**Winter 2021 MTH 440/540 Homework 4**
**Wyatt Whiting**

**Instructions.** Due via Gradescope on Friday February 26. Solutions must be typed. Problems not marked with * are required for everyone. Problems marked with $*$ are required for 540 students and are optional challenge problems for 440 students (they will not be graded for 440 students). Use Sage only on problems which explicitly say to use Sage; solve all other problems without Sage and show your calculations.)

**24.** Let $p$ be a prime number and let $n \geq 1$ be an integer. Show that there exists an integer of order $n$ mod $p$ if and only if $p \equiv 1 \pmod{n}$.

Assume $p \equiv 1 \pmod{n}$, which implies $n \mid p-1$, and therefore $p - 1 = n \cdot q$ for some integer $q$. It then follows that $\frac{p-1}{n}$ is an integer. We can then select some $g$ such that $g$ is a primitive root mod $p$, and by definition has order $p-1$ mod $p$ and $g^{p-1} \equiv 1 \pmod{p}$. If we now consider the element $m = g^{\frac{p-1}{n}}$, then $m^n \equiv 1 \pmod{p}$ by construction. Furthermore, $n$ is the smallest positive integer for which this holds because the $g$ has order $p-1$ mod $p$, so no terms $g^k$ for $1 \leq k < p-1$ will be congruent to 1 (mod $p$). We are also guaranteed that $g^{\frac{p-1}{n}}$ exists since we have established that $\frac{p-1}{n}$ is an integer which must be less than $p-1$. Thus, the element $m = g^{\frac{p-1}{n}}$ exists and has order $n$ mod $p$.

Assume there exists some $m$ such that $m$ has order $n$ mod $p$. So then $m^n \equiv 1 \pmod{p} \implies p \nmid m$. We can then apply Fermat's Little Theorem and state that $m^{p-1} \equiv 1 \pmod{p}$. Since we have both $m^{p-1} \equiv 1 \pmod{p}$ and $m$ has order $n$ mod $p$, then $n \mid p-1$. It then follows that $p - 1 \equiv 0 \pmod{n} \implies p \equiv 1 \pmod{n}$.

We have now shown if there exits some integer of order $n$ mod $p$, then it follows that $p \equiv 1 \pmod{n}$, and if we have $p \equiv 1 \pmod{n}$, there must exist some integer with order $n$ mod $p$. Therefore, there exists an integer of order $n$ mod $p$ if and only if $p \equiv 1 \pmod{n}$.

**25.** By hand using only the quadratic reciprocity law, state whether each of the following congruences has a solution for $x$. You may assume that all moduli are prime numbers.

**(a)** $x^2 \equiv 17 \pmod{67}$

By the law of quadratic reciprocity, we have

$$\begin{aligned}\left(\frac{17}{67}\right)\left(\frac{67}{17}\right) &= (-1)^{\frac{67-1}{1}\frac{17-1}{2}} \\ &= (-1)^{33\cdot 8} = (-1)^{264} \\ &= 1\end{aligned}$$

We can then calculate the Legendre symbol $(\frac{67}{17})$. We see that $67 = 3 \cdot 17 + 16 \implies 67 \equiv 16 \pmod{17}$, and taking $x = 4$ give us $4^2 \equiv 67 \pmod{17}$. The Legendre symbol $(\frac{67}{17})$ therefore is equal to 1, and now we know the symbol $(\frac{17}{67}) = 1$, so the congruence $x^2 \equiv 17 \pmod{67}$ has a solution for $x$.

**(b)** $x^2 \equiv 3 \pmod{67}$

$$\left(\frac{3}{67}\right)\left(\frac{67}{3}\right) = (-1)^{\frac{67-1}{1}\frac{3-1}{2}}$$
$$= (-1)^{33\cdot 1} = -1$$

$$67 = 22 \cdot 3 + 1 \implies 67 \equiv 1 \pmod 3$$

We can take $x = 1$ to get $x^2 = 1^2 \equiv 1 \pmod 3 \equiv 67 \pmod 3$, so $(\frac{67}{3}) = 1$, and therefore $(\frac{3}{67}) = -1$, so there is no solution to the congruence $x^2 \equiv 3 \pmod{67}$.

