



Congratulations! You passed!

TO PASS 80% or higher

Keep Learning

GRADE
100%

Week 5 - Programming Assignment [Optional]

TOTAL POINTS 1

1. Your goal this week is to write a program to compute discrete log modulo a prime p . Let g be some element in \mathbb{Z}_p^* and suppose you are given h in \mathbb{Z}_p^* such that $h = g^x$ where $1 \leq x \leq 2^{40}$. Your goal is to find x . More precisely, the input to your program is p, g, h and the output is x .

1 / 1 point

The trivial algorithm for this problem is to try all 2^{40} possible values of x until the correct one is found, that is until we find an x satisfying $h = g^x$ in \mathbb{Z}_p . This requires 2^{40} multiplications. In this project you will implement an algorithm that runs in time roughly $\sqrt{2^{40}} = 2^{20}$ using a meet in the middle attack.

Let $B = 2^{20}$. Since x is less than B^2

we can write the unknown x base B as $x = x_0 B + x_1$

where x_0, x_1 are in the range $[0, B - 1]$. Then

$$h = g^x = g^{x_0 B + x_1} = (g^B)^{x_0} \cdot g^{x_1} \quad \text{in } \mathbb{Z}_p.$$

By moving the term g^{x_1} to the other side we obtain

$$\boxed{h/g^{x_1} = (g^B)^{x_0}} \quad \text{in } \mathbb{Z}_p.$$

The variables in this equation are x_0, x_1 and everything else is known: you are given g, h and $B = 2^{20}$. Since the variables x_0 and x_1 are now on different sides of the equation we can find a solution using meet in the middle ([Lecture 3.3](#) at 14:25):

- First build a hash table of all possible values of the left hand side h/g^{x_1} for $x_1 = 0, 1, \dots, 2^{20}$.
- Then for each value $x_0 = 0, 1, 2, \dots, 2^{20}$ check if the right hand side $(g^B)^{x_0}$ is in this hash table. If so, then you have found a solution (x_0, x_1) from which you can compute the required x as $x = x_0 B + x_1$.

The overall work is about 2^{20} multiplications to build the table and another 2^{20} lookups in this table.

Now that we have an algorithm, here is the problem to solve:

```
p = 134078079299425970995740249982058461274793658205923933 \
    77723561443721764030073546976801874298166903427690031 \
    858186486050853753882811946569946433649006084171
g = 11717829880366207009516117596335367088558084999998952205 \
    59997945906392949973658374667057217647146031292859482967 \
    5428279466566527115212748467589894601965568
h = 323947510405045044356526437872806578864909752095244 \
    952783479245297198197614329255807385693795855318053 \
    2878928001494706097394108577585732452307673444020333
```

Each of these three numbers is about 153 digits. Find x such that $h = g^x$ in \mathbb{Z}_p .

To solve this assignment it is best to use an environment that supports multi-precision and modular arithmetic. In Python you could use the [gmpy2](#) or [numbthy](#) modules. Both can be used for modular inversion and exponentiation. In C you can use [GMP](#). In Java use a BigInteger class which can perform mod, modPow and modInverse operations.

375374217830

Correct