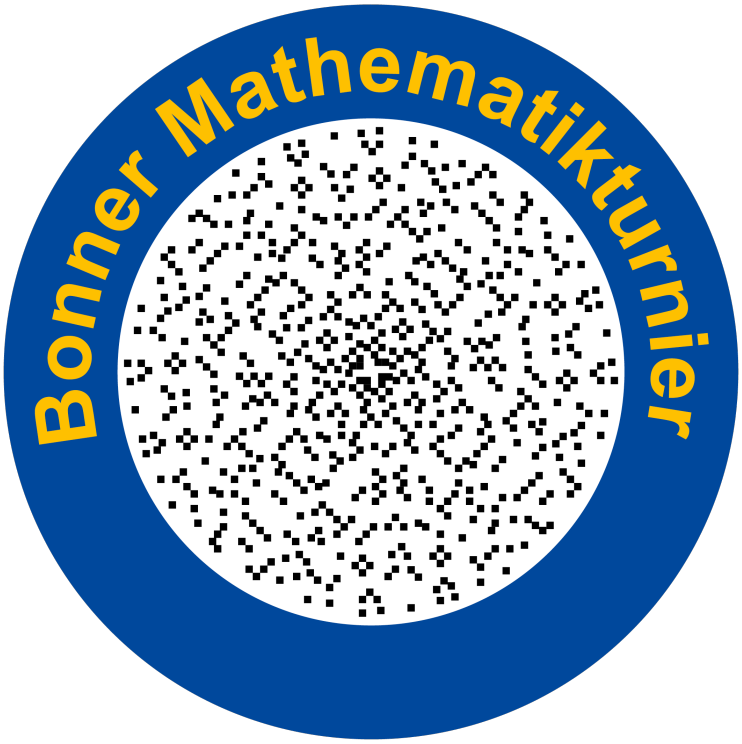


Kryptographie und Zahlentheorie

Internationales Mathematikturnier 2026
Vorbereitungsmaterial

18. September 2026

Liebe Teilnehmer*innen des Mathematikturniers 2026,

in ein paar Wochen ist es so weit, und ihr könnt am Mathematikturnier teilnehmen. Die erste Runde dieses Turniers heißt „Sum of Us“ und behandelt in diesem Jahr das Thema „Kryptographie und Zahlentheorie“. Hierauf könnt ihr euch mit dem Übungsmaterial dieses Dokuments vorbereiten. Eure Lehrkräfte sind bestimmt bereit, euch hierbei zu helfen!

Folgende Hilfsmittel sind in diesem Jahr zugelassen:

- Schmierpapier, Stifte, Bleistift, Lineal.
- Das Vorbereitungsmaterial (ausgedruckt oder auf einem Tablet).

Bitte beachtet, dass keine Taschenrechner erlaubt sind, ihr also insbesondere die Modulo-Berechnungen von Hand durchführen müsst!

Die Unterlagen wurden in diesem Jahr von Víctor González Alonso, Michael J. Gruber, Thorsten Holm und Florian Leydecker von der Leibniz Universität Hannover geschrieben.

Die Organisationsteams,
Stefan Hartmann & Rainer Kaenders (Universität Bonn)
Víctor González Alonso, Michael J. Gruber, Thorsten Holm & Florian Leydecker (Leibniz Universität Hannover)
Niels Bonneux & Joeri Van der Veken (KU Leuven)
Manus Visser (Radboud Universiteit Nijmegen)
Michael Eichmair, Robin Gludovatz & Dmytro Rzhemuvskyi (Universität Wien)

Inhaltsverzeichnis

1. Kryptographie - gestern und heute 4
1.1. Grundbegriffe 4
1.2. Symmetrische Verfahren 5
1.3. Asymmetrische Verfahren 6
1.4. Kombinierte Verfahren 9
2. Ganze Zahlen und Teilbarkeit 9
2.1. Teilbarkeit in den ganzen Zahlen 10
2.2. Primzahlen 10
2.3. Division mit Rest und größte gemeinsame Teiler 13
2.4. Der Euklidische Algorithmus 14
3. Modulare Arithmetik und Kryptographie 19
3.1. Rechnen mit Kongruenzen 19
3.2. Teilbarkeitsregeln 22
3.3. Anwendung: Prüfziffern bei Strichcodes, Kreditkarten und IBAN 24
3.4. Modulare Inverse 27
3.5. Einfache symmetrische Verfahren 30
3.6. Diffie-Hellman-Schlüsselaustausch 34
3.7. ElGamal-Algorithmus 36
3.8. Diskreter Logarithmus 38
3.9. Die Eulersche φ-Funktion 43
3.10. Das RSA-Verfahren zur Verschlüsselung 47
4. Lösungen zu den Aufgaben 53

1. Kryptographie - gestern und heute

In einer zunehmend digitalisierten Welt, in der Informationen ständig über das Internet ausgetauscht werden, spielt die Sicherheit dieser Daten eine entscheidende Rolle. Hier setzt die Kryptographie an – die Wissenschaft der Verschlüsselung und Sicherheit von Informationen. Bereits in der Antike wurden kryptographische Methoden verwendet, um geheime Nachrichten zu übermitteln. Im Laufe der Jahrhunderte haben sich diese Methoden stark weiterentwickelt und heute sind sie ein unverzichtbares Werkzeug in der Informationstechnologie und -sicherheit. Zwei Hauptarten der Kryptographie sind die symmetrische und die asymmetrische Kryptographie. Beide Ansätze haben ihre eigenen Anwendungsfälle und spezifische Vorteile sowie Herausforderungen. Oft werden sie kombiniert, um die Vorteile zu bündeln und die jeweiligen Nachteile zu vermeiden. In diesem Kapitel lernen wir Grundbegriffe und ein paar einfache (symmetrische) Verfahren kennen. Bevor wir uns mit komplexeren (asymmetrischen) Verfahren beschäftigen können, stellen wir die hierfür notwendigen mathematischen Werkzeuge bereit: Modulare Arithmetik.

1.1. **Grundbegriffe.**

Kryptographie umfasst verschiedene Techniken, um Daten vor unbefugtem Zugriff zu schützen. Dies wird typischerweise durch Verschlüsselung erreicht, bei der Daten (auch „plaintext“ genannt) in eine verschlüsselte Form umgewandelt werden, die ohne den richtigen Schlüssel unverständlich ist. Der umgekehrte Prozess, bei dem die ursprünglichen Informationen aus der verschlüsselten Form zurückgewonnen werden, heißt Entschlüsselung. Etwas mathematischer formuliert: Ein kryptographisches Verfahren besteht aus einer Verschlüsselungsfunktion encrypt und einer Entschlüsselungsfunktion decrypt mit der Eigenschaft

$$\text{decrypt}\big(\text{encrypt}(m)\big) = m \quad \text{für jede Nachricht } m.$$

encrypt bildet Nachrichten auf Kryptotexte ab, decrypt umgekehrt, und die obige Eigenschaft heißt in der Mathematik: decrypt ist Linksinverse von encrypt.

Beispiel 1.1

Alice möchte Bob eine geheime Nachricht schicken. Die beiden haben vereinbart, folgende Verschlüsselungsfunktion zu benutzen:

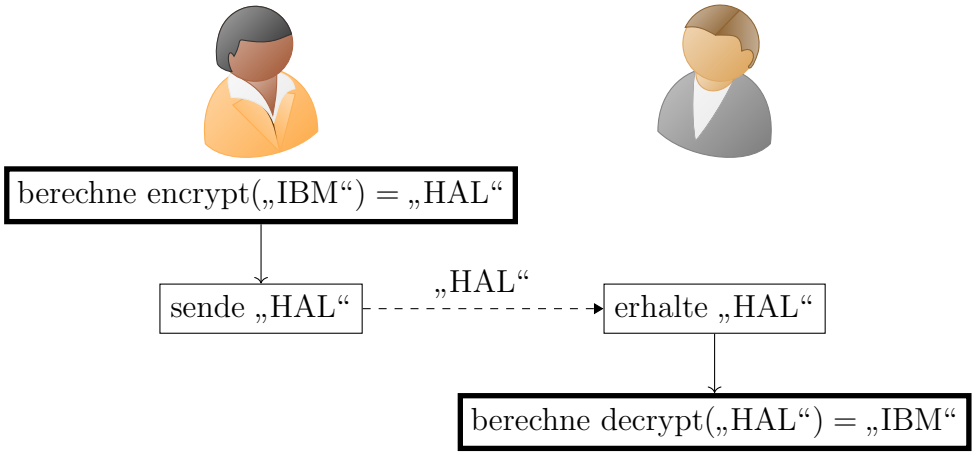
encrypt = „ersetze jeden Buchstaben durch den im Alphabet davor stehenden“

Sie kennen daher auch die Entschlüsselungsfunktion, da die Verschlüsselung so einfach rückgängig zu machen ist:

decrypt = „ersetze jeden Buchstaben durch den im Alphabet danach stehenden“

Alice möchte Bob die Marke ihres Computers mitteilen, aber das soll geheim bleiben. Das Übermittlungsprotokoll sieht dann wie in Abbildung 1 aus.

Abbildung 1: Alphabetverschiebung um 1



1.2. **Symmetrische Verfahren.** Bei der **symmetrischen Kryptographie** verwenden Sender und Empfänger einen gemeinsamen Schlüssel sowohl für die Verschlüsselung als auch für die Entschlüsselung. Ein klassisches Beispiel ist die Verschiebe-Chiffre, bei der Buchstaben eines Textes im Alphabet um eine feste Anzahl von Positionen verschoben werden. Der Schlüssel K ist bei diesem Verfahren die Anzahl der Verschiebungen und

$\text{encrypt}_K =$ „ersetze jeden Buchstaben durch den im Alphabet K Positionen davor“
$\text{decrypt}_K =$ „ersetze jeden Buchstaben durch den im Alphabet K Positionen danach“

In Beispiel 1.1 verwendeten Alice und Bob den Schlüssel $K = 1$, auf den sie sich vorab einigten. Für andere K ist der Ablauf des Verfahrens der gleiche, nur mit Verschiebung um K Positionen.

Bei einem symmetrischen Verfahren sieht das Protokoll allgemein wie in Abbildung 2 aus. Hier teilt Alice die Nachricht m mit Bob, indem sie die verschlüsselte Nachricht m' übermittelt. Es gibt nur einen Schlüssel K, und das Verfahren beruht auf:

$$\text{decrypt}_K\big(\text{encrypt}_K(m)\big) = m \quad \text{für jede Nachricht } m$$

Aufgrund der für Sprache typischen Buchstabenhäufigkeit ist der Schlüssel K der Verschiebe-Chiffre bei längeren Texten einfach aus dem verschlüsselten Text m' (der bei der Übertragung abgefangen werden kann) zu erraten. Moderne symmetrische Algorithmen, wie der Advanced Encryption Standard (AES), bieten deutlich höhere Sicherheit und werden häufig zur sicheren Datenübertragung und Datenspeicherung verwendet. Die Herausforderung bei symmetrischen Algorithmen besteht dann darin, den gemeinsamen Schlüssel sicher zwischen den Kommunikationsparteien zu verteilen, also den ersten Schritt abzusichern, wenn K und m' über den gleichen Weg übermittelt werden (Abbildung 3).

Abbildung 2: Ablauf bei symmetrischen Verfahren

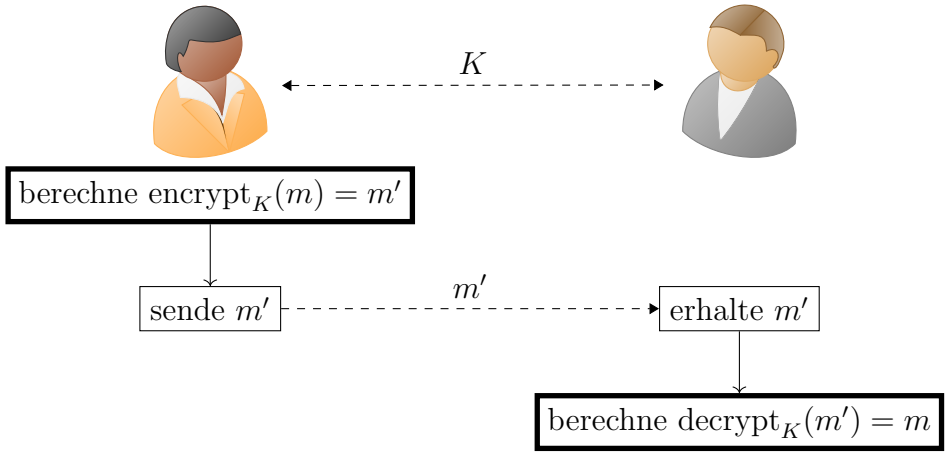
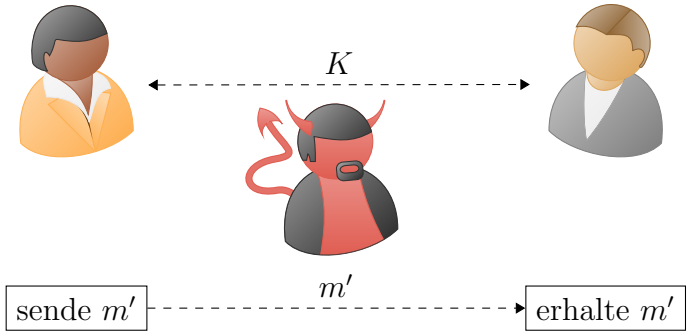


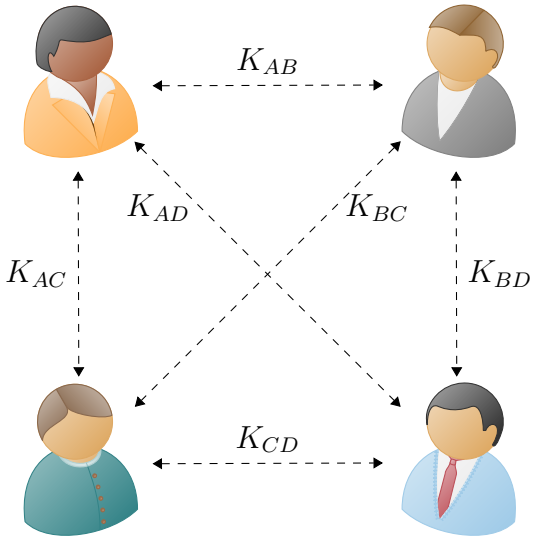
Abbildung 3: Abhörgefahr bei symmetrischen Verfahren



Neben der Schwierigkeit des sicheren Schlüsselaustauschs gibt es bei symmetrischen Verfahren eine weitere Herausforderung: Möchten mehrere Personen miteinander (paarweise) verschlüsselt kommunizieren, so müssen sie für jede Paarung einen neuen Schlüssel vereinbaren. Dies wird sehr schnell komplex und unhandlich (Abbildung 4): Bei N Personen muss jede Person einen Schlüssel pro Kommunikationspartner verwalten, also $N - 1$; insgesamt sind $N \cdot (N - 1)/2$ symmetrische Schlüssel im Einsatz.

1.3. **Asymmetrische Verfahren.** Die **asymmetrische Kryptographie**, auch als Public-Key-Kryptographie bekannt, revolutionierte die Kryptographie in den 1970er Jahren. Im Gegensatz zur symmetrischen Kryptographie verwenden die Parteien hier ein Schlüsselpaar: einen öffentlichen Schlüssel K, der frei verteilt werden kann, und einen privaten Schlüssel K', der geheim gehalten wird. Nachrichten, die mit dem

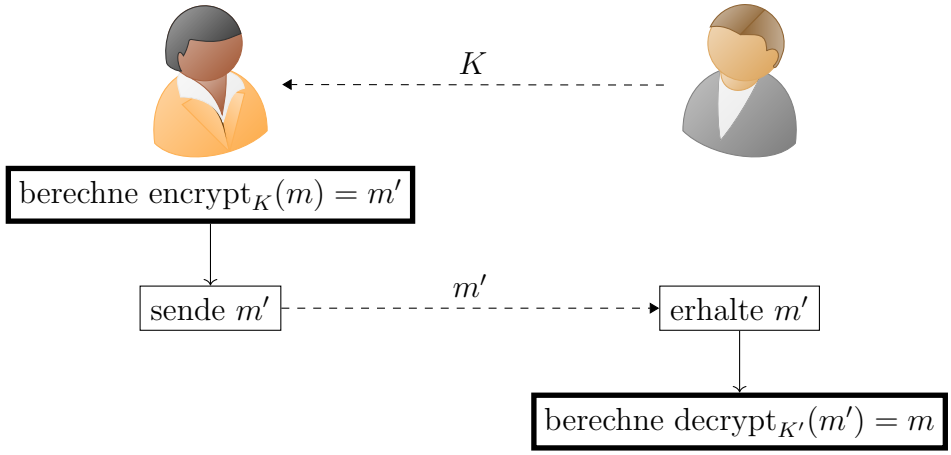
Abbildung 4: Komplexität bei symmetrischen Verfahren



öffentlichen Schlüssel verschlüsselt sind, können nur mit dem entsprechenden privaten Schlüssel entschlüsselt werden. Mit anderen Worten beruht das Verfahren auf

$$\text{decrypt}_{K'}\big(\text{encrypt}_K(m)\big) = m \quad \text{für jede Nachricht } m.$$

Abbildung 5: Ablauf bei asymmetrischen Verfahren (Verschlüsselung)



Dies löst das Abhörproblem völlig, da nur K abgehört werden kann, zur Entschlüsselung aber K' benötigt wird. Etwas genauer: Das Abhörproblem ist gelöst, falls es (zumindest in der Praxis) unmöglich ist, den geheimen Schlüssel K' aus dem öffentlichen Schlüssel K zu bestimmen.

Tatsächlich ist so auch das Komplexitätsproblem gelöst: Abbildung 6 hat zwar mehr Verbindungen als Abbildung 4, aber jede Person muss nur ein Schlüsselpaar erzeugen und den eigenen öffentlichen Schlüssel einmal für alle zur Verfügung stellen. Umgekehrt kann jede Person den öffentlichen Schlüssel anderer Personen jederzeit erhalten, ohne dass eine gesicherte Übertragung nötig wäre. Jede Person muss also nur 2 Schlüssel permanent speichern und 1 Schlüssel veröffentlichen; insgesamt sind $2N$ Schlüssel im Einsatz.

