Number theory is the branch of mathematics that studies properties of the integers. It is full of conjectures (propositions that have not been proven and have not been refuted) that are very simple to state, but extremely difficult to resolve. We'll stick to a lighter side of number theory that gives us a nice playground to exercise the concepts we just learned – logic, proofs, induction, invariants. We'll also see how this concepts help us figure out correctness and termination of algorithms.

1 Die hard, once and for all

You are standing next to a water source. You have two empty jugs: A 3 litre jug and a 5 litre jug with no marks on them. You must fill the larger jug with precisely 4 litres of water. Can you do it?

This scenario is straight out of the 1995 classic "Die Hard 3: With a Vengeance" featuring Bruce Willis and Samuel L. Jackson. Should they fail to complete this task within 5 minutes, a bomb goes off and New York City is obliterated. In the nick of time, Bruce and Samuel come up with a solution:

small jug	large jug	action
—	—	fill up large jug from source
—	5ℓ	fill up small jug from large jug
3ℓ	2ℓ	empty small jug
—	2ℓ	pour large jug contents into small jug
2ℓ	—	fill up large jug from source
2ℓ	5ℓ	fill up small jug from large jug
3ℓ	4ℓ	done!

The rumour is that in the sequels, Bruce and Samuel will be asked to measure — always 4 litres — with a 21 litre and a 26 litre jug; with an 899 litre and 1,147 litre jug; and finally, in the last movie of the series, "Die once and for all", with a 3 litre and 6 litre jug. What should they do? To help them out, let us first come up with a *mathematical model* of their problem.

The water jug problem. Let a, b be two positive integers with and v be another positive integer. Given two initially empty jugs – jug A and jug B – with capacities a litres and b litres, respectively, and an infinite source of water, does there exist a sequence of pouring steps so that one of the jugs ends up with v litres of water?

To be precise, let us clearly specify the rules for pouring water:

Rules for pouring water. We are allowed to perform the following steps:

1. **Empty out a jug:** The emptied out jug now contains 0 litres. The contents of the other jug stay the same.

2. **Fill up a jug from the source:** The filled up jug now contains as much water as its capacity. The contents of the other jug stay the same.
3. **Transfer water between jugs:** Water can be poured from jug X into jug Y until jug Y is full *or* jug X is empty.

Greatest common divisors and integer combinations

Let a, b be two integers. We say a *divides* b if $ak = b$ for some integer k. For example, 2 divides 4 and -6 but 2 does not divide 5; 4 does not divide 2; and every number divides zero.

The *greatest common divisor (GCD)* of two integers a, b is the largest integer k so that k divides a and k divides b. We write $\gcd(a, b)$ for the GCD of a and b. For example, $\gcd(2, 6) = 2$, $\gcd(4, 6) = 2$, $\gcd(3, 5) = 1$.

A number c is an *integer combination* of a and b if there exist integers s and t such that $c = s \cdot a + t \cdot b$. For example,

- 2 is an integer combination of 2 and 6 because $2 = 1 \cdot 2 + 0 \cdot 6$.
- 2 is an integer combination of 4 and 6 because $2 = (-1) \cdot 4 + 1 \cdot 6$.
- 1 is an integer combination of 3 and 5 because $1 = 2 \cdot 3 + (-1) \cdot 5$.

It looks like GCDs and integer combinations are closely related. This is not an accident.

Lemma 1. *For all integers a and b, $\gcd(a, b)$ divides every integer combination of a and b.*

Proof. Let c be an integer combination of a and b. Then we can write

$$c = s \cdot a + t \cdot b \tag{1}$$

for some integers a and b. Since $\gcd(a, b)$ divides a, we can write $a = a' \cdot \gcd(a, b)$ for some integer a'. Since $\gcd(a, b)$ divides b, we can write $b = b' \cdot \gcd(a, b)$ for some integer b'. Substituting into (1) this gives

$$c = s \cdot a' \cdot \gcd(a, b) + t \cdot b' \cdot \gcd(a, b) = (sa' + tb') \cdot \gcd(a, b).$$

Therefore $\gcd(a, b)$ divides c. □

An invariant

We can now state an *invariant* about the die hard problem:

Lemma 2. *The amount of water in each jug is always an integer combination of a and b.*

Proof. We prove this lemma by induction on the number of pourings n.