**(c)** $2x^2 + 5x + 1 \equiv 0 \pmod{67}$

Given this quadratic congruence, a solution exists if and only if the discriminant $b^2 - 4ac$ is a square modulo 67. We then inspect the discriminant:

$$b^2 - 4ac = 5^2 - 4 \cdot 2 \cdot 1$$
$$= 25 - 8$$
$$= 17$$

So the congruence $2x^2 + 5x + 1 \equiv 0 \pmod{67}$ has a solution for $x$ if and only if there exists some $n$ such that $n^2 \equiv 17 \pmod{67}$. From part (a), we already know this is the case. Therefore, the quadratic congruence has a solution for $x$.

**(d)** $x^2 \equiv 65 \pmod{101}$

We begin by factoring the top of the Legendre symbol:

$$\left(\frac{65}{101}\right) = \left(\frac{5 \cdot 13}{101}\right) = \left(\frac{5}{101}\right)\left(\frac{13}{101}\right)$$

We can then apply quadratic reciprocity to each of the terms:

$$\left(\frac{5}{101}\right) = (-1)^{\frac{5-1}{2}\frac{101-1}{2}}\left(\frac{101}{5}\right)$$
$$= (1)\left(\frac{101}{5}\right) = \left(\frac{1}{5}\right)$$
$$= 1,$$

and

$$\left(\frac{13}{101}\right) = (-1)^{\frac{13-1}{2}\frac{5-1}{2}}\left(\frac{101}{13}\right)$$
$$= (1)\left(\frac{101}{13}\right) = \left(\frac{10}{13}\right)$$
$$= 1.$$

Putting these together, we see

$$\left(\frac{65}{101}\right) = \left(\frac{5}{101}\right)\left(\frac{13}{101}\right) = 1 \cdot 1 = 1.$$

Thus, the system $x^2 \equiv 65 \pmod{101}$ has a solution for some $x$.

**(e)** $x^2 \equiv 5 \pmod{234985724985724985724938572398723984723987423984339823 91}$

Let $l =$ [that very large number above]. Then,

$$\begin{aligned}\left(\frac{5}{l}\right)\left(\frac{l}{5}\right) &= (-1)^{\frac{l-1}{1}\frac{5-1}{2}} \\ &= (-1)^{\frac{l-1}{2}\cdot 2} = 1\end{aligned}$$

Since the last digit of $l$ ends in 1, then $l \equiv 1 \pmod{5}$. Then taking $x = 1$ gives us $x^2 = 1^2 \equiv 1 \pmod{5}$, so then $(\frac{l}{5}) = 1$ and therefore $(\frac{5}{l}) = 1$, so the congruence $x^2 \equiv 5 \pmod{l}$ has a solution for some $x$.

**26.** For each equation in problem **25** which has a solution, find a solution. (You may use Sage but do not use Sage's `sqrt` command, which does work mod $p$ but is too powerful and makes this problem too easy. As always, show how you arrived at each answer. Hint: In some cases, it might help to use the remarks near the top of page 87 of Stein.)

I wrote the following function in Sage to solve the systems:

```
def solve_quad_resid(q, p):
    x = 1
    q = mod(q, p)
    while not mod(x^2, p) == q: x += 1
    return x
```

This function returns the smallest positive integer $x$ such that $x^2 \equiv q \pmod{p}$. We can then call the function for several values. In part (a), we take $q = 17$ and $p = 67$. Then,

```
solve_quad_resid(17,67)
```

```
> 33
```

so $33^2 \equiv 17 \pmod{67}$. Then for part (d):

```
solve_quad_resid(65, 101)
```

```
> 41
```

so we have $41^2 \equiv 65 \pmod{101}$.

This algorithm works fine when $p$ is relatively small, but will be extremely time consuming when $p$ is large as in part (e). However, we can use fact that if $p \equiv 5 \pmod{4}$ and $a$ is a quadratic residue mod $p$, then $a^{\frac{p+1}{2}}$ is a square root of $a$. This relies on $p \equiv 3 \pmod{4}$, which we can also check with Sage:

```
l=23498572498572498572493857239872398472398742398433982391
if mod(l,4) == 3:
    print(mod(5,l)^((l + 1) / 4))
```

```
> 136821633892045566729666238362270248822397330691214934 41.
```

So the condition holds and we find the desired square root. For the sake of not having giant, unreadable numbers written out, that value squared is congruent to 5 modulus the giant modulus from the problem.

