MATH 470: Communications and Cryptography

Homework 3

Due date: 13 September 2023 *Name: Huy Lai*

Problem 1. Let $p = 587$ and numbers $a = 345$, compute $a^{-1} \mod p$ in two ways:

(i) Use the extended Euclidean algorithm.

(ii) Use the fast power algorithm and Fermat's little theorem.

Solution:
Using the Extended Euclidean Algorithm

q	r	u	v
	587	0	1
	345	1	0
1	242	-1	1
1	103	2	-1
2	36	-5	3
2	31	12	-7
1	5	-17	10
6	1	114	-67
5	0	u	v

According to the EEA, the integers $(u, v) = (114, -67)$.
$354(114) + 587(-67) = 1$
$345^{-1} \equiv 114 \mod 587$

Using the fast power algorithm and Fermat's Little Theorem.

Fermat's Little Theorem states

$$a^{p-1} \equiv 1 \mod p$$

Multiplying both sides of this by a^{-1} results in

$$a^{p-2} \equiv a^{-1} \mod p$$

Therefore,

$$a^{-1} \equiv a^{p-2} \mod p$$

$p - 2 = 585_{10} = 1001001001_2$

$a^{2^0} \equiv a^1 \equiv 345 \mod p$
$a^{2^1} \equiv a^2 \equiv 451 \mod p$
$a^{2^2} \equiv a^4 \equiv 299 \mod p$
$a^{2^3} \equiv a^8 \equiv 177 \mod p$
$a^{2^4} \equiv a^{16} \equiv 218 \mod p$
$a^{2^5} \equiv a^{32} \equiv 564 \mod p$
$a^{2^6} \equiv a^{64} \equiv 529 \mod p$
$a^{2^7} \equiv a^{128} \equiv 429 \mod p$
$a^{2^8} \equiv a^{256} \equiv 310 \mod p$
$a^{2^9} \equiv a^{512} \equiv 419 \mod p$

$345^{585} = 345^{2^9+2^6+2^3+2^0} \equiv 419 \cdot 529 \cdot 177 \cdot 345 \equiv 114 \mod 587$
$345^{-1} \equiv 114 \mod 587$

Problem 2. Let p be a prime and let q be a prime that divides $p-1$. Let $a \in \mathbb{F}_p^*$ and let $b = a^{\frac{p-1}{q}}$. Prove that either $b = 1$ or else b has order q.

Solution:
The proof is as follows.

Proof. Let k be the order of b.
Raising the definition of b to the power of q will result in

$$b^q = a^{p-1}$$

By Fermat's Little Theorem, $b^q \equiv a^{p-1} \equiv 1 \mod p$
By Proposition 1.29 in the Textbook, $k \mid q$. Since q is prime, then $k = 1$ or $k = q$.
As a result either b has order q, or it has order 1. □

Problem 3. Let p be a prime such that $q = \frac{1}{2}(p-1)$ is also prime. Suppose that g is an integer satisfying

$$g \not\equiv 0 \mod p, g \not\equiv \pm 1 \mod p, g^q \not\equiv 1 \mod p$$

Prove that g is a primitive root modulo p.

Solution:
The proof is as follows.

Proof. Let k be the order of g. Then by proposition, $k \mid (p-1)$.
Since $p - 1 = 2q$ with q prime, this means that

$$k = 1 \text{ or } k = 2 \text{ or } k = q \text{ or } k = 2q$$

Since $g \not\equiv \pm 1 \mod p$, then $k \neq 1$ and $k \neq 2$.
Since $g^q \not\equiv 1 \mod p$, then $k \neq q$.
As a result $k = 2q \Rightarrow k = p - 1$.
With this, the order of g is $p - 1$, this satisfies the definition of a primitive root. Therefore, g is a primitive root modulo p. □

Problem 4. Let p be an odd prime number and let b be an integer with $p \nmid b$. Prove that either b has two square roots modulo p or else b has no square roots modulo p. In other words, prove that the congruence

$$X^2 \equiv b \mod p$$

has either two solutions or no solutions in $\mathbb{Z}/p\mathbb{Z}$. (What happens for $p = 2$? What happens if $p \mid b$?)

Solution:
The proof is as follows.

Proof. Let a_1, a_2 be solutions to the congruency.
Then by proposition of modulo, $p \mid \left(a_1^2 - b\right)$ and $p \mid \left(a_2^2 - b\right)$.
By proposition of divisibility, $p \mid \left[\left(a_1^2 - b\right) - \left(a_2^2 - b\right)\right]$
This can be rewritten as

$$p \mid (a_1 - a_2)(a_1 + a_2)$$

From this, p must divide either $a_1 - a_2$ or $a_1 + a_2$.
If $p \mid (a_1 - a_2)$, then $a_1 \equiv a_2 \mod p$. If $p \mid (a_1 + a_2)$, then $a_1 \equiv -a_2 \mod p$.
As a result, there are a most two solutions.

