Fiber: 光ファイバー網の整備

input ファイル “`fiber.in`”
output 標準出力
ソースファイル `fiber.c/fiber.cpp/Fiber.java`
時間制限 1 秒 / データ

クロアチアでは国内の全ての都市を光ファイバー網で結ぶ計画が進行している．光ファイバーを用いた通信回線は極めて高品質かつ高速であり，複数の都市を経由していても，ほとんど速度が低下することなく，高速な通信を行うことができる．例えば，k 個の都市 $A_1, \ldots, A_k$ に対し，都市 A_i と A_{i+1} $(1 \leqq i \leqq k-1)$ の間に光ファイバーが敷設されていれば，都市 A_1 と A_k の間で高速通信を行うことができる．

クロアチア政府は，国全体に光ファイバー網を張り巡らせて，全ての都市と都市の間で高速通信が行えるようにしたいと考えている．ところが困ったことに，現在では，規制緩和の影響で複数の光ファイバー敷設業者が乱立しており，国全体の光ファイバー網がどうなっているのか誰も把握していない．そこで，各業者から提出された光ファイバーの敷設情報を元に，全ての都市と都市の間の高速通信を可能にするためには，あと何本の光ファイバーを新たに敷設する必要があるのかを調べることにした．

光ファイバーの敷設情報が与えられたときに，全ての都市と都市の間で高速通信を可能にするために，あと何本の光ファイバーを新たに敷設する必要があるのかを求めるプログラムを作成せよ．

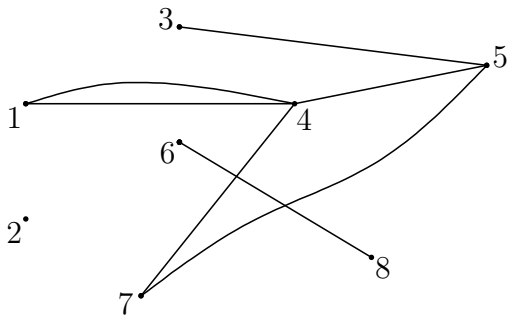
Input. 入力ファイル `fiber.in` の 1 行目には，クロアチア国内の都市の個数 n $(1 \leqq n \leqq 10000)$ が書かれている．各都市には 1 から n までの番号が付けられている．2 行目には，敷設されている光ファイバーの本数を表す整数 m $(1 \leqq m \leqq 30000)$ が書かれている．続く m 行 (3 行目 〜 $m+2$ 行目) は光ファイバーの敷設情報を表す．$i+2$ 行目 $(1 \leqq i \leqq m)$ には 2 つの異なる整数 a_i, b_i $(1 \leqq a_i, b_i \leqq n,\ a_i \neq b_i)$ が空白を区切りとして書かれている．これは，都市 a_i と都市 b_i の間に光ファイバーが敷設されていることを意味する．

Output. 出力は，標準出力に行うこと．全ての都市と都市の間で高速通信を可能にするために新たに敷設が必要な光ファイバーの本数の最小値を出力せよ．もし，現在のままでも全ての都市と都市の間で高速通信が可能な場合は，0 を出力せよ．

例

fiber.in	標準出力
8 7 3 5 4 1 5 4 7 5 4 7 1 4 6 8	2

この入力例に対応する 8 個の都市間の光ファイバー網を図示すると，以下の通りである．



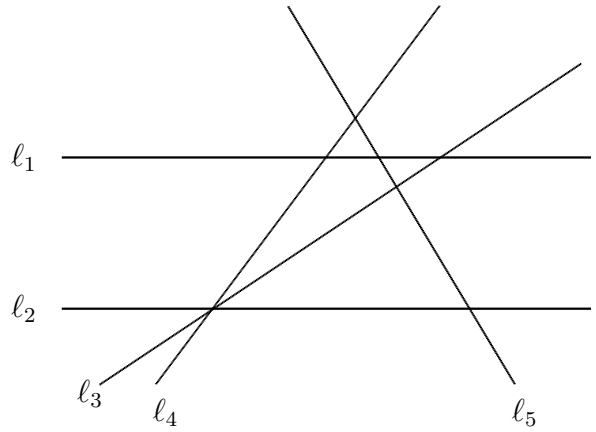
2 本の光ファイバーを新たに敷設すれば，全ての都市と都市の間で高速通信が可能になる．例えば，都市 2 と 1，都市 6 と 4 の間に敷設すればよい．これが新たに必要な光ファイバーの本数の最小値である．

Lines: 直線

input ファイル “`lines.in`”
output 標準出力
ソースファイル `lines.c/lines.cpp/Lines.java`
時間制限 1 秒 / データ

平面上の N 本の直線 $\ell_1, \ell_2, \ldots, \ell_N$ が入力として与えられる．これらの直線によって平面を分割したときの領域の個数を求めるプログラムを作成せよ．ただし， これらの直線には重複があるかもしれない．