Abbildung 6: Komplexität bei asymmetrischen Verfahren

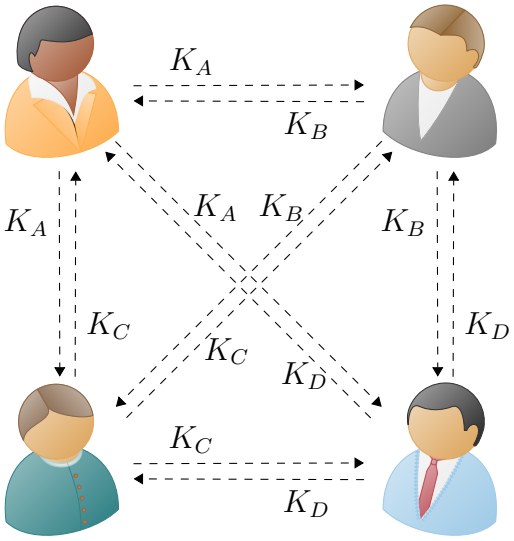
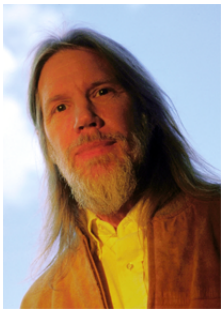
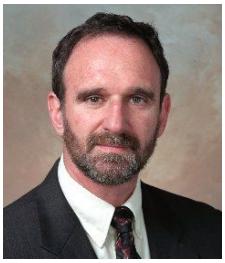


Abbildung 7: Whitfield Diffie, Martin Hellman.

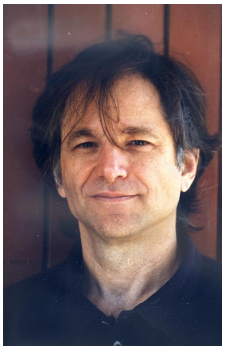
(Wikimedia Commons, published under Licenses CC BY 2.0 and CC BY-SA 3.0. No changes were made.)

Die Geburtsstunde der Public-Key-Kryptographie ist ein Artikel von W. Diffie und M. E. Hellman aus dem Jahr 1976. Dort beschrieben sie ihre revolutionäre Idee eines

asymmetrischen Verschlüsselungssystems. Die entscheidende Frage, ob ein solches Verfahren tatsächlich realisierbar ist, wurde kurze Zeit später in einem bahnbrechenden Artikel von R. Rivest, A. Shamir und L. Adleman aus dem Jahr 1978 beantwortet. Das dort vorgeschlagene RSA-Verfahren ist bis heute für technische Anwendungen im Internet-Zeitalter grundlegend. Es basiert auf der mathematischen Schwierigkeit, große Zahlen in ihre Primfaktoren zu zerlegen. Wir beschäftigen uns damit, sobald wir die dafür notwendige Mathematik behandelt haben: modulare Arithmetik, insbesondere der Euklidische Algorithmus (Satz 2.18), die Eulersche φ-Funktion (Definition 3.52 und Satz 3.55) und der kleine Satz von Fermat (Satz 3.60).

Abbildung 8: Ronald Rivest, Adi Shamir, Leonard Adleman.





(Wikimedia Commons, published under Licenses CC BY-SA 4.0 and CC BY-SA 3.0. No changes were made.)

1.4. **Kombinierte Verfahren.** Asymmetrische Verfahren werden häufig nicht direkt zur Verschlüsselung einer Nachricht eingesetzt, sondern zum sicheren Austausch eines gemeinsamen Geheimnisses, das dann wiederum als Schlüssel für ein symmetrisches Verfahren eingesetzt werden kann, mit dem die eigentliche Nachricht verschlüsselt wird. Besonders deutlich wird das beim Diffie-Hellman-Verfahren (D-H-*Schlüssel*-austausch), denn hier wird ein gemeinsames Geheimnis ausgetauscht, das vom Verfahren erzeugt wird und nicht gewählt werden kann. Es wird später (im ElGamal-Verfahren) als symmetrischer Schlüssel genutzt.

Wir werden uns meist auf jeweils einen Aspekt fokussieren, also z. B. (nur) Nachrichten mit RSA verschlüsseln.

2. Ganze Zahlen und Teilbarkeit

Bevor wir uns mit kryptographischen Verfahren beschäftigen können, benötigen wir zunächst ein paar mathematische Grundlagen. Wir bezeichnen mit $\mathbb{N} = \{1, 2, 3, \ldots\}$ die natürlichen Zahlen und verwenden die Notation $\mathbb{N}_0 = \mathbb{N} \cup \{0\}$, wenn die 0 hinzugenommen werden soll. Damit wir neben der Addition auch subtrahieren können,

benötigen wir die Menge der ganzen Zahlen

$$\mathbb{Z} = \{\ldots, -3, -2, -1, 0, 1, 2, 3, \ldots\}$$

mit den üblichen Rechenregeln für die Addition und Multiplikation.

2.1. **Teilbarkeit in den ganzen Zahlen.** In diesem Abschnitt werden die grundlegenden Begriffe und Resultate zur Teilbarkeit im Zahlbereich $\mathbb{Z}$ der ganzen Zahlen behandelt. Natürlich wisst ihr eigentlich, was es bedeutet, dass eine Zahl eine andere teilt. Um mit so einem Begriff mathematisch arbeiten zu können, ist jedoch eine präzise Vereinbarung (eine Definition) nötig.

Definition 2.1: Teilbarkeit in $\mathbb{Z}$

Seien a, b ganze Zahlen. Wir sagen, a *teilt* b (oder a *ist ein Teiler von* b), wenn es eine ganze Zahl n gibt, so dass $b = an$. Notation: $a \mid b$.
Ist a kein Teiler von b, so schreiben wir $a \nmid b$.

Beispiel 2.2

Die Menge aller Teiler der Zahl 36 ist:

$$\{\pm 1, \pm 2, \pm 3, \pm 4, \pm 6, \pm 9, \pm 12, \pm 18, \pm 36\}.$$

Es gibt viele schöne Beobachtungen und Regeln rund um den Begriff der Teilbarkeit. Für uns wird später die folgende Regel wichtig sein.

Bemerkung 2.3. *Seien a, b, d ganze Zahlen. Wenn $d \mid a$ und $d \mid b$ gilt, dann gilt auch $d \mid sa + tb$ für alle ganzen Zahlen s und t.*

Begründung. *Der entscheidende Punkt ist, dass wir d aus a und b ausklammern können. Nach Voraussetzung ist $d \mid a$ und $d \mid b$, d.h. es gibt ganze Zahlen n, m, so dass $a = dn$ und $b = dm$ ist. Mit den üblichen Rechenregeln für ganze Zahlen erhalten wir*

$$sa + tb = sdn + tdm = d(sn + tm)$$

und das bedeutet, dass $d \mid sa + tb$.

2.2. **Primzahlen.** Wir kommen jetzt zu den fundamentalen Bausteinen der natürlichen und der ganzen Zahlen, den Primzahlen.

Definition 2.4: Primzahlen

Eine natürliche Zahl p heißt *Primzahl*, wenn sie genau zwei natürliche Zahlen als Teiler hat, nämlich 1 und p.

Bemerkung 2.5.

(i) *1 ist keine Primzahl. (Denn die 1 hat nur einen Teiler in* $\mathbb{N}$*.)*
(ii) *2 ist die einzige gerade Primzahl.*
(iii) *Die ersten Primzahlen sind*

$$2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 53, 59, 61, 67, 71, 73, 79, 83, \ldots$$

Satz 2.6: Existenz eines Primteilers

Jede natürliche Zahl $n > 1$ besitzt einen *Primteiler*, d.h. eine Primzahl p mit $p \mid n$.

Begründung. *Da $n > 1$ ist, besitzt n eine natürliche Zahl > 1, die n teilt, nämlich n selbst. Unter allen natürlichen Zahlen > 1, die n teilen, sei p die kleinste. Dann ist p ein Primteiler von n, denn wäre p keine Primzahl, dann gäbe es einen Teiler b von p (und damit auch von n) mit $1 < b < p$, was aber nicht sein kann, da p die kleinste solche Zahl war.*

Dieser Satz garantiert die Existenz eines Primteilers, die Begründung liefert aber kein Verfahren, einen Primteiler zu berechnen. Tatsächlich ist das für große Zahlen sehr schwierig, und für sehr große Zahlen selbst mit den schnellsten Computern nicht immer möglich. Darauf beruht die Sicherheit vieler wichtiger Verschlüsselungsverfahren wie dem RSA-Verfahren, das wir noch ausführlich behandeln werden.

Bemerkung 2.7. *Die größte bekannte Primzahl (Stand März 2026) ist*

$$2^{136279841} - 1,$$

eine Zahl mit 41024320 Dezimalstellen. Gefunden wurde diese Primzahl am 12.10.2024 im Rahmen der „Great Internet Mersenne Primes Search“, abgekürzt GIMPS [1]*. In diesem Projekt stellen Freiwillige Rechenzeit auf ihren Computern zur Verfügung, um mit Hilfe von frei verfügbarer Software Zahlen der Form $2^p - 1$ zu testen, ob sie prim sind.*
Eine Primzahl der Form $2^p - 1$ heißt Mersenne-Primzahl [2]*. Ist $2^p - 1$ eine Mersenne-Primzahl, dann muss der Exponent p auch eine Primzahl sein, denn*

$$2^{ab} - 1 = (2^{a(b-1)} + 2^{a(b-2)} + \ldots + 2^{a \cdot 2} + 2^a + 1) \cdot (2^a - 1)$$

ist eine Faktorisierung in kleinere Zahlen, wenn a und b beide nicht 1 sind. Es ist aber nicht umgekehrt jede Zahl $2^p - 1$ mit p Primzahl, auch eine Primzahl, z.B. ist $2^{11} - 1 = 2047 = 23 \cdot 89$.
Der Grund, warum Zahlen der Form $2^p - 1$ gut geeignet sind, um große Primzahlen zu finden, ist, dass es einen effizienten Test auf Primalität für diese Zahlen gibt, den Lucas-Lehmer-Test[3]*.*

Aus der Definition 2.4 ist zunächst gar nicht klar, wie viele Primzahlen es überhaupt gibt. Zumindest hört die Liste von Primzahlen nicht irgendwann auf, wie das folgende berühmte Resultat zeigt, das Euklid[4] schon im dritten Jahrhundert vor Christus bewiesen hat.

Satz 2.8

Es gibt unendlich viele Primzahlen.

Begründung. *Wir zeigen, dass es zu jeder endlichen Menge von Primzahlen stets eine weitere Primzahl gibt. Für endlich viele Primzahlen $\{p_1, \ldots, p_r\}$ betrachten wir die natürliche Zahl*

$$n = 1 + p_1 \cdot p_2 \cdot \ldots \cdot p_r.$$

Nach Satz 2.6 besitzt n einen Primteiler p. Aber p kann nicht einer der Zahlen $p_1, \ldots, p_r$ sein, denn sonst würde p die Zahl $n - p_1 \cdot p_2 \cdot \ldots \cdot p_r = 1$ teilen, was nicht sein kann. Damit haben wir eine neue Primzahl gefunden und gezeigt, dass die Menge der Primzahlen nicht endlich sein kann.

Wir wollen die Konstruktion aus der eben gegebenen Begründung konkret anwenden. Wir starten mit der Primzahl 2 und betrachten $n = 1+2 = 3$ und erhalten 3 als weitere Primzahl. Jetzt betrachten wir die Menge $\{2, 3\}$ und bilden $1 + 2 \cdot 3 = 7$, wieder eine Primzahl. Für $\{2, 3, 7\}$ betrachten wir $1 + 2 \cdot 3 \cdot 7 = 43$, wieder eine Primzahl. Für $\{2, 3, 7, 43\}$ betrachten wir $1 + 2 \cdot 3 \cdot 7 \cdot 43 = 1807$. Dies ist keine Primzahl, denn $1807 = 13 \cdot 139$, aber wichtig ist ja nur, dass wir unter den Primteilern von 1807 eine neue Primzahl finden, hier zum Beispiel 13 (oder 139, das ist auch eine Primzahl).

Das folgende Resultat macht deutlich, in welchem Sinne die Primzahlen die Bausteine für alle ganzen Zahlen sind: Jede natürliche Zahl lässt sich als ein Produkt von Primzahlen schreiben.

Satz 2.9: Primfaktorzerlegung

Sei n eine natürliche Zahl, $n \geq 1$. Dann existieren eine natürliche Zahl s und Primzahlen $p_1, \ldots, p_s$ (nicht notwendig verschieden), so dass

$$n = p_1 p_2 \ldots p_s.$$

Diese Zerlegung ist bis auf die Reihenfolge der Faktoren eindeutig.

---

[1] `https://en.wikipedia.org/wiki/Great_Internet_Mersenne_Prime_Search`

[2] Benannt nach Marin Mersenne (1588-1648).

[3] Benannt nach Édouard Lucas (1842-1891) und Derrick Henry Lehmer (1905-1991). Weitere Details: `https://de.wikipedia.org/wiki/Lucas-Lehmer-Test`.

[4] Euklid von Alexandria (ca. 350 v. Chr. – 270 v. Chr.)

Begründung. *Nach Satz 2.6 besitzt n einen Primteiler p_1. Wenn $n = p_1$ eine Primzahl ist, dann ist $n = p_1$ bereits eine Primfaktorzerlegung. Wenn n keine Primzahl ist, dann haben wir die Zerlegung $n = p_1 \cdot \frac{n}{p_1}$, wobei $1 < \frac{n}{p_1} < n$.*
Jetzt wenden wir dasselbe Verfahren auf die kleinere Zahl $\frac{n}{p_1}$ an. Diese hat einen Primteiler p_2. Wenn $\frac{n}{p_1} = p_2$ eine Primzahl ist, dann ist $n = p_1 \cdot \frac{n}{p_1} = p_1 \cdot p_2$ eine Zerlegung in Primfaktoren. Sonst erhalten wir eine Zerlegung $\frac{n}{p_1} = p_2 \cdot \frac{n}{p_1 p_2}$ mit einer kleineren Zahl $\frac{n}{p_1 p_2} < \frac{n}{p_1}$.
Da die auftretenden Zahlen immer kleiner werden, erhalten wir nach endlich vielen Schritten eine Zerlegung von n als Produkt von Primzahlen.
Die Eindeutigkeit ist etwas mühsamer und wird daher hier nicht extra begründet.

Beispiel 2.10

Die Primfaktorzerlegung der Zahl 540 ist

$$540 = 2 \cdot 2 \cdot 3 \cdot 3 \cdot 3 \cdot 5 = 2^2 \cdot 3^3 \cdot 5.$$

Die Primfaktorzerlegung in dem Beispiel konnten wir nur finden, weil wir einige der Teiler von 540 direkt erkennen konnten. Aber allgemein ist es sehr schwierig, die Primfaktorzerlegung einer Zahl zu finden. Das ist allerdings ganz wichtig für einige Verschlüsselungsverfahren (siehe Satz 3.68 für den Fall des RSA-Verfahrens).

Aufgabe 2.11

Findet die Primfaktorzerlegung der Zahl 630.

2.3. **Division mit Rest und größte gemeinsame Teiler.** Für die Kryptographie relevant ist der Begriff des größten gemeinsamen Teilers von ganzen Zahlen. Den werden wir in diesem Abschnitt einführen und dann einen Algorithmus kennenlernen, mit dem der größte gemeinsame Teiler schnell berechnet werden kann.

Der Algorithmus beruht auf dem folgenden grundlegenden Verfahren der Division mit Rest, wie man es schon aus der Schule kennt[5].

Satz 2.12: Division mit Rest

Seien a, b ganze Zahlen, wobei $b \neq 0$. Dann gibt es eindeutig bestimmte ganze Zahlen q, r, so dass

$$a = qb + r \quad \text{und} \quad 0 \leq r < |b|.$$

[5] ...jedenfalls für natürliche Zahlen. Hieraus folgt das Verfahren für alle ganzen Zahlen.

Beispiel 2.13

Die Zahlen q und r lassen sich mit schriftlicher Division berechnen (oder mit dem Taschenrechner). Zum Beispiel ergibt sich für $a = 1111$ und $b = 78$ als Division mit Rest

$$1111 = 14 \cdot 78 + 19$$

also $q = 14$ und $r = 19$.

Ein auch aus der Schule bekannter Begriff ist der des größten gemeinsamen Teilers, zumindest in den natürlichen Zahlen.

Definition 2.14: Größter gemeinsamer Teiler

Seien a, b ganze Zahlen.

(i) Eine ganze Zahl d heißt *gemeinsamer Teiler* von a und b, wenn $d \mid a$ und $d \mid b$.

(ii) Der *größte gemeinsame Teiler* von a und b ist die größte natürliche Zahl d, die a und b teilt. Wir schreiben dann $\mathrm{ggT}(a, b) = d$.

(iii) Die Zahlen a und b heißen *teilerfremd*, wenn 1 der größte gemeinsame Teiler von a und b ist.

Falls man von zwei Zahlen die Primfaktorzerlegungen kennt, so lässt sich der größte gemeinsame Teiler wie folgt bestimmen. Seien $a = p_1^{e_1} \cdot \ldots \cdot p_s^{e_s}$ und $b = p_1^{f_1} \cdot \ldots \cdot p_s^{f_s}$, wobei wir als Exponenten e_i, f_i auch 0 erlauben, damit in beiden Produkten dieselben Primzahlen stehen. Die (positiven) Teiler von a sind alle Zahlen der Form $p_1^{e_1'} \cdot \ldots \cdot p_s^{e_s'}$ mit $0 \leq e_i' \leq e_i$, und entsprechend für die Teiler von b. Dann ist

$$\mathrm{ggT}(a, b) = p_1^{\min(e_1, f_1)} \cdot \ldots \cdot p_s^{\min(e_s, f_s)}.$$

Für jeden Primteiler nimmt man also jeweils den kleineren der beiden Exponenten. Auf diese Weise wird manchmal der größte gemeinsame Teiler in der Schule berechnet.

Beispiel 2.15

Für $a = 32670 = 2 \cdot 3^3 \cdot 5 \cdot 11^2$ und $b = 5460 = 2^2 \cdot 3 \cdot 5 \cdot 7 \cdot 13$ ist

$$\mathrm{ggT}(a, b) = 2 \cdot 3 \cdot 5 = 30.$$

2.4. **Der Euklidische Algorithmus.** Da die Primfaktorzerlegung einer Zahl nicht so einfach zu bestimmen ist (und für sehr große Zahlen auch mit den schnellsten Computern nicht), bleibt die grundsätzliche Frage, wie man größte gemeinsame Teiler konkret berechnen kann. Das geht mit dem **Euklidischen Algorithmus**. Grundlage für diesen ist das folgende Resultat über die größten gemeinsamen Teiler von Zahlen, die bei der Division mit Rest auftauchen.