We prove that one solution is not possible by contradiction.
Let a be the solution to the congruency.
Then, $a^2 \equiv b \mod p$.
We can generate another unique solution by completing the square as follows

$$p^2 + 2ap + a^2 \equiv b \mod p$$

This gives that $(p + a)^2 \equiv b \mod p$. However, $p + a \neq a$.
Therefore, if one solution exists, another can be found.
This proves that there are zero or two solutions. □

Problem 5. Problem 5

Subproblem 1. Let $p = 13$ and let $g = 2$. Note that p is prime and that g is a primitive root modulo p. Make a list of the powers of g and their orders modulo p (i.e., for each $a \in \{1, 2, 3, \cdots, 12\}$, write down $g^a \mod p$ and the order of $g^a \mod p$). What are all the primitive roots modulo p? Compute $\phi(p-1)$, where ϕ is the Euler's totient function.

Solution:

a	$2^a \mod 13$	Order
1	2	12
2	4	6
3	8	4
4	3	3
5	6	2
6	12	2
7	11	12
8	9	3
9	5	6
10	10	12
11	7	12
12	1	1

The primitive roots modulo p are:

$2^1 \mod p = 2$
$2^{10} \mod p = 10$
$2^7 \mod p = 11$
$2^{11} \mod p = 7$

$\phi(n-1) = \phi(12) = 4$

Subproblem 2. Let p be a prime, let g be a primitive root modulo p, and let a be an integer. Prove that the order of $g^a \mod p$ is exactly $\dfrac{p-1}{\gcd(a, p-1)}$. Explain why this implies that the number of primitive roots modulo p is exactly $\phi(p-1)$, assuming that a primitive root g modulo p exists. (Looking over your work in part (a) may help you gain some intuition).

Solution:
Let $n = \dfrac{p-1}{\gcd(a, p-1)}$ To prove that the order of $g^a \mod p$ is exactly n we must show that the order cannot be smaller than this number and that the order cannot be larger than this number.

First prove that the order cannot be smaller

Proof. Let k be the order of $g^a \mod p$
This means that k is the smallest positive integer such that $(g^a)^k \equiv 1 \mod p$.
Assume that $k < n$.

By Fermat's Little Theorem, $g^{p-1} \equiv 1 \mod p$.
Raising both sides by the power of n results in

$$\left(g^{p-1}\right)^n \equiv 1^n \equiv 1 \mod p$$

Consider

$$(g^a)^n \equiv g^{an} \mod p$$

Multiplying the exponent by $\dfrac{p-1}{p-1}$ results in

$$g^{an} \equiv g^{(p-1)\cdot\frac{an}{p-1}} \mod p$$

Using Fermat's Little Theorem gives

$$g^{an} \equiv 1^{\frac{an}{p-1}} \mod p$$

This implies that the order of $g^a \mod p$ is at most n.
This contriducts with the assumption that the order is smaller □

Problem 6. You may assume that the following integers p and q are primes:

$$p = 12345678901234567890123456789012345678901234567890123456789012345678901234567890123459287$$
$$q = 61728394506172839450617283945061728394506172839450617283945061728394506172839450617283945061729643$$

Also note that $p = 2q + 1$. Find the smallest positive integer g that is a primitive root modulo p.

Solution:

```python
# from sympy.ntheory.residue_ntheory import primitive_root
from math import sqrt
from typing import List, Set


def fast_pow(base, power, modulo):
    result = 1
    base %= modulo

    while power > 0:
        if power & 1:
            result = (result * base) % modulo
        base = (base * base) % modulo
        power >>= 1

    return result


def is_primitive_root(g: int, p: int) -> bool:
    # Factors were found using Wolfram Alpha
    # https://www.wolframalpha.com/input?i=Factors+of+%5B%2F%2Fquantity%3A123456
    factors = [
        2, 617283945061728394506172839450617283945061728394506172839450617283945
    for q in factors:
        if (fast_pow(g, (p - 1) // q, p) == 1):
            return False
        return True
```

```python
def primitive_root(p: int) -> int:

    for g in range(2, p):
        print(f"Checking: {g}")
        if (is_primitive_root(g, p)):
            return g

    return -1


def main() -> None:
    p = 12345678901234567890123456789012345678901234567890123456789012345678901234567890123
    # q = 617283945061728394506172839450617283945061728394506172839450617283945061728394506
    proot = primitive_root(p)
    print(f"The smallest primitive root modulo p is: {proot}")


if __name__ == "__main__":
    main()
```

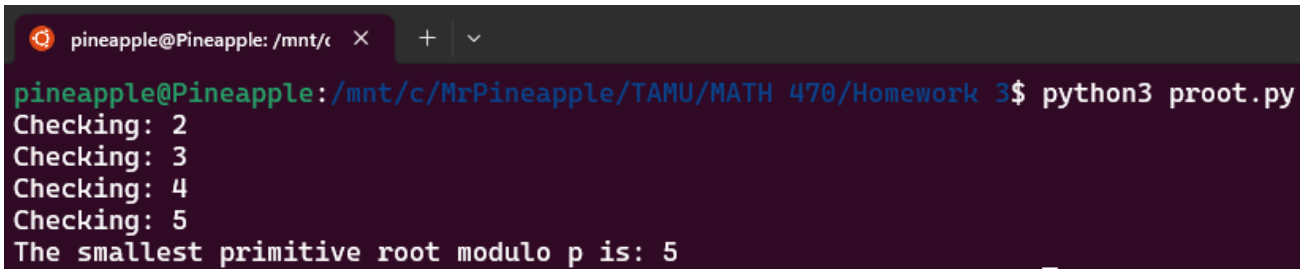

Figure 1: Output