Jörgen Sjögren

TALTEORI OCH KRYPTOGRAFI

Skövde
Maj 00

Innehållsförteckning

0.	Inledning	3
1.	Mängdteoretiska grundbegrepp …..	4
2.	Funktionsbegreppet	8
3.	Något om komplexitet	13
4.	Euklides´ algoritm	17
5.	Kongruensaritmetik	24
6.	Eulers sats mm	32
7.	Allmänt om kryptografi	42
8.	Några exempel på chiffer	49
9.	RSA-systemet	56
10.	Blandade uppgifter till kapitel 4 - 9 ….. …..	62
11.	Tips och lösningsförslag till vissa övningsuppgifter	65
12.	Litteratur	66

0 Inledning

Detta kompendium är ett försök att på ett någorlunda enkelt sätt förklara för vissa kryptografiska system viktig matematik. Huruvida jag har lyckats i detta avseende är det läsarens sak att avgöra. Några avsnitt är svårare än andra. Så är exempelvis avsnitt tre om komplexitet svårt. Avsnittet om informationsteori (ss 44f) är också svårt. Båda är viktiga för kryptografi, men kan hoppas över vid första genomläsningen. Kompendiet innehåller dessutom ett avsnitt om kryptografiska principer, samt genomgång av några kryptografiska system.

Det är viktigt att läsaren inser att det följande i huvudsak är matematisk text, och skall studeras som sådan. Detta innebär att man noggrant själv går genom resonemang och bevis, samt ser till att man tillgodogör sig definitioner. Saknar man förståelse av definitioner kan man inte förstå efterföljande text. När man arbetar med en matematisk text gör man detta aktivt med penna i hand. Regelbundet bör man dessutom kontrollera att man kan definitioner och förstår bevis. Detta görs lämpligen genom att man själv (utan att ha kompendiet tillgängligt) formulerar definitioner och satser samt genomför bevis.

De flesta avsnitten avslutas med några uppgifter. Det är givetvis lämpligt att lösa några uppgifter då och då efter hand som läsningen fortskrider. För de allra flesta av övnings-uppgifterna gäller att det är enkelt att kontrollera huruvida lösningen är korrekt. För dylika uppgifter finns inget "svar" i "facit". För vissa andra uppgifter finns lösningstips i "facit". Se för övrigt ingressen på s 63.

För den som vill gå vidare finns avslutningsvis några förslag till vidare läsning.

1 Mängdteoretiska grundbegrepp

All matematik kan formuleras i mängdteorins språk. Här skall vi dock bara formulera några enstaka bekväma begrepp, som vi behöver i framställningen nedan. Mängdbegreppet är primitivt i den bemärkelsen att det inte låter sig definieras med hjälp av enklare begrepp. Vi kan emellertid karakterisera en mängd som en samling objekt, dess element. Denna samling tänker vi oss havande en egen existens. Så består t ex en skolklass av en samling individer, men vi kan också tänka oss klassen, samlingen individer, som en självständig enhet.

En mängd kan beskrivas genom att i en lista räkna upp alla element i mängden. Vi sätter elementen inom mängdklamrar och skriver t ex {0, 1, 2, 3, 4}. Observera att det inte spelar någon roll i vilken ordning elementen räknas upp. En skolklass består exempelvis av samma individer oavsett hur de är ordnade i en klasslista. En mängd kan också beskrivas genom att ange en egenskap som alla mängdens element har. Vi skriver då $\{x : P(x)\}$ som utläses "mängden av alla element x så att $P(x)$". Mängden ovan skulle då kunna skrivas $\{x : x$ är ett naturligt tal och $x < 5\}$. Vi kan uttrycka att ett objekt x är ett element i en mängd A genom att skriva $x \in A$. På motsvarande sätt betyder $x \notin A$, att x inte är element i A.

Nu till några definitioner.

Definition 1.1: A är en *delmängd* till B, $A \subseteq B$, om och endast om $\forall x(x \in A \rightarrow x \in B)$. Detta utläses ”För alla x gäller att om $x \in A$, så $x \in B$, vilket betyder att alla element i A är element i B”.

Exempel 1.1: $\{0, 1, 2, 3, 4\} \subseteq \{x : x$ är ett heltal$\}$.

Definition 1.2: Två mängder A och B är *lika* om och endast om $A \subseteq B$ och $B \subseteq A$. Vi skriver då $A = B$.

Exempel 1.2: $\{0, 1, 2, 3, 4\} = \{x : x$ är ett naturligt tal och $x < 5\}$

Nedanstående mängdbeteckningar är viktiga och vanliga.
N betecknar mängden naturliga tal.
Z, Q, R och **C** betecknar mängden av hela, rationella, reella respektive komplexa tal.
$\mathbf{Z}_n = \{0, 1, \ldots , n-1\}$ för $n > 0$.
Den mängd som saknar element kallas *tomma mängden* och betecknas $\varnothing$.

Exempel 1.3: Exempel 1.2 ovan kan då skrivas $\mathbf{Z}_5 = \{x : x \in \mathbf{N} \ \& \ x < 5\} =$
$= \{x \in \mathbf{N} : x < 5\}$

Givet en uppsättning mängder kan vi med hjälp av mängdoperationen bilda nya mängder.

Definition 1.3: $A \cup B = \{x : x \in A$ eller $x \in B\}$ kallas *unionen* av A och B.

Definition 1.4: $A \cap B = \{x : x \in A$ och $x \in B\}$ kallas *snittet* av A och B.

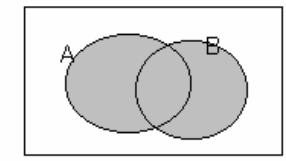
Definition 1.5: $A - B = \{x : x \in A$ och $x \notin B\}$ kallas *differensen* av A och B.

Definition 1.6: $A^C = G - A = \{x : x \in G$ och $x \notin A\}$ kallas *komplement* till A och bildas givet en grundmängd (ett universum) G.

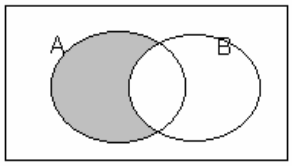
Grundmängden är ofta underförstådd. Andra vanliga beteckningar för komplement är $\overline{A}$ och A'.

Ovanstående definitioner kan illustreras med hjälp av Venndiagram.

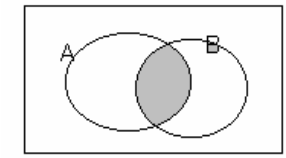
Union
$A \cup B$



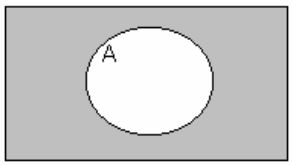
Differens
A - B



Snitt
$A \cap B$



Komplement
A^C



Rektangeln symboliserar grundmängden, det universum inom vilket operationen äger rum.

Definition 1.7: *Potensmängden* till A, P(A), är mängden av alla delmängder till A, dvs $P(A) = \{B : B \subseteq A\}$.

Exempel 1.4: Låt G = **N**, A = $\mathbf{Z}_4$, B = $\{x : x = 2y \ \& \ y \in \mathbf{Z}_3\}$, dvs A = {0, 1, 2, 3} och B = {0, 2, 4}.
Då gäller det att
$A \cup B = \{0, 1, 2, 3, 4\} = \mathbf{Z}_5$
$A \cap B = \{0, 2\}$
A - B = {1, 3}
$A^C = \{4, 5, 6, 7, ...\}$
$P(B) = \{\varnothing, \{0\}, \{2\}, \{4\}, \{0, 2\}, \{0, 4\}, \{2, 4\}, B\}$
Sista raden gäller eftersom $\varnothing \subseteq A$ för varje mängd A, och därför $\varnothing \subseteq B$. Dessutom är $A \subseteq A$ för varje mängd A, varför $B \subseteq B$. Därför måste både $\varnothing \in P(B)$ och $B \in P(B)$.

Mängder är ickeordnade samlingar av objekt. Om objekten är ordnade skriver vi $\langle a_1, a_2, \ldots, a_n \rangle$ för att beteckna *den ordnade n-tippeln av elementen* $a_1, a_2, \ldots, a_n$

Definition 1.8: *Produktmängden* av A och B, $A \times B$, är mängden av alla ordnade par där förstakomponenten väljs från A och andrakomponenten väljs från B, dvs $A \times B = \{\langle a, b \rangle : a \in A \text{ och } b \in B\}$.

Exempel 1.5: Låt A = {1, 2} och B = {c, d, e}.
Då är $A \times B = \{\langle 1, c \rangle, \langle 2, c \rangle, \langle 1, d \rangle, \langle 2, d \rangle, \langle 1, e \rangle, \langle 2, e \rangle\}$.
Observera att det i allmänhet gäller att $A \times B \neq B \times A$.

Exempel 1.6: Visa att $A \cap B \subseteq A$.

Låt $x \in A \cap B$ (x är ett godtyckligt element i $A \cap B$).
Då måste $x \in A$ och $x \in B$, varför $x \in A$.
Vi har därför att $x \in A \cap B \rightarrow x \in A$.
Eftersom x är ett godtyckligt element kan vi skriva $\forall x(x \in A \cap B \rightarrow x \in A)$.
$\therefore$ $A \cap B \subseteq A$. (Symbolen "$\therefore$" utläses "alltså".)

Exempel 1.7: Visa att $(A \cap B)^C = A^C \cup B^C$.

Vi har att visa att (i) $(A \cap B)^C \subseteq A^C \cup B^C$ och (ii) $A^C \cup B^C \subseteq (A \cap B)^C$ (se def 1.2).

(i)	Låt	$x \in (A \cap B)^C$	
	Då	$x \notin (A \cap B)$, vilket ger	
	att	$x \notin A$ eller $x \notin B$	
	dvs	$x \in A^C$ eller $x \in B^C$	
		$x \in A^C \cup B^C$	(Rita!)
	$\therefore$	$(A \cap B)^C \subseteq A^C \cup B^C$.	
(ii)	Låt	$x \in A^C \cup B^C$.	
	då	$x \in A^C$ eller $x \in B^C$	
		$x \notin A$ eller $x \notin B$	
		$x \notin A \cap B$	
		$x \in (A \cap B)^C$	(Rita!)
	$\therefore$	$(A \cap B)^C \subseteq A^C \cup B^C$.	

(i), (ii) och definition 1.2 ger då $(A \cap B)^C = A^C \cup B^C$.

Övningsuppgifter

1.1 Låt $G = \mathbf{Z}$, $A = \mathbf{Z}_7$, $B = \{x : x = 2y \ \& \ y \in \mathbf{N}\}$.
Ange A och B på listform
Ange $A \cap B$, $A - B$, $P(A - B)$, $P(A) - P(B)$.

1.2 Låt $G = \mathbf{N}$. Ange på listform

a)	$\{x : x \neq x\}$.	b)	$\{x : x = x\}$.
c)	$\{y : y = 2x \ \& \ x \in \mathbf{N}\}$.	d)	$\{y : y = 2x + 1 \ \& \ x \in \mathbf{N}\}$.

1.3 Visa att $A \subseteq A$ för varje A, och att $\varnothing \subseteq A$ för varje A.

1.4 Antag att vi vet att $A \cap C = B \cap C$. Gäller det då att $A = B$? Motivera!

1.5 Som uppgift 1.4 men med union i stället för snitt.

1.6 Ge exempel på mängder A och B så att både $A \in B$ och $A \subseteq B$.

1.7 Antag att $P(A) = P(B)$. Gäller det då att $A = B$? Motivera!

1.8 Om A är en ändlig mängd med n element säger vi att A:s kardinalitet är n och skriver $|A| = n$. Kan man säga något om $|A \cup B|$, om $|A| = n$ och $|B| = m$?

1.9 Vi skriver A^n för $A \times A \times ... \times A$, dvs för produktmängden av n stycken mängder A. Beräkna $|\mathbf{Z}_2^{56}|$.

1.10 Visa att $|P(A)| = 2^n$, om $|A| = n$.

1.11 Låt $A = \{1, 2, 3, 4, 5\}$ och $B = \{0, 3, 6\}$. Bestäm
$A \cup B$, $A \cap B$, $A - B$, $B - A$.

1.12 Bestäm A och B, om $A - B = \{1, 5, 7, 8\}$, $B - A = \{2, 10\}$ och
$A \cap B = \{3, 6, 9\}$.

1.13 Visa att $A \subseteq A \cup B$ och $A - B \subseteq A$.

1.14 Vad kan man säga om mängderna A och B, om man vet att
a) $A \cup B = A$ b) $A \cap B = A$
c) $A - B = A$ d) $A \cap B = B \cap A$?

1.15 Visa att $A \subseteq B$ om och endast om $B^C \subseteq A^C$.

1.16 Visa att $A \subseteq C$, om $A \subseteq B$ och $B \subseteq C$.

2 Funktionsbegreppet

Funktioner används för att para ihop element i olika mängder.

Definition 2.1: En *funktion* (avbildning) f från en mängd A till en mängd B är en regel som till varje element $a \in A$ ordnar ett entydigt bestämt element $b \in B$. Detta element kallas *värdet* av a och betecknas f(a). Vi skriver $f: A \to B$ om f är en funktion från A till B. b kallas *bild* och a kallas *urbild*.

Denna definition är egentligen något oklar, och kan leda läsaren i felaktiga banor, på grund av ordet "regel". Det får inte förstås i betydelsen "algebraiskt uttryck" eller något liknande. Varje hopparning, som till varje element i A ordnar ett unikt element i B, är en funktion. Detta kan preciseras med följande variant av definitionen ovan.

Definition 2.2: $f: A \to B$ om och endast om

(i) $f \subseteq A \times B$,

(ii) $\forall a \in A\ \exists b \in B\ (\langle a, b\rangle \in f)$ och

(iii) om $\langle a, b_1\rangle \in f$ och $\langle a, b_2\rangle \in f$, så $b_1 = b_2$.

(Uttrycket "$\exists b \in B$" utläses "det finns ett b sådant att $b \in B$".)

I definitionerna ovan kallas A *definitionsmängd* eller *avsändarmängd*. B kan vi kalla *mottagarmängd*.

Om $f: A \to B$ och $S \subseteq A$, så är $f(S) = \{b \in B : \exists a \in S(b = f(a))\}$, dvs f(S) är mängden av bilder under funktionen f till element i S.

Om $f: A \to B$, kallas f(A) *värdemängden* till f.

Exempel 2.1: Låt $A = B = \mathbf{Z}_+ = \{n \in \mathbf{Z} : n > 0\}$ och låt f(n) = det n:te primtalet

Då är f(1) = 2
f(2) = 3
f(3) = 5
f(4) = 7
osv.

Hur skulle en "regel" som definierar denna funktion kunna se ut?
Definitionsmängden till funktionen f är $D_f = \mathbf{Z}_+$.
Värdemängden till funktionen f är $V_f = \{x : x \text{ är ett primtal}\}$.
$f(\{1, 2, 3\}) = \{2, 3, 5\}$.

