**Strange addition**

Attempted by: **1637**

/

Accuracy: **69%**

/

Maximum Score: **20**

/

13 Votes

Tag(s):

Ad-Hoc, Easy, Math

**PROBLEM**

**EDITORIAL**

**MY SUBMISSIONS**

**ANALYTICS**

HackerMan has a message that he has coded in form of digits, which means that the message contains only numbers and nothing else. He is fearful that the enemy may get their hands on the secret message and may decode it. HackerMan already knows the message by heart and he can simply destroy it.

But he wants to keep it incase its needed in a worse situation. He wants to further encode the message in such a format which is not completely reversible. This way if enemies gets hold of the message they will not be completely able to decode the message.

Since the message consists only of number he decides to flip the numbers. The first digit becomes last and vice versa. For example, if there is 37693769 in the code, it becomes 96739673 now. All the leading zeros are omitted e.g. 1590015900 gives 951951. So this way the encoding can not completely be deciphered and has some loss of information.

HackerMan is further thinking of complicating the process and he needs your help. He decides to add the two flipped numbers and print the result in the encoded (flipped) form. There is one problem in this method though. For example, 134134 could be 431431, 43104310 or 4310043100 before reversing. Hence the method ensures that no zeros were lost, that is it can be assumed that the original number was 431431.

**Input**

The input consists of TT test cases. The first line of the input contains only positive integer TT. Then follow the cases. Each case consists of exactly one line with two positive integers separated by space. These are the flipped numbers you are to add.

**Output**

For each case, print exactly one line containing only one integer - the flipped sum of two flipped numbers.

**Constraints**

* The value of TT will be less than 10001000
* The value of digits will be less than 500000500000

**SAMPLE INPUT**

3

353 575

238746 39857

890 231

**SAMPLE OUTPUT**

829

527327

32

**Explanation**

There are 33 test cases in the sample input. Following it are three lines that contains two numbers each, so the output also contains three lines that contain the reverse of the sum of the two reversed numbers.

**Time Limit:**2.0 sec(s) for all input files combined.

**Memory Limit:**256 MB

**Source Limit:**1024 KB

**Marking Scheme:**Marks are awarded when all the testcases pass.

**Allowed Languages:**C, C++, Clojure, C#, D, Erlang, F#, Go, Groovy, Haskell, Java, Java 8, JavaScript(Rhino), JavaScript(Node.js), Lisp, Lisp (SBCL), Lua, Objective-C, OCaml, Octave, Pascal, Perl, PHP, Python, Python 3, R(RScript), Racket, Ruby, Rust, Scala, Scala 2.11.8, Swift, Visual Basic

<https://www.hackerearth.com/practice/basic-programming/implementation/basics-of-implementation/practice-problems/algorithm/the-reversed-numbers/>

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace ConsoleApplication1

{

class Program

{

static void Main(string[] args)

{

int t = int.Parse(Console.ReadLine());

while (t-- > 0)

{

//string s = "238746 39857";

string s = Console.ReadLine();

char[] a = s.Split(' ')[0].ToCharArray();

char[] b = s.Split(' ')[1].ToCharArray();

Array.Reverse(a);

Array.Reverse(b);

int res = int.Parse(new string(a)) + int.Parse(new string(b));

char[] rev\_res = res.ToString().ToCharArray();

Array.Reverse(rev\_res);

Console.WriteLine(int.Parse( new string(rev\_res)));

}

Console.ReadLine();

}

}

}