Satz 2.16

Seien a, b ganze Zahlen. Für alle ganzen Zahlen q, r mit $a = qb + r$ gilt

$$\mathrm{ggT}(a, b) = \mathrm{ggT}(b, r).$$

Begründung. *Wir zeigen, dass die gemeinsamen Teiler von a und b genau dieselben sind wie die gemeinsamen Teiler von b und r. Dann sind insbesondere auch die größten gemeinsamen Teiler gleich.*
Ist d ein gemeinsamer Teiler von a und b, dann teilt d (nach Bemerkung 2.3) auch $a - qb = r$, ist also ein gemeinsamer Teiler von b und r.
Ist d ein gemeinsamer Teiler von b und r, dann teilt d (nach Bemerkung 2.3) auch $qb + r = a$, ist also gemeinsamer Teiler von a und b.

Beispiel 2.17

In Beispiel 2.13 haben wir folgende Division mit Rest betrachtet:

$$1111 = 14 \cdot 78 + 19.$$

Mit Satz 2.16 folgt $\mathrm{ggT}(1111, 78) = \mathrm{ggT}(78, 19)$. Diesen Ansatz können wir wiederholen, und wir betrachten Division mit Rest von 78 durch 19, also $78 = 4 \cdot 19 + 2$. Satz 2.16 liefert $\mathrm{ggT}(78, 19) = \mathrm{ggT}(19, 2)$. Eine weitere Division mit Rest $19 = 9 \cdot 2 + 1$ ergibt dann $\mathrm{ggT}(19, 2) = \mathrm{ggT}(2, 1) = 1$. Insgesamt haben wir also jetzt durch mehrfache Division mit Rest herausgefunden, dass $\mathrm{ggT}(1111, 78) = 1$ ist.
Dies kann auch mit Hilfe der Primfaktorzerlegungen überprüft werden: Da 78 offensichtliche kleine Primfaktoren hat, lässt sich die Primfaktorzerlegung $78 = 2 \cdot 3 \cdot 13$ leicht berechnen. 1111 ist weder durch 2, 3 oder 13 teilbar, und folglich haben 1111 und 78 keinen gemeinsamen Primfaktor, also sind 1 (und -1) die einzigen gemeinsamen Teiler.

Das Vorgehen in diesem Beispiel führt uns zu dem angekündigten Verfahren zur Berechnung des größten gemeinsamen Teilers.

Satz 2.18: Euklidischer Algorithmus [6]

Seien a, b ganze Zahlen und $b \neq 0$. Wir setzen $r_0 = a$, $r_1 = |b|$ und betrachten die folgende Kette von Divisionen mit Rest:

$$\begin{aligned} a &= q_1 b + r_2 \quad \text{mit} \quad 0 < r_2 < r_1 \\ b &= q_2 r_2 + r_3 \quad \text{mit} \quad 0 < r_3 < r_2 \\ r_2 &= q_3 r_3 + r_4 \quad \text{mit} \quad 0 < r_4 < r_3 \\ &\vdots \\ r_{m-2} &= q_{m-1} r_{m-1} + r_m \quad \text{mit} \quad 0 < r_m < r_{m-1} \\ r_{m-1} &= q_m r_m + 0 \end{aligned}$$

Dann gilt:

(i) Das obige Verfahren bricht nach endlich vielen Schritten ab.
(ii) Es ist $\mathrm{ggT}(a, b) = r_m$, der letzte von 0 verschiedene Rest.

Begründung.

(i) Falls $b \mid a$, so bricht das Verfahren im ersten Schritt ab. Sei also jetzt $b \nmid a$. Dann ist $r_2 \neq 0$ und die Reste bilden eine echt absteigende Folge $r_2 > r_3 > \ldots$ von Zahlen aus $\mathbb{N}_0$. Nach endlich vielen Schritten muss man mit den Resten beim kleinsten Element 0 angekommen sein.

(ii) Nach Satz 2.16 erhalten wir aus den Gleichungen im Euklidischen Algorithmus die folgende Kette von Gleichungen:

$$\begin{aligned} \mathrm{ggT}(a, b) = \mathrm{ggT}(b, r_2) &= \mathrm{ggT}(r_2, r_3) = \mathrm{ggT}(r_3, r_4) \\ &= \ldots = \mathrm{ggT}(r_{m-1}, r_m) = r_m \end{aligned}$$

wobei die letzte Gleichung gilt, da $r_{m-1} = q_m r_m$. und Teil (b) ist gezeigt. □

Folgerung 2.19: Bézout-Koeffizienten

Seien a, b ganze Zahlen, wobei $a \neq 0$ oder $b \neq 0$. Dann gibt es ganze Zahlen s, t, so dass

$$\mathrm{ggT}(a, b) = sa + tb.$$

Solche Zahlen s, t heißen *Bézout-Koeffizienten* [7].

Bemerkung 2.20. *Die Bézout-Koeffizienten s, t sind nie eindeutig: zum Beispiel, für jede ganze Zahl k gilt*

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

[6] Benannt nach Euklid von Alexandria (ca. 350 v.Chr.-270 v.Chr.).
[7] Benannt nach Étienne Bézout (1730-1783).

Aus einem Paar s, t von Bézout-Koeffizienten erhalten wir also unendlich viele Paare $s + kb, t - ka$ von Bézout-Koeffizienten.

Bézout-Koeffizienten lassen sich berechnen, indem man die Gleichungen im Euklidischen Algorithmus von unten nach oben ineinander einsetzt, wie folgendes Beispiel veranschaulicht.

Beispiel 2.21

Wir wollen den größten gemeinsamen Teiler der Zahlen $a = 14351$ und $b = 12317$ berechnen. Mit dem Euklidischen Algorithmus erhalten wir folgende Divisionen mit Rest:

$$14351 = 1 \cdot 12317 + 2034$$
$$12317 = 6 \cdot 2034 + 113$$
$$2034 = 18 \cdot 113 + 0.$$

Der ggT ist nach Satz 2.18 der letzte von 0 verschiedene Rest, also

$$\mathrm{ggT}(14351, 12317) = 113.$$[8]

Wir berechnen nun noch Bézout-Koeffizienten (vgl. Folgerung 2.19). In diesem Fall lautet die vorletzte Gleichung nach Umstellen

$$113 = 12317 - 6 \cdot 2034.$$

Den Faktor 2034 ersetzen wir durch die erste Gleichung und erhalten

$$\begin{aligned} 113 &= 12317 - 6 \cdot 2034 = 12317 - 6 \cdot (14351 - 1 \cdot 12317) \\ &= (-6) \cdot 14351 + 7 \cdot 12317 = \underbrace{(-6)}_{s} \cdot a + \underbrace{7}_{t} \cdot b. \end{aligned}$$

Damit sind -6 und 7 mögliche Bézout-Koeffizienten.

Aufgabe 2.22

Berechnet mit dem Euklidischen Algorithmus den größten gemeinsamen Teiler von $a = 1737$ und $b = 117$, sowie Bézout-Koeffizienten $s, t \in \mathbb{Z}$, so dass $\mathrm{ggT}(a, b) = sa + tb$.

Als Anwendung der Bézout-Koeffizienten können wir etwas über die Lösbarkeit ganzzahliger Gleichungen sagen.

[8] Sicherheitshalber kann man leicht überprüfen, dass 113 tatsächlich ein gemeinsamer Teiler von a und b ist: $a = 14351 = 127 \cdot 113$ und $b = 109 \cdot 113$ sind beide Vielfache von 113. Darüberhinaus, da 109 eine Primzahl und offenbar kein Teiler von 127 ist, haben a und b keine weiteren gemeinsamen Primfaktoren, also 113 ist in der Tat $\mathrm{ggT}(a, b)$.

Folgerung 2.23

Die Gleichung $ax + by = c$ mit ganzen Zahlen a, b, c hat genau dann ganzzahlige Lösungen x, y, wenn $\mathrm{ggT}(a, b)$ ein Teiler von c ist. Insbesondere ist $\mathrm{ggT}(a, b)$ die kleinste positive Zahl von der Form $ax + by$ mit ganzen Zahlen x, y.

Begründung. *Nehmen wir zuerst an, dass $c = ax + by$ mit ganzen Zahlen x, y gilt. Nach Bemerkung 2.3 wissen wir, dass jeder Teiler d von a und b auch ein Teiler von $c = ax + by$ ist. Insbesondere ist $\mathrm{ggT}(a, b)$ ein Teiler von c.*
Sei umgekehrt $\mathrm{ggT}(a, b)$ ein Teiler von c, d.h. $c = k \cdot \mathrm{ggT}(a, b)$ für eine ganze Zahl k. Nach Folgerung 2.19 können wir $\mathrm{ggT}(a, b) = sa + tb$ mit geeigneten Bézout-Koeffizienten schreiben. Zusammen erhalten wir

$$c = k \cdot \mathrm{ggT}(a, b) = \underbrace{(ks)}_{x} a + \underbrace{(kt)}_{y} b,$$

und damit ganzzahlige Lösungen der Gleichung.

Beispiel 2.24

Die Begründung in der vorigen Bemerkung besagt auch, dass sich Lösungen von Gleichungen der Form $ax + by = c$ mit dem euklidischen Algorithmus berechnen lassen. Als konkretes Beispiel betrachten wir die Gleichung

$$1309\,x + 1001\,y = 231.$$

Mit dem euklidischen Algorithmus berechnen wir den ggT von $a = 1309$ und $b = 1001$.

$$\begin{aligned} 1309 &= 1 \cdot 1001 + 308 \\ 1001 &= 3 \cdot 308 + 77 \\ 308 &= 4 \cdot 77 + 0 \end{aligned}$$

Also ist $\mathrm{ggT}(a, b) = 77$ und Bézout-Koeffizienten berechnen wir durch Rückwärtseinsetzen:

$$\mathrm{ggT}(1309, 1001) = 77 = 1001 - 3 \cdot (1309 - 1001) = \underbrace{-3}_{s} \cdot 1309 + \underbrace{4}_{t} \cdot 1001.$$

Es ist $\mathrm{ggT}(a, b) = 77 \mid 3 \cdot 77 = 231 = c$, also gibt es nach Folgerung 2.23 ganzzahlige Lösungen der Gleichung. Eine Lösung erhalten wir mit $x = \frac{c}{\mathrm{ggT}(a,b)} s = 3 \cdot (-3) = -9$ und $y = \frac{c}{\mathrm{ggT}(a,b)} t = 3 \cdot 4 = 12$.

Aufgabe 2.25

(i) Für welche der folgenden Zahlen c hat die Gleichhung $1338x+4668y = c$ ganzzahligen Lösungen?

(a) $c = 22$, (b) $c = 24$ (c) $c = 27$.

(ii) Bestimmt eine Lösung für x mit $c = 96$. Gibt es weitere Lösungen?

3. Modulare Arithmetik und Kryptographie

Die Grundidee der modularen Arithmetik ist es, zwei ganze Zahlen als „gleich“ zu behandeln, wenn sie denselben Rest bei Division durch eine gegebene Zahl n lassen. Dies führt zu neuen Zahlbereichen (mit endlich vielen Elementen), welche einige überraschende Eigenschaften haben.

3.1. Rechnen mit Kongruenzen.

Wir schauen uns die Division mit Rest noch einmal an und fassen alle Zahlen zusammen, die bei Division durch eine natürliche Zahl n denselben Rest lassen.

Definition 3.1: Kongruenz modulo n

Seien n eine (positive) natürliche Zahl und a, b zwei ganze Zahlen. Wir sagen *a ist kongruent zu b modulo n*, wenn $n \mid a - b$, d.h. die Differenz $a - b$ ist ein Vielfaches von n. In diesem Fall schreiben wir

$$a \equiv b \bmod n.$$

Beispiel 3.2

(i) Sei $n = 2$. Dann ist $a \equiv b \bmod 2$ genau dann, wenn $2 \mid a - b$. Dies bedeutet, dass a und b beide gerade oder beide ungerade sind.

(ii) Sei $n = 3$. Dann ist $a \equiv 0 \bmod 3$ genau dann, wenn a durch 3 teilbar ist. Es ist $a \equiv 1 \bmod 3$ genau dann, wenn $a - 1 = 3k$ ein Vielfaches von 3 ist (hier ist k eine ganze Zahl). Dies bedeutet $a = 1 + 3k$, und wenn wir k über alle möglichen ganzen Zahlen laufen lassen, bekommen wir $a = ..., -8, -5, -2, 1, 4, 7, 10, ...$
Analog kann man zeigen, dass $a \equiv 2 \bmod 3$ genau dann gilt, wenn $a = \ldots, -7, -4, -1, 2, 5, 8, 11, \ldots$

Bemerkung 3.3. *Die Kongruenz $a \equiv b \bmod n$ ist gleichbedeutend damit, dass a und b bei Division mit Rest durch n denselben Rest lassen.*

Begründung. *Wir schreiben $a = q_1 n + r_1$ mit $0 \leq r_1 < n$ und $b = q_2 n + r_2$ mit $0 \leq r_2 < n$. Dann ist $a - b = (q_1 - q_2)n + (r_1 - r_2)$. Die rechte Seite ist genau dann durch n teilbar, wenn $n \mid r_1 - r_2$. Da die Reste kleiner oder gleich n sind, gilt $-n6 < r_1 - r_2 < n$. In diesem Bereich ist aber nur eine Zahl durch n teilbar, nämlich die Null. Daher gilt $n \mid r_1 - r_2$ genau dann, wenn $r_1 - r_2 = 0$, also $r_1 = r_2$, wie behauptet.*

Kongruenz modulo einer festen Zahl n hat folgende einfache Eigenschaften[9]:

- Jede Zahl a erfüllt $a \equiv a \bmod n$, da $a - a = 0 = 0n$ ein Vielfaches von n ist.
- Falls $a \equiv b \bmod n$, so ist auch $b \equiv a \bmod n$. Warum?
- Falls $a \equiv b \bmod n$ und $b \equiv c \bmod n$, so gilt auch $a \equiv c \bmod n$. Warum?

Wir wollen nun die üblichen Rechenoperationen mit ganzen Zahlen nur „bis auf Kongruenz“ modulo einer festen Zahl n betrachten. Das heißt, wenn $a \equiv b \bmod n$ ist, möchten wir gerne a und b als „gleich“ betrachten. Insbesondere sollte es bei den Operationen egal sein, ob wir mit a oder mit b rechnen. Zum Beispiel, wenn wir die Addition mit einer dritten ganzen Zahl c betrachten, sollen $a+c$ und $b+c$ auch „gleich“, also kongruent modulo n sein. Dies ist zum Glück der Fall (sonst wäre die modulare Arithmetik wenig sinnvoll!), denn folgendes gilt:

Satz 3.4: Rechnen modulo n

Seien n, a, b, c, d ganze Zahlen, wobei $n > 0$. Falls $a \equiv b \bmod n$ und $c \equiv d \bmod n$, so gelten

$$a + c \equiv b + d \bmod n \qquad \text{und} \qquad ac \equiv bd \bmod n.$$

Beispiel 3.5

Wir rechnen modulo $n = 7$. Dann ist zum Beispiel

$$3 \equiv 10 \equiv 17 \equiv -4 \equiv -11 \bmod 7,$$

da die Differenzen $3 - 10 = 10 - 17 = -7$, $17 - (-4) = 21$ und $(-4) - (-11) = 7$ Vielfache von 7 sind. Genauso haben wir

$$4 \equiv 11 \equiv 18 \equiv -3 \equiv -10 \bmod 7.$$

Dann gelten zum Beispiel

$$3 + 4 = 7 \equiv 0 \bmod 7 \qquad \text{und} \qquad 3 \cdot 4 = 12 \equiv 5 \bmod 7.$$

Wir können aber zum Beispiel 3 durch -4 und 4 durch 18 ersetzen, was die gleichen Ergebnisse (modulo 7) liefert:

$$(-4) + 18 = 14 \equiv 0 \bmod 7 \qquad \text{und} \qquad (-4) \cdot 18 = -72 = -77 + 5 \equiv 5 \bmod 7.$$

[9] In der Mathematik sagt man, dass Kongruenz modulo n eine Äquivalenzrelation ist.

Da jede Zahl a kongruent zu seinem Rest bei der Division durch n ist, genügt es, die Zahlen $\{0, 1, \ldots, n-1\}$ zu betrachten. Die Addition und Multiplikation werden wie gewöhnlich berechnet, und das Ergebnis wird dann modulo n reduziert (d.h. durch seinen Rest modulo n ersetzt).

Beispiel 3.6

Modulo $n = 4$ erhalten wir folgende Additions- und Multiplikationstafel:

$+$	0	1	2	3
0	0	1	2	3
1	1	2	3	0
2	2	3	0	1
3	3	0	1	2

$\cdot$	0	1	2	3
0	0	0	0	0
1	0	1	2	3
2	0	2	0	2
3	0	3	2	1

Wir können modulo n fast alle Eigenschaften aus den ganzen Zahlen nutzen, wie zum Beispiel die Assoziativ-, Kommutativ- und Distributivgesetze. Auch Addition von 0 und Multiplikation mit 1 (oder beliebige dazu kongruente Zahlen) haben keinen Effekt. Es gibt aber einige überraschende Neuigkeiten, die im obigen Beispiel modulo 4 schon auftauchen:

- Das Produkt von zwei nicht zu Null kongruenten Zahlen kann zu Null kongruent sein: z.B. $2 \not\equiv 0 \bmod 4$, aber $2 \cdot 2 = 4 \equiv 0 \bmod 4$.
- Das Produkt von zwei „großen“ Zahlen kann zu 1 kongruent werden: z.B. $3 \cdot 3 = 9 \equiv 1 \bmod 4$.

Bei größerem n oder mehreren Rechenoperationen ist es besser, auch die Zwischenergebnisse so früh wie möglich modulo n zu reduzieren, um die auftauchenden Zahlen möglichst klein zu halten. Keinesfalls sollten Ausdrücke erst komplett als ganze Zahlen ausgerechnet und ganz am Ende modulo n reduziert werden.

Beispiel 3.7

Wir wollen mit möglichst wenig Rechenaufwand das Produkt von 1986 und 1117 modulo 30 berechnen:

- Zuerst berechnen wir die Reste bei Division durch 30:

$$1986 = 66 \cdot 30 + 6 \equiv 6 \bmod 30 \quad \text{und} \quad 1117 = 37 \cdot 30 + 7 \equiv 7 \bmod 30.$$

- Dann berechnen wir das Produkt

$$1986 \cdot 1117 \equiv 6 \cdot 7 = 42 \bmod 30.$$

- Zuletzt reduzieren wir auch das Ergebnis modulo 30: $42 = 30 + 12 \equiv 12 \bmod 30$.