Exempel 2.2: Vilka av följande samband definierar funktioner? A och B är tänkta att vara avsändarmängd respektive mottagarmängd. Ange i förekommande fall värdemängd.

a) $A = \mathbf{Z}_8$, $B = \mathbf{N}$, $f(n) = n^2$.

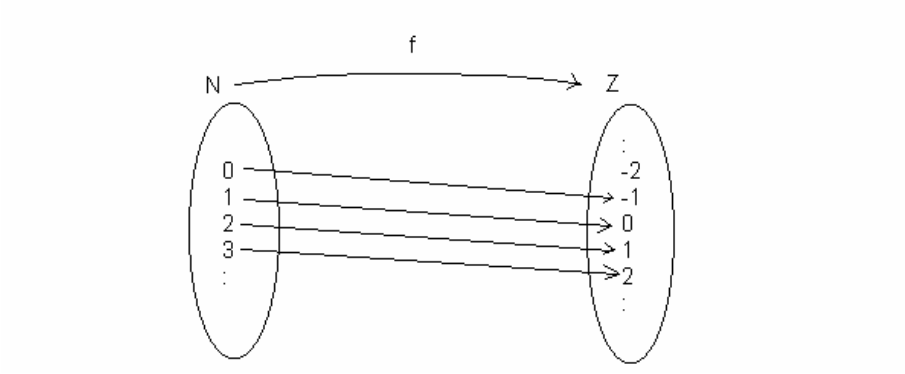
b) $A = \mathbf{N}$, $B = P(\mathbf{N})$, $f(n) = \mathbf{Z}_n$.

c) $A = P(\mathbf{N})$, $B = \mathbf{N}$, f(S) = summan av elementen i S.

d) $A = \mathbf{N}$, $B = P(\mathbf{N})$, f(n) = en delmängd till $\mathbf{N}$ med elementens summa = n.

Uttrycken i a, b och c definierar funktioner, men inte det i d, ty för t ex n = 3 gäller både att f(3) = {3} och f(3) = {1, 2}, varför bilden till 3 under f inte är unik.

Funktioner illustreras lämpligen med diagram enligt följande. Låt f: **N** $\rightarrow$ **Z** definierad av f(n) = n - 1



Vi har då en illustrativ bild av vad funktionen f "gör med" elementen i avsändarmängden.

Funktioner kan klassificeras med avseende på vilka egenskaper de har.

Definition 2.3 En funktion f: A $\rightarrow$ B är *injektiv* (en-entydig, 1-1) om och endast om $f(a_1) = f(a_2) \Rightarrow a_1 = a_2$, för alla par av element a_1 och a_2 i avsändarmängden A. (Pilen kan utläsas "medför att" eller "implicerar".)

En alternativ, ekvivalent, formulering är att $a_1 \neq a_2 \Rightarrow f(a_1) \neq f(a_2)$, dvs olika urbilder avbildas på olika bilder. I ett diagram enligt ovan betyder detta att inget element i mottagarmängden får träffas av mer än högst en pil.

Exempel 2.3 f: **N** $\rightarrow$ **Z**, $f(n) = n^2$
f är injektiv, ty antag att $n^2 = m^2$. Då måste n = m, ty D_f = **N** (rita!)

g: **Z** $\rightarrow$ **Z**, $g(n) = n^2$
g är ej injektiv, ty g(2) = g(-2) = 4, dvs två olika element i avsändarmängden avbildas på samma element i mottagarmängden.

Definition 2.4: En funktion f: A $\rightarrow$ B är *surjektiv* (på) om och endast om f(A) = B.

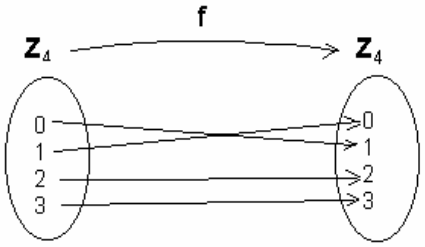
Detta innebär att varje element i mottagarmängden träffas av minst en pil.

Exempel 2.4: f: **Z** $\rightarrow$ **Z**, f(n) = n + 1.
Då är f surjektiv, ty antag att m är ett element i mottagarmängden. Då gäller att f(m - 1) = m, och m - 1 tillhör avsändarmängden för varje m $\in$ **Z**. (rita!)

Exempel 2.5: f: **N** $\rightarrow$ **N**, f(n) = n + 1 är inte surjektiv, ty 0 är inte bild av något element. Detta inser vi av att ekvationen n + 1 = 0 inte löses av något element i **N**.

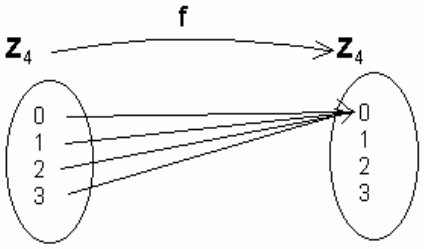
Exempel 2.6: Följande exempel är illustrativa.

a.



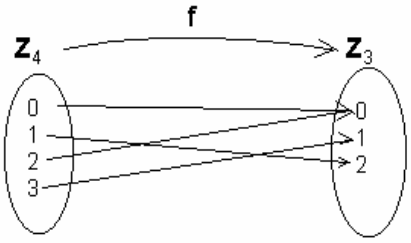
Injektiv & surjektiv

b.



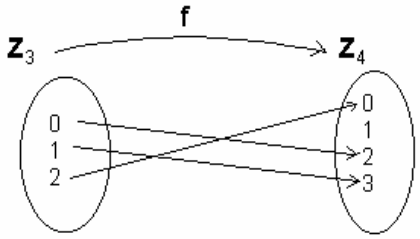
Varken injektiv eller surjektiv

c.



Surjektiv, ej injektiv

d.



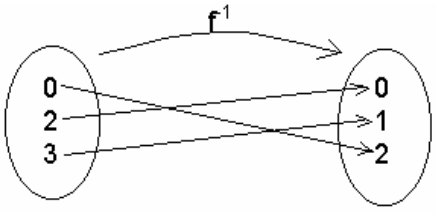
Injektiv, ej surjektiv

Om en funktion är både injektiv och surjektiv, dvs *bijektiv*, så erhåller vi en funktion även om vi "vänder på pilarna". Denna iakttagelse leder till följande definition.

Definition 2.5: Låt f vara en bijektion från A på B. *Inversen* till f, betecknad f^{-1}, är då en funktion från B på A definierad av $f^{-1}(b) = a$ om $f(a) = b$.

Exempel 2.7: I exempel 2.6 a är avbildningen i en bijektion. Då är inversen till f definierad av f^{-1}: $\mathbf{Z}_4 \to \mathbf{Z}_4$, $f^{-1}(0) = 1$, $f^{-1}(1) = 0$, $f^{-1}(2) = 2$, $f^{-1}(3) = 3$. (Rita!)

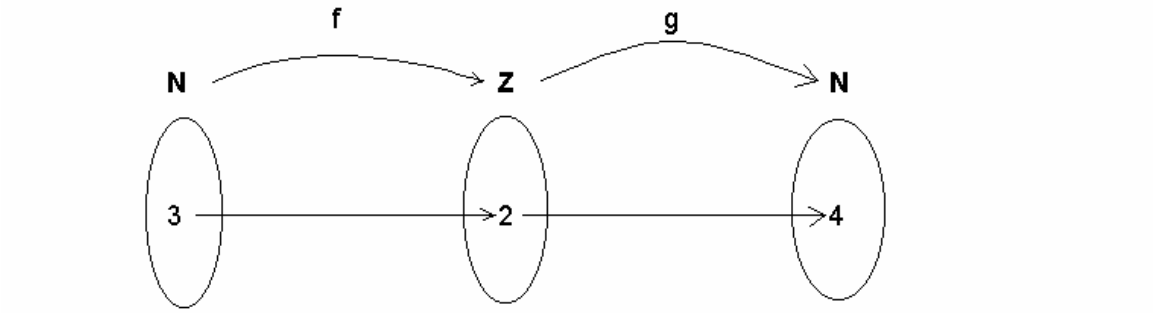
I själva verket behöver vi normalt inte bry oss om villkoret att f skall vara surjektiv, eftersom om $f: A \to B$ och f är injektiv, så är $f: A \to f(A)$ bijektiv. Betrakta exempel 2.6 d. Där gäller att $f(\mathbf{Z}_3) = \{0, 2, 3\}$. Då är $g: \mathbf{Z}_3 \to \{0, 2, 3\}$ bijektiv. Vi kan uppfatta f och g som samma funktion trots att de har olika mottagarmängd, eftersom man säger att två funktioner är *lika* om de har samma avsändarmängd, och samma funktionsvärde för varje element i avsändarmängden. Detta ger att vi sätter $f^{-1} = g^{-1}$, och kan därmed definiera f^{-1} med följande tabell.



Vi behöver också kunna sätta samman avbildningar. Antag till exempel att vi har följande två funktioner

f: $\mathbf{N} \to \mathbf{Z}$, $f(n) = n - 1$
g: $\mathbf{Z} \to \mathbf{N}$ $g(n) = n^2$.

Antag dessutom att vi först vill applicera f på ett tal och sedan g på bilden under f av detta tal. Vi kan illustrera med följande figur.



Vad vi är ute efter är en avbildning h som direkt avbildar t ex 3 på 4 enligt ovanstående regler. Det är uppenbart att i exemplet ovan $h(n) = (n - 1)^2$ är den tänkta avbildningen. Varför?

Definition 2.6: Låt f: $A \to B$ och g: $B \to C$. *Sammansättningen* av f och g, $g \circ f$: $A \to C$, definieras av $g \circ f(n) = g(f(n))$. Uttrycket $g \circ f$ läses "g ring f".

Vi har alltså från ovanstående resonemang att $h = g \circ f$ och h: $\mathbf{N} \to \mathbf{N}$.

Vi betecknar *identitetsfunktionen*, den funktion som avbildar varje element i en mängd på sig själv, med id. Det gäller då att $id(x) = x$ för varje $x \in D_{id}$. Följande samband gäller mellan en funktion och dess invers $f \circ f^{-1} = f^{-1} \circ f = id$.

Exempel 2.8: Låt $f_1, f_2, f_3 : \mathbf{Z}_2^8 \to \mathbf{Z}_2^8$ definierade av

f_1 reverserar (vänder på) en bitsträng av längd 8, t ex

$f_1(10111001) = 10011101$.

f_2 byter plats på par av intilliggande bitar, t ex

$f_2(10011101) = 01101110$.

f_3 adderar 1 bitvis till varje bit enligt $0 + 0 = 1 + 1 = 0$, $0 + 1 = 1 + 0 = 1$, t ex

$f_3(01101110) = 10010001$.

Då är $f_3 \circ f_2 \circ f_1$ en avbildning som överför t ex 10111001 på 10010001. Vad blir bilden av 00101110 under denna avbildning?

För att bestämma den inversa avbildningen $(f_3 \circ f_2 \circ f_1)^{-1}$ till $f_3 \circ f_2 \circ f_1$, noterar vi först att $(f_3 \circ f_2 \circ f_1)^{-1} = f_1^{-1} \circ f_2^{-1} \circ f_3^{-1}$ (se övning 2.8). Observera sedan att samtliga funktioner ovan är sin egen invers, dvs $f_1^{-1} = f_1$, $f_2^{-1} = f_2$ och $f_3^{-1} = f_3$.

Vi får då att $(f_3 \circ f_2 \circ f_1)^{-1} = f_1^{-1} \circ f_2^{-1} \circ f_3^{-1} = f_1 \circ f_2 \circ f_3$, och vi skulle få tillbaka ett element om vi applicerade samma avbildning i omvänd ordning. Kontrollera att detta är fallet med bitsträngarna ovan.

Övningsuppgifter

2.1 Bestäm $f \circ g$ och $g \circ f$, då $f(n) = n^2 + 1$ och $g(n) = n + 2$. Observera att $f \circ g \neq g \circ f$.

2.2 Vilka av följande funktioner från **Z** till **Z** är injektiva respektive surjektiva?
a) $f(n) = n - 1$ b) $f(n) = n^2 + 1$ c) $f(n) = n^3$.

2.3 Låt $f(x) = 2x$. Bestäm $f(\mathbf{N})$, $f(\mathbf{Z})$ och $f(\mathbf{R})$.

2.4 Låt $f: A \to B$, och låt S respektive T vara delmängder till A. Visa att $f(S \cap T) \subseteq f(S) \cap f(T)$. Ge ett exempel som visar att det inte behöver vara likhet.

2.5 Ge ett exempel på en funktion från **N** till **N** som är
a. injektiv men inte surjektiv,
b. surjektiv men inte injektiv,
c. varken injektiv eller surjektiv,
d. både injektiv och surjektiv (men skiljd från id).

2.6 Bestäm respektive värdemängd för följande funktioner från **R** till **R**.

A) $f(x) = 2x + 1$ b) $f(x) = x^2 + 1$ c) $f(x) = \dfrac{x^2+1}{x^2+2}$.

Vilka av dem är inverterbara?

2.7 Låt $f(x) = 3x + 1$ och $g(x) = 1/x$. Bestäm $f^{-1} \circ g^{-1}$ och $(g \circ f)^{-1}$.

2.8 Bevisa att $f^{-1} \circ g^{-1} = (g \circ f)^{-1}$.

3 Något om komplexitet

3.1 Algoritmers komplexitet

Med en *algoritm* förstår vi en bestämd procedur som i ett ändligt antal steg genererar utdata givet någon form av indata. Som mönsterexempel kan vi använda de vanliga algoritmerna för multiplikation och division. Typiskt för en algoritm är att ju "större" indata är enligt något sätt att mäta, desto fler operationer (additioner, multiplikationer, jämförelser etc) måste utföras. Ju större tal som multipliceras med varandra, desto fler operationer måste ju utföras. Vi kan tänka oss att antalet operationer, mätt på något väsentligt sätt, är en funktion f av storleken n hos indata. f är då normalt en positiv, växande funktion. Om en algoritm skall vara särskilt användbar, är det inte tillräckligt att utdata genereras efter ett ändligt antal steg. Det är inte säkert att vi är beredda att vänta t ex 10^6 år på ett resultat. Vi behöver därför en teknik att mäta algoritmers *komplexitet*. Man brukar skilja mellan *tidskomplexitet* - hur lång tid en algoritm tar mätt t ex som antalet operationer som krävs, och *rumskomplexitet* - hur mycket minne, papper etc som åtgår. Nedan skall vi koncentrera oss på tidskomplexitet, och skall därför skaffa oss ett sätt att mäta hur snabbt funktioner växer.