Base case $n = 0$: Initially, each jug has 0 liters of water and $0 = 0 \cdot a + 0 \cdot b$.

Inductive step: Assume the amount of water in each jug is an integer combination of a and b after n pourings. Specifically, assume jug A has $x = s_1 a + t_1 b$ liters and jug B has $y = s_2 a + t_2 b$ litres. We consider cases depending on the action taken in the $(n+1)$st pouring:

- **Case 1.** One of the jugs is emptied: Then the emptied jug has 0 litres and the other one has the same amount of water it had before. They are both integer combinations of a and b.
- **Case 2.** One of the jugs is filled up from the source. We consider two subcases: If jug A is filled up from the source, then jug A has a litres and $a = 1 \cdot a + 0 \cdot b$. The contents of jug B don't change, so their contents are both integer combinations of a and b. If jug B is filled up from the source, the argument is *analogous*.[1]
- **Case 3.** Water is poured from one jug into the other. We consider three subcases:
  - Water is poured from A to B and A becomes empty. Then B will have $x + y$ litres of water and $x + y = (s_1 + s_2)a + (t_1 + t_2)b$, which is an integer combination of a and b.
  - Water is poured from A to B and B becomes full. Then A will have $x + (b - y)$ litres of water remaining and $x + (b - y) = (s_1 - s_2)a + (t_1 - t_2 + 1)b$, which is an integer combination of a and b.
  - Water is poured from B to A. The argument is analogous to the previous two cases.

It follows that in all cases, the amount of water in each jug is an integer combination of a and b after $n + 1$ pourings. □

Now from Lemma 14 and Lemma 1, we can obtain a useful corollary:

Corollary 3. $\gcd(a, b)$ *always divides the amount of water in each jug.*

In particular, if we have a 3 litre jug and a 6 litre jug, we can never end up with 4 litres of water:

Corollary 4. *In "Die once and for all", Bruce dies.*

2 Euclid's algorithm

Euclid's algorithm is a procedure for calculating the GCD of two positive integers. It was invented by the Euclideans around 3,000 years ago and it still the best known procedure for calculating GCDs. To explain Euclid's algorithm, we need to recall division with remainder.

Theorem 5. *Let n and d be integers with $d > 0$. There exists unique integers q and r such that*

$$n = q \cdot d + r \quad \textit{and} \quad 0 \leq r < d.$$

[1] "Analogous" means that it follows by the exact same logic as the previous case by a change of notation. Writing it out and reading through it is tedious, but you should be comfortable doing this if the need arose.

For example, if $n = 13$ and $d = 3$, we can write $13 = 4 \cdot 3 + 1$. Moreover, $q = 4$ and $r = 1$ are unique assuming r is in the range $0 \leq r < d$.

The number q can be calculated using the usual "division rule" you learn in school (stopping at the decimal point). The number r is the remainder you obtain after you subtract qd from n.

Proof. First, we show existence: Let q be the largest integer such that $qd \leq n$. That means $(q+1)d > n$. Then $0 \leq n - qd < d$. Set $r = n - qd$.

Now we show uniqueness: Suppose we can write n in the desired form in two different ways:

$$n = qd + r, 0 \leq r < d$$
$$n = q'd + r', 0 \leq r' < d.$$

Then $qd + r = q'd + r'$, so $(q - q')d = r' - r$. Therefore $r' - r$ divides d. Since $0 \leq r, r' < d$ we must have $-d < r' - r < d$. The only number in this range that divides d is zero, so $r' - r = 0$ and $q' - q = 0$. It follows that $q' = q$ and $r' = r$, so the two representations are the same. □

Euclid's algorithm is a *recursive* algorithm for calculating the GCD of positive integers.

Euclid's algorithm $E(n, d)$, where n, d are integers such that $n > d \geq 0$.
If $d = 0$, return n.
Otherwise, write $n = qd + r$ where $0 \leq r < d$. Return $E(d, r)$.

