CS F211
Data Structures and Algorithms
Assignment - 0

Allow language: **C**

General Tips

- Try to use functions as much as possible in your code. Functions increase reusability and the pass-by-value feature provides a significant help sometimes. Modularizing your code also helps you to debug efficiently.
- Use `scanf` to read characters/strings from STDIN. Avoid using `getchar`, `getc` or `gets`. Try to read up about character suppression in `scanf` as it will be very helpful in some of the problems.
- Use `printf` instead of `putc`, `putchar` or `puts` to print character/string output on STDOUT.
- Indent your code appropriately and use proper variable names. These increase readability and writability of the code. Also, Use comments wherever necessary.
- Use a proper IDEs like Sublime Text or VSCode as they help to run and test your code on multiple test-cases easily.

A: Big Sum

Depending on the platform, the largest integer data type in C will allow you to store numbers that are tens of digits long. In this question, you will write a program that will enable you to add non-negative integers that are at most a thousand digits long. To this end, create two strings that can be used to store upto 1000 digit positive integers in base 10; Each digit will be a character in the usual positional number system. Obtain these numbers as input from the user and assume that the first number is greater than or equal to the second and that the numbers are non-negative. Your program should compute their sum and store it in a string without padding zeros to the left. Also, you should print the sum. Write the entire program in the `main()` function without having to write separate functions. Further, you are not allowed to use library functions from the `string.h` library.

Input

The first line of input contains the integer A ($0 \leq A \leq 10^{1000}$) represented as a string. The second line of input contains the integer B ($A \leq B \leq 10^{1000}$) represented as a string.

Output

Print a single string, representing the sum of two long integers provided to you. Note that, the output must not be zero padded on the left.

```
input
78977
98173

output
177150
```

```
input
123
1

output
124
```

```
input
68730457693724357452985234523765
11974275824875928729875504587907

output
80704733518600286182860739111672
```