Definition 3.1: Låt f och g vara två positiva funktioner (dvs två funktioner med positiva funktionsvärden) definierade på $A \subseteq \mathbf{N}$. Vi säger att f är *stort ordo* g, $f(n) = O(g(n))$, om det finns två tal c och K så att $f(n) \leq c \cdot g(n)$ om $n > K$.

I definitionen är f tänkt att vara en funktion som mäter någon algoritms komplexitet, och g en jämförelsefunktion. Observera att "=" i uttrycket "$f(n) = O(g(n))$" inte är ett vanligt likhetstecken, utan har en helt annan betydelse, som är fastlagd i definitionen.

Exempel 3.1: Låt $f(n) = 17n + 384$. Då gäller att $f(n) = O(n)$, ty $17n + 384 \leq 18n$, om $n > 400$. Vi har satt $c = 18$ och $K = 400$ i definitionen.
Det gäller också att $f(n) = O(2n)$. Välj t ex $c = 9$ och $K = 400$.

Exemplet ovan ger alltså att stortordouppskattningar inte är entydigt bestämda! Vi vill förstås ha så bra uppskattningar som möjligt.

Exempel 3.2: Låt $f(n) = 3n^2 + 1000$. Då gäller $f(n) = O(n^2)$, ty $3n^2 + 1000 \leq 4n^2$ bara $n > 50$.

I dessa stortordouppskattningar utnyttjar vi en uppsättning standardjämförelsefunktioner, och använder följande terminologi.

g(n)	f:s komplexitet
1	Konstant
log n	Logaritmisk
n	Linjär
n^2	Kvadratisk
n^3	Kubisk
n^p, $p > 0$	Polynomiell
2^n	Exponentiell

Tidsåtgången i sekunder vid olika inputstorlek, om vi förutsätter 10^9 operationer per sekund illustreras i följande tabell.

inputstorlek	komplexitet			
n	n	n^2	n^3	2^n
10^2	10^{-7}	10^{-5}	10^{-3}	10^{14} år
10^4	10^{-5}	10^{-1}	10^3	$*$
10^6	10^{-3}	10^3	33 år	$*$

"$*$" betecknar att beräkningen tar väldigt lång tid.

3.2 Problemkomplexitet

Ett givet, lösbart problem kan lösas av många olika algoritmer, som kan vara mer eller mindre effektiva vad gäller tidsåtgång eller minnesutnyttjande. I *Komplexitetsteori* klassificerar man problem efter den tids- eller rumsåtgång som krävs för att lösa problemet. Vad detta innebär kan definieras precist, men vi avstår från detta här, och hänvisar den intresserade läsaren till exempelvis [Bovet] (se litteraturlistan). Problem som är lösbara inom polynomtid kallas *hanterbara* (tractable), eftersom de kan lösas för rimligt stora indatamängder. Problem som inte kan lösas systematiskt inom polynomtid kallas *ohanterbara* (intractable), eller *svåra*, eftersom de är omöjliga att lösa inom rimlig tid.

Klassen **P** består av alla problem, som är lösbara inom polynomtid. Klassen **NP** (icke-deterministisk polynomiell) innehåller de problem, som är sådana att en gissad lösning kan verifieras inom polynomtid. Vi har emellertid ingen garanti för vi medelst gissningar verkligen kan hitta en lösning inom polynomtid. För att systematiskt lösa vissa problem i **NP** verkar det åtgå exponentiell tid.

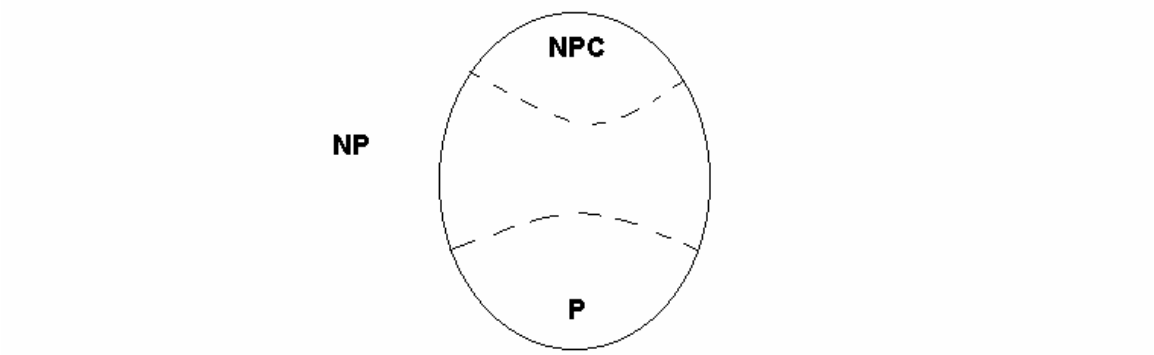
Exempel 3.3: Ett exempel på ett problem, som förmodligen hör till **NP** - **P** är "the knapsack problem". Givet en mängd av n heltal $A = \{a_1, a_2, \ldots, a_n\}$ och ett givet heltal S, skall man avgöra huruvida det finns en delmängd till A vars element har summan S. Observera att A har 2^n delmängder (övn 1.10). Det är lätt att kontrollera om elementen i en given delmängd till A har summan S, dvs det är lätt att kontrollera att en gissad lösning verkligen är en lösning, men det kan vara svårt att hitta en dylik delmängd. De bästa algoritmerna för att lösa detta problem har tidskomplexitet $O(2^{n/2})$ och rumskomplexitet $O(2^{n/4})$.

Exempel 3.4: Problemet att multiplicera två naturliga tal hör till komplexitetsklassen **P**. Varför är det så?

Exempel 3.5: Ett annat viktigt exempel på ett problem som verkar ha exponentiell tidskomplexitet är problemet att avgöra huruvida en given satslogisk formel satsifieras av någon sanningsvärdestilldelning. Detta problem kallas satisfierbarhetsproblemet. Om en satslogisk formel innehåller n satsparametrar, så finns 2^n möjliga sanningsvärdestilldelningar. Ingen algoritm är känd som kan lösa detta problem inom polynomtid. Man tror att problemet hör till **NP** - **P**.

Uppenbart är **P** en delklass till **NP**. Om det dessutom vore så att alla **NP**-problem vore lösbara inom polynomtid, skulle det gälla att **P** = **NP**, trots att många problem i klassen **NP** verkar "svårare". Det är okänt huruvida $\mathbf{P} \neq \mathbf{NP}$ eller **P** = **NP**. Detta problem är ett av den moderna matematikens viktigaste, ännu olösta, problem.

Det gäller att satisfierbarhetsproblemet har egenskapen att alla andra **NP**-problem kan reduceras, översättas, till detta inom polynomtid. Detta betyder att om satisfierbarhets-problemet är lösbart inom polynomtid, så är varje problem i **NP** lösbart inom polynomtid, eller, annorlunda uttryckt, om något **NP**-problem är ohanterbart, så är också satisfier-barhetsproblemet ohanterbart. De problem (inklusive "the knapsack problem"), som är ekvivalenta med satisfierbarhetsproblemet i ovannämnda mening, kallas **NP**-fullständiga problem (**NPC**). Dessa problem har egenskapen att om ett av dem är i **P**, så är alla **NP**-problem i **P**, dvs **P = NP**. De **NP**-fullständiga problemen är de "svåraste" i **NP**. De snabbaste algoritmer vi känner för att systematiskt lösa dessa problem har exponentiell "worst-case"-komplexitet.
Nedanstående figur illustrerar hur några komplexitetsklasser förmodligen är relaterade till varandra



Exempel 3.6: Följande två besläktade problem tros vara utanför både **P** och **NPC**, men i **NP**. Det ena är primtalsproblemet; avgör huruvida ett givet heltal n är ett primtal, och det andra är problemet att avgöra huruvida ett givet heltal är sammansatt (se nedan s 19 för definition av begrepp). Observera att det kan vara mycket svårare att hitta faktorerna i ett sammansatt tal än att avgöra huruvida talet i fråga är sammansatt.

Att dessa problem förmodligen ligger utanför **P** är av synnerlig vikt för det så kallade RSA-systemet (nedan avsnitt 9).

Vi avslutar med en kort kommentar om två olika typer av funktioner, som är intressanta inom kryptografin.

Definition 3.2: En funktion $f: A \rightarrow B$ är en *envägsfunktion* (one-way function) om och endast om $f^{-1}(y)$ är "svår" att beräkna för nästan alla $y \in B$, medan det är "lätt" att beräkna $f(x)$ för alla $x \in A$.

Definition 3.3: En funktion $f: A \rightarrow B$ är en *envägsfunktion med fallucka* (trapdoor one-way function) om det är "lätt" att beräkna f^{-1} givet någon ytterligare information.

Denna extra information, som krävs i ovanstående definition, är inom kryptografi förstås en hemlig nyckel. Man vet inte om det verkligen finns äkta envägsfunktioner med fallucka. Kandidater för dylika ges av "knapsack"-problemet och faktoriseringsproblemet.

Övningsuppgifter

3.1 Visa att $2x^3 + x^2 + 7$ är stort ordo x^3.

3.2 Vad betyder det att en funktion är $O(1)$?

3.3 Antag att $f_1(x) = O(g_1(x))$ och $f_2(x) = O(g_2(x))$. Visa att $f_1(x) + f_2(x) = O(\max(g_1(x), g_2(x)))$.

3.4 Visa att $\sum_{i=1}^{n} i^k = O(n^{k+1})$.

3.5 Vilken tidskomplexitet har den vanliga algoritmen för att multiplicera två n-siffriga tal i det decimala systemet. Mät tiden först i hur många additioner och multiplikationer med ensffriga tal vi måste utföra, och sedan i hur många multiplikationer vi behöver utföra. Är det någon skillnad?
Det finns snabbare algoritmer för detta. Se till exempel [Rosen 1995].

3.6 Låt mängden A i exempel 3.3 vara A = {17, 5, 231, 52, 11, 28, 211, 58, 92, 3, 68, 301, 158, 12, 76, 16} och låt S = 1032. Hitta en delmängd till A vars element har summan S.

4 Euklides´ algoritm

4.1 Divisionsalgoritmen

Exempel 4.1: Om vi utgår från två heltal, t ex 38 och 7, så gäller 38/7 = 5 + 3/7, eller 38 = 5·7 + 3, där 5 kallas *kvot* och 3 *rest*. Resten kan alltid väljas så att 0 ≤ resten < nämnaren.

Allmänt formuleras detta i följande sats.

Sats 4.1: Om a och b är heltal med $a \neq 0$, så finns entydigt bestämda heltal q och r sådana att $b = qa + r$, där $0 \leq r < |a|$.

Bevis: Sätt $r = b - qa$ för olika heltal q. Då är r ett heltal. För olika värden på q erhålls följande möjliga r.

$$\ldots , b - (-2)a, b - (-1)a, b, b - a, b - 2a, \ldots$$

Som synes är avståndet mellan två rester |a|. Alltså finns exakt en rest r bland talen 0, 1, 2, ... , |a| -1, dvs detta r satisfierar $0 \leq r < |a|$.

Exempel 4.2: Med b = 351 och a = 28 får vi
351 = 13·28 + (-13) = 12·28 + 15 = 11·28 + 43 = 10·28 + 71 osv.
Vi väljer den minsta ickenegativa resten r = 15, vilket ger q = 12.

4.2 Delare

Definition 4.1: Heltalet a kallas *delare* till heltalet b om det finns ett heltal q så att b = qa. Vi skriver a|b, om a är delare till b, och $a \nmid b$, om a inte är delare till b.

Vi säger också att a är *faktor* i b, att b är *(jämnt) delbart* med a och att b är en *multipel* av a.

Exempel 4.3: 5|60, ty 60 = 5·12.
$5 \nmid 61$, ty det finns inget heltal som multiplicerat med 5 ger 61.

Varje heltal b har de *triviala delarna* ±1 och ±b. Övriga delare kallas *äkta*. Vi anger några egenskaper hos relationen att vara delare i följande sats.

Sats 4.2: Låt a, b och c vara heltal. Då gäller
(i) Om a|b och b|c, så a|c.
(ii) Om a|b och a|c, så a|xb + yc för alla heltal x och y.
(iii) Om a|b, b|a och $a \neq 0 \neq b$, så är $a = \pm b$. Om både a och b är positiva så gäller a = b.

Bevis (i) Antag att a|b och b|c.
Då finns heltal k_1 och k_2 så att $b = a\,k_1$ och $c = b\,k_2$. Detta ger att $c = b\,k_2 = (a\,k_1)\,k_2 = a(k_1\,k_2)$, och vi har hittat ett heltal $k = k_1\,k_2$ sådant att c = ak. Då gäller a|c.

(ii) Antag att $a|b$ och $a|c$.
Då finns heltal k_1 och k_2 så att $b = a\,k_1$ och $c = a\,k_2$. För godtyckliga heltal x och y gäller då att
$xb + yc = x(a\,k_1) + y(a\,k_2) = a(x\,k_1 + y\,k_2)$, och vi har hittat ett heltal som multiplicerat med a ger $xb + yc$.
Alltså gäller $a|xb + yc$ för alla heltal x och y.

(iii) Antag att $a|b$ och $b|a$.
Då finns som vanligt heltal k_1 och k_2 så att $b = a\,k_1$ och $a = b\,k_2$. Vi får då att $b = a\,k_1 = (b\,k_2)\,k_1 = b(k_2k_1)$. Eftersom det enligt förutsättning gäller att $b \neq 0$, har vi att $k_2\,k_1 = 1$.
Detta ger då att $k_1 = k_2 = \pm 1$, eftersom k_1 och k_2 är heltal. Då gäller $a = \pm b$. Om a och b dessutom båda är positiva, måste $k_1 = k_2 = 1$, vilket ger att $a = b$.

Exempel 4.4: $3|6$ och $3|9$ ger att $3|2\cdot 6 + (-4)\cdot 9$, som förstås stämmer eftersom $2\cdot 6 + (-4)\cdot 9 = -24 = (-8)\cdot 3$.

4.3 Största gemensamma delare

Definition 4.2: Det största heltal d som delar både a och b kallas den *största gemensamma delaren* till a och b, och betecknas $\gcd(a, b)$. Här får inte både a och b vara noll.

Exempel 4.5: $\gcd(38, 27) = 1$, ty talen saknar gemensamma faktorer.
$\gcd(16, 12) = 4$, ty $4|12$, $4|16$ och inget tal större än fyra delar både 12 och 16.
$\gcd(-21, 14) = 7$, $\gcd(17, 0) = 17$.
Observera att $\gcd(0, 0)$ ej är definierat.

Definition 4.3: Om $\gcd(a, b) = 1$, sägs a och b vara *relativt prima.*

Exempel 4.6: $\gcd(-11, 8) = 1$, $\gcd(6, 35) = 1$.

