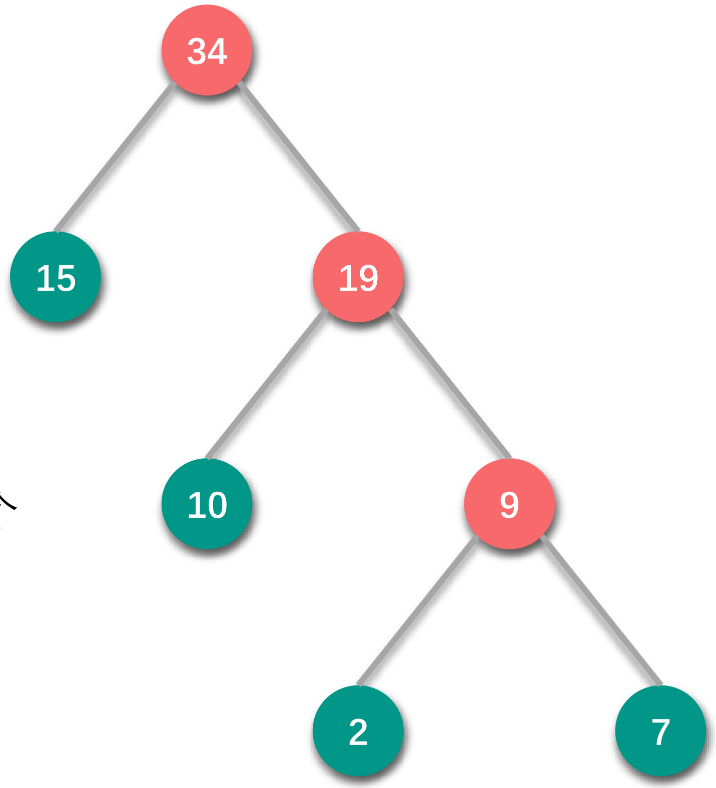
对于$n = 1$ 或 $n = 2$ 时结论显然成立

假设 $n = k$ 时该算法正确，考虑证明 $n = k + 1$ 时该算法也正确

可以发现：对于任意最优解，最小数一定在最深的那一层（否则可以将最小数与最深的那一层的某个比它大的数交换，这样得到的总代价是更小的），显然在最深的层的节点先合并的

综上 $n = k + 1$ 时第一步一定是合并最小的两个点

然后就转化到 $n = k$ 时场景，由归纳法，命题得证



#908 合并果子

对于$n=1$或$n=2$结论显然成立

设对于n堆,选取两堆重量合并代价最小且当前合并总代价为C

对于$n-1$堆,记任选两堆合并的代价为ΔC,要使得当前$C+\Delta C$最小,那么ΔC为最小两堆重量之和

对于每次取出两个最小值,加入一个新的值,需要维护动态有序性

若每一轮重新排序时间复杂度$O(\mathrm{nlog}n)$,总时间复杂度$O(n^2\mathrm{log}n)$

可以使用 $priority\_queue$(堆)进行维护,总时间复杂度$O(\mathrm{nlog}n)$

该模型可解决Huffman编码问题

```
priority_queue<int, vector<int>, greater<int>> q;
for (int i = 0; i < n; i++)
  q.push(a[i]);
while (q.size() > 1)
{
  int a = q.top();
  q.pop();
  int b = q.top();
  q.pop();
  ans += a + b;
  q.push(a + b);
}
```

#998 均分纸牌

实验舱 青少年编程 LAB 走近科学 走进名校

【题目描述】

有 n 堆纸牌，编号分别为 1，2，$\ldots, n$。每堆上有若干张，但纸牌总数必为 n 的倍数。可以在任一堆上取若干张纸牌，然后移动。

移牌规则为：在编号为 1 的堆上取的纸牌，只能移到编号为 2 的堆上；在编号为 n 的堆上取的纸牌，只能移到编号为 $n-1$ 的堆上；其他堆上取的纸牌，可以移到相邻左边或右边的堆上。