Let's apply Euclid's algorithm to calculate the GCD of 1147 and 899:

$$\begin{aligned}
E(1147, 899) &= E(899, 248) && \text{because } 1147 = 1 \cdot 899 + 248 \\
&= E(248, 155) && \text{because } 899 = 3 \cdot 248 + 155 \\
&= E(155, 93) && \text{because } 248 = 1 \cdot 155 + 93 \\
&= E(93, 62) && \text{because } 155 = 1 \cdot 93 + 62 \\
&= E(62, 31) && \text{because } 155 = 1 \cdot 93 + 62 \\
&= E(31, 0) && \text{because } 62 = 2 \cdot 31 + 0 \\
&= 31.
\end{aligned}$$

How can we be sure that Euclid's algorithm indeed outputs the GCD of n and d? How can we even be sure that Euclid's algorithm *terminates*? This is a theorem that we will have to prove:

Theorem 6. *For every pair of integers n, d such that $n > d \geq 0$, $E(n, d)$ terminates and outputs* $\gcd(n, d)$.

To prove this theorem, we'll need a little lemma.

Lemma 7. *For all integers a, b, and t,* $\gcd(a, b) = \gcd(a + tb, b)$.

Proof of Theorem 11. We will prove the theorem by strong induction on d.

Base case: When $d = 0$, the algorithm returns $n = \gcd(n, 0)$ and terminates.

Inductive step: Now assume the algorithm terminates and outputs $\gcd(n, d')$ for all input pairs (n, d'), where $n > d'$ and $0 \leq d' \leq d$. Consider what the algorithm does on input $(n, d+1)$: It writes $n = q(d+1)+r$ with $0 \leq r < d+1$ and returns $E(d+1, r)$. By our inductive assumption, $E(d+1, r)$ must terminate since r is between 0 and d and $E(d+1, r)$ outputs $\gcd(r, d+1)$. Therefore, $E(n, d+1)$ must terminate as well. Moreover, $E(n, d+1)$ outputs $\gcd(r, d+1) = \gcd(n - q(d+1), d+1)$. By Lemma 7, this number equals $\gcd(n, d+1)$. □

Proof of Lemma 7. We will show that, for every integer k, k divides a and b if and only if k divides $a+tb$ and b. In particular, this means that the *largest* divisors of the two pairs of numbers – namely, their GCDs – must be equal.

First, assume k divides both a and b. Then we can write $a = a'k$ and $b = b'k$ for integers a', b'. We get that $a + tb = (a' + tb')k$, so k also divides $a' + tb'$.

For the other direction, assume k divides both $a + tb$ and b and write $a + tb = a'k$ and $b = b'k$ for integers a', b'. Then $a = a'k - tb'k = (a' - tb')k$, so k also divides a. □

3 How to solve any water jug problem

Corollary 3 is *sufficient* condition for Bruce to die: If $\gcd(a, b)$ is not 1, 2, or 4, then Bruce dies. This happens in the 3 litre + 6 litre scenario, as well as the 1147 litre + 899 litre one. What if we have a 21 litre and a 26 litre jug? The GCD of 21 and 26 is 1, so Corollary 3 does not rule out the possibility that Bruce may survive. But can he, actually, survive?

The key to survival is the following converse to Lemma 1.

Lemma 8. *Let a, b be any two nonnegative[2] integers. Then $\gcd(a, b)$ can be written as an integer combination of a and b.*

Corollary 9. *Let a, b be any two positive integers. Then there exist nonnegative integers s and t such that $\gcd(a, b) = s \cdot a - t \cdot b$.*

Proof of Corollary 9. By Lemma 8, there exist integers s and t such that $\gcd(a, b) = s \cdot a + t \cdot b$. Then for every integer k,

$$(s + kb) \cdot a + (t - ka) \cdot b = sa + tb = \gcd(a, b).$$

No matter what the values of s and t are, when k is sufficiently large, $s + kb$ is positive and $t - ka$ is negative. This gives us a representation of $\gcd(a, b)$ of the desired form. □

[2]The assumption that the integers are nonnegative is not important; in fact, it is not necessary, but we will make it because it makes the arguments a bit simpler.

For example, if $a = 21$ and $b = 26$, then $\gcd(21, 26) = 1$ and we can write

$$1 = 5 \cdot 21 - 4 \cdot 26.$$

Multiplying both sides by 4, we get

$$4 = 20 \cdot 21 - 16 \cdot 26.$$

Now here is how Bruce can handle a 21 litre and a 26 litre jug: Keep filling up the 21 litre jug for a total of 20 times. Whenever it becomes full, pour all the water into the 26 litre jug. If that one becomes full, empty it out and continue pouring into it.