4.4 Euklides´ algoritm för hela tal

Euklides´ algoritm är mycket viktig och ger oss en metod att enkelt beräkna största gemensamma delaren till två givna tal. Motsvarande teknik kan även användas för att hitta gemensamma faktorer till två polynom. Vi inleder med ett exempel innan vi teoretiskt bevisar att algoritmen gör det vi förväntar oss av den.

Exempel 4.7: Vi skall bestämma $\gcd(1848, 4914)$. Med hjälp av divisionsalgoritmen genererar vi följande räkneschema:

$$4914 = 2\cdot 1848 + 1218$$
$$1848 = 1\cdot 1218 + 630$$
$$1218 = 1\cdot 630 + 588$$
$$630 = 1\cdot 588 + 42$$
$$588 = 14\cdot 42$$

Den sista icke försvinnande resten är 42.
Då är $\gcd(1848, 4914) = 42$.

Allmänt gäller följande. Antag att vi söker gcd(a, b), där a inte är delare till b, och b inte är delare till a. Om a|b (eller b|a) så är problemet trivialt, eftersom det då gäller att gcd(a, b) = |a|. Vi antar godtyckligt att a < b och arrangerar räkningarna som följer.

(1)	$b = q_1 a + r_1$	där	$0 < r_1 < \|a\|$
(2)	$a = q_2 r_1 + r_2$		$0 < r_2 < r_1$
(3)	$r_1 = q_3 r_2 + r_3$		$0 < r_3 < r_2$
⋮	⋮		⋮
(n)	$r_{n-2} = q_n r_{n-1} + r_n$		$0 < r_n < r_{n-1}$
(n+1)	$r_{n-1} = q_{n+1} r_n$		

Eftersom resterna $r_1, r_2, ..., r_n$ bildar en strängt avtagande följd av positiva heltal, måste det finnas en sista positiv rest, och vi formulerar följande sats.

Sats 4.3: Den sista icke försvinnande resten i Euklides´ algoritm är lika med den största gemensamma delaren till a och b, dvs $\gcd(a, b) = r_n$.

Innan vi bevisar detta resultat skall vi formulera och bevisa följande lemma. Ett lemma är en hjälpsats, ett resultat som används i beviset av en sats.

Lemma 4.4: Om $a = qb + r$, så $\gcd(a, b) = \gcd(b, r)$. Här är a, b, q och r heltal som satisfierar villkoren för att respektive största gemensamma delare skall vara väldefinerade.

Bevis: Sätt $\gcd(a, b) = d_1$ och $\gcd(b, r) = d_2$.
Då gäller att $d_1|a$ och $d_1|b$. Enligt sats 4.2 måste då gälla att $d_1|a - bq$, dvs $d_1|r$. Då gäller att d_1 delar både b och r. Men eftersom d_2 är den största gemensamma delaren till b och r, så måste $d_1 \leq d_2$ (observera att största gemensamma delaren till ett tal alltid är positiv).
Det gäller dessutom att $d_2|b$ och $d_2|r$, varför d_2 måste dela $a = bq + r$. Vi får då att $d_2 \leq d_1$, eftersom $d_1 = \gcd(a, b)$. Alltså måste det gälla att $d_1 = d_2$.

Med hjälp av detta lemma bevisar vi nu enkelt sats 4.3.

Bevis av sats 4.3: Av lemmat följer $\gcd(b, a) = \gcd(a, r_1) = \gcd(r_1, r_2) = ... = \gcd(r_{n-1}, r_n) = \gcd(r_n, 0) = r_n$.

I handböcker finns snabbare varianter av Euklides´ algoritm. Den är utomordentligt viktig i många tillämpningar, t ex inom kryptografi.

Exempel 4.8: Beräkna gcd(189, 336)

Euklides´ algoritm ger

$336 = 1 \cdot 189 + 147$
$189 = 1 \cdot 147 + 42$
$147 = 3 \cdot 42 + 21$
$42 = 2 \cdot 21$

Den sista icke försvinnande resten är 21.
Alltså är gcd(189, 336) = 21.

4.5 Den diofantiska ekvationen ax + by = c

Ekvationer där koefficienter och variabler endast får anta heltalsvärden brukar kallas diofantiska. Vi begränsar oss till typen i rubriken, och inleder med ett exempel.

Exempel 4.9: Hitta en lösning till 336x + 189y = 21. Jämför med föregående exempel! Euklides´ algoritm "baklänges" ger

21 = 147 - 3·42 = 147 - 3(189 - 147) = 4·147 - 3·189 = 4(336 - 189) - 3·189 =
= 4·336 - 7·189 = 336·4 + 189·(-7)

Sammanfattar vi har vi alltså fått att

$$336 \cdot 4 + 189 \cdot (-7) = 21,$$

dvs x = 4, y = -7 är en lösning till ekvationen. Det finns många fler.

Sats 4.5: Ekvationen ax + by = gcd(a, b) är lösbar.

Bevis: Satsen är trivial, om a|b eller om b|a. (Varför?)
Om så icke är fallet kan vi hitta en lösning med hjälp av Euklides´ algoritm som vi skisserat ovan.

Sats 4.6: Ekvationen ax + by = c är lösbar om och endast om gcd(a, b)|c.

Bevis: $\Rightarrow$/ Antag att ekvationen är lösbar. Då finns heltal x_0 och y_0 som satisfierar ekvationen, dvs $a x_0 + b y_0 = c$. Det gäller att gcd(a, b)|a och gcd(a, b)|b, varför gcd(a, b)|$a x_0 + b y_0$ (sats 4.2), dvs gcd(a, b)|c.

$\Leftarrow$/ Antag att gcd(a, b)|c. Då finns heltal q så att c = q·gcd(a, b).
Sats 4.5 ger att det finns heltal m och n så att am + bn = gcd(a, b).
Multiplikation med q ger a(qm) + b(qn) = q·gcd(a, b) = c, dvs x = qm och y = qn löser ekvationen, som alltså är lösbar.

Korollarium 4.7:Ekvationen ax + by = 1 är lösbar om och endast om gcd(a, b) = 1.

Ett korollarium är en följdsats som direkt följer av en tidigare sats. Vi lämnar slutligen följande generella resultat, som vi avstår från att bevisa.

Sats 4.8: Om $x = x_0$ och $y = y_0$ är en lösning till ax + by = c, så är för ett godtyckligt heltal n

$$x = x_0 + n \cdot \frac{b}{\gcd(a,b)} \quad \text{och} \quad y = y_0 - n \cdot \frac{a}{\gcd(a,b)} \quad \text{också en lösning,}$$

och varje lösning kan framställas på denna form.

Vi avslutar så med några exempel.

Exempel 4.10: Lös ekvationen 1056x + 627y = 165.

Euklides´ algoritm ger

1056 = 1·627 + 429

$627 = 1\cdot429 + 198$
$429 = 2\cdot198 + 33$
$198 = 6\cdot33$

Den sista icke försvinnande resten är 33, och alltså är gcd(1056, 627) = 33. Då är ekvationen lösbar, ty $33|165$. Vi börjar med att hitta en lösning med hjälp av Euklides´ algoritm. Algoritmen baklänges ger

$33 = 429 - 2\cdot198 = 429 - 2(627 - 1\cdot429) = -2\cdot627 + 3\cdot429 =$
$= -2\cdot627 + 3(1056 - 1\cdot627) = 3\cdot1056 - 5\cdot627 = 1056\cdot3 + 627\cdot(-5).$

Vi har alltså att $1056\cdot3 + 627\cdot(-5) = 33$.

Multiplikation med 5 ger $1056\cdot15 + 627\cdot(-25) = 165$,
och vi ser att $x = 15$ och $y = -25$ är en lösning.

Allmänt får vi av föregående sats följande lösning.

$$x = 15 + n\cdot\frac{627}{33} = 15 + 19n \text{ och } y = -25 - n\cdot\frac{1056}{33} = -25 - 32n,$$

där n är ett godtyckligt heltal.

Exempel 4.11: Lös ekvationen $5x + 15y = 1$.

Här gäller att gcd(5, 15) = 5. Då saknar ekvationen lösning, ty 5 är inte delare till 1.

4.6 Något om primtal

Definition 4.4: Ett heltal p är ett *primtal*, om $p > 1$, och p inte har några andra delare än ± 1 och $\pm p$. Ett heltal är *sammansatt*, om det inte är ett primtal.

Exempel 4.12: De första primtalen är 2, 3, 5, 7, 11, ...
569287 är sammansatt, ty $569287 = 997\cdot571$.

Primtalen är de hela talens "byggstenar" och är av fundamental betydelse för talteorin. Vi formulerar avslutningsvis *aritmetikens fundamentalsats*, som var känd redan under antiken.

Sats 4.9 Varje positivt heltal kan, bortsett från ordningsföljden, skrivas som en produkt av primtal på ett och endast ett sätt.

Exempel 4.13: $2100 = 2^2\cdot3\cdot5^2\cdot7$.

Nedanstående berömda resultat bevisades redan av Euklides.

Sats 4.10: Det finns oändligt många primtal.

Bevis: Antag motsatsen, dvs att det bara finns ett ändligt antal primtal, och låt dessa vara $p_1, p_2, \ldots, p_n$, där vi godtyckligt kan anta att p_n är det största. Vi bildar sedan $a = p_1\cdot p_2\cdot \ldots \cdot p_n + 1$. Då gäller det att $a > p_n$.

Det finns nu två möjligheter

(i) a är ett primtal.

Detta motsäger att p_n är det största primtalet.

(ii) a är sammansatt.

Då kan inget av talen $p_1, p_2, \dots, p_n$ dela a, ty om så vore fallet, så skulle något av dem, p_i säg, dela $a - p_1 \cdot p_2 \cdot \dots \cdot p_n = 1$, eftersom p_i då delar både a och $p_1 \cdot p_2 \cdot \dots \cdot p_n$. Detta är orimligt, eftersom $p_i > 1$. Detta innebär då att a inte delas av något primtal, men då måste a själv vara ett primtal, eller innehålla primtalsfaktorer som är större än p_n, vilket motsäger våra förutsättningar.

Båda möjligheterna leder sålunda till motsägelse, varför vårt ursprungliga antagande måste vara felaktigt. Det finns alltså oändligt många primtal.

Övningsuppgifter

4.1 Bestäm samtliga positva delare till 28.

4.2 Undersök om 414 och 331 är relativt prima.

4.3 Bestäm med hjälp av Euklides´ algoritm

a. gcd(111, 201) b. gcd(1001, 1331)
c. gcd(12345, 54321)

4.4 Hitta en lösning till följande diofantiska ekvationer. Undersök först om respektive ekvation är lösbar.

a. 707x + 1155y = 8603
b. 36x + 45 y = 1
c. 318x - 13y = 1
d. 858x + 847y = 176

4.5 Formulera den allmänna lösningen för de diofantiska ekvationerna i uppgift 4.4.

4.6 Vad blir minsta icke-negativa rest och tillhörande kvot då

a. 19 delas med 7,
b. 1001 delas med 13?

4.7 Dela upp följande tal i primtalsfaktorer

a.	101	b.	143	c.	189
d.	899	e.	7429	f.	1328717
g.	99991	h*.	527253443		

4.8 Bestäm primtalsuppdelningen av 10!.

4.9 Bevisa att om $a|b$ och $b|c$, så $a|c$.

4.10* Talet 28 är *perfekt*. Detta innebär att summan av de positiva delarna, bortsett från 28, är just 28 (jämför uppgift 4.1). Bestäm nästa perfekta tal.

4.11 Visa att om $a|b$ och $a|c$, så $a|b + c$.

5 Kongruensaritmetik

Alla tal i följande avsnitt är heltal, om inget annat utsägs.

Definition 5.1: a *är kongruent med* b *modulo* m om och endast om m är delare till a - b.

Att a är kongruent med b modulo m skrivs med symboler $a \equiv b(\text{mod } m)$. Om modulen framgår av sammanhanget skriver vi ibland enbart $a \equiv b$.

Exempel 5.1: $21 \equiv 0(\text{mod } 7)$, ty $7|21 - 0$.
$23 \equiv 16(\text{mod } 7)$, ty $7|23 - 16$.

En alternativ definition skulle vara $a \equiv b(\text{mod } m)$ om och endast om a och b har samma minsta ickenegativa rest vid division med m, där vi förutsätter att $m > 0$. Vi såg tidigare, när vi behandlade divisionsalgoritmen ovan, att det finns en entydig rest r, där $0 \leq r < m$, vid division av a med m. De tal som är möjliga rester bildar en mängd. Denna mängd av rester kallar vi mängden av *minsta ickenegativa rester modulo m*. I denna mängd tillåts bara tal som inte är kongruenta med något annat tal i mängden. En rest från denna mängd betecknas a mod m, eller $[a]_m$. Observera att denna funktion definieras annorlunda i vissa programmeringsspråk. Vi formulerar som en sats att de två definitionerna är ekvivalenta.

Sats 5.1: Det gäller att $a \equiv b(\text{mod } m)$ om och endast om $a \bmod m = b \bmod m$.

Bevis: Övning

Detta ger att vi fritt kan tala om kongruenser eller rester.

Exempel 5.2: $23 \bmod 7 = 2$ (observera att det inte är 16, se ex 5.1)
$[251]_2 = [1]_2$.

Om vi dividerar ett heltal med 7 enligt divisionsalgoritmen kan vi få någon av resterna 0, 1, 2, 3, 4, 5 eller 6. Denna mängd kallar vi enligt ovan för mängden av minsta ickenegativa rester modulo 7. Andra mängder av inkongruenta rester är {7, 8, 9, 10, 11, 12, 13}, {-3, -2, -1, 0, 1, 2, 3} och {-3, 7, 1, 13, 9, 38, -11} etc. Kontrollera att resterna i den sista mängden är parvis inkongruenta modulo 7!

5.1 Räkneregler för kongruenser

Sats 5.2: Om $a \equiv b(\text{mod } m)$, så gäller följande för alla heltal x att

(i) $a + x \equiv b + x(\text{mod } m)$
(ii) $ax \equiv bx(\text{mod } m)$

Bevis: (i) Enligt definition av kongruens, gäller
$a \equiv b(\text{mod } m) \Leftrightarrow \exists k(a - b = km)$.
Vi får då $km = a - b = a + x - x - b = (a + x) - (b + x)$, som säger att
$a + x \equiv b + x(\text{mod } m)$.

(ii) Övning

Notera att (i) ger tillgång även till subtraktion, eftersom $a - b = a + (-b)$. Division är emellertid mer komplicerat. Så gäller t ex inte strykningslagen generellt. Den gäller dock om ett visst villkor är uppfyllt.

Sats 5.3: Antag att $\gcd(x, m) = 1$. Då gäller att $ax \equiv bx \pmod m \Rightarrow a \equiv b \pmod m$.