Also $1986 \cdot 1117 \equiv 12 \bmod 30$.

Beispiel 3.8

Welcher Rest bleibt bei Division mit Rest von 17^{341} *durch* 5*?*
Wir rechnen also modulo 5. Es ist vollkommen unpraktikabel, zunächst 17^{341} auszurechnen, das ist eine Zahl mit mehr als 400 Dezimalstellen. Stattdessen nutzen wir die üblichen Potenzgesetze und reduzieren einige Zwischenergebnisse modulo 5.

- Zuerst reduzieren wir 17 modulo 5: $17 \equiv 2 \bmod 5$, und somit
$$17^{341} \equiv 2^{341} \bmod 5.$$
- Danach kann man bemerken, dass $2^2 \equiv -1 \bmod 5$ gilt. Also kann jedes Quadrat von 17 durch -1 ersetzt werden. Daher ist es sinnvoll, bei der Potenz $2^{341} = 2 \cdot 2 \cdot \ldots \cdot 2$ so viele Paare von Faktoren wie möglich zu gruppieren, indem der Exponent durch 2 dividiert wird:
$$2^{341} = 2^{2 \cdot 170 + 1} = (2^2)^{170} \cdot 2 \equiv (-1)^{170} \cdot 2 = 2 \bmod 5.$$

Der gesuchte Rest ist also 2.

Beispiel 3.9

Welcher Rest entsteht bei Division mit Rest von 3^{3719} *durch* 8*?*
In diesem Fall ist $3^2 = 9 \equiv 1 \bmod 8$. Also ist es wieder sinnvoll, den Exponent durch 2 zu teilen:
$$3^{3719} = 3^{2 \cdot 1859 + 1} = (3^2)^{1859} \cdot 3^1 \equiv 1^{1859} \cdot 3 = 3 \bmod 8.$$

Es entsteht also Rest 3.

Aufgabe 3.10

Was ist der Rest der Division von $4^{92} - 44598 \cdot 95623$ durch 21?

3.2. **Teilbarkeitsregeln.** Bevor wir uns als nächstes mit den modularen Inversen beschäftigen, wollen wir als Anwendung der Kongruenzrechnung einige Teilbarkeitsregeln präsentieren. Es geht also um die Frage, wie man an der Dezimaldarstellung einer Zahl erkennen kann, ob sie durch eine vorgegebene Zahl teilbar ist oder nicht. Einige dieser Regeln sind sicher aus der Schule bekannt, wir werden diese Regeln hier aber auch begründen und damit verstehen können.

Wir betrachten eine natürliche Zahl a in Dezimaldarstellung

$$a = a_r a_{r-1} \ldots a_2 a_1 a_0 \quad \text{mit Ziffern } a_i \text{ aus der Menge } \{0, 1, \ldots, 9\}.$$

Mit Zehnerpotenzen ausgeschrieben hat a also die Form

$$a = a_r \cdot 10^r + a_{r-1} \cdot 10^{r-1} + \ldots + a_2 \cdot 10^2 + a_1 \cdot 10 + a_0.$$

3.2.1. *Teilbarkeit durch 3 bzw. 9.* Sei $d = 3$ oder $d = 9$. Für die Zehnerpotenzen gilt nach den Rechenregeln modulo d, dass

$$10^k \equiv 1^k \equiv 1 \bmod d \text{ für alle } k \in \mathbb{N}_0.$$

Also erhalten wir

$$\begin{aligned} a &\equiv a_r \cdot 10^r + a_{r-1} \cdot 10^{r-1} + \ldots + a_2 \cdot 10^2 + a_1 \cdot 10 + a_0 \bmod d \\ &\equiv a_r + a_{r-1} + \ldots + a_1 + a_0 \bmod d. \end{aligned}$$

Auf der rechten Seite steht die Summe der Ziffern von a, die sogenannte *Quersumme* von a. Als Teilbarkeitsregel ergibt sich:

Teilbarkeitsregel 3.11: Teilbarkeit durch 3 und 9

Eine natürliche Zahl ist genau dann durch 3 bzw. 9 teilbar, wenn ihre Quersumme durch 3 bzw. 9 teilbar ist.

Beispiel 3.12

Zum Beispiel ist die Zahl $a = 2719842$ durch 3 (aber nicht durch 9) teilbar, weil ihre Quersumme $2 + 7 + 1 + 9 + 8 + 4 + 2 = 33$ durch 3 (aber nicht durch 9) teilbar ist.

3.2.2. *Teilbarkeit durch 4.* Für die Zehnerpotenzen gilt $10 \equiv 2 \bmod 4$ und $10^k \equiv 0 \bmod 4$ für alle $k \geq 2$. Damit erhalten wir

$$a \equiv a_r \cdot 10^r + \ldots + a_2 \cdot 10^2 + a_1 \cdot 10 + a_0 \equiv a_1 \cdot 10 + a_0 \bmod 4.$$

Der Ausdruck $a_1 \cdot 10 + a_0$ gibt die Zahl aus den letzten beiden Ziffern der Dezimaldarstellung von a, also erhalten wir als Teilbarkeitsregel:

Teilbarkeitsregel 3.13: Teilbarkeit durch 4

Eine natürliche Zahl ist genau dann durch 4 teilbar, wenn die Zahl aus den letzten beiden Ziffern ihrer Dezimaldarstellung durch 4 teilbar ist.

3.2.3. *Teilbarkeit durch 11.* Für die Zehnerpotenzen gilt für alle $k \in \mathbb{N}_0$:

$$10^r \equiv (-1)^r \bmod 11.$$

Es folgt

$$\begin{aligned} a &\equiv a_r \cdot 10^r + a_{r-1} \cdot 10^{r-1} + \ldots + a_1 \cdot 10 + a_0 \bmod 11 \\ &\equiv (-1)^r a_r + (-1)^{r-1} a_{r-1} + \ldots - a_1 + a_0 \bmod 11. \end{aligned}$$

Die Zahl $(-1)^r a_r + (-1)^{r-1} a_{r-1} + \ldots - a_1 + a_0$ nennen wir die *alternierende Quersumme* von a. Als Teilbarkeitskriterium erhalten wir

Teilbarkeitsregel 3.14: Teilbarkeit durch 11

Eine natürliche Zahl ist genau dann durch 11 teilbar, wenn ihre alternierende Quersumme durch 11 teilbar ist.

3.3. Anwendung: Prüfziffern bei Strichcodes, Kreditkarten und IBAN.

3.3.1. *Strichcodes.* Auf praktisch allen Waren finden sich Strichcodes zur eindeutigen Identifizierung. Der Strichcode und die zugehörige Zahlenfolge in Abbildung 9 ist ein Beispiel für eine „European Article Number“, abgekürzt EAN. Dieses zunächst in Europa eingeführte System ist inzwischen Teil des weltweiten Systems der „Global Trade Item Number“ (GTIN).

Eine EAN besteht aus 13 Ziffern. Die letzte Ziffer ist dabei eine Prüfziffer, die wie folgt berechnet wird. Sei

$$a_1\, a_2\, \ldots\, a_{12}\, a_{13}$$

eine EAN, mit Ziffern $a_i \in \{0, 1, 2, \ldots, 9\}$. Die ersten 12 Ziffern werden, links beginnend, abwechselnd mit 1 und 3 multipliziert und dann aufaddiert. Die Prüfziffer wird dann so gewählt, dass die Gesamtsumme kongruent zu 0 modulo 10 wird, es ist also

$$a_{13} \equiv -(a_1 + 3a_2 + a_3 + 3a_4 + a_5 + 3a_6 + a_7 + 3a_8 + a_9 + 3a_{10} + a_{11} + 3a_{12}) \bmod 10.$$

Wir überprüfen das für die EAN in Abbildung 9:

$$\begin{aligned} a_{13} &\equiv -(4 + 3\cdot 0 + 1 + 3\cdot 2 + 3 + 3\cdot 4 + 6 + 3\cdot 2 + 7 + 3\cdot 8 + 0 + 3\cdot 0) \bmod 10 \\ &\equiv -9 \;\equiv\; 1 \bmod 10 \end{aligned}$$

Abbildung 9: European Article Number (EAN) auf einer Teepackung.

(Photo from Wikipedia. Published under License Creative Commons CC0 1.0 Universal Public Domain Dedication.)

Warum wird überhaupt eine Prüfziffer in die EAN integriert? Der Vorteil ist, dass an der Prüfziffer gewisse Fehler bei der EAN erkannt werden können und in dem Fall eine Fehlermeldung ausgegeben werden kann. Zum Beispiel kann es leicht passieren, dass bei der Eingabe der EAN ein Tippfehler passiert oder beim Scannen des Strichcodes

eine Ziffer falsch wird (durch eine Verschmutzung am Strichcode oder einen kleinen Kratzer oder Knick im Etikett).

Eine falsche Ziffer in einer EAN wird durch die Prüfziffer erkannt. Zur Begründung sei

$$a_1 \ldots a_i \ldots a_{13}$$

eine korrekte EAN und

$$a_1 \ldots a_{i-1}\, \tilde{a}_i\, a_{i+1} \ldots a_{13}$$

der 13-stellige Code, der durch einen Fehler an der i-ten Position entsteht.

Für die korrekte EAN ist die Prüfziffer-Summe

$$Z := a_1 + 3a_2 + \ldots + a_{11} + 3a_{12} + a_{13} \equiv 0 \bmod 10.$$

Für den fehlerhaften 13-stelligen Code erhalten wir die Prüfziffer-Summe $\tilde{Z}$:

$$\begin{aligned} \tilde{Z} &= \begin{cases} Z + 3(\tilde{a}_i - a_i) & \text{wenn } i \text{ gerade} \\ Z + (\tilde{a}_i - a_i) & \text{wenn } i \text{ ungerade} \end{cases} \\ &\equiv \begin{cases} 3(\tilde{a}_i - a_i) \bmod 10 & \text{wenn } i \text{ gerade} \\ \tilde{a}_i - a_i \bmod 10 & \text{wenn } i \text{ ungerade} \end{cases} \end{aligned}$$

In beiden Fällen kann die entstandene Summe $\tilde{Z}$ nicht durch 10 teilbar sein, da sonst $\tilde{a}_i - a_i = 0$ wäre (wegen $\mathrm{ggT}(3, 10) = 1$). Also ist $\tilde{Z} \not\equiv 0 \bmod 10$, d.h. an der Prüfziffer wird erkannt, dass der fehlerhafte 13-stellige Code keine korrekte EAN sein kann.

Ebenso werden Vertauschungen zweier direkt benachbarter Ziffern meist erkannt, außer die Differenz der vertauschten Ziffern ist 5 (d.h. Vertauschungen 0 5/5 0 und 1 6/6 1 und 2 7/7 2 und 3 8/8 3 und 4 9/9 4 werden nicht erkannt).

Aufgabe 3.15

(i) Kann 4 0 3 6 0 3 1 7 5 5 6 0 7 eine zulässige EAN sein?
(ii) Bestimmt die Prüfziffer in der folgenden EAN: 4 0 1 4 4 8 1 5 1 1 3 1 ?

3.3.2. *Kreditkarten-Nummern.* Auch bei Kreditkarten-Nummern (und einigen weiteren Nummern, die zur eindeutigen Identifikation benutzt werden) ist die letzte Ziffer eine Prüfziffer. Sie wird mit dem „Luhn-Algorithmus“ [10] berechnet.

Algorithmus 3.16: Luhn-Algorithmus

Gegeben sei eine Ziffernfolge beliebiger Länge $a_n\, a_{n-1} \ldots a_2\, a_1\, a_0$.

1. Startend mit der vorletzten Ziffer, gehe nach links und multipliziere alle Ziffern $a_1, a_3, \ldots$ mit ungeradem Index mit 2. Wenn eine Zahl ≥ 10 entsteht, ziehe 9 ab.

[10] Benannt nach Hans Peter Luhn (1896-1964), Wissenschaftler bei IBM.

2. Bilde die Summe S aller Ziffern aus Schritt 1 (inklusive der nicht-verdoppelten Ziffern).
3. Wenn $S + a_0 \equiv 0 \bmod 10$, Ausgabe „Nummer korrekt“.
   Wenn $S + a_0 \not\equiv 0 \bmod 10$, Ausgabe „Nummer nicht korrekt“.

Als Beispiel überprüfen wir, ob die Ziffernfolge

$$a_{15}\, a_{14}\, \ldots\, a_2\, a_1\, a_0 = 1\,2\,3\,4\,5\,6\,7\,8\,9\,0\,1\,2\,3\,4\,5\,6$$

eine korrekte Kreditkarten-Nummer sein kann. Die Summe S ist

$$\begin{aligned} S &= 2\cdot 1 + 2 + 2\cdot 3 + 4 + (2\cdot 5 - 9) + 6 + (2\cdot 7 - 9) + 8 \\ &\quad + (2\cdot 9 - 9) + 0 + 2\cdot 1 + 2 + 2\cdot 3 + 4 + (2\cdot 5 - 9) \\ &\equiv 8 \bmod 10. \end{aligned}$$

Da $S + a_0 \equiv 8 + 6 \not\equiv 0 \bmod 10$ ist, gibt der Luhn-Algorithmus als Ausgabe „Nummer nicht korrekt“ aus.

Durch die mit dem Luhn-Algorithmus berechnete Prüfziffer werden einzelne falsche Ziffern immer erkannt. Darüber hinaus erkennt der Luhn-Algorithmus Vertauschungen zweier direkt benachbarter Ziffern, mit Ausnahme von 0 9 und 9 0.

Aufgabe 3.17

(i) Kann 9 8 7 6 5 4 3 2 1 0 9 8 7 6 5 4 eine zulässige Kreditkartennummer sein?
(ii) Bestimmt die Prüfziffer in der Kreditkartennummer

5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 ?

3.3.3. *IBAN.* Die „International Bank Account Number“, abgekürzt IBAN, ist eine internationale Standard-Notation für Kontonummern. Die IBAN ist wie folgt zusammengesetzt:

- 2-stelliger Ländercode (z.B. DE für Deutschland),
- 2-stellige Prüfsumme,
- Maximal 30-stellige Folge von Buchstaben (A-Z) und Ziffern zur Identifikation des Kontos (in Deutschland aus 18 Ziffern bestehend, 8 für die Bankleitzahl, 10 für die Kontonummer).

Die zwei Stellen der Prüfsumme werden durch eine Rechnung modulo 97 ermittelt. Wir beschreiben das Verfahren der Einfachheit halber nur für die in Deutschland verwendete Form der IBAN (der Algorithmus verläuft aber analog für alle Länder).

Algorithmus 3.18: Überprüfung der IBAN

Sei

$$\text{DE}??' \underbrace{a_1a_2\dots a_8}_{\text{Bankleitzahl}} \underbrace{a_9a_{10}\dots a_{18}}_{\text{Kontonummer}}$$

eine IBAN, für die die beiden Prüfziffern ? und ?′ berechnet werden sollen.

1. Schreibe die ersten vier Stellen an das Ende, es entsteht
$$a_1a_2\dots a_8a_9a_{10}\dots a_{18}\text{DE}??'.$$
2. Die Buchstaben werden ersetzt durch zweistellige Zahlen, entsprechend $A = 10$, $B = 11$, ..., $Z = 35$. Es entsteht also die Folge
$$a_1a_2\dots a_8a_9a_{10}\dots a_{18}1314??'.$$
3. Die Prüfziffern werden jetzt so gewählt, dass die Zahl mit obiger 24-stelliger Dezimaldarstellung kongruent zu 1 modulo 97 ist.

Für weitere Informationen zur IBAN verweisen wir auf die Wikipedia-Seiten
`https://de.wikipedia.org/wiki/Internationale_Bankkontonummer`
`https://en.wikipedia.org/wiki/International_Bank_Account_Number`

3.4. **Modulare Inverse.** Bei der Lösung von linearen Gleichungen mit rationalen oder reellen Zahlen ist es in der Regel notwendig, durch Zahlen zu teilen. Zum Beispiel hat die Gleichung $7x = 5$ die rationale Zahl $x = \frac{5}{7}$ als Lösung. Dies folgt daraus, dass es zu der Zahl $a = 7$ eine andere rationale Zahl b gibt, mit $ab = 1$, nämlich $b = \frac{1}{7}$. Division durch 7 kann also als Multiplikation mit $\frac{1}{7}$ interpretiert werden.

Wenn wir aber ganzzahlige Lösungen suchen, geraten wir in Probleme, da es keine ganze Zahl b mit $7b = 1$ gibt. Wir sagen, dass 7 als ganze Zahl *nicht invertierbar* ist. Allgemein nennt man ein Element a eines Zahlbereichs *invertierbar*, wenn es ein anderes Element b in dem Zahlbereich gibt, so dass $a \cdot b = 1$ ist. Zum Beispiel sind in den ganzen Zahlen 1 und -1 invertierbar, denn $1 \cdot 1 = 1$ und $(-1) \cdot (-1) = 1$.

Wenn wir Arithmetik modulo einer Zahl n betrachten, haben wir etwas mehr Spielraum als bei den ganzen Zahlen.

Definition 3.19: Invertierbare Zahlen modulo n

Eine ganze Zahl a heißt *invertierbar modulo* n, falls es eine ganze Zahl b gibt, mit
$$ab \equiv 1 \bmod n.$$
Die Zahl b (oder eine dazu modulo n kongruente Zahl) heißt *modulare Inverse* von a (modulo n). Wir schreiben auch $b = a^{-1} \bmod n$.

Beachtet: jetzt muss das Produkt ab nicht unbedingt gleich 1 sein, sondern nur Rest 1 bei der Division durch n haben. Also muss $ab = 1 + kn$ für irgendeine ganze Zahl

k gelten. Diese zusätzliche Flexibilität, die Zahl k variieren zu können, gibt viel mehr invertierbare Zahlen modulo n als 1 und -1.

Beispiel 3.20

Modulo 4 sind nur 1 und 3 (und alle dazu kongruenten Zahlen) invertierbar. Dies erkennt man an der Multiplikationstafel im Beispiel 3.6. In der Tat findet sich nur in den Zeilen dieser beiden Elemente als Ergebnis der Multiplikation eine 1, und sogar nur einmal in jeder Zeile. Die 2 ist also modulo 4 nicht invertierbar.