At the end, the 26 litre jug will contain exactly 4 litres. Here is why: We poured a total of $20 \cdot 21$ litres from the source, and some multiple of 26 litres out of the larger jug, so the amount of water left in the larger jug is of the form $20 \cdot 21 - t \cdot 26$. Let's see what these numbers look like:

t	$\cdots$	14	15	16	17	18	$\cdots$
$20 \cdot 21 - t \cdot 26$	$\cdots$	56	30	4	-22	-48	$\cdots$

So there *must* be 4 liters left in the 26 liter jug: This is the only number in the list which is within the capacity of the jug!

We can now specify the exact conditions under which the water jug problem has a solution:

Theorem 10. *Assume a, b, v are positive integers and $a \leq b$. The water jug problem with jugs of capacity a and b and target v has a solution* if and only if $\gcd(a, b)$ *divides v and $0 \leq v \leq b$.*

Proof. Assume the water jug problem has a solution. By Corollary 3, $\gcd(a, b)$ must divide v. Clearly v must also be within the size of the larger bin.

Now assume v is a multiple of $\gcd(a, b)$ and $0 \leq v \leq b$. The cases $v = 0$ and $v = b$ are easy, so from here on we'll assume $0 < v < b$. By Corollary 9 we can write

$$\gcd(a, b) = sa - tb$$

for some nonnegative integers s, t. Since v divides $\gcd(a, b)$, we can write $v = k\gcd(a, b)$ for some k and

$$v = (ks)a - (kt)b.$$

To obtain v litres, we proceed like this: Keep filling jug A for a total of ks times. Whenever it becomes full, pour all the water into jug B. If that one becomes full, empty it out and continue pouring into it.

Suppose jug B was emptied a total of t' times. Then the amount of water left inside jug B in the end is

$$(ks)a - t'b = (ks)a - (kt)b + (kt - t')b = v + (kt - t')b$$

If $kt - t' < 0$, this number is negative; if $kt - t' > 0$, then it is larger than b. In either case, this is not a valid amount of water to be left in jug B. It must be that $kt - t' = 0$ and there are exactly v litres left in jug B. □

In fact, we do not need to fill jug A exactly ks times; we can interrupt the procedure as soon as we get 4 litres into jug B. I wrote a program that implements this strategy. You can play with it.

We are almost done – all that remains is to prove Lemma 8. This follows by analysing an extension of Euclid's algorithm, which we won't do here. The purpose of the extended Euclid's algorithm is to compute the numbers s and t from Lemma 8.

Extended Euclid's algorithm $X(n, d)$, where n, d are integers such that $n > d \geq 0$.
If $d = 0$, return $(1, 0)$.
Otherwise,
Write $n = qd + r$ where $0 \leq r < d$.
Calculate $(s, t) = X(d, r)$.
Return $(t, s - q \cdot t)$.

Lemma 8 is now a consequence of the following Theorem:

Theorem 11. *For every pair of integers n, d such that $n > d \geq 0$, $X(n, d)$ terminates and outputs a pair of integers (s, t) such that $s \cdot n + t \cdot d = \gcd(n, d)$.*

4 Prime numbers

I am sure you already know quite a bit about prime numbers, so let me just state two useful facts. If you want to read the proofs, you can find them in the textbook.

Lemma 12. *If p is a prime and p divides $a \cdot b$, then p divides a or p divides b.*

Theorem 13 (Fundamental theorem of arithmetic)**.** *Every positive integer n can be written uniquely as a product of primes.*

For example, $15 = 3 \cdot 5$, $8 = 2 \cdot 2 \cdot 2$, $12 = 2 \cdot 2 \cdot 3$, and $17 = 17$.