Bevis: $ax \equiv bx \pmod m \Rightarrow \exists k(ax - bx = km)$.
För detta k gäller då att $x(a - b) = km$.
Eftersom x och m är relativt prima enligt förutsättning, så måste x vara faktor i k (se övn 5.7).
Då finns heltal l så att $k = xl$, vilket ger att $a - b = lm$.
Alltså gäller det att $a \equiv b \pmod m$.

Exempel 5.4: Det gäller att $13 \equiv 2 \pmod{11}$. Då måste $13 + 3 \equiv 2 + 3 \pmod{11}$.
Om $x + 2 \equiv 5 \pmod 7$, så $x + 2 - 2 \equiv 5 - 2 \pmod 7$, vilket ger att $x \equiv 3 \pmod 7$.

$4x \equiv 14 \pmod 5$ ger att $2x \equiv 7 \pmod 5$, eftersom $\gcd(2, 5) = 1$.

5.2 Räkning med rester

Vi har följande räkneregler för rester

Sats 5.4: Låt $a_1 \bmod m = r_1$ och $a_2 \bmod m = r_2$. Då gäller
(i) $(a_1 + a_2) \bmod m = (r_1 + r_2) \bmod m$,
(ii) $(a_1 \cdot a_2) \bmod m = (r_1 \cdot r_2) \bmod m$.

Bevis: Förutsättningen innebär att $a_1 = mq_1 + r_1$ för något heltal q_1, och $a_2 = mq_2 + r_2$ för något heltal q_2.

(i) I kraft av sats 5.1 har vi att bevisa att $a_1 + a_2 \equiv r_1 + r_2 \pmod m$.
Eftersom $a_1 + a_2 - (r_1 + r_2) = (m q_1 + r_1) + (m q_2 + r_2) - (r_1 + r_2) = m(q_1 + q_2)$, måste $m \mid a_1 + a_2 - (r_1 + r_2)$, dvs $a_1 + a_2 \equiv r_1 + r_2 \pmod m$.

(ii) Vi har att visa att $a_1 \cdot a_2 \equiv r_1 \cdot r_2 \pmod m$.
Då $a_1 \cdot a_2 - r_1 \cdot r_2 = (m q_1 + r_1)(m q_2 + r_2) - r_1 \cdot r_2 =$
$= m^2 q_1 q_2 + m q_1 r_2 + r_1 m q_2 + r_1 r_2 - r_1 r_2 = m(q_1 q_2 + q_1 r_2 + r_1 q_2)$, måste
$m \mid a_1 \cdot a_2 - r_1 \cdot r_2$, dvs $a_1 \cdot a_2 \equiv r_1 \cdot r_2 \pmod m$.

Exempel 5.5: Bestäm den minsta ickenegativa resten modulo sex då
$b = 934 \cdot 8975 + 517 \cdot 7192 - 697$ divideras med sex.

Det gäller att $934 \bmod 6 = 4$, $8975 \bmod 6 = 5$, $517 \bmod 6 = 1$,
$7192 \bmod 6 = 4$, $697 \bmod 6 = 1$.
Vi får då $b \bmod 6 = (4 \cdot 5 + 1 \cdot 4 - 1) \bmod 6 = 23 \bmod 6 = 5$.

Exempel 5.6: Beräkna $28^{52} \bmod 15$.

Vi får $28^{52} \bmod 15 = (-2)^{52} \bmod 15 = 2^{52} \bmod 15 = (2^4)^{13} \bmod 15 =$
$= 16^{13} \bmod 15 = 1^{13} \bmod 15 = 1$.

Observera att det följer av sats 5.4 (ii) att $a^n \bmod m = (a \bmod m)^n \bmod m$. Det gäller dock inte allmänt att $a^{n \bmod m} \bmod m = a^n \bmod m$. Så är t ex $2^5 \bmod 3 = 32 \bmod 3 = 2$, men

$2^{5 \bmod 3} \bmod 3 = 2^2 \bmod 3 = 4 \bmod 3 = 1$. Detta beror på att exponenten räknar antal faktorer vi har i basen.

Vitsen vid räkning med rester är att samma resultat erhålles om man först beräknar uttrycket och därefter bestämmer resten, som om varje i uttrycket ingående tal, dock ej exponenter, reduceras till minsta rest innan respektive operation utförs. Centralt nedan kommer att vara att smidigt kunna beräkna rester för potenser med stor bas och stor exponent. Vi skall därför beskriva en algoritm som åstadkommer detta.

5.2 Modulär exponentiering medelst upprepad kvadrering

Vi inleder med ett exempel.

Exempel 5.7: Beräkna $3^{19} \bmod 7$. Vi skriver först talet 19 på binär form.

$19 = 16 + 2 + 1 = 1 \cdot 16 + 0 \cdot 8 + 0 \cdot 4 + 1 \cdot 2 + 1 \cdot 1 = (10011)_2$.
Då är $3^{19} = 3^1 \cdot 3^2 \cdot 3^{16}$, vilket vi ju kunde se omedelbart, men vi behöver binärformframställningen av exponenten i algoritmen.
Vi beräknar nu rester för kvadrater till potenser av tre.

$3^1 \bmod 7 = 3$
$3^2 \bmod 7 = 9 \bmod 7 = 2$
$3^4 \bmod 7 = 9^2 \bmod 7 = 2^2 \bmod 7 = 4$
$3^8 \bmod 7 = 4^2 \bmod 7 = 16 \bmod 7 = 2$
$3^{16} \bmod 7 = 2^2 \bmod 7 = 4$

Vi får slutligen $(3^1 \cdot 3^2 \cdot 3^{16}) \bmod 7 = (3 \cdot 2 \cdot 4) \bmod 7 = 24 \bmod 7 = 3$.

Vi formulerar så själva algoritmen för beräkning av $a^n \bmod m$.

1. Bestäm först binär representation av exponenten n, dvs

$$n = n_0 \cdot 1 + n_1 \cdot 2 + n_2 \cdot 2^n + \ldots + n_j \cdot 2^j.$$

2. Initiera variabeln r, som skall innehålla slutresultatet, $r = 1$.

3. Om $n_0 = 1$, så sätt $r = (r \cdot a) \bmod m$
Om $n_0 = 0$, så sätt $r = r$
Beräkna $a_1 = a^2 \bmod m$

4. Om $n_1 = 1$, så sätt $r = (r \cdot a_1) \bmod m$
Om $n_1 = 0$, så sätt $r = r$
Beräkna $a_2 = a_1^2 \bmod m$

⋮

k. Om $n_j = 1$, så sätt $r = (r \cdot a_j) \bmod m$
Om $n_j = 0$, så sätt $r = r$ (Detta skall inte inträffa, varför?)

Då är $r = a^n \bmod m$.

Exempel 5.8: Beräkna $1023^{91} \bmod 337$.

Vi noterar inledningsvis att $1023 \bmod 337 = 12$, varför det gäller att $1023^{91} \bmod 337 = 12^{91} \bmod 337$.

$91 = 64 + 16 + 8 + 2 + 1 = (1011011)_2$.

Sätt $r = 1$

$n_0 = 1$ Sätt $r = 12$
$a_1 = 12^2 \bmod 337 = 144$

$n_1 = 1$ $r = (12 \cdot 144) \bmod 337 = 43$
$a_2 = 144^2 \bmod 337 = 179$

$n_2 = 0$ $r = 43$
$a_3 = 179^2 \bmod 337 = 26$

$n_3 = 1$ $r = (43 \cdot 26) \bmod 337 = 107$
$a_4 = 26^2 \bmod 337 = 2$

$n_4 = 1$ $r = (107 \cdot 2) \bmod 337 = 214$
$a_5 = 2^2 \bmod 337 = 4$

$n_5 = 0$ $r = 214$
$a_6 = 4^2 \bmod 337 = 16$

$n_6 = 1$ $r = (214 \cdot 16) \bmod 337 = 54$

Alltså $1023^{91} \bmod 337 = 54$.

5.3 Lösning av linjära kongruenser

Vi skall hitta en metod att bestämma $x \in \{0, 1, 2, \ldots, m - 1\}$ då $ax + b \equiv c \pmod m$. Först en observation. Betrakta följande multiplikationstabeller:

a. Multiplikation modulo 7

·	0	1	2	3	4	5	6
0	0	0	0	0	0	0	0
1	0	1	2	3	4	5	6
2	0	2	4	6	1	3	5
3	0	3	6	2	5	1	4
4	0	4	1	5	2	6	3
5	0	5	3	1	6	4	2
6	0	6	5	4	3	2	1

b. Multiplikation modulo 6

·	0	1	2	3	4	5
0	0	0	0	0	0	0
1	0	1	2	3	4	5
2	0	2	4	0	2	4
3	0	3	0	3	0	3
4	0	4	2	0	4	2
5	0	5	4	3	2	1

I tabell b är resultatet av multiplikationen med t ex 4 ej entydigt bestämd, eftersom $4 \cdot 2 = 4 \cdot 5 = 2$. Då är ekvationen $4x \equiv 2 \pmod 6$ ej entydigt lösbar i $\{0, 1, 2, 3, 4, 5\}$. Både $x = 2$ och $x = 5$ duger! Däremot är resultaten i tabell a entydiga. Detta gör att ekvationen $4x \equiv 2 \pmod 7$ är entydigt lösbar i $\{0, 1, 2, 3, 4, 5, 6\}$ med lösningen $x = 4$. Den avgörande skillnaden mellan a och b är att 7 är ett primtal, medan 6 inte är ett primtal. Det gäller att $4x \equiv 2 \pmod 6$ ej är entydigt lösbar eftersom $\gcd(6, 4) \neq 1$, medan ekvationen $4x \equiv 2 \pmod 7$ är entydigt lösbar eftersom $\gcd(7, 4) = 1$.

Definition 5.2: Talet x är *multiplikativ invers* till a om och endast om $ax \equiv xa \equiv 1 (\mathrm{mod}\ m)$. Vi skriver då $x = a^{-1}$ eller ibland $1/a$.

Sats 5.5: Om modulen m är ett primtal p, så har a en entydigt bestämd multiplikativ invers i $\{1, 2, \ldots, p - 1\}$. (Egentligen är det tillräckligt att $\gcd(a, m) = 1$.)

Exempel 5.8: Lös kongruensen $4x \equiv 2 (\mathrm{mod}\ 7)$.

I multiplikationstabellen ovan ser vi att $4 \cdot 2 \equiv 1 (\mathrm{mod}\ 7)$. Vi multiplicerar därför båda sidor i kongruensen med 2 och får

$$8x \equiv 4 (\mathrm{mod}\ 7), \text{ som ger att}$$
$$1x \equiv 4 (\mathrm{mod}\ 7), \text{ dvs}$$
$$x \equiv 4 (\mathrm{mod}\ 7).$$

Det är uppenbart att kongruensen $ax + b \equiv c (\mathrm{mod}\ m)$ kan reduceras till $ax \equiv d (\mathrm{mod}\ m)$. Den senare ekvationen kan lösas genom att den multiplikativa inversen till a bestäms, och lösningen blir kan då skrivas $x \equiv a^{-1}d (\mathrm{mod}\ m)$. Detta under förutsättningen att $\gcd(a, m) = 1$.

Exempel 5.9: Lös kongruensen $8x - 1 \equiv 3 (\mathrm{mod}\ 5)$.

Observera först att det finns en entydig lösning i $\{0, 1, 2, 3, 4\}$, eftersom $\gcd(8, 5) = 1$. Vi får först att

$$8x \equiv 4 (\mathrm{mod}\ 5)$$

Det gäller att $2 \cdot 8 = 16 \equiv 1 (\mathrm{mod}\ 5)$, dvs 2 är multiplikativ invers till 8, och vi kan skriva $8^{-1} = 2$, eller $1/8 = 2$. Multiplikation med 2 ger

$$16x \equiv 8 (\mathrm{mod}\ 5)$$
$$1x \equiv 3 (\mathrm{mod}\ 5), \text{ dvs}$$
$$x \equiv 3 (\mathrm{mod}\ 5).$$

Hur kan man då på ett systematiskt sätt, och inte bara gissningsvis, bestämma multiplikativ invers till ett tal modulo m? Observera att kongruensen $ax \equiv d (\mathrm{mod}\ m)$ kan lösas med hjälp av Euklides´ algoritm, eftersom

$$ax \equiv d (\mathrm{mod}\ m) \Leftrightarrow m | ax - d \Leftrightarrow \exists y(my = d - ax) \Leftrightarrow \exists y(ax + my = d).$$

Den sista ekvationen är lösbar om och endast om $\gcd(a, m) | d$. Detta gör att kongruenser med fördel kan lösas med hjälp av Euklides´ algoritm. Vi förfar t ex enligt följande.

$ax \equiv d (\mathrm{mod}\ m)$
$ax - d = (-y)m$ för något y.
$ax + my = d$, som löses på vanligt sätt.

Exempel 5.10: Lös kongruensen $23x \equiv 1 (\mathrm{mod}\ 59)$.

Vi skall om möjligt hitta ett $x \in \{0, 1, \ldots, 58\}$ så att $23x + 59y = 1$ för något y som vi inte behöver bestämma.

Euklides´ algoritm ger

$$59 = 2 \cdot 23 + 13$$

$$23 = 1\cdot 13 + 10$$
$$13 = 1\cdot 10 + 3$$
$$10 = 3\cdot 3 + 1$$
$$3 = 3\cdot 1$$

Eftersom gcd(23, 59) = 1, så är den diofantiska ekvationen lösbar. Algoritmen baklänges ger

$$1 = 10 - 3\cdot 3 = 10 - 3\cdot(13 - 1\cdot 10) = -3\cdot 13 + 4\cdot 10 = -3\cdot 13 + 4(23 - 1\cdot 13) =$$
$$= 4\cdot 23 - 7\cdot 13 = 4\cdot 23 - 7(59 - 2\cdot 23) = -7\cdot 59 + 18\cdot 23.$$

Sammanfattningsvis får vi sålunda

$$23\cdot 18 + 59\cdot(-7) = 1.$$

Vi kan då välja $x = 18$ eftersom $18 \in \{0, 1, \ldots, 58\}$, och får

$$x \equiv 18 (\mathrm{mod}\ 59).$$

En alternativ lösning skulle var med lite trixande följande

$$23x \equiv 1 (\mathrm{mod}\ 59)$$
$$69x \equiv 3 (\mathrm{mod}\ 59)$$
$$10x \equiv 3 (\mathrm{mod}\ 59)$$
$$60x \equiv 18 (\mathrm{mod}\ 59)$$
$$1x \equiv 18 (\mathrm{mod}\ 59)$$
$$x \equiv 18 (\mathrm{mod}\ 59)$$

Användande av ”trixande” (jmf ex 5.10) via multiplikationer vid lösning av kongruenser kräver en viss försiktighet. Den normala strategin när man löser t ex vanliga ekvationer eller ekvationssystem är att man gör successiva omskrivningar där de på varandra följande ekvationerna (ekvationssystemen) har exakt samma lösningsmängd. Detta indikeras ibland med att man skriver en ekvivalenssymbol ($\Leftrightarrow$) mellan leden. Detta kan tolkas som att ekvationerna (ekvationssystemen) har exakt samma lösningsmängd.