Beispiel 3.21

Modulo 5 sind alle Zahlen invertierbar, die keine Vielfache von 5 sind. Also ist a genau dann modulo 5 invertierbar, wenn $a \not\equiv 0 \bmod 5$. Die modularen Inversen können aus den folgenden Kongruenzen entnommen werden:

$$1 \cdot 1 = 1 \bmod 5, \quad 2 \cdot 3 = 6 \equiv 1 \bmod 5 \quad \text{und} \quad 4 \cdot 4 = 16 \equiv 1 \bmod 5.$$

Also modulo 5 gilt $1^{-1} = 1$ (wie immer), $2^{-1} = 3$, $3^{-1} = 2$ und $4^{-1} = 4$. Das letzte Inverse ist auch nicht überraschend, denn $4 \equiv -1 \bmod 5$, und die inverse von -1 ist immer -1 (also wiederum 4 in diesem Fall).

Der folgende Satz gibt eine vollständige Antwort zur Frage nach den invertierbaren Zahlen modulo einer festen Zahl n, und erklärt damit auch die Beobachtungen in den obigen Beispielen.

Satz 3.22: Invertierbare Elemente modulo n

Sei n eine natürliche Zahl. Eine ganze Zahl a ist genau dann modulo n invertierbar, wenn $\mathrm{ggT}(a, n) = 1$ gilt.

Begründung. *Wir nehmen zunächst an, dass a invertierbar modulo n ist. Dann gibt es eine ganze Zahl b mit $ab \equiv 1 \bmod n$, also $n \mid ab - 1$ (vgl. Definition 3.1). Es gibt daher eine Zahl $r \in \mathbb{Z}$, so dass $ab - 1 = rn$. Sei d nun ein beliebiger gemeinsamer Teiler von a und n. Dann teilt d auch $ab - rn = 1$. Es sind also $d = \pm 1$ die einzigen gemeinsamen Teiler von a und n und wir haben gezeigt, dass* $\mathrm{ggT}(a, n) = 1$.
Angenommen umgekehrt $\mathrm{ggT}(a, n) = 1$, *so gibt es nach Folgerung 2.19 Bézout-Koeffizienten s, t, so dass*

$$1 = \mathrm{ggT}(a, n) = sa + tn.$$

Modulo n erhalten wir

$$1 = sa + tn \equiv sa + t0 = sa \bmod n\ .$$

Also ist a modulo n invertierbar, mit modularer Inverse s (oder irgendeine zu s modulo n kongruente Zahl).

Beispiel 3.23

Mit Satz 3.22 ergeben sich zum Beispiel folgende invertierbare Reste:

- Modulo 6: 1 und 5.
- Modulo 7: 1, 2, 3, 4, 5 und 6.
- Modulo 8: 1, 3, 5 und 7.
- Modulo 9: 1, 2, 4, 5, 7 und 8:

Bemerkung 3.24. *Allgemeiner gilt: genau dann sind alle $a \not\equiv 0 \bmod n$ invertierbar, wenn n eine Primzahl ist.*

Begründung. *Nach Satz 3.22 sind genau dann alle Reste ungleich 0 invertierbar, wenn n zu allen Zahlen kleiner als n teilerfremd ist. Dies ist aber genau dann der Fall, wenn n eine Primzahl ist.*

Bemerkung 3.25 (Berechnung von modularen Inversen)**.** *In der Begründung von Satz 3.22 haben wir gesehen, dass die modularen Inversen gegeben sind durch Bézout-Koeffizienten. Zur Erinnerung: Ist $\mathrm{ggT}(a,n) = 1$, so gibt es ganze Zahlen s,t, so dass $1 = \mathrm{ggT}(a,n) = sa + tn$. Modulo n ergibt sich dann*

$$1 \equiv sa \bmod n, \qquad \text{d.h. } a^{-1} = s \bmod n.$$

Bézout-Koeffizienten lassen sich mit dem Euklidischen Algorithmus berechnen (durch Rückwärtseinsetzen, vgl. Folgerung 2.19). Es ist nicht nötig, vorab zu prüfen, ob eine Zahl invertierbar modulo n ist, denn das ergibt sich sowieso aus dem Euklidischen Algorithmus, je nachdem ob sich als Ergebnis $\mathrm{ggT}(a,n) = 1$ ergibt oder nicht.

Beispiel 3.26

Entscheidet, ob die jeweilige Zahl invertierbar (bzgl. des angegebenen Moduls) ist und berechnet ggf. das modulare Inverse.

(i) Wir betrachten die Zahl 31 modulo 42. Entsprechend der vorigen Bemerkung führen wir den Euklidischen Algorithmus für $n = 42$ und $a = 31$ aus.

$$\begin{aligned} 42 &= 1 \cdot 31 + 11 \\ 31 &= 2 \cdot 11 + 9 \\ 11 &= 1 \cdot 9 + 2 \\ 9 &= 4 \cdot 2 + 1 \\ 2 &= 2 \cdot 1 + 0. \end{aligned}$$

Es ergibt sich, dass $\mathrm{ggT}(31,42) = 1$, und folglich ist 31 invertierbar modulo 42 (siehe Satz 3.22). Durch Rückwärtseinsetzen ermitteln wir

Bézout-Koeffizienten.

$$\begin{aligned} 1 &= 9 - 4 \cdot 2 = 9 - 4 \cdot (11 - 9) = 5 \cdot 9 - 4 \cdot 11 \\ &= 5 \cdot (31 - 2 \cdot 11) - 4 \cdot 11 = 5 \cdot 31 - 14 \cdot 11 \\ &= 5 \cdot 31 - 14 \cdot (42 - 31) = 19 \cdot 31 - 14 \cdot 42. \end{aligned}$$

Modulo 42 ergibt sich $1 \equiv 19 \cdot 31$, also ist $31^{-1} \equiv 19 \bmod 42$.

(ii) Wir betrachten 391 modulo 527. Der Euklidische Algorithmus hat die folgenden Schritte.

$$\begin{aligned} 527 &= 1 \cdot 391 + 136 \\ 391 &= 2 \cdot 136 + 119 \\ 136 &= 1 \cdot 119 + 17 \\ 119 &= 7 \cdot 17 + 0. \end{aligned}$$

Als Ergebnis ist $\mathrm{ggT}(391, 527) = 17 \neq 1$, d.h. 391 ist nicht invertierbar modulo 527.

Aufgabe 3.27

Untersucht welche der folgenden Zahlen modulo 63 invertierbar sind, und bestimmt ggf. die modularen Inversen:

(i) 19, (ii) 35, (iii) 40.

3.5. **Einfache symmetrische Verfahren.** Die traditionellen Verschlüsselungsverfahren bedienen sich zweier ähnlicher Transformationen, die den Klartext bearbeiten und sich gegenseitig aufheben.

3.5.1. *Verschiebe-Chiffre.* In Abschnitt 1.2 hatten wir bereits das Beispiel der Verschiebung im Alphabet um s Positionen:

$\mathrm{encrypt}_s$ = „ersetze jeden Buchstaben durch den im Alphabet s Positionen danach“
$\mathrm{decrypt}_s$ = „ersetze jeden Buchstaben durch den im Alphabet s Positionen davor“

Dabei fangen wir am Ende des Alphabets wieder bei A an, z. B. bei $s = 3$:

Klartext	A	B	C	D	E	F	G	H	I	J	K	L	M
	1	2	3	4	5	6	7	8	9	10	11	12	13
	4	5	6	7	8	9	10	11	12	13	14	15	16
Geheimtext	D	E	F	G	H	I	J	K	L	M	N	O	P
Klartext	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
	14	15	16	17	18	19	20	21	22	23	24	25	26
	17	18	19	20	21	22	23	24	25	26	1	2	3
Geheimtext	Q	R	S	T	U	V	W	X	Y	Z	A	B	C

Die Leerzeichen zwischen den Worten und Groß- und Kleinschreibung werden dabei ignoriert. Denken wir uns die Buchstaben nummeriert, so haben wir 26 Zeichen, und die Verschiebung im Alphabet entspricht einer Addition modulo 26:

$$\text{encrypt}_s(m) \equiv m + s \bmod 26$$
$$\text{decrypt}_s(m) \equiv m - s \bmod 26$$

Diese Methode heißt **Verschiebe-Chiffre**. Die Verschiebung um $s = 3$ wird Julius Cäsar[11] zugeschrieben und heißt deshalb auch **Cäsar-Chiffre**.

Beispiel 3.28: Cäsar-Chiffre

Der Satz „ATTACK AT DAWN“ wird für $s = 3$ zu „DWWDFNDWGDZQ“.

Der Algorithmus ist sehr einfach, und es gibt so wenige Schlüssel, dass man leicht alle durchprobieren kann. Möchte man ein größeres Alphabet verwenden (z. B. einschließlich Ziffern), so nummeriert man von 1 bis n (z. B. $n = 36$) und rechnet analog:

$$\text{encrypt}_s(m) \equiv m + s \bmod n$$
$$\text{decrypt}_s(m) \equiv m - s \bmod n$$

Die Umkehrtransformation ist dabei einfach zu finden, da wir in der modularen Arithmetik immer eine Subtraktion haben, oder mathematischer gesprochen: Es gibt immer ein Inverses $-s$ bezüglich der Addition modulo n, so dass $s + (-s) \equiv 0 \bmod n$.

3.5.2. *Tausch-Chiffre.* Man könnte versuchen, statt der Addition in der Verschiebe-Chiffre eine Multiplikation mit t modulo 26 zu benutzen, d. h. $\text{encrypt}(m) \equiv m \cdot t \bmod 26$.

Beispiel 3.29

Mit dem Faktor $t = 2$ wird dann aus „ATTACK AT DAWN“ der Geheimtext „BNNBFVBNHBTB“.
Jetzt hat die Empfängerseite ein Problem: „B“ kann sowohl für „A“, als auch für „N“ stehen. Diese Chiffre ist nicht praktikabel.

Beispiel 3.30

Mit dem Faktor $t = 3$ wird aus „ATTACK AT DAWN“ jetzt der Geheimtext „CHHCIGCHLCQP“. Hier haben wir also Erfolg!

Der Unterschied zwischen den Beispielen liegt daran, ob t und 26 teilerfremd zueinander sind. Wir wissen, dass dies genau die Bedingung dafür ist, dass es eine modulare Inverse x von t bezüglich der Multiplikation gibt, also ein x mit $t \cdot x \equiv 1 \bmod 26$.

[11] Cäsar, Gaius Julius (100 v.Chr. – 44 v.Chr.), römischer Feldherr.

Definition 3.31: Tausch-Chiffre

Eine **Tausch-Chiffre** besteht aus einer Kombination aus einer Verschiebe-Chiffre mit anschließender Multiplikation, beides in modularer Arithmetik modulo n. Der Schlüssel einer Tausch-Chiffre ist das Paar (s, t), die Verschlüsselung ist:

$$\text{encrypt}_{(s,t)}(m) \equiv (m + s) \cdot t \bmod n$$

Ist $t \cdot x \equiv 1 \bmod n$, so können wir die Transformation mit modularer Arithmetik wie folgt umkehren. Die verschlüsselte Nachricht hat die Form

$$m' \equiv (m + s) \cdot t \bmod n.$$

Multiplikation mit x auf beiden Seiten ergibt

$$m' \cdot x \equiv (m + s) \cdot t \cdot x \equiv m + s \bmod n.$$

Durch Addition von $-s$ erhalten wir daraus

$$m' \cdot x - s \equiv m \bmod n,$$

und dies liefert die Entschlüsselungsfunktion

$$\text{decrypt}_{(s,x)}(m') \equiv m' \cdot x - s \bmod n.$$

Modulare Inverse existieren nur für t mit $\text{ggT}(t, n) = 1$, also im Fall $n = 26$ für

$$1, 3, 5, 7, 9, 11, 15, 17, 19, 21, 23, 25$$

Multiplikative Schlüssel und ihre modulare Inverse sind dann:

Schlüssel	1	3	5	7	9	11	15	17	19	21	23	25
modulare Inverse	1	9	21	15	3	19	7	23	11	5	17	25

Damit haben wir eine große Anzahl von Tausch-Chiffren, insgesamt $26 \cdot 12 = 312$, die man kaum noch per Hand entschlüsseln kann.

Beispiel 3.32: Tausch-Chiffre

Wir chiffrieren „ATTACK AT DAWN“ mit einer Tausch-Chiffre mit dem Schlüssel $(3, 5)$ modulo 26.
Zuerst erfolgt die Cäsar-Chiffre und ergibt „DWWDFNDWGDZQ“.
Dies multiplizieren wir noch mit dem Faktor $t = 5$ und versenden schließlich den Text „TKKTDRTKITZG“.
Der Empfänger muss zunächst die Multiplikation rückgängig machen durch Multiplikation mit $t^{-1} = 21 \bmod 26$, danach wird 3 subtrahiert:

Geheimtext	T	K	K	T	D	R	T	K	I	T	Z	G
	20	11	11	20	4	18	20	11	9	20	26	7
$\cdot\, 21 \bmod 26$	4	23	23	4	6	14	4	23	7	4	0	17
$-\, 3 \bmod 26$	1	20	20	1	3	11	1	20	4	1	23	14
Klartext	A	T	T	A	C	K	A	T	D	A	W	N

Aufgabe 3.33

Alice und Bob einigen sich auf eine Tausch-Chiffre für Großbuchstaben (d.h. mit Rechnen modulo 26) mit geheimem Schlüssel $(s, t) = (12, 7)$.

(i) Alice will das Wort „KONGRUENZ“ verschlüsseln. Wie lautet der Geheimtext?

(ii) Bob hat von Alice den Geheimtext „POQLOBVBOSH“ erhalten. Was ist der Klartext?

Bei der Cäsar-Chiffre kann der geheime Schlüssel entdeckt werden, wenn die Verschlüsselung eines einzelnen Buchstabens bekannt ist. Bei der Tausch-Chiffre ist es nicht so einfach, aber auch nicht besonders kompliziert: Es kann reichen, die Verschlüsselungen von zwei Buchstaben zu kennen. Wie folgendes Beispiel zeigt, gibt jede bekannte Verschlüsselung eine „Kongruenzgleichung“ für die zwei Teile des Schlüssels (s, t). Wenn genügend viele Koeffizienten invertierbar modulo dem gewählten n sind, können s und t daraus bestimmt werden.

Beispiel 3.34

Bei einer Tausch-Chiffre modulo 26 werden folgende Verschlüsselungen bekannt: $7 \mapsto 20$ und $12 \mapsto 23$. D.h. der Schlüssel (s, t) erfüllt

$$t \cdot (7 + s) \equiv 20 \bmod 26 \quad \text{und} \quad t \cdot (12 + s) \equiv 23 \bmod 26.$$

Ausmultiplizieren und Isolieren von st in beiden Kongruenzen liefert

$$st \equiv 20 - 7t \bmod 26 \quad \text{und} \quad st \equiv 23 - 12t \bmod 26,$$

also insbesondere $20 - 7t \equiv 23 - 12t \bmod 26$, und somit

$$5t \equiv 3 \bmod 26.$$

Multiplikation mit der modularen Inverse $5^{-1} = 21 \bmod 26$ liefert $t \equiv 21 \cdot 3 \equiv 11 \bmod 26$. Um s zu bestimmen nutzen wir nun eine der ursprünglichen Kongruenzen mit dem schon bestimmten Wert von t. Zum Beispiel

$$11 \cdot (7 + s) \equiv 20 \bmod 26,$$

also $11s \equiv 20 - 11 \cdot 7 = -57 \equiv 21 \bmod 26$. Multiplikation mit $11^{-1} = 19 \bmod 26$ gibt $s \equiv 19 \cdot 21 = 399 \equiv 9 \bmod 26$. Der Schlüssel ist dann $(s, t) = (9, 11)$.

Aufgabe 3.35

Bei einer Tausch-Chiffre mit Großbuchstaben als Zahlen modulo 26 wie in der Tabelle werden die Buchstaben E, M und R als W, C und J verschlüsselt. Was ist der Schlüssel (s, t)?

Solche oder ähnliche Verschlüsselungsverfahren setzen jeweils auf Sender- wie auf Empfängerseite einen (passenden) geheimen Schlüssel voraus. Dadurch bleibt das Problem, wie der **geheime** Schlüssel vom Sender zum Empfänger kommt, erhalten. Bei N Kommunikationspartnern muss jeder mit jedem einen Schlüssel teilen. Hierzu braucht man $N(N-1)/2$ Schlüssel. Wie werden diese Schlüssel verwaltet und verteilt?

3.6. **Diffie-Hellman-Schlüsselaustausch.** Um ein gemeinsames Geheimnis auszutauschen, das dann z. B. als Schlüssel in einem symmetrischen Verfahren benutzt werden kann, bietet sich der Diffie-Hellman-Schlüsselaustausch (nach Whitfield Diffie[12] und Martin Hellman[13] 1976) an. Die Grundidee ist dabei, dass folgende aus den natürlichen Zahlen bekannte Fundamentalgleichung auch beim Rechnen modulo einer Primzahl p gilt:

$$(g^x)^y \equiv g^{xy} \equiv g^{yx} \equiv (g^y)^x \bmod p.$$

Um also g^{xy} zu berechnen, benötigt man außer $X = g^x$ nur y (linke Seite), alternativ außer $Y = g^y$ nur x (rechte Seite), und beide Rechnungen liefern das Gleiche:

$$X^y \equiv Y^x \bmod p.$$

Dies ist das beim Diffie-Hellman-Verfahren verschlüsselt ausgetauschte gemeinsame Geheimnis $s = X^y = Y^x$. Eine Eigenheit des Verfahrens ist, dass s nicht gewählt werden kann, sondern sich aus dem Verfahren ergibt. Das gemeinsame Geheimnis s kann dann wiederum als Schlüssel benutzt werden, um mit einem symmetrischen Verfahren eine beliebige Nachricht zu verschlüsseln. Der Algorithmus läuft wie folgt:

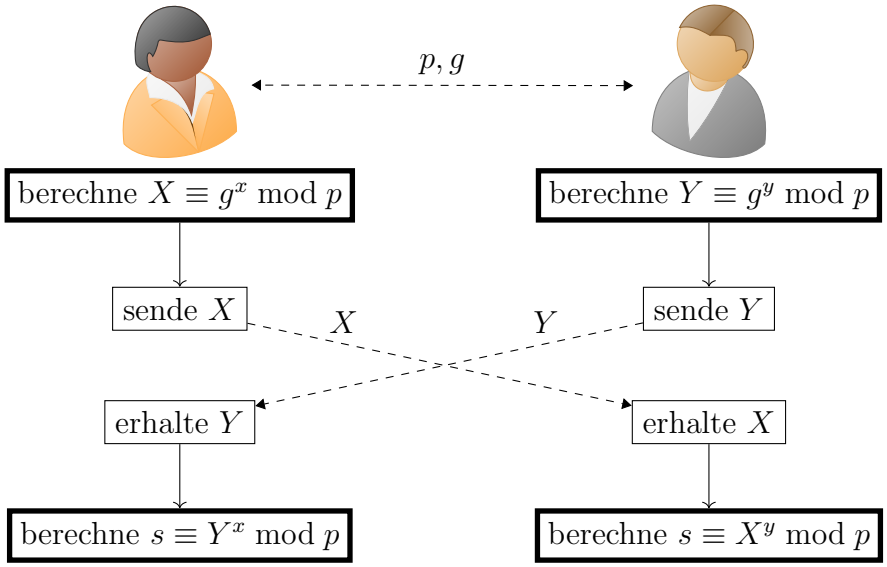
Algorithmus 3.36: Diffie-Hellman Schlüsselaustausch

1. Alice und Bob einigen sich auf eine große Primzahl p und eine Zahl $2 \leq g < p$. Diese Zahlen dürfen öffentlich bekannt sein.
2. Alice wählt eine große Zufallszahl x (bleibt geheim) und sendet an Bob (öffentlich)
$$X \equiv g^x \bmod p.$$
3. Bob wählt eine große Zufallszahl y (bleibt geheim) und sendet an Alice (öffentlich)
$$Y \equiv g^y \bmod p.$$

[12] Diffie, Whitfield (*1944), amerikanischer Kryptologe und Mathematiker (Sun Microsystems).
[13] Hellman, Martin (*1945), amerikanischer Kryptologe, eigentlich Elektrotechniker.

