**5 kyu**

**Last digit of a large number**

2354688% of 49520of 3,616[dburgoyne](https://www.codewars.com/users/dburgoyne)

C++

* [TRAIN AGAIN](https://www.codewars.com/kata/last-digit-of-a-large-number/train/cpp)
* [NEXT KATA](https://www.codewars.com/trainer/cpp)

Details

[Solutions](https://www.codewars.com/kata/last-digit-of-a-large-number/solutions/cpp)

[Forks (12)](https://www.codewars.com/kata/last-digit-of-a-large-number/forks/cpp)

[Discourse (105)](https://www.codewars.com/kata/last-digit-of-a-large-number/discuss/cpp)

* Add to Collection
* |
* Share this kata:

Define a function that takes in two numbers a and b and returns the last decimal digit of a^b. Note that a and b may be very large!

For example, the last decimal digit of 9^7 is 9, since 9^7 = 4782969. The last decimal digit of (2^200)^(2^300), which has over 10^92 decimal digits, is 6.

The inputs to your function will always be non-negative integers.

Examples

last\_digit("4", "1") // returns 4

last\_digit("4", "2") // returns 6

last\_digit("9", "7") // returns 9

last\_digit("10","10000000000") // returns 0

Remarks

**JavaScript, C++, R**

Since these languages don't have native arbitrarily large integers, your arguments are going to be strings representing non-negative integers.

<https://www.codewars.com/kata/last-digit-of-a-large-number/cpp>

// C++ code to find last digit of a^b

//#include <bits/stdc++.h>

#include <iostream>

#include <stdio.h>

#include <math.h>

using namespace std;

// Function to find b % a

int Modulo(int a, std::string b)

{

// Initialize result

int mod = 0;

// calculating mod of b with a to make

// b like 0 <= b < a

for (int i = 0; i < b.length(); i++)

mod = (mod \* 10 + b[i] - '0') % a;

return mod; // return modulo

}

int last\_digit(std::string str1, std::string str2) {

// return 0;

int len\_a = str1.length(), len\_b = str2.length();

// if a and b both are 0

if (len\_a == 1 && len\_b == 1 && str2[0] == '0' && str1[0] == '0')

return 1;

// if exponent is 0

if (len\_b == 1 && str2[0] == '0')

return 1;

// if base is 0

if (len\_a == 1 && str1 [0] == '0')

return 0;

// if exponent is divisible by 4 that means last

// digit will be pow(a, 4) % 10.

// if exponent is not divisible by 4 that means last

// digit will be pow(a, b%4) % 10

int exp = (Modulo(4, str2) == 0) ? 4 : Modulo(4, str2);

// Find last digit in 'a' and compute its exponent

int res = pow(str1 [len\_a - 1] - '0', exp);

// Return last digit of result

return res % 10;

}

// Driver program to run test case

int main()

{

// char a[] = "117", b[] = "3";

string a = "117";

string b = "3";

cout << last\_digit(a, b);

return 0;

}