Scientific calculation often needs to operate values with many digits or with high precision. However, the built-in data types only provide a fixed size for representing values. A common way to address the issue is to provide a class for big integer, which can manipulate integers with arbitrary number of digits.
Your goal is to provide a class for representing the big integer with a string of digits and a sign. The class should also provide the following functionalities:
Use stream insertion operator (<<) to output the class object.
Use stream extraction operator (>>) to input the class object.
Use addition operator (+) to add two class objects.
Use subtraction operator to subtract two class objects.
Use relational operators (> and <), and equality operators (== and !=) to compare two class objects.
The above overloaded operators should behave as if they perform on the built-in integer types.
The implementation of addition and subtraction can be simplified as two cases A + B and A – B, where A and B are both positive integers. For addition, we have:
-A + B = B – A, A + (-B) = A – B, -A + (-B) = -(A+B),
and for subtraction, we have:
-A - B = -(A+B), A – (-B) = A + B, -A – (-B) = B – A
Therefore the eight cases of addition and subtraction are reduced to two cases, and the sign of the result is determined by the magnitude.
Requirement: Provide a class satisfying the above requirements. Prepare your class with appropriate constructors and overloaded operators and encapsulate the required methods. Separate your program in files of three kinds: the class header file (.h), the class source code file (.cpp), and the file containing main function (.cpp).
Prohibited: Use C-style input/output.

Input

Each case contains an two integers n_1 and n_2. The input ends with an asterisk (*).

Output

For each case, output the result of n_1 - n_2, followed by an end line stream manipulator.

Sample Input

-80948824551711527614232216857618927954312334113874277931986502860248 65
-80613893446066184963788291359840763615420973726016575412001460 71

-57335998182660380125094783512016406189841439880877838371073 4
91099683484992553337438088068198972282890781586124258626 53

46182119762952003918195325258677229419698255491250839396 7
-56935766582544161633553282536186214629150364929344059634288758 1

12726471321638328601260546793478816387617237858587331081092491573342
82770241037395972028670818303620284183758170488136789555663008 82

*

Sample Output

-80948743937818081548047253069327568113548718692900551915411090858787 94
-58246995017510305658469164392698395912670347696739080957338 7
56935812764663924585557200731511473306379784627599550885128154 8
12718194297534589004057679711648454359198862041538517402136925272460