现在要求找出一种移动方法，用最少的移动次数使每堆上纸牌数都一样多。

例如 $n=4$，4 堆纸牌数分别为：① 9 ② 8 ③ 17 ④ 6

移动 3 次可达到目的：

从 ③ 取 4 张牌放到④（9 8 13 10）->从③取3张牌放到 ②（9 11 10 10）-> 从②取1张牌放到①（10 10 10 10）。

【输入】

n（n 堆纸牌，$1 \leq n \leq 100$）

$a_1\ a_2\ \ldots\ a_n$（n 堆纸牌，每堆纸牌初始数，$l \leq a_i \leq 10000$）。

【输出】

所有堆均达到相等时的最少移动次数。

【输入样例】

```
4
9 8 17 6
```

【输出样例】

```
3
```

#998 均分纸牌

注意到每对相邻的纸牌最多只会移动一次我们记 x_i 为 i 到 $i-1$ 移动的纸牌数量

注意 x_i 有 $2 \le i \le n$，且 x_i 可以是负数,表示反方向移动

设平均数为 $\bar{a}$

x_i 的计算方式如下:

$$a_1 + x_2 = \bar{a}$$

$$a_2 - x_2 + x_3 = \bar{a} \iff x_2 = a_2 + x_3 - \bar{a}$$

$$a_{n-1} + x_n - x_{n-1} = \bar{a} \iff x_{n-1} = a_{n-1} + x_n - \bar{a}$$

$$\cdots$$

$$a_n - x_n = \bar{a} \iff x_n = a_n - \bar{a}$$

从左到右考虑 i若 x_i 不为 0,说明 a_i 到 a_{i+1}至少存在一次转移,最少移动次数为

$$\sum_{i=2}^{n} (x_i \ne 0)$$

```cpp
#include <bits/stdc++.h>
using namespace std;
int n, ans, a[101], ave;
int main()
{
    cin >> n;
    for (int i = 0; i < n; i++)
        cin >> a[i], ave += a[i];
    ave /= n;
    for (int i = 0; i < n; i++)
    {
        ans += (a[i] != ave);
        a[i + 1] -= ave - a[i];
    }
    cout << ans;
    return 0;
}
```

#2141、货仓选址

题目描述

在一条数轴上有 N 家商店，它们的坐标分别为 $A_1 \sim A_N$ 。

现在需要在数轴上建立一家货仓，每天清晨，从货仓到每家商店都要运送一车商品。

为了提高效率，求把货仓建在何处，可以使得货仓到每家商店的距离之和最小。

输入格式

第一行输入整数 N 。

第二行 N 个整数 $A_1 \sim A_N$ 。

输出格式

输出一个整数，表示距离之和的最小值。

输入样例：

```
4
6 2 9 1
```

输出样例：

```
12
```

数据范围

对于全部的数据 $1 \le N \le 100000$,坐标不超过 2^{31}

#2141、货仓选址

给定一个数列,中位数有这样的性质： 所有数与中位数的绝对差之和最小

记商店排序后的位置为 $x_1,x_2,x_3,\cdots,x_{n-1},x_n$,货仓位置为$x$,距离之和为$d$

$$
\begin{aligned}
d &= |x_1 - x| + |x_2 - x| + \cdots + |x_{n-1} - x| + |x_n - x| \\
&= (|x_n - x| + |x_1 - x|) + (|x_{n-1} - x| + |x_2 - x|) + \cdots \qquad (1) \\
&\geq x_n - x_1 + x_{n-1} - x_2 + \cdots
\end{aligned}
$$

不难发现,对于任意一对 $x_i > x_j$,当 $x > x_i$或$x < x_j$, $|x_i - x| + |x_j - x| \geq x_i - x_j$

要使得d取最小值,需要使得(1)中括号内每一组都满足$x_j \leq x \leq x_i$

当n为奇数时, x 为数列中位数

n为偶数时?

```
#include <bits/stdc++.h>
using namespace std;
long long n, a[100005], ans, mid;
int main()
{
  scanf("%lld", &n);
  for (int i = 0; i < n; i++)
    scanf("%lld", a + i);
  sort(a, a + n);
  for (int i = 0; i < n; i++)
    ans += abs(a[n / 2] - a[i]);
  printf("%lld\n", ans);
  return 0;
}
```

