Kick Start 2022 - Coding Practice with Kick Start Session #2

Sherlock and Parentheses

Problem

Download the Starter Code!

Sherlock and Watson have recently enrolled in a computer programming course. Today, the tutor taught them about the balanced parentheses problem. A string S consisting only of characters `(` and/or `)` is *balanced* if:

- It is an empty string, or:
- It has the form (S), where S is a balanced string, or:
- It has the form S_1S_2, where S_1 is a balanced string and S_2 is a balanced string.

Sherlock coded up the solution very quickly and started bragging about how good he is, so Watson gave him a problem to test his knowledge. He asked Sherlock to generate a string S of `L + R` characters, in which there are a total of $\mathbf{L}$ left parentheses `(` and a total of $\mathbf{R}$ right parentheses `)`. Moreover, the string must have as many different balanced non-empty substrings as possible. (Two substrings are considered different as long as they start and end at different indexes of the string, even if their content happens to be the same). Note that S itself does not have to be balanced.
Sherlock is sure that once he knows the maximum possible number of balanced non-empty substrings, he will be able to solve the problem. Can you help him find that maximum number?

Input

The first line of the input gives the number of test cases, $\mathbf{T}$. $\mathbf{T}$ test cases follow.
Each test case consists of one line with two integers: $\mathbf{L}$ and $\mathbf{R}$.

Output

For each test case, output one line containing `Case #`x`:` y, where x is the test case number (starting from 1) and y is the answer, as described above.

Limits

Time limit: 20 seconds.
Memory limit: 1 GB.
$1 \leq \mathbf{T} \leq 100$.

Test Set 1

$0 \leq \mathbf{L} \leq 20$.
$0 \leq \mathbf{R} \leq 20$.
$1 \leq \mathbf{L} + \mathbf{R} \leq 20$.

Test Set 2

$0 \leq \mathbf{L} \leq 10^5$.
$0 \leq \mathbf{R} \leq 10^5$.
$1 \leq \mathbf{L} + \mathbf{R} \leq 10^5$.

Sample

Sample Input	Sample Output
3 1 0 1 1 3 2	Case #1: 0 Case #2: 1 Case #3: 3

In Sample Case #1, the only possible string is `(`. There are no balanced non-empty substrings.

In Sample Case #2, the optimal string is `()`. There is only one balanced non-empty substring: the entire string itself.

In Sample Case #3, both strings `()()(` and `(()()` give the same optimal answer.
For the case `()()(`, for example, the three balanced substrings are `()` from indexes 1 to 2, `()` from indexes 3 to 4, and `()()` from indexes 1 to 4.