PRACTICE-09_PTR-MAT

Monster Multiply Revisited (p1v1d1)

You will be given two non-negative integers a and b in two lines of the input. You have to print their product in the output.
The numbers a and b may contain upto 50 digits. However, for your convenience, we will pad the numbers with zeros to the left so that we always give you 50 digits for each number.

Note, however, that your output must not contain leading zeros (i.e. do not output 00023, output 23 instead). Be careful that your output may contain more than 50 digits. If the product is zero, simply print a single zero in the output.

The numbers in this question will not fit even inside long variables. You should use arrays to solve this problem.

All Test Cases (Visible + Hidden)

Input	Output
00000000000000000000000000000000000000000000001000 00000000000000000000000000000000000000000000000323	323000
00000000000000000000123456789012345678901234567890 00000000000000000000000000000000000000000000000000	0
00000023827889475437571832584970549769864590123843 00000000007504054608468557543849324032940329403894	1788057838282367510943644394990419354362134067686869025297444009450112279278264446
99999999999999999999999999999999999999999999999999 99999999999999999999999999999999999999999999999999	9999999999999999999999999999999999999999999999999800000000000000000000000000000000
11111111111111111111111111111111111111111111111111 11111111111111111111111111111111111111111111111111	1234567901234567901234567901234567901234567901234543209876543209876543209876543209
10000000000000000000000000000000000000000000000000 10000000000000000000000000000000000000000000000000	1000000000000000000000000000000000000000000000000000000000000000000000000000000000

Proper Case (p1v2d1)

In a single line of input, you will be given a string containing only English alphabet letters, punctuation marks (the list of punctuation marks for this question is fullstop . comma , exclamation mark ! colon : semicolon ; hyphen -) and spaces and tabs. There will be no newline characters in the string. The string will contain at most 999 characters. However, the person writing the string has untidy capitalization in the string. The capitalization rule for this question is given below.

1. If the first character in the line is an alphabet character, it should be in upper case.
2. If an alphabet character occurs after a punctuation mark or a space/tab, it should be in upper case.
3. All other occurrences of alphabet characters should be in lower case.

Rewrite and print the string with proper capitalization.

All Test Cases (Visible + Hidden)

Input	Output
hello WORLD	Hello World
HI,my name is MR C AND i can teach you programming.Are you ready!	Hi,My Name Is Mr C And I Can Teach You Programming.Are You Ready!
WoW! a::B is the same as A::b	Wow! A::B Is The Same As A::B
I can show you the world. sHINING shiMMering SPlenDid!Tell me princess-now when did you last let your heart decide	I Can Show You The World. Shining Shimmering Splendid!Tell Me Princess-Now When Did You Last Let Your Heart Decide
Many researchers choose this way of capitalization for their research-papers.	Many Researchers Choose This Way Of Capitalization For Their Research-Papers.
tAlE aS oLd As TiMe! TuNe As OlD aS sOnG.	Tale As Old As Time! Tune As Old As Song.

Num2Word (p1v3d1)

A long variable takes 8 bytes to store whereas a char takes 1 byte. This means a long variable has enough space to store 8 characters inside it. In the input, you will be given a single non-negative integer. Extract the 8 characters stored inside that long integer and store them inside a character array with 9 characters - the 9th character should be set by you to the NULL character. Print the character array as a string.

Example:
'W' = 87 = 01010111
'O' = 79 = 01001111
'\0' = 0 = 00000000 (in binary)

So the binary number 01010111 01001111 01010111 00000000 00000000 00000000 00000000 00000000
(spaces between bytes are just for ease of reading) which corresponds to the long integer
6291342861971488768
actually represents the sequence of 8 characters 'W' 'O' 'W' '\0' '\0' '\0' '\0' '\0'

HINT: There are at least three ways to solve this problem - let us know on Piazza if you find more

1. Method 1: use remainder and divide trick to extract bytes out of the long number
2. Method 2: a convenient way of doing method 1 using bitshift operators (search online to learn more - we will do them in class later).
3. Method 3: use a char* pointer to access individual bytes of the long variable. The week 8 tutorial will cover how to perform arithmetic using pointers. This can be used to extract characters from a long integer. WARNING: if using this method, be careful that the Prutor server is a *Little Endian* machine (search online to learn more) which means it actually stores the bytes in the reverse order i.e. the long number 6291342861971488768 will be stored as
00000000 00000000 00000000 00000000 00000000 01010111 01001111 01010111

All Test Cases (Visible + Hidden)

Input	Output
6291342861971488768	WOW
5217737160284717824	Hi Mr C
29590389553906432	
5795967168569242112	Pointer
2531905900372128256	###
4714531563309524736	Amazing

Mr C meets Matrices (p1v4d1)

Just as Mr C can store arrays that look like vectors, he can also store matrices as a 2D array. He can also store higher order tensors as 3D, 4D arrays. Look at the code given to you which declares a matrix with 4 rows and 6 columns, reads elements from the input and prints them in row wise fashion.

Modify this code to print the elements of the matrix in column wise order i.e. print column 1 elements first then column 2 elements and so on. Elements of a column should be printed on the same line, separated by a single space. Elements of different columns should be printed on different lines.

All Test Cases (Visible + Hidden)

Input	Output

	1 10 100 1000
	2 20 200 2000
	3 30 300 3000
	4 40 400 4000
	5 50 500 5000
	6 60 600 6000

	1 10 100 1000
	2 20 200 2000
	3 30 300 3000
	4 40 400 4000
	5 50 500 5000
	6 60 600 6000