Exempel 5.11: Lös ekvationen $(x + 1)^2 = (x + 1)(x - 1)$.

Vi får då följande svit av ekvivalenta ekvationer

$$(x + 1)^2 = (x + 1)(x - 1)$$
$$\Leftrightarrow$$
$$x^2 + 2x + 1 = x^2 - 1$$
$$\Leftrightarrow$$
$$2x = -2$$
$$\Leftrightarrow$$
$$x = -1.$$

Här gäller det att den första och den fjärde ekvationen har exakt samma rötter, nämligen -1. Däremot har inte ekvationerna

(1) $(x + 1)^2 = (x + 1)(x - 1)$ och
(2) $x + 1 = x - 1$

samma rötter. Ekvation (2) saknar lösning medan ekvation (1) har den ovan nämnda.

Man skulle kunna tro att man kan erhålla ekvation (2) från ekvation (1) i exemplet ovan, men så är inte fallet eftersom division på båda sidor av likhetstecknet med faktorn $x + 1$ inte är tillåten för varje värde på x. Däremot följer ekvation (1) av ekvation (2), eftersom det är tillåtet att multiplicera på båda sidor i en likhet med ett och samma tal. Betrakta sedan följande exempel med kongruenser.

Exempel 5.12: Betrakta kongruensen

(1) $2x \equiv 3 (\mathrm{mod} 14)$

Här gäller det att $\gcd(2, 14) = 2$, som inte delar 3, varför kongruensen saknar lösning. Om vi multiplicerar båda sidor i kongruensen med 2 (en tillåten operation), erhåller vi

(2) $4x \equiv 6 (\mathrm{mod} 14)$

Denna kongruens har lösningen

$x \equiv 5 (\mathrm{mod} 14)$

Kongruenserna (1) och (2) har alltså inte samma lösningar. Förhållandet mellan dem är

$2x \equiv 3 (\mathrm{mod} 14) \Rightarrow 4x \equiv 6 (\mathrm{mod} 14)$,

dvs om ett tal löser kongruens (1), så löser det också kongruens (2). Att implikationen inte gäller åt andra hållet beror på att strykningslagen (sats 5.3) inte gäller generellt.

Vill man, i kraft av insikterna i exempel 5.12, vara på den säkra sidan, så multiplicerar man endast med tal som är relativt prima med modulen. Gör man så, gäller ekvivalens mellan kongruenserna, eftersom strykningslagen då gäller. Multiplicerar man med ett tal som inte är relativt prima med modulen, måste man kontrollera i den ursprungliga kongruensen om erhållna möjliga lösningar verkligen löser den givna kongruensen.

Övningsuppgifter

5.1 Beräkna

a. $(523 \cdot 18 - 83^3 + 3211) \bmod 7$.
b. $(618 \cdot 613 - 7511 \cdot 381) \bmod 21$

5.2 Beräkna

a. $5^{17} \bmod 11$
b. $38^{35} \bmod 29$
c. $182^{64} \bmod 183$
d. $181^{70} \bmod 511$
e. $1904^{543} \bmod 3127$
f. $1084^{511} \bmod 3127$

5.3 Skriv upp tabeller för multiplikation modulo 8 och modulo 11.

5.4 Lös följande kongruenser. Försök använda både Euklides´ algoritm och metoden med multiplikativ invers.

a.	$3x \equiv 2 \pmod 7$	b.	$5x \equiv 3 \pmod{11}$
c.	$2x \equiv 1 \pmod 3$	d.	$128x \equiv 833 \pmod{1001}$
e.	$17x \equiv 14 \pmod{21}$	f.	$987x \equiv 610 \pmod{1597}$
g.	$521x \equiv 1 \pmod{3016}$	h.	$543x \equiv 1 \pmod{3016}$

5.5 Bevisa sats 5.1.

5.6 Bevisa sats 5.2b.

5.7* I beviset för sats 5.3 utnyttjas följande resultat.
Om $a|bc$ och $\gcd(a, b) = 1$, så gäller att $a|c$.
Bevisa detta.
Bevisa också att, om p är ett primtal och $p|bc$, så $p|b$ eller $p|c$.

6 Eulers sats med mera

6.1 Eulers funktion

I detta avsnitt skall vi bekanta oss med en talteoretisk funktion, som är viktig i kryptografiska sammanhang, samt ytterligare några resultat vi behöver senare. En *talteoretisk* funktion är en funktion från A till B, där A och B är delmängder till **Z**.

Definition 6.1: Låt $\varphi(n)$ = antal positiva heltal mindre än eller lika med n och som är relativt prima med n. Då är φ en funktion från $\mathbf{Z}_+$ till $\mathbf{Z}_+$. Denna funktion brukar kallas *Eulers funktion.* Formellt kan funktionen skrivas
$\varphi(n) = |\{m \in \mathbf{Z}_+ : m \leq n \ \& \ \gcd(m, n) = 1\}|$.

Exempel 6.1: $\varphi(1) = 1$, $\varphi(2) = 1$, $\varphi(3) = 2$,
$\varphi(4) = 2$, $\varphi(5) = 4$, $\varphi(6) = 2$,
$\varphi(7) = 6$, etc

Följande sats visar hur man kan beräkna $\varphi(n)$ för godtyckligt argument.

Sats 6.1: För alla primtal p gäller

(i) $\varphi(p) = p - 1$,
(ii) $\varphi(p^n) = p^n - p^{n-1} = p^n(1 - 1/p)$,
(iii) φ är multiplikativ, dvs $\gcd(a, b) = 1 \Rightarrow \varphi(ab) = \varphi(a)\varphi(b)$.

Bevis: (i) Uppenbart är alla tal mindre än p relativt prima med p, eftersom p är ett primtal.

(ii) Vi skall räkna hur många tal från 1 till p^n, som inte är relativt prima med p^n. De är

$$1p, 2p, 3p, \ldots, p\cdot p, (p + 1)p, (p + 2)p, \ldots, (2p)p, \ldots, p^{n-1}\cdot p,$$

dvs alla tal som innehåller minst en faktor p. Som synes är det p^{n-1} stycken tal. Då måste $\varphi(p^n) = p^n - p^{n-1} = p^n(1 - 1/p)$, eftersom det finns p^n tal som är mindre än eller lika med p^n, och p^{n-1} av dessa är inte relativt prima med p.

(iii) Vi visar det enklare resultatet att för p och q primtal gäller

$$\varphi(pq) = \varphi(p)\varphi(q).$$

Observera att $(p - 1)(q - 1) = pq - p - q + 1$.
Vi räknar upp de tal som inte är relativt prima med pq.

$1p, 2p, 3p, \ldots, qp$ dvs q stycken tal som inte är relativt prima med p.

$1q, 2q, 3q, \ldots, pq$ dvs p stycken tal som inte är relativt prima med q.

Ovan är talet pq räknat två gånger, varför vi får att
$\varphi(pq) = pq - q - p + 1 = (p - 1)(q - 1) = \varphi(p)\varphi(q)$.

Exempel 6.2: $\varphi(6) = \varphi(2\cdot 3) = \varphi(2)\varphi(3) = 1\cdot 2 = 2$
$\varphi(15) = \varphi(3\cdot 5) = \varphi(3)\varphi(5) = 2\cdot 4 = 8$
$\varphi(8) = \varphi(2^3) = 2^3 - 2^2 = 8 - 4 = 4$
$\varphi(36) = \varphi(4\cdot 9) = \varphi(4)\varphi(9) = \varphi(2^2)\varphi(3^2) = (4 - 2)(9 - 3) = 12$.

Generellt kan $\varphi(n)$ beräknas på följande sätt med utgångspunkt från ovanstående sats. Skriv $n = p_1^{\alpha_1}\cdot p_2^{\alpha_2}\cdot\ldots\cdot p_k^{\alpha_k}$. Detta kan göras entydigt på grund av aritmetikens fundamentalsats. Sats 6.1 ger då

$$\varphi(n) = \varphi(p_1^{\alpha_1})\varphi(p_2^{\alpha_2})\ldots\varphi(p_k^{\alpha_k}) = p_1^{\alpha_1}(1 - \frac{1}{p_1})p_2^{\alpha_2}(1 - \frac{1}{p_2})\ldots p_k^{\alpha_k}(1 - \frac{1}{p_k}) =$$

$$= n(1 - \frac{1}{p_1})(1 - \frac{1}{p_2})\ldots(1 - \frac{1}{p_k}).$$

Vi formulerar detta praktiska resultat i ett korollarium.

Korollarium 6.1: $\varphi(n) = n(1 - \frac{1}{p_1})(1 - \frac{1}{p_2})\ldots(1 - \frac{1}{p_k})$.

Exempel 6.3: $\varphi(36) = \varphi(2^2\cdot 3^2) = 36(1 - 1/2)(1 - 1/3) = 12$.

6.2 Eulers och Fermats satser

I detta avsnitt skall vi formulera och bevisa en sats av Euler som är av fundamental betydelse vid konstruktion av det så kallade RSA-systemet (se nedan).

Definition 6.2: En *reducerad mängd av rester modulo m* är en mängd av $\varphi(m)$ heltal sådan att

(i) varje element i mängden är relativt prima med m, och
(ii) inga par av olika element i mängden är kongruenta modulo m.

Exempel 6.4: Mängden av de minsta icke-negativa resterna modulo 7 är {0, 1, 2, 3, 4, 5, 6}. Här är alla tal utom noll relativt prima med 7. En reducerade mängden är {1, 2, 3, 4, 5, 6}, eftersom dessutom i denna mängd det gäller att inga par av element är kongruenta modulo 7. Notera att $\varphi(7) = 6$. Mängden av minsta ickenegativa rester modulo 12 är {0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11}. En reducerad mängd är A = {1, 5, 7, 11}. Kom ihåg att elementen skall vara parvis inkongruenta modulo 12. Det gäller att $\varphi(12) = \varphi(2^2)\varphi(3) = 4(1-1/2)\cdot 2 = 4$. Mängden B = {5, 25, 35, 55} är en annan mängd av reducerade rester modulo 12. Mängden uppfyller villkoren i definition 6.2. Observera att vart och ett av talen i B är kongruent med exakt ett av talen i A. Kontrollera detta! Talet 5 är relativt prima med 12. B har erhållits från A genom att varje tal i A multiplicerats med 5. Används t ex 6 i stället, så erhålls ingen reducerad mängd av rester. Vi får då mängden {6, 30, 42, 66}, och det gäller ju att $6 \equiv 30 \pmod{12}$.

Följande resultat gäller för dylika reducerade mängder av rester.

Sats 6.2: Om $\{r_1, r_2, \dots, r_{\varphi(m)}\}$ är en reducerad mängd av rester modulo m och gcd(a, m) = 1, så är också $\{ar_1, ar_2, \dots, ar_{\varphi(m)}\}$ en reducerad mängd av rester modulo m.

Bevis: Vi skall visa att (i) $\gcd(ar_j, m) = 1$ för $j = 1, 2, \dots, \varphi(m)$, och att (ii) inga par av olika element från $\{ar_1, ar_2, \dots, ar_{\varphi(m)}\}$ är kongruenta modulo m.

(i) Antag att $\gcd(ar_j, m) > 1$ för godtyckligt j. Då finns primtal p så att $p \mid \gcd(ar_j, m)$, dvs $p \mid a$ och $p \mid m$, eller $p \mid r_j$ och $p \mid m$. Men p kan inte dela både a och m enligt förutsättning, och inte heller kan p dela både r_j och m, eftersom $\{r_1, r_2, \dots, r_{\varphi(m)}\}$ är en reducerad mängd av rester. Eftersom antagandet att $\gcd(ar_j, m) > 1$ leder till en motsägelse, så måste $\gcd(ar_j, m) = 1$.

(ii) Antag att $ar_j \equiv ar_k \pmod m$ för $j \neq k$.
Eftersom gcd(a, m) = 1, så gäller $r_j \equiv r_k \pmod m$, men detta motsäger återigen att $\{r_1, r_2, \dots, r_{\varphi(m)}\}$ är en reducerad mängd av rester. Då måste ar_j och ar_k vara inkongruenta modulo m.

Därmed är beviset klart.

Sats 6.3: (Euler) Om m är ett positivt heltal och a ett heltal sådant att gcd(a, m) = 1, så $a^{\varphi(m)} \equiv 1 \pmod m$.

Bevis: Låt $\{r_1, r_2, \dots, r_{\varphi(m)}\}$ vara den reducerade mängd av positiva rester i vilken alla element är mindre än m, dvs $0 < r_j < m$ för alla j. Eftersom gcd(a, m) = 1, gäller att $\{ar_1, ar_2, \dots, ar_{\varphi(m)}\}$ också är en reducerad mängd av rester. Då måste det för varje j gälla att $ar_j \equiv r_k \pmod m$ för exakt ett k (se övning 6.5). Multiplikation ger

$$ar_1 \cdot ar_2 \cdot \dots \cdot ar_{\varphi(m)} \equiv r_1 \cdot r_2 \cdot \dots \cdot r_{\varphi(m)} \pmod m.$$

Detta ger

$$a^{\varphi(m)} r_1 \cdot r_2 \cdot \dots \cdot r_{\varphi(m)} \equiv r_1 \cdot r_2 \cdot \dots \cdot r_{\varphi(m)} \pmod m,$$

som i sin tur ger

$$a^{\varphi(m)} \equiv 1 \pmod m,$$

eftersom samtliga r_j är relativt prima med m.

Vi har sålunda bevisat att $a^{\varphi(m)} \equiv 1 \pmod m$.

Sats 6.4: (Fermat) Om p är ett primtal och p inte delar a, så $a^{p-1} \equiv 1 \pmod p$.

Bevis: Detta följer direkt av Eulers sats och av att $\varphi(p) = p - 1$.

Eulers sats kan användas för att hitta multiplikativa inverser. Betrakta kongruensen $ax \equiv b \pmod m$, där gcd(a, m) = 1. Vi får då om vi multiplicerar med $a^{\varphi(m)-1}$ att

$$a^{\varphi(m)-1}ax \equiv a^{\varphi(m)-1}b \pmod m$$
$$a^{\varphi(m)}x \equiv a^{\varphi(m)-1}b \pmod m.$$

Eftersom $a^{\varphi(m)} \equiv 1 \pmod m$, så gäller det att $x \equiv a^{\varphi(m)-1}b \pmod m$.