下図では 14 個の領域に分割されている．

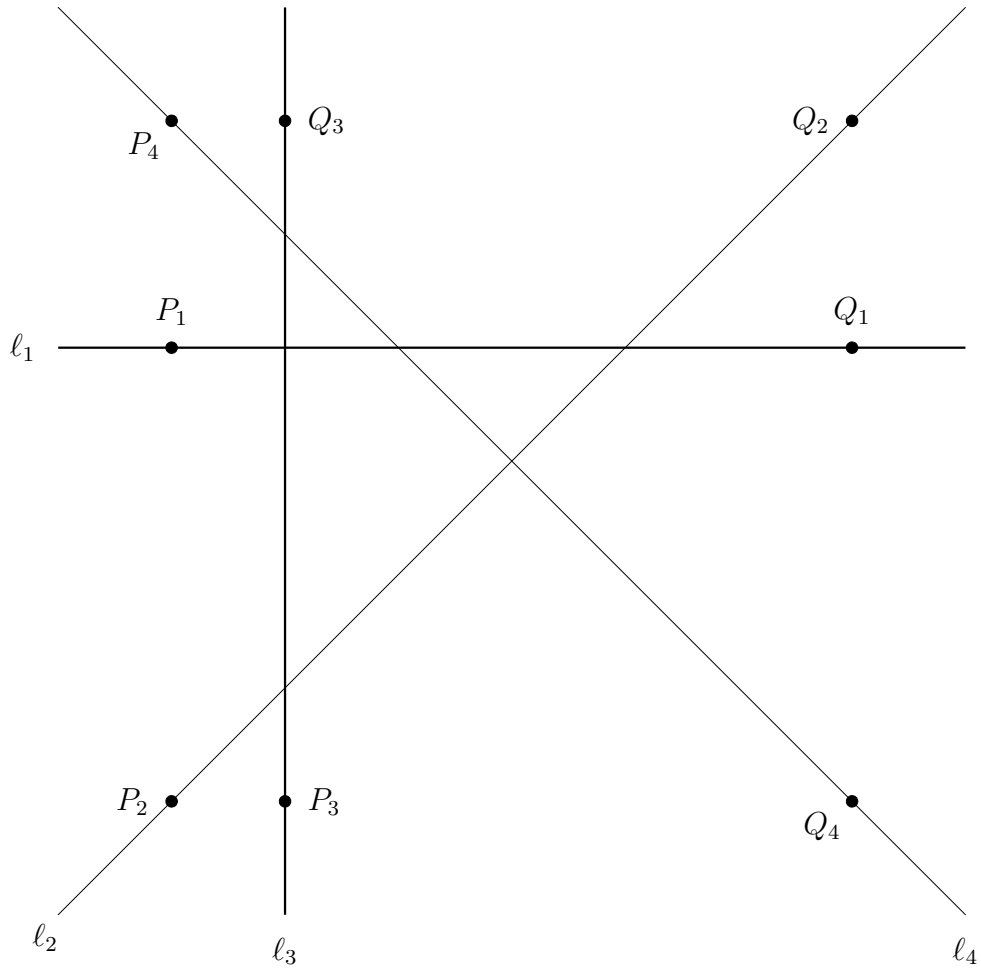


Input．入力はファイルは `lines.in` である．入力は $N+1$ 行からなる．最初の行に N $(1 \leqq N \leqq 1000)$ の値が書かれている．$i+1$ 行目 $(1 \leqq i \leqq N)$ には，4 つの整数 a_i, b_i, c_i, d_i $(0 \leqq a_i, b_i, c_i, d_i \leqq 1000, (a_i, b_i) \neq (c_i, d_i))$ が空白を区切りとして書かれている．これは，直線 ℓ_i が，点 $P_i(a_i, b_i)$ と点 $Q_i(c_i, d_i)$ を結ぶ直線であることを意味している．

Output．標準出力に 1 行で領域の個数を出力せよ．

例

lines.in	標準出力
4 0 4 6 4 0 0 6 6 1 0 1 6 0 6 6 0	11



Packing: 半導体工場

出力のみの課題 (OUTPUT ONLY TASK)・相対評価

input 問題文中で与えられる.
output ファイル "`packing-out`k`.txt`" ($k = 1, 2, 3, 4, 5$)

あなたの勤めている半導体工場では，最新鋭の装置を用いてシリコン板を加工し，半導体チップの材料として出荷している．シリコン板の加工に用いる装置は極めて高精度であり，加工位置を 10 ナノメートル単位で調整することができる (10 ナノメートル = 100,000,000 分の 1 メートル).

ある顧客からの注文により，あなたは，辺の長さが 1 メートルの正方形のシリコン板から，n 枚のシリコン円板を切り抜くことになった．切り抜くシリコン円板は全て同じ大きさでなければならない．また，貴重なシリコン板を無駄にしないよう，できるだけ大きい円板を切り抜くことが望ましい.

正方形のシリコン板から，できるだけ大きな n 枚のシリコン円板を切り抜く方法を求めよ.

Input. 以下の表の n の値を，入力として用いよ.