#1102 加勒比海盗

题目描述

在北美洲东南部，有一片神秘的海域，那里碧海蓝天、阳光明媚，这正是传说中海盗最活跃的加勒比海（$Caribbean\ Sea$）。17 世纪时，这里更是欧洲大陆的商旅舰队到达美洲的必经之地，所以当时的海盗活动非常猖獗，海盗不仅攻击过往商人，甚至攻击英国皇家舰……

有一天，海盗们截获了一艘装满各种各样古董的货船，每一件古董都价值连城，一旦打碎就失去了它的价值。虽然海盗船足够大，但载重量为 C，每件古董的重量为 w_i，海盗们该如何把尽可能多数量的宝贝装上海盗船呢?

输入描述:

第一行是两个整数 $c, n(1 < c, n < 10000)$ 表示载重量 c 及古董的个数 n。
第二行是 n 个数，分别表示第 i 个古董的重量。

输出描述:

输出能装入的古董最大数量。

输入样例

```
30 8
4 10 7 11 3 5 14 2
```

输出样例

```
5
```

#1102 加勒比海盗(最优装载问题)

实验舱 青少年编程 LAB 走近科学 走进名校

采用重量轻先装的贪心选择策略,可产生该问题的最优解

设集装箱依其重量从小到大排序,$(x_1, x_2, \cdots, x_n)$是其最优解,$x_i = \{0,1\}$,设x_k是第一个等于1的

- $k = 1$,满足贪心选择性质
- 如$k \neq 1$,用x_1替换x_k,构造的新解同原解最优值相同,故也是最优解,满足贪心选择性质

该具有最优子结构性质

记$T(S, W)$为$S \sim n$个物品装上船(载重量为W)的最大数量,当$W < \mathrm{w}_s$时$T(s, W) = 0$

假设第一个物品可以选取,$T(1, W) = 1 + T(2, W - w_1)$

因$T(1, W)$是最优解,则$T(2, W - w_1)$一定是最优解

如果$T(2, W - w_1)$不是该问题的最优解,则存在最优解$T'(2, W - w_1) > T(2, W - w_1)$

使得$1 + T'(2, W - w_1) = T'(1, W) > T(1, W)$,与$T(1, W)$是最优解相矛盾,故$T(2, W - w_1)$一定是最优值

#1102 加勒比海盗(最优装载问题)

我们也可以通过强化结论+反证法来证明

强化解的比较规则：对于两个可行解 $(p_1,p_2,\cdots,p_a)$ 和 $(q_1,q_2,\cdots,q_b)$，若 $a \neq b$ 我们认为数量少的解更优，若 $a=b$ 我们认为总重量少的解更优，即比较 $\sum p_i$ 和 $\sum q_i$ 谁更小

断言：采用重量轻先装的贪心选择策略一定能得到最优解

反证：假设贪心策略不一定正确，即可能存在某个解 $(p_1,p_2,\cdots,p_a)$，在我们定的强化规则下比贪心解严格更优

将这个最优解排序，我们能知道，至少有一个未被装的物品 t 满足 $t < p_a$（否则这个最优解就是贪心解了）

我们找到第一个 k 满足 $p_k > t$然后用 t 替换 p_k，用 p_k 替换 p_{k+1}，…，用 p_{a-1} 替换 p_a

显然新构造的解严格比原解更优，产生了矛盾，说明不存在解比贪心解严格更优

#609 活动安排

题目描述

设有 n 个活动的集合 $E=\{1,2,..,n\}$ ，其中每个活动都要求使用同一资源，如演讲会场等，而在同一时间内只有一个活动能使用这一资源。

每个活动 i 都有一个要求使用该资源的起始时间 s_i 和一个结束时间 f_i ，且 $s_i<f_i$ 。如果选择了活动 i ，则它在时间区间 $[s_i,f_i)$ 内占用资源。