5 Modular arithmetic

Let's fix an integer p, which we will call the *modulus*. Theorem 5 says that for every integer n there exists a unique integer between 0 and $p - 1$ such that $n = qp + r$ for some q. We call r the *remainder* of n *modulo* p — in short n modulo p — and we write

$$r = n \bmod p.$$

We can do arithmetic using the numbers $0, 1, \ldots, p - 1$: additions, subtractions, multiplications, divisions. The arithmetic remains valid as long as we take the remainders of all expressions modulo p. Taking remainders all the time is tedious so instead we'll take advantage of the concept of congruence: Two integers m and n are *congruent* modulo p if p divides $m - n$. We write $m \equiv n$

$(\mathrm{mod}\ p)$ for "m and n are congruent modulo p." Congruences behave nicely with respect to additions and multiplications:

$$\text{If} \quad x \equiv x' \ (\mathrm{mod}\ p) \quad \text{and} \quad y \equiv y' \ (\mathrm{mod}\ p), \qquad \begin{array}{l} \text{then} \quad x + y \equiv x' + y' \ (\mathrm{mod}\ p) \\ \text{and} \quad x \cdot y \equiv x' \cdot y' \ (\mathrm{mod}\ p). \end{array}$$

Addition and subtraction Addition and subtraction modulo p is fairly easy: For example, if I want to calculate $3+8$ modulo 9 first I add 3 and 8 as integers to get 11 then I take the remainder of 11 modulo 9 to get 2:

$$(3+8) \bmod 9 = 11 \bmod 9 = 2.$$

The *additive inverse* of a modulo p is the number $(-a) \mod p$. Say the modulus is 5. Then the additive inverse of 0 is 0, the additive inverse of 1 is 4, the additive inverse of 2 is 3, and vice versa. Subtraction can then be done by replacing each negative number with its additive inverse.

$$(3-7) \bmod 8 = (3+1) \bmod 8 = 4 \bmod 8 = 4.$$

Multiplication and division From here on we will assume that the modulus p is a prime number. (If it isn't, some of the following statements won't be true.) Just like addition, to multiply two numbers modulo p, first we multiply them as integers and then we take the remainder modulo p, for example

$$3 \cdot 4 \bmod 7 = 12 \bmod 7 = 5.$$

What about division? First, let's show how to calculate *multiplicative inverses*, namely ratios of the form $1/x$. This is more commonly written as x^{-1}. Division by zero is forbidden, but otherwise we have a nice lemma:

Lemma 14. *For every x between 1 and $p-1$ there exists a unique y between 1 and $p-1$ such that $xy \equiv 1 \ (\mathrm{mod}\ p)$.*

Proof. First we show existence of y. Take any x between 1 and $p-1$. Since p is prime, $\gcd(x,p) = 1$. By Lemma 8 there exist integers s, t for which $s \cdot x + t \cdot p = 1$. Taking both sides modulo p, we get that $s \cdot x \equiv 1 \ (\mathrm{mod}\ p)$. Take $y = s \bmod p$. Then $y \cdot x \equiv s \cdot x \ (\mathrm{mod}\ p)$, so $y \cdot x \equiv 1 \ (\mathrm{mod}\ p)$.

Now we show uniqueness. We just saw that for every x between 1 and $p-1$ there is *at least one* y in the same range such that $xy \equiv 1 \ (\mathrm{mod}\ p)$. Now we show there is at most one such y. Suppose $xy \equiv 1 \ (\mathrm{mod}\ p)$ and $xy' \equiv 1 \ (\mathrm{mod}\ p)$. Then $x(y-y') \equiv 0 \ (\mathrm{mod}\ p)$, so p divides $x(y-y')$. By Lemma 12, p divides x or p divides $y-y'$. Since $x < p$, p must divide $y-y'$. But $y-y'$ is a number between $-(p-2)$ and $p-2$, so this is only possible if $y = y'$, namely $y - y' = 0$. □

The proof of Lemma 14 not only tells us that x^{-1} exists, but also how to calculate it: First, use the Extended Euclid's algorithm to find s and t such that $s \cdot x + t \cdot p = 1$. Then output $s \bmod p$. For example, to get the multiplicative inverse of 11 modulo 17, I run $X(11,17)$ to get $(-3,2)$, namely $(-3) \cdot 11 + 2 \cdot 17 = 1$. Therefore

$$11^{-1} \bmod 17 = -3 \bmod 17 = 14.$$

To divide two numbers, we first take the multiplicative inverse of the denominator, then multiply by the numerator, for instance to divide 3 by 11 modulo 17 we calculate

$$3 \cdot 11^{-1} \bmod 17 = 3 \cdot 14 \bmod 17 = 42 \bmod 17 = 8.$$

References

This lecture is based on Chapter 4 of the text *Mathematics for Computer Science* by E. Lehman, T. Leighton, and A. Meyer.