**27.** Alice and Bob are communicating via a Diffie-Hellman key exchange. Alice sends the following data to Nikita:

$$p = 49253 \qquad g = 2 \qquad g^a \equiv 558 \pmod{p}$$

Bob responds to Alice with:

$$g^b \equiv 32288 \pmod{p}$$

You intercept both transmissions. Using Sage and a brute-force attack on the discrete logarithm problem, find Alice and Bob's shared secret key $s = g^{ab} \pmod{p}$.

```
def break_encrypt(p,g,ga,gb):
    a = 1
    b = 1
    # loop until g^a mod p == ga
    # and until g^b mod p == gb
    while(mod(g,p)^a != ga): a += 1
    while(mod(g,p)^b != gb): b += 1

    # return g^(ab) mod p
    return mod(g,p)^(a*b)

p = 49253
g = 2
ga_modp = 558
gb_modp = 32288

print(break_encrypt(p,g,ga_modp,gb_modp))
```

```
> 43739
```

So we have $s = 43739 \equiv g^{ab} \pmod{p}$.

**28.** Use Sage and the method of Fermat factorization to factor $41156989185107 = pq$ for primes $p$ and $q$. You may use Sage's `is_square( )`, `sqrt( )`, `int( )` and `is_prime( )` commands, but do not use powerful commands such as `factor( )` or `prime_factors( )`.

```
def fermat_factor(n):
    # round square root of n up to get a
    # set b what it would need to be
    a = ceil(sqrt(n))
    b2 = a^2 - n

    # loop until b is a square
    while not is_square(b2):
        # increment a, reclculate b2
        a += 1
        b2 = a^2 - n

    # loop ends, so now we have a, b
    # such that b^2 = a^2 - n -> n = a^2 - b^2
    # so return factor (a - b)
    return a - sqrt(b2)

pq = 41156989185107
p = fermat_factor(pq)
q = pq / p
print(p, q)

> 6409511 6421237
```

So we have $41156989185107 = 6409511 \cdot 6421237$.

**29.** Alice wants to send Bob an email containing her login password for a certain account. Her password is her cat's name[1]. Of course, she wants to keep this secret from everyone else. Bob sends to Alice the RSA public key $(n, e) = (41156989185107, 7)$; notice that $n$ occurs in problem **28**. Alice writes down her cat's name, which is 7 letters, and converts it into a block $x$ of 14 numerical digits using the rule $A = 01$, $B = 02$, $C = 03$, ..., $Z = 26$. Using the encryption function $E(x) \equiv x^e \pmod{n}$, Alice encrypts her message and sends the encrypted message $y = E(x)$ to Bob over email.

For example, if Alice wanted to encrypt BEAVER, she would first convert BEAVER to the plaintext block $x = 020501220518$ using $B = 02$, $E = 05$, etc, she would calculate

$$y \equiv E(020501220518) \equiv 20501220518^7 \equiv 15602263727108 \pmod{41156989185107}$$

and send the encrypted message 15602263727108 to Bob.

The actual encrypted message Alice sends to Bob is

17792272918826

which you have intercepted. Use Sage and your answer to problem **28** to crack the cryptosystem and decrypt the message. What is the name of Alice's cat?

Following is my Sage code:

```
p = 6409511
q = 6421237
e = 7
# result of Euler's Totient function
phi_n = (p - 1) * (q - 1)
# d has inverse of e mod phi_n
d = xgcd(e, phi_n)[1]
# message string
message = 17792272918826
# now raise message to the power of d mod pq
undone = mod(message, p*q)^(d)
undone

> 3080118120905
```

We then prepend a 0, resulting in the character pairs `03 08 01 18 12 09 05`. Applying the encoding key, we then get the name of Alice's cat, `C H A R L I E`. Hurrah!

[1]This is actually my cat's name. (But it is not my password!)