データ番号	シリコン円板の枚数 n
1	6
2	11
3	17
4	22
5	90

Output. 出力データをファイルとして提出せよ．出力データの作成方法は問わない．プログラムに直接生成させてもよいし，必要に応じて，それをテキストエディタ等で加工してもよい．出力データの作成に用いたプログラムを提出する必要は無い.

データ番号 k の入力に対する出力ファイル名は `packing-out`k`.txt` である ($k = 1, 2, 3, 4, 5$).

出力データは n 行からなる．n は切り抜くシリコン円板の枚数であり，Input の項で与えられている．i 行目 ($1 \leqq i \leqq n$) には，i 枚目のシリコン円板の中心の位置を表す 2 つの整数 x_i, y_i を空白を区切りとして出力せよ．これは，i 枚目のシリコン円板の中心が，正方形板の左端から $x_i \times 10$ ナノメートル，上端から $y_i \times 10$ ナノメートルの場所であることを意味する．x_i, y_i は $0 \leqq x_i, y_i \leqq 100,000,000$ をみたす整数である.

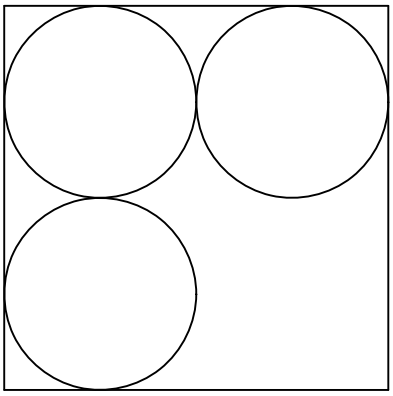
切り抜くシリコン円板の大きさは，装置により自動的に計算されるので，出力データの中に書く必要はない.

モデル解法　この問題の一つの解法は次のようなものである．$(m-1)^2 < n \leqq m^2$ をみたす整数 m を求める．正方形板を $m \times m$ の小正方形板に分割する．そこから n 個の小正方形板を選び，内接する円板を切り抜く．これにより，半径 $\frac{1}{2m}$ の円板を切り抜く解法が得られる．この解法を「モデル解法」と呼ぶことにする．モデル解法は必ずしも効率の良い解法ではない．

例 1　$n = 3$ の場合に，「モデル解法」によって得られた出力データは以下の通りである．

```
25000000 25000000
75000000 25000000
25000000 75000000
```

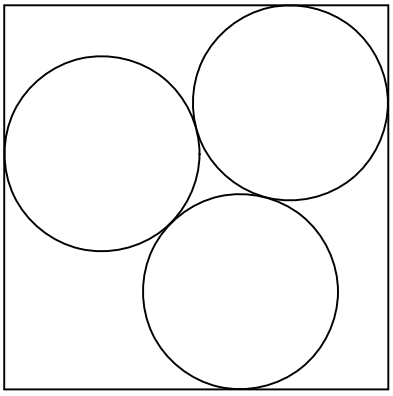
これを図示すると下図の通りである．



例 2　例 1 は最適解ではない．例えば，

```
74540929 25404070
25474639 38662789
61538179 74593059
```

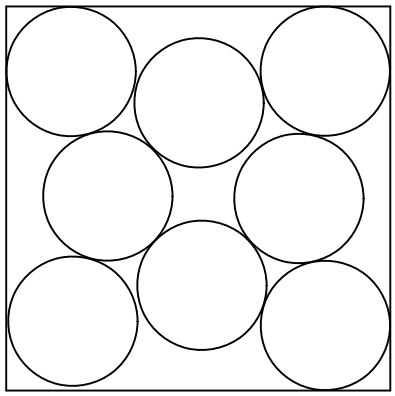
の方が良い結果を与える．これを図示すると以下の通りである．



例 2 以下は，$n = 8$ の場合の出力データの例である．このデータは,「モデル解法」よりも良い結果を与える．

```
50993849 72645350
17399669 81993059
83110119 83073720
50245039 25147290
83063380 16879409
16948019 16965430
26471179 49375439
76212739 50011850
```

これを図示すると以下の通りである．



採点について 出力データ毎に配点を定め，以下に従い，相対評価で採点を行う．

- 出力データが誤っていた場合や未提出の場合は，その出力データに関する得点は 0 点である．また，半径が「モデル解法」以下の出力データも 0 点である．
- 「モデル解法」よりも半径の大きなデータが提出された場合，その得点は次のようにして計算される.「モデル解法」の半径を d_0 とおく．提出された出力データの半径のうち，最も大きいものを d_1 $(d_1 > d_0)$ とおく．半径 d $(d_1 \geqq d \geqq d_0)$ の出力データには，配点のうち，

$$\left(\frac{d - d_0}{d_1 - d_0} \times 100\right) \ \%$$

の得点が与えられる．

備考 計算途中でのオーバーフローに注意すること．$2^{32} = 4,294,967,296$ である．C/C++の場合は，必要に応じて，内部で double 型や long long int 型を用いて演算を行うと良いだろう．