若区间 $[s_i,f_i)$ 与区间 $[s_j,f_j)$ 不相交，则称活动 i 与活动 j 是相容的。也就是说，当 $f_i\leq s_j$ 或 $f_j\leq s_i$ 时，活动 i 与活动 j 相容。选择出由互相兼容的活动组成的最大集合。

输入格式

第一行一个整数 n ；

接下来的 n 行，每行两个整数 s_i 和 f_i 。

输出格式

输出互相兼容的最大活动个数。

数据范围与提示

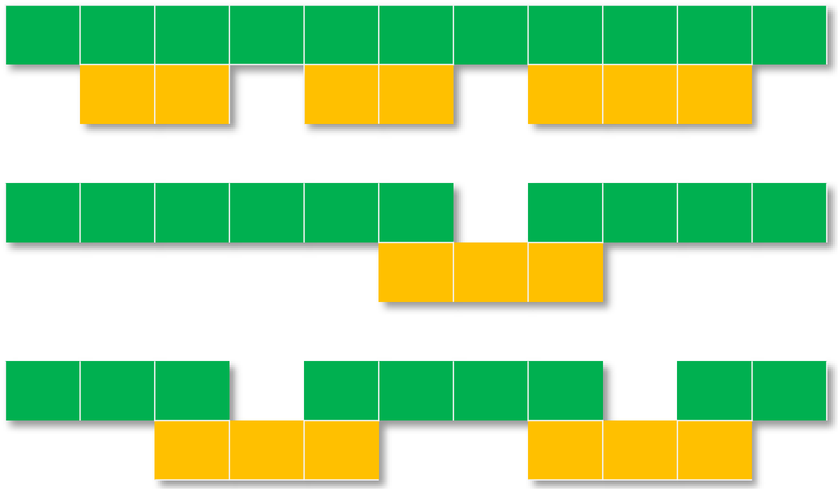
对于全部的数据 $1\leq n\leq 1000$

样例输入

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

样例输出

```
2
```



#609 活动安排

对结束时间进行升序排序,若当前活动不与上一个活动产生交集那么选取,可产生该问题的最优解

设活动按结束时间从早到晚排序,$(x_1, x_2, \cdots, x_n)$是其最优解,$x_i = \{0,1\}$,设x_k是第一个等于1的

- $k = 1$,满足贪心选择性质
- 如$k \neq 1$,用x_1替换x_k,构造的新解同原解最优值相同,故也是最优解,满足贪心选择性质

该具有最优子结构性质

尝试证明

```cpp
#include <bits/stdc++.h>
using namespace std;
struct node
{
  int s, e;
  bool operator<(const node a) const
  {
    return e < a.e;
  }
} a[1005];
int n, ans = 1, pre;
int main()
{
  cin >> n;
  for (int i = 0; i < n; i++)
    cin >> a[i].s >> a[i].e;
  sort(a, a + n);
  pre = a[0].e;
  for (int i = 0; i < n; i++)
    if (a[i].s >= pre)
      ans++, pre = a[i].e;
  cout << ans << endl;
  return 0;
}
```

#1508、区间合并

题目描述

给定 n 个闭区间 $[a_i,\ b_i]$，其中 $i=1,2,...,n$。任意两个相邻或相交的闭区间可以合并为一个闭区间。例如，$[1,2]$ 和 $[2,3]$ 可以合并为 $[1,3]$，$[1,3]$ 和 $[2,4]$ 可以合并为 $[1,4]$，但是 $[1,2]$ 和 $[3,4]$ 不可以合并。
如果这些区间可以合并，请将他们合并

输入格式

第一行为一个整数 n，$3 \le n \le 100000$。表示输入区间的数量。
之后 n 行，在第 i 行上（$1 \le i \le n$），为两个整数 a_i 和 b_i，整数之间用一个空格分隔，表示区间 $[a_i,b_i]$（其中 $-10^9 \le a_i \le b_i \le 10^9$）。