4. Alice berechnet das gemeinsame Geheimnis durch
$$s \equiv Y^x \bmod p.$$
5. Bob berechnet das gemeinsame Geheimnis durch
$$s \equiv X^y \bmod p.$$

Abbildung 10: Diffie-Hellman-Austausch



Die Sicherheit des Verfahrens beruht darauf, dass es schwierig ist, bei bekanntem p, g und X aus $X \equiv g^x \bmod p$ das geheim gehaltene x (und damit s) zu bestimmen. Hätten wir es mit (positiven) reellen Zahlen zu tun, so könnten wir einfach den Logarithmus zur Basis g benutzen! Wir werden im Abschnitt 3.8 sehen, ob und wie schwierig dieses Problem modulo p lösbar ist. Die Empfehlung lautet dann: Die Primzahl p muss möglichst groß sein. Die Zahl g sollte so gewählt sein, dass die Menge
$$\{g^0,\ g^1, \ldots,\ g^{p-1} \bmod p\}$$
möglichst viele verschiedene Zahlen enthält.

Beispiel 3.37

Mit $p = 29$ und $g = 5$ wählen Alice bzw. Bob die Exponenten $x = 3$ bzw. $y = 8$. Alice sendet Bob die Zahl
$$X \equiv g^x = 5^3 = 125 \equiv 9 \bmod 29,$$

und Bob sendet Alice die Zahl

$$Y \equiv g^y = 5^8 = \left(5^2\right)^4 \equiv (-4)^4 = 256 \equiv 24 \bmod 29.$$

Bob berechnet dann die geheime Zahl s als

$$X^y = 9^8 = (81)^4 \equiv (23)^4 \equiv (-6)^4 \equiv 36^2 \equiv 7^2 = 49 \equiv 20 \bmod 29,$$

und Alice berechnet

$$Y^x = 24^3 \equiv (-5)^3 = -125 \equiv -9 \equiv 20 \bmod 29.$$

Beide kommen wie erwartet zum gleichen Ergebnis $s = 20$.

Aufgabe 3.38

Alice und Bob wollen eine gemeinsame geheime Zahl modulo 37 mit Diffie-Hellmann erzeugen. Dafür wählen sie 5 als öffentliche Basis. Alice wählt den Exponent 11, und Bob wählt den Exponent 19.

(i) Welche Zahl X sendet Alice an Bob?
(ii) Und welche Zahl Y sendet Bob an Alice?
(iii) Was ist die gemeinsame geheime Zahl s?

3.7. **ElGamal-Algorithmus.** Der ElGamal [14]–Algorithmus (1985) ist eine Verallgemeinerung des Diffie-Hellman-Verfahrens. Die Grundidee ist dabei, das gemeinsame Geheimnis $s \equiv X^y \equiv Y^x \bmod p$ aus dem Diffie-Hellman-Schlüsselaustausch mittels der modularen Inversen zur Verschlüsselung einer beliebigen Nachricht m einzusetzen: Setzen wir $m' = mY^x$, so gilt

$$m'\left(X^y\right)^{-1} \equiv mY^x\left(X^y\right)^{-1} \equiv mX^y\left(X^y\right)^{-1} \equiv m \bmod p.$$

Mit den Grundbegriffen aus Abschnitt 1.1 gilt für das ElGamal-Verfahren also:

$$\text{encrypt}_{Y^x}(m) = mY^x \qquad \text{encrypt}_{X^y}(m) = mX^y$$
$$\text{decrypt}_{X^y}(m') = m'\left(X^y\right)^{-1} \qquad \text{decrypt}_{Y^x}(m') = m'\left(Y^x\right)^{-1}$$

Der Algorithmus läuft wie folgt:

Algorithmus 3.39: ElGamal Verschlüsselung

1. Alice und Bob einigen sich auf eine große Primzahl p und eine Zahl $2 \leq g < p$. Diese Zahlen dürfen öffentlich bekannt sein.

[14] ElGamal, Taher (*1956), amerikanischer Kryptologe, Informatiker (RSA Security, Inc.)

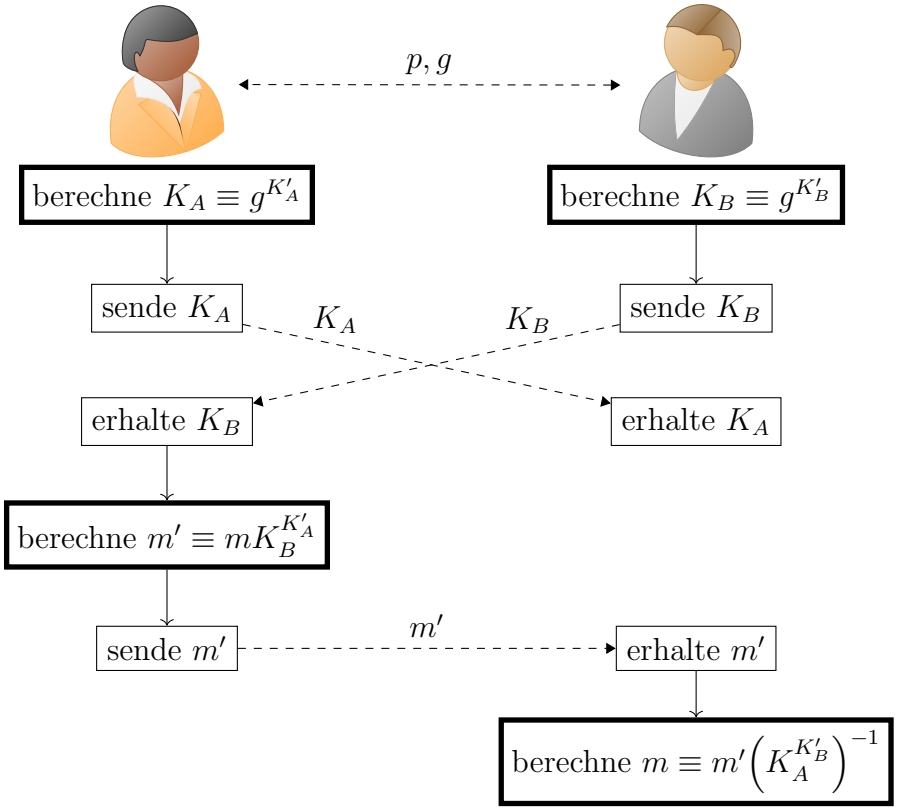
2. Bob wählt als privaten Schlüssel eine große Zufallszahl $K'_B = y$ und sendet an Alice seinen öffentlichen Schlüssel
$$K_B = Y \equiv g^y \bmod p.$$
3. Alice wählt als privaten Schlüssel eine große Zufallszahl $K'_A = x$ und sendet an Bob ihren öffentlichen Schlüssel
$$K_A = X \equiv g^x \bmod p.$$
4. Alice verschlüsselt die Nachricht m durch
$$m' \equiv mY^x \bmod p$$
und sendet sie an Bob.
5. Bob entschlüsselt die Nachricht durch
$$m \equiv m'\left(X^y\right)^{-1} \bmod p.$$

Alice und Bob verwenden zur Ver- bzw. Entschlüsselung also den gleichen geheimen symmetrischen Schlüssel ($X^y \equiv Y^x \bmod p$), den sie jeweils aus ihrem eigenen privaten asymmetrischen Schlüssel (x bzw. y) und dem öffentlichen asymmetrischen Schlüssel des Kommunikationspartners (Y bzw. X) berechnen. Ohne Kenntnis eines der privaten asymmetrischen Schlüssel bleibt der symmetrische Schlüssel geheim.

Beispiel 3.40

Alice und Bob nutzen die gemeinsame Geheimzahl (und die entsprechenden geheimen Exponenten) aus Beispiel 3.37 um weitere Zahlen geheim auszutauschen. Alice will Bob die Zahl $m = 13$ mitteilen. Dafür berechnet sie
$$m' \equiv mY^x = 13 \cdot 20 = 260 \equiv 28 \bmod 29,$$
und sendet dies zu Bob.
Um die Zahl m zu bestimmen, muss Bob zuerst die modulare Inverse der Geheimzahl $s = X^y = 20 \bmod 29$ berechnen. Dies kann zum Beispiel mit dem Euklidischen Algorithmus und Rücksubstitution gemacht werden (siehe Bemerkung 3.25). Man kann aber auch die Faktorisierung $20 = 2 \cdot 10$ nutzen, da beide Inverse von 2 und 10 modulo 29 ziemlich einfach zu bestimmen sind: aus $1 \equiv 30 = 3 \cdot 10 = 15 \cdot 2 \bmod 29$ folgen $2^{-1} = 15 \bmod 29$ und $10^{-1} \equiv 3 \bmod 29$. Daher gilt in diesem Fall
$$(X^y)^{-1} = 20^{-1} \equiv 2^{-1} \cdot 10^{-1} = 15 \cdot 3 = 45 \equiv 16 \bmod 29$$
und somit kann Bob folgendes berechnen:
$$m = m' \cdot (X^y)^{-1} = 28 \cdot 16 \equiv (-1) \cdot 16 = -16 \equiv 13 \bmod 29.$$

Abbildung 11: ElGamal-Verschlüsselung (alles modulo p)



Aufgabe 3.41

Jetzt verwenden Alice und Bob die Zahlen aus Aufgabe 3.38.

(i) Alice will Bob die Zahl 13 mitteilen. Welche Zahl soll sie Bob senden?
(ii) Alice erhält von Bob die Zahl 12. Welche Zahl hat Bob verschlüsselt?

3.8. **Diskreter Logarithmus.** Ist p eine Primzahl, so wissen wir aus dem Abschnitt 3.4 (modulare Inverse), dass jede natürliche Zahl $1 \leq a \leq p-1$ invertierbar modulo p ist, da $\mathrm{ggT}(p, a) = 1$. Dies gilt dann auch für deren Produkte. Mathematisch gesprochen bilden sie zusammen eine Gruppe, die Einheitengruppe modulo p.

Insbesondere können wir alle Potenzen $a^k \bmod p$ einer festen Zahl a betrachten. In manchen Fällen erhalten wir alle möglichen invertierbaren Reste modulo p, aber nicht immer. Dies motiviert folgende Definition.

Definition 3.42: Erzeugendes Element

Eine natürliche Zahl $2 \leq a < p$ heißt *erzeugendes Element modulo* p, falls die Potenzen von a alle möglichen Reste modulo p (bis auf 0) ergeben.

Beispiel 3.43

Für $p = 5$ sind dies alle möglichen Potenzen a^k modulo 5:

k	0	1	2	3	4
1^k	1	1	1	1	1
2^k	1	2	4	3	1
3^k	1	3	4	2	1
4^k	1	4	1	4	1

Dass die Zeile für $a = 1$ nur die 1 enthält, wussten wir natürlich vorher. Aber auch die Zeile für $a = 4$ enthält nicht alle möglichen Werte modulo 5. Die Zeilen für $a = 2$ und $a = 3$ enthalten jeweils alle natürlichen Zahlen von 1 bis 4.
Also die erzeugende Elemente modulo $p = 5$ sind genau 2 und 3.

Im Beispiel 3.43 haben wir beim Exponent $k = 4 = p - 1$ aufgehört und festgestellt, dass wir wie für $k = 0$ nur den Wert 1 erhalten. Im Allgemeinen können wir schon bei $k = p - 2$ aufhören:

Begründung. *Es gibt nur $p - 1$ mögliche Ergebnisse modulo p (0 kann nicht vorkommen). Daher erhalten wir (beginnend mit $a^0 = 1$ spätestens bei a^{p-1} schon ein wiederholtes Element, d.h. $a^r \equiv a^s \bmod p$ für Exponenten $0 \leq s < r \leq p - 1$. Da a invertierbar ist, können wir s-mal mit der modularen Inversen $a^{-1} \bmod p$ multiplizieren und erhalten $a^{r-s} \equiv 1 \bmod p$. Also erhalten wir schon bei einem Exponenten $k < p$, dass $a^k \equiv 1 \bmod p$, und für $r \geq k$ ist die Potenz $a^r \equiv a^{r-k} \bmod p$ schon mit kleinerem Exponenten vorgekommen.*

Bemerkung 3.44. *In der Tat ist der kleinste Exponent $k > 0$ mit $a^k \equiv 1 \bmod p$ immer ein Teiler von $p - 1$. Um dies zu begründen sind aber einige Resultate der Gruppentheorie notwendig.*

Wir können uns nun folgendes natürliches Problem stellen: gegeben zwei Zahlen $1 \leq a, b < p$, finde eine natürliche Zahl x mit

$$y \equiv a^x \bmod p$$

Das ist das **diskrete Logarithmus-Problem**.

Wenn so ein Exponent x existiert, schreibt man

$$x \equiv \log_a y \bmod p$$

Bemerkung 3.45.

(i) *Die Existenz von x ist nur garantiert, wenn a ein erzeugendes Element modulo p ist. Sonst kann es gut passieren, dass die gegebene Zahl b keine Potenz von a (modulo p) ist. Im Beispiel 3.43 wird klar, dass 3 keine Potenz von 4 modulo 5 ist, d.h. $\log_4 3 \bmod 5$ existiert nicht.*

Im extremen Fall $a = 1$ existiert kein $\log_1 b$ für $2 \leq b < p$. Und für $b = 1$ kann man immer $x = 0$ nehmen. Es ist daher sinnvoll, sich auf $2 \leq a, b < p$ zu beschränken.

(ii) *Falls es existiert, ist $\log_a b \bmod p$ nicht eindeutig. Modulo 5 gilt es zum Beispiel $4 \equiv 4^3 \equiv 4 \bmod 5$, also $\log_4 4$ könnte 1 oder 3 sein (oder 5, 7,...).*

Wenn a aber ein erzeugendes Element ist, sind alle Potenzen $1 = a^0, a^1, \ldots, a^{p-2} \bmod p$ unterschiedlich. Man kann dann $\log_a b \bmod p$ eindeutig als den einzigen Exponenten $0 \leq x \leq p-2$ mit $b = a^x$ definieren.

Beispiel 3.46

Für $p = 5$ und $a = 2$ können wir aus den beiden Zeilen

k	0	1	2	3
2^k	1	2	4	3

des obigen Beispiels direkt die diskreten Logarithmen modulo 5 zur Basis 2 ablesen:

y	1	2	3	4
$\log_2 y$	0	1	3	2

Bemerkung 3.47. *Im Bereich der (positiven) reellen Zahlen existiert der Logarithmus immer, da dort die Potenzfunktion monoton wachsend und stetig ist. Zum Beispiel kann die Dezimaldarstellung von $\log_2 100$ bis zur beliebigen Präzision wie folgt bestimmt werden: Da*

$$2^0 = 1 \ (< 100), \quad 2^{20} = 1048576 \ (> 100)$$

gilt $0 < \log_2 100 < 20$. Wenn man dann die Werte 2^y für ganzahlige y zwischen 0 und 20 berechnet, findet man

$$2^6 = 64 < 100 < 128 = 2^7$$

und somit $6 < \log_2 100 < 7$, also $\log_2 100 = 6, \ldots$. Man betrachtet dann 2^y für y zwischen 6 und 7 im Abstand von $\frac{1}{10}$ und findet

$$2^{6,6} = 97,005\ldots < 100 < 2^{6,7} = 103,968\ldots$$

Daraus folgt $6{,}6 < \log_2 100 < 6{,}7$, also $\log_2 100 = 6{,}6\ldots$. Wenn man jetzt die Werte von 2^y von y in kleineren Abständen vergleicht, kann man weitere Dezimalstellen von $\log_2 100$ bestimmen.

Bei Modulo-Rechnung war aber schon bei unserem kleinen Beispiel ($p = 5, a = 2$) die Monotonie verschwunden, und wir waren auf eine Tabelle angewiesen. Für große p wird diese nicht mehr handhabbar, und man geht davon aus, dass das diskrete Logarithmus-Problem „schwer“, für praktische Zwecke also nicht lösbar ist[15].

Es gibt aber eine Methode, den diskreten Logarithmus zu bestimmen: den **Babystep-Giantstep Algorithmus**. Dieser wird von folgender Beobachtung motiviert: wenn man weiß, dass die gegebene Basis a höchstens m^2 unterschiedliche Potenzen hat (z.B. mit $m \geq \sqrt{p-1}$), so reicht es, die Exponenten $x = 0, 1, \ldots, m^2 - 1$ zu betrachten. Dank Division mit Rest (durch m), können diese Exponenten als $x = im + j$ mit geeigneten $0 \leq i, j < m$ geschrieben werden. Die Bedingung $a^x = b$ wird dann

$$a^{im+j} = b, \quad \text{also} \quad a^j = b\left(a^{-m}\right)^i.$$

Es reicht dann, die Listen

- $a^0 = 1, a^1, \ldots, a^{m-1}$ (die Babysteps)
- $b = b\left(a^{-m}\right)^0, b\left(a^{-m}\right), \ldots, b\left(a^{-m}\right)^{m-1}$ (die Giantsteps)

zu berechnen und nach einer sogenannten „Kollision“ zu suchen, d.h. nach Werten i, j mit $a^j = b\left(a^{-m}\right)^i$.

