DM549 and DS820

Lecture 13: Primes and Greatest Common Divisors

Kevin Aguyar Brix
Email: kabrix@iomada.sdu.dk

University of Southern Denmark

28 October, 2024

Last Time: Introduction to Modular Arithmetic

Definitions: Divisibility, quotient, remainder.

Definition (Definition 4.1.3)

Let $a, b \in \mathbb{Z}$ and $m \in \mathbb{Z}^+$. Then we have the *congruence* (kongruens)

$$a \equiv b \pmod{m}$$

if and only if m divides $a - b$. We also say that *a and b are congruent (kongruente) modulo m*.

Theorem (only proof sketch, Theorems 4.1.3 and 4.1.4)

Let $a, b \in \mathbb{Z}$ and $m \in \mathbb{Z}^+$. Then the following statements are equivalent:

(i) $a \equiv b \pmod{m}$

(ii) $a \bmod m = b \bmod m$

(iii) There exists $k \in \mathbb{Z}$ with $a = b + km$.

Last Time: Adding and Multiplying Congruences

Theorem (Theorem 4.5.1)

Let $a, b, c, d \in \mathbb{Z}$ and $m \in \mathbb{Z}^+$. If $a \equiv b \pmod{m}$ and $c \equiv d \pmod{m}$, then

$$a + b \equiv c + d \pmod{m} \quad \text{and} \quad a \cdot b \equiv c \cdot d \pmod{m}.$$

Remark:

- In particular, that means that we can add the same number to both sides of a congruence or multiply them with the same number.
- The above statements also hold with subtraction:
  - The theorem shows that $c \equiv d \pmod{m}$ implies $-c \equiv -d \pmod{m}$.
  - We can hence add $-c \equiv -d \pmod{m}$ to subtract $c \equiv d \pmod{m}$.
- We cannot always divide both sides of a congruence by the same number!

A Trick for Faster Computation

Corollary (proof only for addition, Corollary 4.1.2)

Let $a, b \in \mathbb{Z}$ and $m \in \mathbb{Z}^+$. Then

$$(a + b) \bmod m \equiv (a \bmod m) + (b \bmod m) \pmod{m}$$

and

$$(a \cdot b) \bmod m \equiv (a \bmod m) \cdot (b \bmod m) \pmod{m}.$$

A Quiz

Go to `pollev.com/kevs`

Primes and The Fundamental Theorem of Arithmetic

Definition (Definition 4.3.1)

Let $n \in \mathbb{Z}$ with $n \geq 2$. We call n a *prime number* (primtal) if the only positive factors of n are 1 and n. If n is not prime, n is called *composite* (sammensat).

The Fundamental Theorem of Arithmetic (no proof, Theorem 4.3.1)

Let $n \in \mathbb{Z}$ with $n \geq 2$. One can write n as a product of prime numbers in exactly one way (up to rearranging factors).

Theorem (Theorem 4.3.3)

There are infinitely many primes.

A Joke


6'
5'
4'
3'
2357
PRIME SUSPECTS
ONEDUHFOOL
CARTOONS
WWW.KOMEDIE.CO.ZA

The Greatest Common Divisor

Definition (Definition 4.3.2)

Let $a, b \in \mathbb{Z} \setminus \{0\}$. Then

$$\gcd(a, b) = \max \left\{ d \;\middle|\; d \mid a \wedge d \mid b \right\}$$

is called the *greatest common divisor* (største fælles divisor) of a and b.

Definition (Definition 4.3.3)

Let $a, b \in \mathbb{Z} \setminus \{0\}$. We call a, b *relatively prime* (inbyrdes primske) if $gcd(a, b) = 1$.

The Least Common Multiple

Definition (Definition 4.3.2)

Let $a, b \in \mathbb{Z}^+$. Then

$$\operatorname{lcm}(a, b) = \min \left\{ m \;\middle|\; a \mid m \wedge b \mid m \right\}$$

is called the *least common multiple* (mindst fælles multiplum) of a and b.

Theorem (Theorem 4.3.5)

Let $a, b \in \mathbb{Z}^+$. Then

$$a \cdot b = \gcd(a, b) \cdot \operatorname{lcm}(a, b).$$

Proof sketch:

- For each prime p, let a_p (b_p) be the number of times that p occurs as a prime factor in a (b).
- In $\gcd(a, b)$, p occurs $\min(a_p, b_p)$ times; in $\operatorname{lcm}(a, b)$, $\max(a_p, b_p)$ times.
- So in $\gcd(a, b) \cdot \operatorname{lcm}(a, b)$, it occurs $\min(a_p, a_p) + \max(a_p, b_p) = a_p + b_p$ times, just like in $a \cdot b$.

The Euclidean Algorithm

Lemma (Lemma 4.3.1)

Let $a, b, q, r \in \mathbb{Z} \setminus \{0\}$ with $a = bq + r$. Then $\gcd(a, b) = \gcd(b, r)$.

Euclidean Algorithm: To compute $\gcd(a, b)$ assuming $a > b$,

- Start with $x = a$ and $y = b$.
- As long as $y \neq 0$:
  - Replace (x, y) with $(y, x \bmod y)$.
- Return x.

Note: By lemma above, it follows the Euclidean Algorithm is correct.

Theorem (Theorem 4.3.6)

Let $a, b \in \mathbb{Z}^+$. Then there exist $s, t \in \mathbb{Z}$ with $\gcd(a, b) = sa + tb$.

Note: Lemma follows by going through computations in reverse order.