Exempel 6.5: Lös kongruensen $3x \equiv 4 \pmod 5$.

Det gäller att $\varphi(5) = 4$. Multiplikation med 3^{4-1} ger

$$3^{4-1} \cdot 3x \equiv 3^{4-1} \cdot 4 \pmod 5.$$
$$3^4 \cdot x \equiv 3^{4-1} \cdot 4 \pmod 5$$

Eftersom $3^4 \equiv 1 \pmod 5$ får vi

$$x \equiv 3^{4-1} \cdot 4 \equiv 27 \cdot 4 \equiv 3 \pmod 5$$

Alltså $x \equiv 3 \pmod 5$.

Sammanfattningsvis har vi att den linjära kongruensen $ax \equiv b \pmod{(m)}$ har exakt en lösning i $\{0, 1, 2, \ldots, m - 1\}$, om $\gcd(a, m) = 1$. Denna lösning kan hittas med hjälp av Euklides´ algoritm (lös den diofantiska ekvationen $ax + my = b$), eller med hjälp av Eulers sats genom att beräkna $a^{\varphi(m)-1}b \bmod m$. Vilken av dessa metoder är snabbast för stora tal?

6.3 System av linjära kongruenser

Följande problem finns formulerat ca 100 f kr av den kinesiske matematikern Sun Tsu och den grekiske matematikern Nichomachos. Hitta det minsta positiva heltal som vid division med tre ger resten två, vid division med fem ger resten tre och vid division med sju ger resten två. Översatt till kongruenser blir problemet

$$\begin{cases} x \equiv 2 \pmod 3 & (1) \\ x \equiv 3 \pmod 5 & (2) \\ x \equiv 2 \pmod 7 & (3) \end{cases}$$

Problemet är att hitta det minsta ickenegativa heltal som satisfierar samtliga kongruenser.

(1) ger att (4) $x = 3t + 2$ för något t.

(4) insatt i (2) ger
$3t + 2 \equiv 3 \pmod 5$
$3t \equiv 1 \pmod 5$
$t \equiv 2 \pmod 5$ (multiplikation med 2)

dvs (5) $t = 5u + 2$ för något u.

(4) och (5) ger $x = 3(5u + 2) + 2 = 15u + 8$ för något u.

Detta insatt i (3) ger
$15u + 8 \equiv 2 \pmod 7$
$u + 1 \equiv 2 \pmod 7$
$u \equiv 1 \pmod 7$.

Då är $u = 7v + 1$ för något v, vilket ger
$x = 3t + 2 = 3(5u + 2) + 2 = 15u + 8 =$
$= 15(7v + 1) + 8 = 105v + 23$.

Vi får därför $x \equiv 23 \pmod{105}$.

Sats 6.4: (Kinesiska restsatsen) Systemet av t linjära kongruenser

$$\begin{cases} x \equiv a_1 \pmod{m_1} \\ x \equiv a_2 \pmod{m_2} \\ \quad\vdots \\ x \equiv a_t \pmod{m_t} \end{cases}$$

där modulerna är parvis relativt prima har en entydig lösning modulo m, där $m = m_1 \cdot m_2 \cdot \ldots \cdot m_t$. Lösningen ges av

$x \equiv x_0 \pmod{m}$, där $x_0 = (y_1 \frac{m}{m_1} a_1 + y_2 \frac{m}{m_2} a_2 + \ldots + y_t \frac{m}{m_t} a_t) \bmod m$

och y_i är lösningen till $\frac{m}{m_i} \cdot y_i \equiv 1 \pmod{m_i}$ för i = 1, 2, ..., t.

Beviset utnyttjar idén i ovanstående exempel och ingår inte.

Exempel 6.6: Vi löser det inledande exemplet igen.

$$\begin{cases} x \equiv 2 \pmod 3 & (1) \\ x \equiv 3 \pmod 5 & (2) \\ x \equiv 2 \pmod 7 & (3) \end{cases}$$

Vi får då med satsens beteckningar

$m_1 = 3, \quad a_1 = 2$
$m_2 = 5, \quad a_2 = 3$
$m_3 = 7, \quad a_3 = 2$
$m = 105$

$35y_1 \equiv 1 \pmod 3$, $21y_2 \equiv 1 \pmod 5$, $15y_3 \equiv 1 \pmod 7$,
$2y_1 \equiv 1 \pmod 3$,
$y_1 \equiv 2 \pmod 3$, $y_2 \equiv 1 \pmod 5$, $y_3 \equiv 1 \pmod 7$.

$x_0 = (2 \cdot 35 \cdot 2 + 1 \cdot 21 \cdot 3 + 1 \cdot 15 \cdot 2) \bmod 105 = (140 + 63 + 30) \bmod 105 =$
$= 128 \bmod 105 = 23.$

Vi får alltså som tidigare $x \equiv 23 \pmod{105}$.

6.4 System av kongruenser med två obekanta

Vi skall lösa system av typen

$$\begin{cases} ax + by \equiv e \pmod m \\ cx + dy \equiv f \pmod m \end{cases}$$

Observera att modulen är densamma i båda kongruenserna. Vi inleder med ett exempel innan vi visar hur dylika system kan lösas generellt.

Exempel 6.7: Lös följande system av kongruenser.

$$\begin{cases} x - 2y \equiv 2 \pmod{11} \\ 2x + 3y \equiv 1 \pmod{11} \end{cases}$$

Multiplicera den första kongruensen med 2 och subtrahera från den andra kongruensen.

$$\begin{cases} x - 2y \equiv 2 \pmod{11} \\ 7y \equiv -3 \pmod{11} \end{cases}$$

Observera att gcd(2, 11) = 1, varför de två systemen är ekvivalenta och därför har samma lösningsmängd. Löser vi den andra kongruensen får vi efter multiplikation med 8

$$56y \equiv -24 \pmod{11}, \text{ dvs}$$
$$y \equiv 9 \pmod{11}.$$

Sätts detta in i den första kongruensen erhålles

$$x - 18 \equiv 2 \pmod{11}$$
$$x \equiv 20 \pmod{11}$$
$$x \equiv 9 \pmod{11}.$$

Systemet har alltså lösningen

$$\begin{cases} x \equiv 9 \pmod{11} \\ y \equiv 9 \pmod{11} \end{cases}$$

Genom insättning i de ursprungliga kongruenserna kan vi kontrollera att lösningen är korrekt.

Generellt löser vi systemet av kongruenser

$$\begin{cases} ax + by \equiv e \pmod{m} \\ cx + dy \equiv f \pmod{m} \end{cases}$$

på t ex följande sätt. Om vi multiplicera första kongruensen med d, andra med b och subtraherar, får vi

$$(ad - bc)x \equiv (de - bf) \pmod{m}.$$

Multiplicerar vi första kongruensen med c, andra med a och subtraherar, får vi

$$(ad - bc)y \equiv (af - ce) \pmod{m}.$$

För entydighet kräver dessa kalkyler att a, b, c och d samtliga är relativt prima med modulen m. Uttrycket ad - bc känns igen som determinanten för koefficientmatrisen

$$\begin{vmatrix} a & b \\ c & d \end{vmatrix} = D.$$

Om det nu gäller att gcd(D, m) = 1, finns en entydig lösning modulo m. Denna kan teoretiskt, om än inte särskilt praktiskt, erhållas genom att multiplicera koefficientmatrisen med dess invers. Skriver vi systemet på matrisform, får vi

$$\begin{pmatrix} 2 & 3 \\ 1 & -2 \end{pmatrix}\begin{pmatrix} x \\ y \end{pmatrix} \equiv \begin{pmatrix} 1 \\ 2 \end{pmatrix} \pmod{11}$$

Här gäller det att

$$D = \begin{vmatrix} 2 & 3 \\ 1 & -2 \end{vmatrix} = -4 - 3 = -7, \text{ varför } \gcd(-7, 11) = 1.$$

Inversen till $\begin{pmatrix} 2 & 3 \\ 1 & -2 \end{pmatrix}$ kan beräknas till $\begin{pmatrix} 5 & 2 \\ 8 & 6 \end{pmatrix}$. Då gäller det att

$$\begin{pmatrix} x \\ y \end{pmatrix} \equiv \begin{pmatrix} 5 & 2 \\ 8 & 6 \end{pmatrix}\begin{pmatrix} 1 \\ 2 \end{pmatrix} \equiv \begin{pmatrix} 9 \\ 20 \end{pmatrix} \equiv \begin{pmatrix} 9 \\ 9 \end{pmatrix} \pmod{11}.$$

6.5 Invertering av f(x) = (ax + b) mod m

Detta problem dyker upp när vi vill bestämma dechiffreringsfunktionen vid så kallade affina chiffer (se nedan). Vi illustrerar först med ett exempel.

Exempel 6.8: Bestäm $f^{-1}(x)$ då $f(x) = (2x + 3) \bmod 5$.

Sätt $y = f(x)$ och lös ut x ur kongruensen $y \equiv 2x + 3 \pmod 5$

$2x \equiv y - 3 \pmod 5$
$6x \equiv 3y - 9 \pmod 5$ (multiplikation med 3, obs att gcd(3, 5) = 1)
$x \equiv 3y + 1 \pmod 5$

Vi får $f^{-1}(y) = (3y + 1) \bmod 5$, som efter variabelbyte ger
$f^{-1}(x) = (3x + 1) \bmod 5$.

Kontrollräknar vi, ser vi att $f^{-1}(f(x)) = (3(2x + 3) + 1) \bmod 5 =$
$= (6x + 10) \bmod 5 = x \bmod 5$, och vi kan vara förvissade om att vi verkligen har hittat inversen till f.

Allmänt får vi med $f(x) = (ax + b) \bmod m$, där vi förutsätter att $\gcd(a, m) = 1$.

Sätt $y = f(x)$ och lös ut x ur kongruensen $y \equiv ax + b \pmod m$. Vi får då $ax \equiv y - b \pmod m$, som multipliceras med den multiplikativa inversen $a^{\varphi(m) - 1}$ till a, vilket ger

$a^{\varphi(m) - 1}ax \equiv a^{\varphi(m) - 1}(y - b) \pmod m$
$x \equiv (a^{\varphi(m) - 1}y - a^{\varphi(m) - 1}b) \pmod m$, dvs
$f^{-1}(x) = (a^{\varphi(m) - 1}x - a^{\varphi(m) - 1}b) \pmod m$.

Om det i kongruensen $ax \equiv b(\text{mod } m)$ gäller att $\gcd(a, m) > 1$, så är kongruensen inte entydigt lösbar i $\{0, 1, \ldots, m - 1\}$. Då gäller, under förutsättning att kongruensen är lösbar, att det finns exakt $d = \gcd(a, m)$ lösningar i $\{0, 1, \ldots, m - 1\}$. Varför är det så?

Exempel 6.9: Lös kongruensen $4x \equiv 2(\text{mod } 10)$

Det gäller att $\gcd(4, 10) = 2$.

Om kongruensen är lösbar, så finns två inkongruenta lösningar modulo 10. Vi dividerar med två och får

$2x \equiv 1(\text{mod } 5)$
$6x \equiv 3(\text{mod } 5)$ $(\gcd(3, 5) = 1)$
$x \equiv 3(\text{mod } 5)$, som kan skrivas
$x \equiv 3(\text{mod } 10)$ eller $x \equiv 8(\text{mod } 10)$,
om vi vill betona att kongruensen har två lösningar i $\{0, 1, \ldots, 9\}$.Bäst svar är kanske $x \equiv 3(\text{mod } 5)$.

Exempel 6.10: Lös kongruensen $3x \equiv 4(\text{mod } 12)$

Vi har att $\gcd(3, 12) = 3$. Då saknar kongruensen lösning, ty 3 är inte delare till 4.

Jämför detta med den diofantiska ekvationen $3x + 12y = 4$, som också är olösbar.

Exempel 6.11: Lös kongruensen $3x \equiv 6(\text{mod } 12)$

$\gcd(3, 12) = 3$ och $3|6$ ger att kongruensen har tre lösningar i $\{0, 1, \ldots, 11\}$.

Dessa är $x \equiv 2(\text{mod } 4)$.

Exempel 6.12: Lös systemet av kongruensen

$$\begin{cases} 3x + 4y \equiv 7(\text{mod } 14) \\ 5x + 6y \equiv 8(\text{mod } 14) \end{cases}$$

(jmf exempel 5.12 och texten därefter)

Lösning, alternativ 1

$$\begin{cases} 3x + 4y \equiv 7(\text{mod } 14) \\ 5x + 6y \equiv 8(\text{mod } 14) \end{cases}$$

Multiplikation med 5 i den första kongruensen och med -3 (observera att både 5 och -3 är relativt prima med 14) i den andra ger

$$\begin{cases} 15x + 20y \equiv 35(\text{mod } 14) \\ -15x - 18y \equiv -24(\text{mod } 14) \end{cases}$$

$$\begin{cases} 15x + 20y \equiv 35 \pmod{14} \\ \quad 2y \equiv 11 \pmod{14} \end{cases}$$

Dessa tre system av kongruenser är ekvivalenta, dvs har exakt samma lösningsmängd. Observera dock att gcd(2, 14) = 2, som inte delar 11, varför systemet saknar lösning.

Lösning, alternativ 2

(1) $$\begin{cases} 3x + 4y \equiv 7 \pmod{14} \\ 5x + 6y \equiv 8 \pmod{14} \end{cases}$$

Multiplikation med 3 respektive -2 ger

(2) $$\begin{cases} 9x + 12y \equiv 21 \pmod{14} \\ -10x - 12y \equiv -16 \pmod{14} \end{cases}$$

Dessa två system är relaterade via en implikation, dvs om ett par av tal löser system (1), så löser detta par också system (2), men inte nödvändigtvis omvänt.

$$\begin{cases} 9x + 12y \equiv 21 \pmod{14} \\ -x \equiv 8 \pmod{14} \end{cases}$$

Den andra kongruensen ger här att

$$x \equiv -5 \equiv 9 \pmod{14}.$$

Insättning ger sedan att

$$-45 + 12y \equiv 21 \pmod{14}$$
$$12y \equiv 66 \equiv 10 \pmod{14}$$

Eftersom gcd(12, 14) = 2 | 10 får vi

$$6y \equiv 5 \pmod{7}$$
$$y \equiv 2 \pmod{7}.$$

Möjliga lösningar är alltså

$$\begin{cases} x \equiv 9 \pmod{14} \\ y \equiv 2 \pmod{14} \end{cases} \quad \text{eller} \quad \begin{cases} x \equiv 9 \pmod{14} \\ y \equiv 9 \pmod{14} \end{cases}$$

Eftersom vi endast har implikation mellan de två första systemen, måste de möjliga lösningarna kontrolleras i det ursprungliga systemet av kongruenser. Gör vi detta ser vi att ingen av de möjliga lösningarna satisfierar systemet, som därför saknar lösning.