输出格式

输出可能有多行，按区间左边界升序输出合并后的区间
每行一个合并后的区间，输出这个闭区间的左右边界，用单个空格隔开

样例输入

```
5
5 6
1 5
10 10
6 9
8 10
```

样例输出

```
1 10
```

#1508、区间合并

将所有区间按照左端点升序排序

将第一个区间加入 M 数组中,并按顺序依次考虑之后的每个区间:

- 若当前区间的左端点在数组 M 中最后一个区间的右端点之后,那么它们不会重合,可直接将这个区间加入数组 M
- 否则,它们重合,需要用当前区间的右端点更新数组 M 中最后一个区间的右端点,将其置为二者的较大值

在排序后的数组中

两个可合并的区间未合并,说明存在三元组 (i,j,k) 及区间 $a[i],a[j],a[k]$ 满足 $i < j < k$且$a[i],a[k]$能合并,但$a[i],a[j]$和$a[j],a[k]$不能合并

$$a[i].e < a[j].s$$

$$a[j].e < a[k].s$$

$$a[i].e \geq a[k].s$$

显然$a[i].s \leq a[i].e$,联立上述不等式可以得到

$$a[i].s < a[j].s \leq a[i].e \leq a[j].e < a[k].s$$

产生了矛盾！ 这说明假设是不成立的因此,所有能够合并的区间都必然是连续的

```
sort(a, a + n);
int s = a[0].begin, e = a[0].end;
for (int i = 1; i < n; i++)
{
  if (a[i].begin > e)
  {
    printf("%d %d\n", s, e);
    s = a[i].begin;
  }
  if (a[i].end > e)
    e = a[i].end;
}
printf("%d %d\n", s, e);
```

#1199、整数区间

实验舱 青少年编程 LAB 走近科学 走进名校

题目描述

请编程完成以下任务：

> - *读取闭区间的个数及它们的描述；*
> - *找到一个含元素个数最少的集合,使得对于每一个区间,都至少有一个整数属于该集合，输出该集合的元素个数。*

输入格式

首行包括区间的数目 $n(1 \le n \le 10000)$,接下来的 n 行,每行包括两个整数 a, b ,被一空格隔开, $0 \le a \le b \le 10000$,它们是某一个区间的开始值和结束值。

输出格式

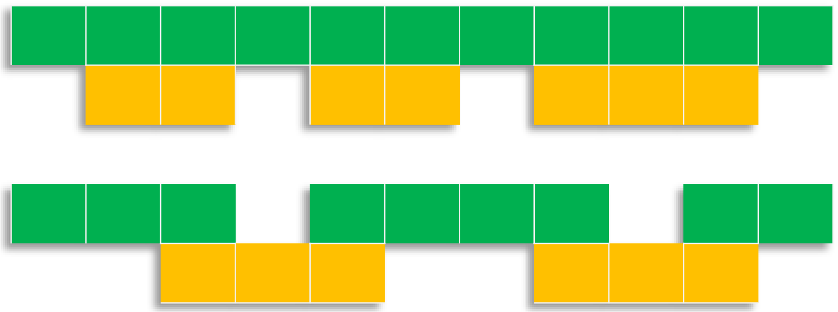
第一行集合元素的个数,对于每一个区间都至少有一个整数属于该区间,且集合所包含元素数目最少。

输入样例

```
4
3 6
2 4
0 2
4 7
```

输出样例

```
2
```



#1199、整数区间

将所有区间按照右端点升序排序

选取第一个区间并将第一个区间的右端点记为pre

- 若当前区间的左端点在pre之后,那么它们不会重合,让一个新数覆盖,并用当前区间右端点更新pre
- 否则,它们重合,不需要选择新的数覆盖

正确性?

```
sort(a, a + n);
pre = a[0].e;
for (int i = 1; i < n; i++)
  if (a[i].s > pre)
    ans++, pre = a[i].e;
printf("%d", ans);
```

实验舱

青少年编程

走近科学 走进名校

谢谢观看