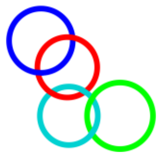
Parachute rings

An early and quite sophisticated (متطور) version of what we now call a parachute is described in Leonardo's *Codex Atlanticus* (ca. 1485). Leonardo's parachute consisted of a sealed linen cloth held open by a pyramid-shaped (هرمي الشكل) wooden structure.

Linked rings

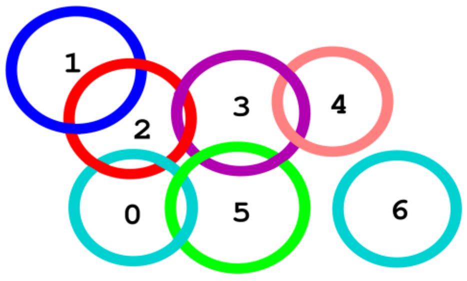
Skydiver Adrian Nicholas tested Leonardo's design more than 500 years later. For this, a modern lightweight structure tied Leonardo's parachute to the human body. We want to use linked rings, which also provide hooks for the sealed linen cloth. Each ring is made of flexible and strong material. Rings can be easily linked together as every ring can be opened and re-closed. A special configuration of linked rings is the *chain*. A *chain* is a sequence of rings in which each ring is only connected to its (at most two) neighbours, as illustrated below. Specifically, a single ring is also a chain.



Other configurations are clearly possible, since a ring can be linked to three or more other rings. We say that a ring is *critical* if after opening and removing it, all remaining rings form a set of chains (or there are no other rings left). In other words, there can be nothing but chains left.

Example

Consider the 7 rings in the next figure, numbered from 0 to 6. There are two critical rings. One critical ring is 2: after its removal, the remaining rings form chains [1], [0, 5, 3, 4] and [6]. The other critical ring is 3: after its removal, the remaining rings form chains [1, 2, 0, 5], [4] and [6]. If we remove any other ring, we do not obtain a set of disjoint chains. For example, after removing ring 5: although we have that [6] is a chain, the linked rings 0, 1, 2, 3 and 4 do not form a chain.



Statement

Your task is to count the number of critical rings in a given configuration that will be communicated to your program.

At the beginning, there are a certain number of disjoint rings. After that, rings are linked together. At any given time, you can be asked to return the number of critical rings in the current configuration. Specifically, you have to implement three routines.

- `Init`(N) — it is called exactly once at the beginning to communicate that there are N disjoint (غير متصلة) rings numbered from 0 to N - 1 (inclusive) in the initial configuration.

- `Link`(A, B) — the two rings numbered A and B get linked together. It is guaranteed that A and B are different and not already linked directly; apart from this, there are no additional conditions on A and B, in particular no conditions arising from physical constraints. Clearly, `Link`(A, B) and `Link`(B, A) are equivalent (متساوي).

- `CountCritical`() — return the number of critical rings for the current configuration of linked rings.

Example

Consider our figure with N = 7 rings and suppose that they are initially unlinked. We show a possible sequence of calls, where after the last call we obtain the situation depicted in our figure.

Call	Returns
`Init`(7)	
`CountCritical`()	7
`Link`(1, 2)	
`CountCritical`()	7
`Link`(0, 5)	
`CountCritical`()	7
`Link`(2, 0)	
`CountCritical`()	7
`Link`(3, 2)	
`CountCritical`()	4
`Link`(3, 5)	
`CountCritical`()	3
`Link`(4, 3)	
`CountCritical`()	2

Subtask 1 [20 points]

- N ≤ 5 000.
- The function `CountCritical` is called only once, after all the other calls; the function `Link` is called at most 5 000 times.

Subtask 2 [17 points]

- N ≤ 1 000 000.
- The function `CountCritical` is called only once, after all the other calls; the function `Link` is called at most 1 000 000 times.

Subtask 3 [18 points]

- N ≤ 20 000.
- The function `CountCritical` is called at most 100 times; the function `Link` is called at most 10 000 times.

Subtask 4 [14 points]

- N ≤ 100 000.
- The functions `CountCritical` and `Link` are called, in total, at most 100 000 times.

Subtask 5 [31 points]

- N ≤ 1 000 000.
- The functions `CountCritical` and `Link` are called, in total, at most 1 000 000 times.

Implementation details

You have to submit exactly one file, called `rings.c`, `rings.cpp` or `rings.pas`. This file implements the subprograms described above using the following signatures.

C/C++ programs

```
void Init(int N);
void Link(int A, int B);
int CountCritical();
```

Pascal programs

```
procedure Init(N : LongInt);
procedure Link(A, B : LongInt);
function CountCritical() : LongInt;
```

These subprograms must behave as described above. Of course you are free to implement other subprograms for their internal use. Your submissions must not interact in any way with standard input/output, nor with any other file.

Sample grader

The sample grader reads the input in the following format:

- line 1: N, L;
- lines 2, …, L + 1:
  - -1 to invoke `CountCritical`;
  - A, B parameters to `Link`.

The sample grader will print all results from `CountCritical`.