MAXimal

added: 10 Jun 2008 18:03
Edited: 15 Dec 2011 21:27

Reverse member ring modulo

Contents [hide]

Definition

Suppose we are given a natural module m, and consider the ring formed by the module (ie, consisting of numbers from $0$before $m-1$). Then for some elements of this ring can be found **an inverse element** .

The inverse of the number $a$modulo $m$called a number $b$that:

$$a \cdot b \equiv 1 \pmod m,$$

and it is often denoted by a^{-1}.

It is clear that for the zero return item does not exist ever; for the remaining elements of the inverse can exist or not. It is argued that the inverse exists only for those elements $a$that **are relatively prime** to the modulus m.

Consider the following two ways of finding the inverse element employed, provided that it exists.

Finally, we consider an algorithm which allows you to find backlinks to all numbers modulo some linear time.

Finding using the Extended Euclidean algorithm

Consider the auxiliary equation (in the unknown $x$and y):

$$a \cdot x + m \cdot y = 1.$$

This linear Diophantine equation of second order . As shown in the corresponding article from the condition $\gcd(a, m) = 1$, it follows that this equation has a solution which can be found using the Extended Euclidean algorithm (hence the same way, it follows that when $\gcd(a, m) \neq 1$, solutions, and therefore the inverse element does not exist).

On the other hand, if we take from both sides of the residue modulo m, we get:

$$a \cdot x = 1 \pmod m.$$

Thus found $x$and will be the inverse of a.

Implementation (including that found $x$necessary to take on the module m, and x may be negative):

```
int x, y;
int g = gcdex (a, m, x, y);
if (g != 1)
        cout << "no solution";
else {
        x = (x % m + m) % m;
        cout << x;
}
```

Asymptotic behavior of the solutions obtained $O(\log m)$.

Finding the binary exponentiation

We use Euler's theorem:

$$a^{\phi(m)} \equiv 1 \pmod m,$$

which is true just in the case of relatively prime $a$and m.

Incidentally, in the case of a simple module m, we get even more simple statement - Fermat's little theorem:

$$a^{m-1} \equiv 1 \pmod m.$$

Multiply both sides of each equation by a^{-1}, we obtain:

- for any module m:

  $$a^{\phi(m)-1} \equiv a^{-1} \pmod m,$$

- for a simple module m:

  $$a^{m-2} \equiv a^{-1} \pmod m.$$

Thus, we have obtained the formula for the direct calculation of the inverse. For practical applications typically use an efficient algorithm for binary exponentiation , which in this case will bring about for exponentiation $O(\log m)$.

This method seems to be a little bit easier as described in the previous paragraph, but it requires knowledge of the values of the Euler function that actually requires the factorization of the module m, which can sometimes be very difficult.

If the factorization of the number is known, then this method also works for the asymptotic behavior $O(\log m)$.

Finding all simple modulo a given linear time

Let a simple module m. Required for each number in the interval $[1; m-1]$ to find its inverse.

Applying the algorithms described above, we get a solution with the asymptotic behavior $O(m \log m)$. Here we present a simple solution to the asymptotic behavior $O(m)$.

The decision is as follows. We denote $r[i]$ the inverse of the desired number of i modulo m. Then the $i > 1$ true identity:

$$r[i] = -\left\lfloor \frac{m}{i} \right\rfloor \cdot r[m \bmod i]. \pmod m$$

The implementation of this amazingly concise solutions:

```
r[1] = 1;
for (int i=2; i<m; ++i)
        r[i] = (m - (m/i) * r[m%i] % m) % m;
```

The proof of this solution is a chain of simple transformations:

We write out the value $m \bmod i$:

$$m \bmod i = m - \left\lfloor \frac{m}{i} \right\rfloor \cdot i,$$

whence, taking both sides modulo m, we get:

$$m \bmod i = - \left\lfloor \frac{m}{i} \right\rfloor \cdot i. \pmod m$$

Multiplying both sides by the inverse of i the inverse to $(m \bmod i)$ obtain the required formula:

$$r[i] = -\left\lfloor \frac{m}{i} \right\rfloor \cdot r[m \bmod i], \pmod m$$

QED.

8 Комментариев e-maxx Войти

Лучшее вначале Поделиться Избранный

Присоединиться к обсуждению...

Дмитрий • год назад

Если я все правильно понял, то этот заголовок неверный: "Нахождение всех ПРОСТЫХ по заданному модулю за линейное время".

7 ^ | v • Ответить • Поделиться ›

e_maxx • 8 месяцев назад

daddssdfgsdgdfg

4 ^ | v • Ответить • Поделиться ›

Serega • 10 месяцев назад

Пусть m=84, а обратный ищется для числа 53. Расширенный алгоритм Евклида вернет в качестве обратного число -19. И если потом к этому числу применить формулу

x = (x % m + m) % m, то полученный таким образом ответ 69 уже не будет корректным обратным элементом.

2 ^ | v • Ответить • Поделиться ›

Ser → Serega • 4 месяца назад

Ваш коммент не корректный. Вернет 1.

^ | v • Ответить • Поделиться ›

Mohammad Samiul Islam • 2 года назад

Finally I understood how Extended Euclidean finds modular inverse.

Btw, Euler theorem is true when a and m are relatively simple. Does that mean their gcd (a, m) is 1?

1 ^ | v • Ответить • Поделиться ›

e_maxx Модератор → Mohammad Samiul Islam • 2 года назад

Yes, "relatively prime" means "these numbers have no common divisor, greater than 1" - in other words, "gcd = 1".

2 ^ | v • Ответить • Поделиться ›

Валера • 2 месяца назад

А можно ли как-то за линейное время посчитать обратные на промежутке, когда m - не простое?

^ | v • Ответить • Поделиться ›

Ilnur • 6 месяцев назад

не понял как здесь писать Вам в личку, поэтому пишу здесь об ошибке(орфографической)