# MA357, Spring 2020 — Problem Set 4

This assignment is due on **Friday, March 6**. More congruences than you can shake a stick at.

**1.** NTG, Exercise 4.7.5.

**2.** NTG, Exercise 4.7.8.

**3.** NTG, Exercise 4.7.11.

**4.** Show that if $n$ is an integer, then so is $\dfrac{n^5}{5} + \dfrac{n^3}{3} + \dfrac{7n}{15}$.

**5.** NTG, Exercise 4.7.25.

**6.** NTG, Exercise 4.7.34.

**7.** A band of 17 pirates, upon dividing their gold coins, found that three coins remained after the coins had been apportioned evenly. In the ensuing brawl, one of the pirates was killed. The wealth was again redistributed equally, and this time ten coins remained. Again an argument broke out and one of the pirates was killed. This time the fortune was distributed evenly among the survivors. What is the least possible value for the number of coins the pirates had initially?

**8.** Let $k$ and $s$ be two positive integers. Show that there exists a sequence of $k$ consecutive integers each of which is divisible by an $s$-th power. (For example, if $k = 3$ and $s = 2$, we are asking for three consecutive integers which are each divisible by a square. Then 48, 49, and 50 are an example, since they are divisible by $4^2$, $7^2$, and $5^2$, respectively. You want to prove that such sequences always exist.)

**9.** In this problem we want to prove the multiplicative structure of $\mathbb{Z}/m\mathbb{Z}$. Remember that we already know that $a$ will have an inverse mod $m$ if and only if $\gcd(a, m) = 1$.

a. How many congruence classes are invertible mod 4? How about mod 12?

b. Suppose $m$ is prime. Show that any $a \not\equiv 0 \pmod{m}$ is invertible mod $m$.

c. Suppose $a$ is invertible mod $m$. Show that there must exist an integer $k \geqslant 1$ such that $a^k \equiv 1 \pmod{m}$.

d. Let $m = 7$. For each $a \equiv 1, 2, 3, 4, 5, 6 \pmod{7}$, find the least $k \geqslant 1$ such that $a^k \equiv 1 \pmod{7}$.

10. NTG, Exercise 4.7.27.

**To Explore:** Given a number $n$, how can we find its factorization? You might start by writing computer program that will factor (small) integers by using trial division. But that takes a long time. Are there better ways?

One problem with trial division is that it is only good if there is a small prime factor. There is a different method, based on the identity $a^2 - b^2 = (a+b)(a-b)$, that works best when there are two factors of about the same size.

Let $n$ be the number to be factored.

a. Let $a = \lfloor \sqrt{n} \rfloor + 1$, $i = 1$, $u = a^2$.

b. If $u - n$ is a square, then set $u - n = b^2$ and then $n = (a-b)(a+b)$, end.

c. If not, let $u = u + 2a + 1$, $a = a + 1$, $i = i + 1$.

d. If $i \leqslant M$, go back to step (b).

e. If $i > M$, report that the algorithm has failed to factor $n$.

As given, the algorithm needs a parameter $M$ that limits the number of iterations it goes through. This allows us to give up when the algorithm is taking too long to find a factorization. The underlying factoring method is usually attributed to Fermat.

Here are some questions to think about:

a. Why does this work?

b. How high would $M$ have to be set in order to be sure that we can factor any number $n$?

c. Does this method improve on trial division?

d. Could we combine this method with trial division to get something more efficient?

Try writing factoring programs using (a) the trial division method, (b) the method outlined above, and (c) some other method (e.g., Pollard's rho). Compare their speeds by attempting to factor various interesting numbers, such as Mersenne numbers $M_k = 2^k - 1$ and "repunits" $R_k = 111\ldots1$ with $k$ digits; if $k$ is composite these always factor, so try prime values of $k$.