Bemerkung 3.48.

- *Wenn es keine Kollision gibt, dann ist b keine Potenz von a. D.h. der Algorithmus kann auch antworten, ob $\log_a b$ existiert.*
- *Im schlimmsten Fall muss man die m Elemente in beiden Listen berechnen, also höchstens $2m$ Elemente. Dies ist deutlich weniger als die ca. m^2 Potenzen von a.*

Der Algorithmus kann expliziter wie folgt beschrieben werden:

Algorithmus 3.49: Babystep-Giantstep

Eingabe: eine Primzahl p und zwei ganze Zahlen $2 \leq a, b < p$.

Ausgabe: ein Exponent x mit $b \equiv a^x \bmod p$, falls es einen gibt. Sonst „Fehler“.

1. Setze $m := \lceil \sqrt{p-1} \rceil$, die kleinste ganze Zahl größer oder gleich $\sqrt{p-1}$, und $c := (a^m)^{-1} \bmod p$.
2. Für alle $j = 0, 1, \ldots, m-1$ berechne a^j.
3. Für alle $i = 0, 1, \ldots, m-1$ berechne bc^i, bis bc^i in den oben berechneten Werten a^j auftaucht.
4. Falls $bc^i = a^j$, gib $x := mi + j$ aus.
5. Falls keine Gleichheit $bc^i = a^j$ vorkommt, gib „Fehler: b ist keine Potenz von a (modulo p)“ aus.

[15] https://de.wikipedia.org/wiki/Diskreter_Logarithmus

Beispiel 3.50

Wir untersuchen mithilfe des Babystep-Giantstep Algorithmus, ob 3 und 6 Potenzen von $a = 7$ modulo $p = 53$ sind.
Wir berechnen zunächst $m = \lceil \sqrt{p-1} \rceil = \lceil \sqrt{52} \rceil = 8$. Um c zu bestimmen, berechnen wir

$$a^8 = 7^8 = 49^4 = (-4)^4 = 256 \equiv 44 \equiv -9 \bmod 53$$

und erhalten mit dem Euklidischen Algorithmus (oder direkt nach einer kurzen Überlegung) $53 = 54 - 1 = (-9)(-6) - 1$, und somit $(-9)(-6) \equiv 1 \bmod 53$. Also

$$c = (-9)^{-1} \equiv -6 \equiv 47 \bmod 53.$$

Alternativ können zuerst $a^{-1} = 7^{-1} \equiv 15 \bmod 53$ und dann $c = 15^8 \bmod 53$ berechnet werden.
Nun berechnen wir die Potenzen $a^0, a^1, \ldots, a^7 \bmod 53$ durch sukzessive Multiplikationen mit $a = 7$ und Reduktion modulo 53:

j	$7^j \bmod 53$
0	1
1	7
2	$7 \cdot 7 = 49 \equiv -4$
3	$(-4) \cdot 7 = -28 \equiv 25$
4	$25 \cdot 7 = 175 \equiv 16$
5	$16 \cdot 7 = 112 \equiv 6$
6	$6 \cdot 7 = 42 \equiv -11$
7	$(-11) \cdot 7 = -77 \equiv 29$

Nun bestimmen wir die Produkte $bc^i \bmod 53$ für $b = 3$ und $b = 6$ bis es eine Kollision mit einer Potenz $a^j \bmod 53$ gibt.

i, j	$7^j \bmod 53$	$3 \cdot 47^i \equiv 3 \cdot (-6)^i \bmod 53$	$6 \cdot 47^i \equiv 6 \cdot (-6)^i \bmod 53$
0	1	3	6
1	7	$3 \cdot (-6) = -18 \equiv 35$	$6 \cdot (-6) = -36 \equiv 17$
2	49	$(-18) \cdot (-6) = 108 \equiv 2$	$17 \cdot (-6) = -102 \equiv 4$
3	25	$2 \cdot (-6) = -12 \equiv 41$	$4 \cdot (-6) = -24 \equiv 29$
4	16	$(-12) \cdot (-6) = 72 \equiv 19$	
5	6	$19 \cdot (-6) = -114 \equiv -8 \equiv 45$	
6	42	$(-8) \cdot (-6) = 48 \equiv -5$	
7	29	$(-5) \cdot (-6) = 30$	

Mit $b = 6$ gibt es eine Kollision für $i = 3$ und $j = 7$ (d.h. $6c^3 \equiv a^7 \equiv 29 \bmod 53$).
Daher gilt $6 = 7^{mi+j} = 7^{8 \cdot 3 + 7} = 7^{31}$.
Mit $b = 3$ gibt es aber keine Kollision. Dies bedeutet, dass 3 keine Potenz von $a = 7$ ist (und somit ist $a = 7$ kein erzeugendes Element modulo 53).

Aufgabe 3.51

Ist 19 eine Potenz von 4 modulo 71? Mit welchem Exponenten?

3.9. **Die Eulersche φ-Funktion.** In diesem Abschnitt beschäftigen wir uns mit der Frage, wie viele invertierbare Reste modulo n es gibt. Diese Anzahlen sind die Werte einer Funktion, die für die Mathematik, aber auch für moderne technische Anwendungen im Bereich Verschlüsselung, sehr wichtig ist.

Definition 3.52: Eulersche φ-Funktion[16]

Die *Eulersche φ-Funktion* ist die Abbildung

$$\varphi : \mathbb{N} \to \mathbb{N}, \ \varphi(n) = |\{a \,|\, 1 \leq a \leq n \text{ und } \operatorname{ggT}(a, n) = 1\}|.$$

Nach Satz 3.22 ist $\varphi(n)$ gleich der Anzahl der invertierbaren Reste modulo n.

Beispiel 3.53

Wir berechnen einige Werte der Eulerschen φ-Funktion.

n	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	...
$\varphi(n)$	1	1	2	2	4	2	6	4	6	4	10	4	12	6	8	8	16	6	18	8	...

Beispiel 3.54

Für alle Primzahlen p gilt $\varphi(p) = p-1$. Dies folgt sofort aus Definition 3.52, denn alle Zahlen $1, 2, \ldots, p-1$ sind zur Primzahl p teilerfremd (und p selbst nicht).

Werte der Eulerschen φ-Funktion sind für große Zahlen nicht leicht zu berechnen, selbst mit Computer-Hilfe nicht, wenn die Zahlen nur groß genug sind. Aber falls man von einer Zahl n die Primfaktorzerlegung kennt (was aber für große Zahlen auch ein schwieriges Problem ist), dann kann man $\varphi(n)$ mit Hilfe des folgenden Resultats berechnen.

Satz 3.55: Berechnung der Eulerschen φ-Funktion

Für die Eulersche φ-Funktion gelten folgende Formeln.

(a) Ist p eine Primzahl und e eine natürliche Zahl, so gilt

$$\varphi(p^e) = p^{e-1}(p-1).$$

[16] Benannt nach Leonhard Euler (1707-1783).

(b) Sind $n_1, \ldots, n_k$ paarweise teilerfremde natürliche Zahlen, so gilt für das Produkt
$$\varphi(n_1 \cdot \ldots \cdot n_k) = \varphi(n_1) \cdot \ldots \cdot \varphi(n_k).$$
(c) Hat die natürliche Zahl $n \geq 2$ die Primfaktorzerlegung
$$n = p_1^{e_1} \cdot \ldots \cdot p_k^{e_k},$$
so gilt
$$\varphi(n) = p_1^{e_1-1}(p_1 - 1) \cdot \ldots \cdot p_k^{e_k-1}(p_k - 1).$$

Begründung. *(a) Wir zählen umgekehrt die nicht zu p^e teilerfremden Zahlen in $\{1, \ldots, p^e\}$. Dies sind genau die durch p teilbaren Zahlen (da p eine Primzahl ist), also jede p-te Zahl. Es folgt*
$$\varphi(p^e) = p^e - \frac{p^e}{p} = p^{e-1}(p-1).$$
Die Begründung von Teil (b) ist zu aufwändig für dieses Vorbereitungsmaterial. Das hierfür entscheidende Resultat ist der „Chinesische Restsatz“ [a].
(c) In einer Primfaktorzerlegung sind die $p_1^{e_1}, \ldots, p_k^{e_k}$ paarweise teilerfremd. Dann erhalten wir aus den Teilen (a) und (b):
$$\begin{aligned}\varphi(n) &= \varphi(p_1^{e_1} \cdot \ldots \cdot p_k^{e_k}) \overset{(b)}{=} \varphi(p_1^{e_1}) \cdot \ldots \cdot \varphi(p_k^{e_k}) \\ &\overset{(a)}{=} p_1^{e_1-1}(p_1 - 1) \cdot \ldots \cdot p_k^{e_k-1}(p_k - 1).\end{aligned}$$

[a] `https://de.wikipedia.org/wiki/Chinesischer_Restsatz`

Beispiel 3.56

Wir berechnen $\varphi(1000)$ mit Hilfe von Satz 3.55 und der leicht zu bestimmenden Primfaktorzerlegung $1000 = 10^3 = 2^3 \cdot 5^3$. Dann ist
$$\varphi(1000) = \varphi(2^3) \cdot \varphi(5^3) = 2^{3-1}(2-1) \cdot 5^{3-1}(5-1) = 4 \cdot 25 \cdot 4 = 400.$$[17]

[17] Zur Kontrolle können wir diese Zahl auch anders bestimmen: Da $1000 = 2^3 \cdot 5^3$ nur 2 und 5 als Primfaktoren hat, sind die zu 1000 teilerfremden Zahlen genau die Zahlen, die weder durch 2 noch durch 5 teilbar sind. Anders gesagt: die ungeraden Zahlen, die in Dezimaldarstellung nicht auf 5 enden. Zwischen 1 und 1000 sind genau die Hälfte der Zahlen gerade (oder ungerade), es gibt also genau 500 ungerade Zahlen: $1, 3, 5, 7, 9, 11, 13, 15, 17, ..., 995, 997, 999$. Davon endet ein Fünftel auf 5, also 100 davon sind Vielfache von 5. Dies lässt genau $500 - 100 = 400$ ungerade Zahlen, die nicht durch 5 teilbar und somit teilerfremd zu 1000 sind, wie erwartet.

Beispiel 3.57

Wir wollen $\varphi(574992)$ berechnen. Dazu suchen wir zunächst die Primfaktorzerlegung von 574992.
Die Zahl 574992 ist gerade, also dividieren wir solange durch 2, bis eine ungerade Zahl entsteht. Es ergibt sich $574992 = 2^4 \cdot 35937$.
Jetzt versuchen wir andere Primzahlen, um den Faktor 35937 weiter zu zerlegen. Die Quersumme $3 + 5 + 9 + 3 + 7 = 27$ ist durch 9 teilbar, also ist 35937 durch 9 teilbar (Teilbarkeitsregel 3.11), und es gilt $35937 = 3^2 \cdot 3993$. Der Faktor 3993 ist wiederum durch 3 teilbar und wir erhalten $35937 = 3^3 \cdot 1331$.
Als Zwischenergebnis haben wir jetzt bereits $574992 = 2^4 \cdot 3^3 \cdot 1331$. Das ist noch nicht die Primfaktorzerlegung, denn 1331 ist keine Primzahl: die alternierende Quersumme ist $-1 + 3 - 3 + 1 = 0$, also nach der Teilbarkeitsregel 3.14 ist 1331 durch 11 teilbar. In der Tat ist $1331 = 11 \cdot 121 = 11 \cdot 11^2 = 11^3$.
Damit haben wir die Primfaktorzerlegung gefunden, nämlich

$$574992 = 2^4 \cdot 3^3 \cdot 11^3.$$

Jetzt können wir die Formeln aus Satz 3.55 benutzen, um den Wert der Eulerschen φ-Funktion zu berechnen:

$$\begin{aligned}\varphi(574992) &= \varphi(2^4 \cdot 3^3 \cdot 11^3) = \varphi(2^4) \cdot \varphi(3^3) \cdot \varphi(11^3) \\ &= 2^{4-1}(2-1) \cdot 3^{3-1}(3-1) \cdot 11^{3-1}(11-1) \\ &= 8 \cdot 18 \cdot 1210 = 174240.\end{aligned}$$

Aufgabe 3.58

Findet die Primfaktorzerlegung von 13860 und berechnet $\varphi(13860)$.

Die Eulersche φ-Funktion ist für das schnelle und effiziente modulo-Rechnen sehr nützlich.

Satz 3.59: Satz von Euler

Sei n eine positive ganze Zahl. Für alle ganzen Zahlen a, die teilerfremd zu n sind, gilt

$$a^{\varphi(n)} \equiv 1 \bmod n.$$

Wir formulieren einen Spezialfall des Satzes von Euler noch einmal separat, dieser geht auf Fermat[18] zurück.

[18] Benannt nach Pierre de Fermat (1607-1665).

Satz 3.60: Kleiner Satz von Fermat

Sei p eine Primzahl. Dann gilt:

(i) Für alle ganze Zahlen $a \not\equiv 0 \bmod p$ gilt $a^{p-1} \equiv 1 \bmod p$.

(ii) Für alle ganze Zahlen a gilt $a^p \equiv a \bmod p$.

Wir illustrieren die Nützlichkeit des Satzes von Euler durch einige Beispiele. Insbesondere können Exponenten durch den Satz von Euler ohne viel Aufwand reduziert werden.

Beispiel 3.61

Was ist der Rest bei Division von 3^{4010} *durch* 2500*?*
Da 3 und 2500 teilerfremd sind, gilt nach dem Satz von Euler, dass

$$3^{\varphi(2500)} \equiv 1 \bmod 2500.$$

Um dies benutzen zu können, berechnen wir $\varphi(2500)$ (mit Satz 3.55):

$$\begin{aligned}\varphi(2500) &= \varphi(25 \cdot 100) = \varphi(2^2 \cdot 5^4) = \varphi(2^2) \cdot \varphi(5^4) \\ &= 2^{2-1}(2-1) \cdot 5^{4-1}(5-1) = 2 \cdot 125 \cdot 4 = 1000.\end{aligned}$$

Also gilt nach dem Satz von Euler, dass $3^{1000} \equiv 1 \bmod 2500$. Mit Hilfe dieser Formel lässt sich der Exponent zerlegen und verkleinern:

$$\begin{aligned}3^{4010} &= 3^{4000+10} = 3^{4000} \cdot 3^{10} = (3^{1000})^4 \cdot 3^{10} \equiv 1^4 \cdot 3^{10} \bmod 2500 \\ &= 3^5 \cdot 3^5 = 243 \cdot 243 = 59049 \equiv 1549 \bmod 2500.\end{aligned}$$

Bei der Division von 3^{4010} durch 2500 bleibt also Rest 1549.

Beispiel 3.62

Was sind die letzten beiden Ziffern der Dezimaldarstellung von 7^{222}*?*
Die letzten beiden Ziffern der Dezimaldarstellung ergeben sich beim Rechnen modulo 100. Da 7 und 100 teilerfremd sind, gilt nach dem Satz von Euler

$$7^{\varphi(100)} \equiv 1 \bmod 100.$$

Es ist

$$\varphi(100) = \varphi(2^2 \cdot 5^2) = \varphi(2^2) \cdot \varphi(5^2) = 2 \cdot 5 \cdot 4 = 40,$$

also gilt $7^{40} \equiv 1 \bmod 100$. Dann erhalten wir

$$\begin{aligned}7^{222} &= 7^{40 \cdot 5 + 22} = (7^{40})^5 \cdot 7^{22} \equiv 1^5 \cdot 7^{22} \bmod 100 \\ &= (7^4)^5 \cdot 7^2 = (2401)^5 \cdot 7^2 \equiv 1^5 \cdot 7^2 = 49 \bmod 100.\end{aligned}$$

Die letzten beiden Ziffern der Dezimaldarstellung von 7^{222} sind also 49.
Eine Anmerkung noch zu diesem Beispiel. Am Schluss haben wir benutzt, dass $7^4 = 2401 \equiv 1 \bmod 100$ ist. Wenn wir das schon vorher beobachtet hätten, hätten

wir die letzten beiden Ziffern der Dezimaldarstellung auch kürzer (und ohne den Satz von Euler) ausrechnen können:

$$7^{222} = 7^{4\cdot 55+2} = (7^4)^{55} \cdot 7^2 \equiv 1^5 \cdot 7^2 \equiv 49 \bmod 100.$$

Der Satz von Euler liefert also nicht unbedingt den kleinsten Exponenten r, so dass $a^r \equiv 1 \bmod n$. Das Wichtige am Satz von Euler ist aber, dass er überhaupt einen solchen Exponenten liefert, denn im Allgemeinen hat man kaum eine Chance, einen solchen Exponenten durch Probieren zu finden.

Aufgabe 3.63

(i) Bestimmt den kleinsten Rest bei Division mit Rest von 3^{52} durch 35.
(ii) Bestimmt die letzten zwei Ziffern in der Dezimaldarstellung von 3^{363}.
(iii) Welcher Rest bleibt bei Division mit Rest von $2^{66666666666666666666}$ durch 77?

3.10. **Das RSA-Verfahren zur Verschlüsselung.** Wie beim Diffie-Hellman-Verfahren aus Abschnitt 3.6 ist der Ausgangspunkt die Fundamentalgleichung für Potenzen:

$$\left(m^e\right)^d \equiv m^{ed} \bmod n.$$

Allerdings ist hier die Basis der Potenz die zu übermittelnde Nachricht m; der Exponent d ist Teil des privaten Schlüssels. Damit $m^{ed} \equiv m \bmod n$ gilt und das Verfahren funktioniert, wird $ed \equiv 1 \bmod \varphi(n)$ gefordert (siehe 3.67 zur Begründung).

Mit den Grundbegriffen aus Abschnitt 1.1 gilt für das RSA-Verfahren also:

$$\text{encrypt}_{n,e}(m) = m^e \bmod n$$
$$\text{decrypt}_{n,d}(m') = (m')^d \bmod n.$$

Der öffentliche Schlüssel ist $K = (n, e)$, der geheime Schlüssel ist $K' = d$ (da das ebenfalls zur Entschlüsselung benötigte n öffentlich bekannt ist). Wegen der Berechnung modulo n sind die Zahlen 0 bis $n-1$ die möglichen Nachrichten.

Alice möchte eine RSA-verschlüsselte Nachricht an Bob schicken. Der Algorithmus läuft wie folgt:

Algorithmus 3.64: RSA-Verschlüsselung

1. Bob wählt zwei Primzahlen p und q. Diese müssen geheim bleiben.
2. Bob berechnet $n = pq$ und $\varphi(n) = (p-1)(q-1)$. [a]
3. Bob wählt eine Zahl $2 \leq e \leq \varphi(n) - 1$, so dass $\text{ggT}(e, \varphi(n)) = 1$. [b]
4. Bob berechnet die Zahl $2 \leq d \leq \varphi(n) - 1$ mit $ed \equiv 1 \bmod \varphi(n)$. [c]
5. Bob sendet an Alice seinen öffentlichen Schlüssel
$$K_B = (n, e).$$

6. Alice verschlüsselt die Nachricht m durch
$$m' \equiv m^e \bmod n$$
und sendet sie an Bob.
7. Bob entschlüsselt die Nachricht durch
$$m \equiv (m')^d \bmod n.$$

[a] Dies ist für Bob kein Problem mit Satz 3.55, da er die Primfaktorzerlegung von n kennt.
[b] Die Teilerfremdheit wird mit dem Euklidischen Algorithmus geprüft.
[c] Die Zahl d ist der kleinste Repräsentant der modularen Inversen von e modulo $\varphi(n)$. Diese existiert, da $\mathrm{ggT}(e_T, \varphi(n_T)) = 1$ (vgl. Satz 3.22) und sie wird mit dem Euklidischen Algorithmus berechnet, siehe die Bemerkung am Ende von Abschnitt 3.4.

Beispiel 3.65

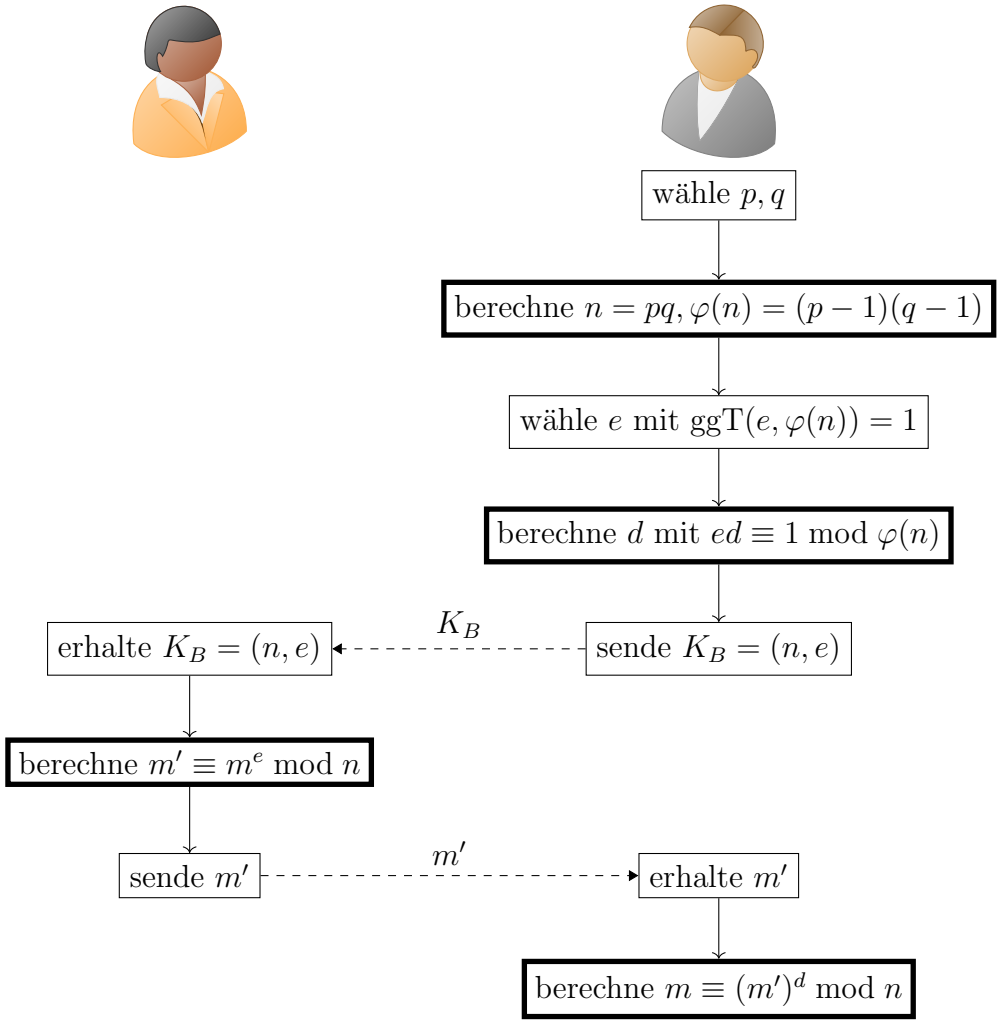
Bob wählt die Primzahlen $p = 5$ und $q = 11$, also er arbeitet mit dem Modul $n = pq = 55$.
Der öffentliche Exponent e muss teilerfremd zu $\varphi(n) = (p-1)(q-1) = 4 \cdot 10 = 40$ sein. Zum Beispiel kann Bob $e = 7$ nehmen, aber nicht $e = 5$. Der zu $e = 7$ entsprechende geheime Schlüssel ist $d = e^{-1} \bmod 40$. Dies kann mithilfe des Euklidischen Algorithmus und Rücksubstitution bestimmt werden: $d = 23$.
Wenn Alice ihm die Zahl $m = 8$ verschlüsselt senden möchte, sollte sie
$$\begin{aligned} m' = m^e = 8^7 &\equiv \left(8^2\right)^3 \cdot 8 = (64)^3 \cdot 8 \equiv 9^3 \cdot 8 \bmod 55 \\ &= 9^2 \cdot (9 \cdot 8) = 81 \cdot 72 \equiv 26 \cdot 17 = 442 \equiv 2 \bmod 55 \end{aligned}$$
senden.
Bob berechnet dann die geheime Nachricht m mithilfe seiner geheimen Schlüssel d als
$$(m')^d = 2^{23} = \left(2^{11}\right)^2 \cdot 2 = (2048)^2 \cdot 2 \equiv 13^2 \cdot 2 \equiv 4 \cdot 2 = 8 \bmod 55.$$

Aufgabe 3.66

Bob hat die Primzahlen $p = 11$ und $q = 17$ gewählt, und schwankt zwischen $e = 23$ und $e = 25$ für den öffentlichen Schlüssel.

(i) Mit welchem Modul n wird Bob arbeiten?
(ii) Welche der zwei Möglichkeiten soll Bob als öffentlichen Schlüssel e nehmen?
(iii) Wie lautet dann der geheime Schlüssel d?
(iv) Alice will Bob die Zahl $m = 33$ mitteilen. Welche Zahl soll Alice senden?
(v) Bob erhält $m' = 97$ von Alice. Welche Zahl steckt dahinter?

Abbildung 12: RSA-Verschlüsselung



Es bleiben zwei fundamentale Aspekte zu diskutieren. Einerseits, die *Korrektheit* des RSA-Verfahrens, d. h. warum der berechtigte Bob nach der Entschlüsselung tatsächlich die Originalnachricht von Alice erhält. Andererseits, worauf die *Sicherheit* des RSA-Verfahrens begründet ist, d. h. warum ein potentieller Angreifer (wahrscheinlich) keine Chance hat, eine abgefangene Nachricht zu entschlüsseln.

Satz 3.67: Korrektheit des RSA-Verfahrens

Mit den obigen Notationen gilt für alle Nachrichten $0 \leq m \leq n-1$, dass $\mathrm{decrypt}_{n,d}\big(\mathrm{encrypt}_{n,e}(m)\big) = m$, d. h. die Hintereinanderausführung von Verschlüsselung und Entschlüsselung liefert wieder die Originalnachricht.

Begründung. *Wir geben die Begründung nur für den Fall, dass die Nachricht m teilerfremd zu n ist (der andere Fall ist etwas aufwändiger). Nach dem Satz von Euler (3.59) gilt $m^{\varphi(n)} \equiv 1 \bmod n$. Da im Verfahren gefordert ist, dass $ed \equiv 1 \bmod \varphi(n)$, also $ed = 1 + j\varphi(n)$ für eine ganze Zahl j, folgt:*

$$(m^e)^d \equiv m^{ed} = m^{1+j\varphi(n)} = m \cdot \left(m^{\varphi(n)}\right)^j \equiv m \cdot 1^j \equiv m \bmod n.$$

Das Verfahren ist also korrekt, wenn es solch ein Paar e, d gibt. Dass es dieses gibt, wissen wir aus dem Abschnitt 3.4 über die modulare Inverse, da die Voraussetzung für Satz 3.22 erfüllt ist.

Wir kommen nun zur Sicherheit des RSA-Verfahrens. Bei der Entschlüsselungsfunktion decrypt wird der private Schlüssel d benutzt, wobei $ed \equiv 1 \bmod \varphi(n)$. Für Bob ist die Berechnung von d kein Problem, da er die Primfaktoren von $n = pq$ und damit auch $\varphi(n) = (p-1)(q-1)$ kennt. Ohne Kenntnis der Primfaktoren von n kann ein Angreifer die Zahl $\varphi(n)$ nicht berechnen:

Satz 3.68: Sicherheit des RSA-Verfahrens

Seien $p, q \geq 3$ Primzahlen und sei $n = pq$. Dann sind folgende Aussagen äquivalent:

(i) n und $\varphi(n)$ sind bekannt.
(ii) Die Primfaktoren von n sind bekannt.

Begründung. *Wenn die Primfaktoren p, q von n bekannt sind, dann können $n = pq$ und $\varphi(n) = (p-1)(q-1)$ direkt berechnet werden.*
Angenommen umgekehrt n und $\varphi(n)$ sind bekannt. Da

$$\varphi(n) = \varphi(pq) = (p-1)(q-1) = pq - p - q + 1 = n - (p+q) + 1,$$

kann die Summe der Primfaktoren aus n und $\varphi(n)$ bestimmt werden:

$$s := p + q = n - \varphi(n) + 1.$$

Da $(x-p)(x-q) = x^2 - (p+q)x + pq$, sind die Primfaktoren p, q einfach die Lösungen der Gleichung $x^2 - sx + n = 0$, welche mit der pq-Formel direkt gefunden werden können:

$$p, q = \frac{s \pm \sqrt{s^2 - 4n}}{2} = \frac{n - \varphi(n) + 1 \pm \sqrt{(n - \varphi(n) + 1)^2 - 4n}}{2}.$$

Nach Satz 3.68 beruht die Sicherheit des RSA-Verfahrens also entscheidend darauf, dass die Faktorisierung großer Zahlen auch mit den schnellsten Computern nicht in realistischer Zeit durchführbar ist. Es ist tatsächlich kein schneller Algorithmus zum Faktorisieren bekannt, es ist aber auch bisher nicht bewiesen, dass solch ein Algorithmus nicht existieren kann.

Das RSA-Verfahren hat weitere Schwachstellen, die in der Praxis berücksichtigt werden müssen. Hier besprechen wir nur einen möglichen Angriff, falls die gleiche Nachricht an zwei Teilnehmende **mit dem gleichen Modul** n und unterschiedlichen öffentlichen Exponenten e_1, e_2 zu Chiffretexten

$$m_1' \equiv m^{e_1} \bmod n$$
$$m_2' \equiv m^{e_2} \bmod n$$

verschlüsselt wird. Also n, m_1', m_2', e_1, e_2 sind allen, insbesondere dem Angreifer, bekannt. Die öffentlichen Exponenten e_1, e_2 werden häufig als Primzahlen gewählt, sind also teilerfremd. Es gibt daher Bézout-Koeffizienten r und s mit

$$r\, e_1 + s\, e_2 = 1,$$

wobei eine der beiden Zahlen negativ sein muss. Angenommen $r < 0$ (sonst vertauschen wir die Rollen von e_1, m_1' und e_2, m_2'), so kann die geheime Nachricht m aus m_1', m_2', r und s folgenderweise bestimmt werden:

$$(m_1'^{-1})^{-r}\, m_2'^{s} \equiv m^{r\, e_1 + s\, e_2} = m \bmod n.$$

wobei die modulare Inverse $m_1'^{-1} \bmod n$ ebenfalls mit dem Euklidischen Algorithmus berechnet werden kann (Bemerkung 3.25).

Beispiel 3.69

Bob und Carl nutzen RSA um verschlüsselte Nachrichten zu erhalten. Bob hat $(n_B, e_B) = (55, 7)$ als öffentlichen Schlüssel (wie in Beispiel 3.65), und Carl nutzt $(n_C, e_C) = (55, 13)$.
Alice will Bob und Carl die gleiche geheime Nachricht mitteilen. Dafür sendet sie $m_B' = 4$ an Bob, und $m_C' = 14$ an Carl.
Da Bob und Carl den gleichen Modul $n_B = n_C = 55$ nutzen, und da die Exponenten die Relation $2e_B - e_C = 2 \cdot 7 - 13 = 1$ erfüllen, kann die Nachricht von Alice ohne die geheimen Schlüsseln von Bob und Carl berechnet werden als

$$m = m_B'^2\, (m_C')^{-1} = 4^2 \cdot 4 = 2^6 = 64 \equiv 9 \bmod 55,$$

wobei $14^{-1} = 4 \mod 55$ direkt aus $14 \cdot 4 = 56 \equiv 1 \bmod 55$ oder mithilfe des Euklidischen Algorithmus bestimmt werden kann.

Aufgabe 3.70

Bob und Carl nutzen den gleichen Modul $n = 11 \cdot 17$ für RSA, und zwar mit öffentlichen Exponenten $e_B = 7$ (wie in Aufgabe 3.66) und $e_C = 11$.
Alice will den beiden die gleiche Zahl m mitteilen, und sendet $m_B' = 93$ an Bob und $m_C' = 53$ an Carl.
Bestimmt die Zahl m ($\bmod n$), ohne die privaten Schlüsseln von Bob oder Carl zu berechnen.

Wir haben hier eine sehr vereinfachte Form des RSA-Verfahrens vorgestellt, um die grundlegenden mathematischen Konzepte klar zu machen. Für eine konkrete technische Umsetzung und Implementierung des RSA-Verfahrens sind noch zahlreiche weitere Sicherheitsaspekte zu beachten, um Angreifern keine Angriffsmöglichkeit zu bieten. Solche Fragen sind ein Thema des hochaktuellen Forschungsgebiets der Kryptographie, an der Schnittstelle von Mathematik, Informatik und Ingenieurwissenschaften.

4. Lösungen zu den Aufgaben

Aufgabe 2.11. $630 = 2 \cdot 3^2 \cdot 5 \cdot 7$.

Aufgabe 2.22. $9 = \mathrm{ggT}(1737, 117) = 6 \cdot 1737 - 89 \cdot 117$.

Aufgabe 2.25

(i) Es ist $\mathrm{ggT}(1338, 4668) = 6$. Die Gleichung $1338x + 4668y = c$ hat genau dann ganzzahlige Lösungen, wenn c ein Vielfaches von 6 ist. Hier also nur für (b) $c = 24$.

(ii) Der Euklidische Algorithmus liefert Bézout-Koeffizienten

$$6 = \underbrace{157}_{s} \cdot 1338 \underbrace{-45}_{t} \cdot 4668.$$

Es ist $c = 96 = 16 \cdot 6$. Also

$$96 = (16 \cdot 157) \cdot 1338 - (16 \cdot 45) \cdot 4668 = \underbrace{2512}_{x} \cdot 1338 \underbrace{-720}_{y} \cdot 4668.$$

Weitere Lösungen ergeben sich durch andere Wahlen von Bézout-Koeffizienten (siehe Bemerkung 2.20). Für jede Zahl ℓ sind $s + \ell b$ und $t - \ell a$ auch Bézout-Koeffizienten. Für $\ell = 1$ ergibt sich hier zum Beispiel die weitere Lösung

$$96 = (16 \cdot (157 + 4668)) \cdot 1338 + (16 \cdot (-45 - 1338)) \cdot 4668 = 77200 \cdot 1338 - 22128 \cdot 4668.$$

Aufgabe 3.10 Es ist $4^3 = 64 \equiv 1 \bmod 21$ und damit folgt $4^{92} = (4^3)^{30} \cdot 4^2 \equiv 16 \bmod 21$. Außerdem ist $44598 \equiv 15 \bmod 21$ und $95623 \equiv 10 \bmod 21$. Insgesamt:

$$4^{92} - 44598 \cdot 95623 \equiv 16 - 15 \cdot 10 \equiv 13 \bmod 21.$$

Also Rest 13.

Aufgabe 3.15.

(i) Nein, die Prüfziffer (letzte Ziffer) müsste eine 8 sein.

(ii) $? = 9$.

Aufgabe 3.17.

(i) Nein, die Prüfziffer (letzte Ziffer) müsste eine 8 sein.

(ii) $? = 8$.

Aufgabe 3.27

(i) Der Euklidische Algorithmus liefert

$$1 = \mathrm{ggT}(63, 19) = -3 \cdot 63 + 10 \cdot 19.$$

Also ist 10 eine modulare Inverse zu 19 modulo 63.

(ii) 35 und 63 sind beide durch 7 teilbar, also hat 35 keine Inverse modulo 63.

(iii) Der Euklidische Algorithmus liefert

$$1 = \mathrm{ggT}(63, 40) = 7 \cdot 63 - 11 \cdot 40.$$

Also ist -11, und damit auch 52, eine Inverse zu 40 modulo 63.

Aufgabe 3.33.

(i) „EGZCBWOZF“,
(ii) „TEILERFREMD“.

Aufgabe 3.35. $(s, t) = (4, 17)$.

Aufgabe 3.38. $X = 2$, $Y = 32$, $X^y = Y^x = 35$.

Aufgabe 3.41. 11, 3.

Aufgabe 3.51. Ja: $4^{13} \equiv 19 \bmod 71$.

Aufgabe 3.58 $13860 = 2^2 \cdot 3^2 \cdot 5 \cdot 7 \cdot 11$ und $\varphi(13860) = 2880$.

Aufgabe 3.63

(i) 11.
(ii) 27.
(iii) 64.

Aufgabe 3.66

(i) 187.
(ii) $e = 23$.
(iii) $d = 7$.
(iv) 33.
(v) 168.

Aufgabe 3.70 $m_M'^2 \left(m_L'^{-1}\right)^3 \equiv 53^2 \cdot (-2)^3 \equiv -32 \equiv 155 \mod 187$.