Övningsuppgifter

6.1 Beräkna

a. $\varphi(25)$ b. $\varphi(100)$
c. $\varphi(2500)$ d. $\varphi(9001)$
e. $\varphi(9197)$ f. $\varphi(9617)$

6.2 Lös följande system av kongruenser.

a. $\begin{cases} x + 2y \equiv 1 \pmod 5 \\ 2x + y \equiv 1 \pmod 5 \end{cases}$ b. $\begin{cases} 3x + 3y \equiv 1 \pmod 5 \\ 3x + 4y \equiv 2 \pmod 5 \end{cases}$

c. $\begin{cases} 4x + y \equiv 2 \pmod 5 \\ 2x + 4y \equiv 1 \pmod 5 \end{cases}$ d. $\begin{cases} 2x + 3y \equiv 5 \pmod 8 \\ x + 5y \equiv 6 \pmod 8 \end{cases}$

6.3 Lös följande system av kongruenser.

a. $\begin{cases} x \equiv 4 \pmod{11} \\ x \equiv 3 \pmod{17} \end{cases}$ b. $\begin{cases} x \equiv 1 \pmod 2 \\ x \equiv 2 \pmod 3 \\ x \equiv 3 \pmod 5 \end{cases}$

6.4 Bestäm f^{-1} då

a. $f(x) = (8x + 3) \bmod 11$
b. $f(x) = (17x + 8) \bmod 26$

6.5* I beviset för Eulers sats utnyttjas följande resultat.
För varje j finns exakt ett k så att $ar_j \equiv r_k \pmod m$.
Bevisa detta under förutsättningarna i Eulers sats.

6.6 Beräkna $13^{35} \bmod 29$.

6.7 Ange multiplikativ invers till 5 modulo 9 och lös därefter kongruensen $5x \equiv 3 \pmod 9$.

6.8 Bestäm multiplikativ invers till 17 modulo 93.

6.9 Lös följande system av kongruenser

a $\begin{cases} x \equiv 3 \pmod 4 \\ x \equiv 1 \pmod 5 \\ x \equiv 2 \pmod 7 \end{cases}$ b. $\begin{cases} 4x + y \equiv 10 \pmod{26} \\ 9x + y \equiv 18 \pmod{26} \\ 13x + y \equiv 14 \pmod{26} \end{cases}$

c. $\begin{cases} 15x + 17y \equiv 3 \pmod{29} \\ 12x + 13y \equiv 1 \pmod{29} \end{cases}$ d. $\begin{cases} x + 2y \equiv 1 \pmod{10} \\ 3x + 8y \equiv 5 \pmod{10} \end{cases}$

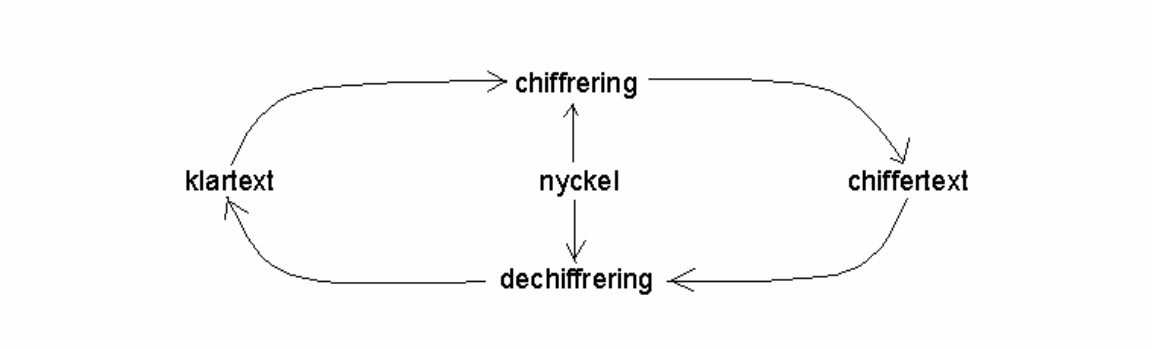
e. $\begin{cases} 13x + y \equiv 2 \pmod{14} \\ 5x + 2y \equiv 3 \pmod{14} \end{cases}$ f. $\begin{cases} 6x + 8y \equiv 3 \pmod{17} \\ 11x + 12y \equiv 6 \pmod{17} \end{cases}$

6.10 Bestäm den inversa funktionen till $f(x) = (17x + 3) \bmod 26$.

7 Allmänt om kryptografi

7.1 Kryptografi

Kryptografi är vetenskapen om och studien av hemlig skrift. Ett *chiffer* är en hemlig metod att skriva genom vilken en *klartext* transformeras till en *chiffertext.* Processen att transformera klartext till chiffertext kallas *chiffrering.* Den omvända processen kallas *dechiffrering*. Både chiffrering och dechiffrering kontrolleras av en kryptografisk *nyckel*.



Det finns två grundläggande typer av chiffer - *transpositionschiffer* (*permutationschiffer*) och *substitutionschiffer.* I ett *transpositionschiffer* arrangerar man om symbolerna eller bitarna som skall chiffreras enligt något schema.

Exempel 7.1: Ordna följande klartext DET HÄR ÄR LÄTT i en 3 x 4 matris.

D E T H
Ä R Ä R
L Ä T T

Kolumnerna läses sedan av i någon ordning, t ex 2 - 4 - 1 - 3, och vi får chiffertexten

ERÄHRTDÄLTÄT.

Ett transpositionschiffer ändrar uppenbart inte frekvensen för en symbol.

I ett *substutitionschiffer* ersätts en symbol eller ett block av symboler med en ny symbol eller ett block av nya symboler.

Exempel 7.2: Caesars chiffer. Låt A = 0, B = 1, ... , Ö = 28 och låt chiffreringsavbildningen vara $f(x) = (x + 3) \bmod 29$ (29 bokstäver i alfabetet).
Då gäller $f(C) = f(02) = 5 \bmod 29 = F$ och klartexten

DET HÄR ÄR LÄTT

chiffreras till

GHW KBU BU OBWW.

Ofta används kombinationer av transpositionschiffer och substutitionschiffer. Dylika chiffer kallas *produktchiffer*.

Exempel 7.3: Kombinerar vi de två exemplen ovan, först transpositionen och sedan substutitionen, erhåller vi chiffertexten

HUBKUWGBOWBW.

Kryptoanalys är vetenskapen om och studien av hur chiffer knäcks. Ett chiffer är *brytbart* om det är möjligt att bestämma klartext eller nyckel från chiffertext, eller att bestämma nyckel från par av klartext/chiffertext.

Kryptoanalytikern har i princip tre attackmetoder i sina försök att bryta ett chiffer. I en *chiffertextattack* måste kryptoanalytikern bestämma nyckel eller en klartext enbart utgående från chiffertexten. Dock kan chiffreringsmetod, klartextspråk, textens ämnesområde etc vara känt.

I en *känd klartextattack* har analytikern tillgång till något par av klartext/chiffertext, och har att bestämma nyckel eller klartext utgående från en annan chiffertext. Ett chiffer kan anses vara säkert endast om det motstår en känd-klartextattack under antagandet att kryptoanalytikern har tillgång till en godtyckligt stor uppsättning av par av klartext/chiffertext.

I en *vald klartextattack* har kryptoanalytikern möjlighet att själv välja klartext som skall chiffreras och har på så sätt tillgång till klartext/chiffertextpar. Detta ger förstås analytikern en än bättre utgångspunkt, eftersom han/hon kan välja klartext på ett klokt sätt för att utvinna så mycket information som möjligt.

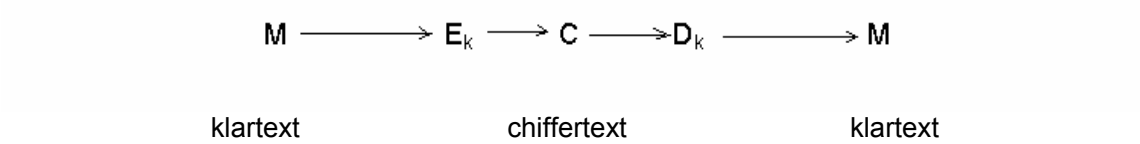
Det kunskapsområde som omfattar både kryptografi och kryptoanalys kallas *kryptologi*.

Ett chiffer är (*obetingat*) *säkert* om det är omöjligt att avslöja klartexten oavsett hur mycket chiffertext man än har tillgång till. Även om det finns säkra chiffer är de besvärliga att hantera, varför man normalt endast är intresserad av chiffer som är *beräkningsmässigt omöjliga* (computationally infeasible) att bryta. Ett chiffer är *beräkningsmässigt säkert* eller *starkt*, om det inte kan brytas med en systematisk analys med tillgängliga resurser (tillgänglig datakraft på rimlig tid). Observera att med ett undantag så är alla chiffer möjliga att forcera om man har tillgång till resurser (se nedan).

Ett *kryptografiskt system* består av följande fem komponenter:

(i)	Klartextrum	$\mathcal{M}$	(message),
(ii)	Chiffertextrum	$\mathcal{C}$	(cipher),
(iii)	Nyckelrum	$\mathcal{K}$	(key),
(iv)	En mängd av chiffreringsavbildningar	$E_k : \mathcal{M} \to \mathcal{C}$, där $k \in \mathcal{K}$,	
(v)	En mängd av dechiffreringsavbildningar	$D_k : \mathcal{C} \to \mathcal{M}$, där $k \in \mathcal{K}$.	

Vi kan illustrera med följande figur:



Här är $M \in \mathcal{M}$ och $C \in \mathcal{C}$. För ett givet k, en given nyckel, är D_k inversen till E_k, varför $D_k(E_k(M)) = M$ för varje klartextmeddelande M.

Ett kryptosystem måste uppfylla följande allmänna krav:

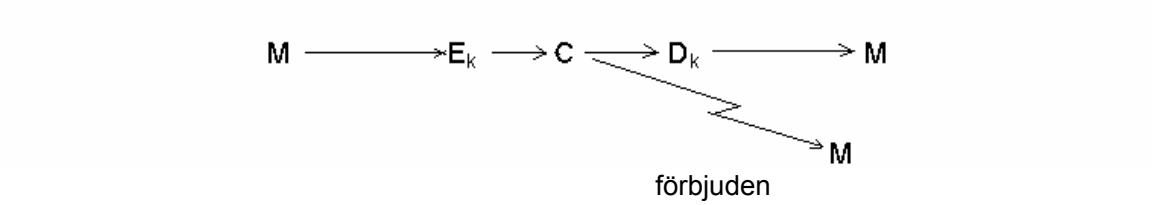
(i) Chiffrerings- och dechiffreringsavbildningarna måste vara effektiva för varje nyckel.
(ii) Systemet måste vara enkelt att använda.
(iii) Systemets säkerhet får bara bero på att nyckeln hålls hemlig. Säkerheten får inte bero på ett eventuellt hemlighållande av algoritmerna E eller D.

Det första kravet är centralt i datortillämpningar, eftersom chiffrering och dechiffrering ofta äger rum i samband med överföring av data, varför avbildningarna inte får vara flaskhalsar. Det andra villkoret innebär att det måste vara enkelt att bestämma avbildningarna utifrån nyckeln för dem som skall använda systemet. Krav (iii) ger att chiffrerings- och dechiffreringsavbildningarna måste vara starka. Man skall alltså inte behöva hålla hemligt vilket kryptosystem som används! Kryptoanalytikern får gärna kunna avslöja E_k eller D_k, om han känner till k, men inte omvänt.

Det finns i princip två olika typer av kryptografiska system, system med *hemlig nyckel* och system med *offentlig nyckel*. I system med hemlig nyckel är både E_k och D_k, eller egentligen k, hemliga. Varje par av användare måste på något säkert sätt utbyta nyckel med varandra. Om n användare skall kommunicera säkert med varandra, så går det åt $n(n - 1)/2$ nycklar (varför?). I system med offentlig nyckel är E_{k_1} offentlig medan D_{k_2} är hemlig. E_{k_1} och D_{k_2} är varandras inverser, men de (eller åtminstone E_{k_1}) skall vara en envägsfunktion med fallucka ("trapdoor one-way function"). Den individ som vill kommunicera i ett system med offentlig nyckel offentliggör sin chiffreringsnyckel E_{k_1}, och hemlighåller sin dechiffreringsnyckel D_{k_2}. Om n användare skall kommunicera åtgår därför n nycklar (varför?).

Ett kryptografiskt system skall kunna garantera både *sekretess* och *autenticitet*. Sekretess innebär att kryptoanalytikern inte skall kunna bestämma klartext från uppsnappad chiffertext.

Sekretess



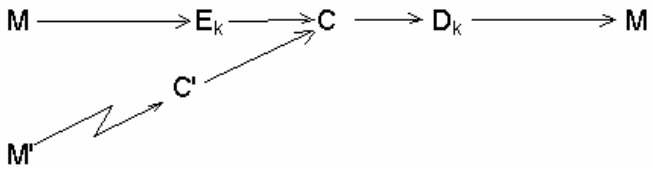
Sekretesskraven innebär att

(i) Det skall vara beräkningsmässigt omöjligt för en kryptoanalytiker att systematiskt bestämma dechiffreringsnyckeln k i dechiffreringsavbildningen D_k från uppsnappad chiffertext C, även om motsvarande klartext M är känd.
(ii) Det skall vara beräkningsmässigt omöjligt för en kryptoanalytiker att systematiskt bestämma klartext från uppsnappad chiffertext C.

Sekretesskravet innebär att dechiffreringsavbildningen D_k (eller egentligen nyckeln k) skyddas. Chiffreringsavbildningen E_k (inklusive nyckeln k) behöver inte vara hemlig, om man inte genom ett offentliggörande av E_k också avslöjar D_k (nyckeln).

Autenticitetskravet innebär att en kryptoanalytiker inte skall kunna ersätta en chiffertext C med en falsk chiffertext C'.

Autenticitet



förbjuden

Formellt innebär kraven följande.

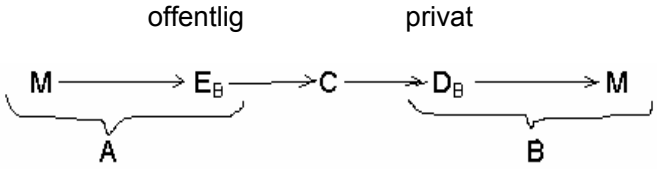
(i) Det skall vara beräkningsmässigt omöjligt för en kryptoanalytiker att systematiskt bestämma chiffreringsavbildningen E_k (nyckeln) givet C, även om motsvarande klartext M är känd.

(ii) Det skall vara beräkningsmässigt omöjligt för en kryptoanalytiker att systematiskt bestämma en chiffertext C' så att $D_k(C')$ är en giltig klartext i $\mathcal{M}$.

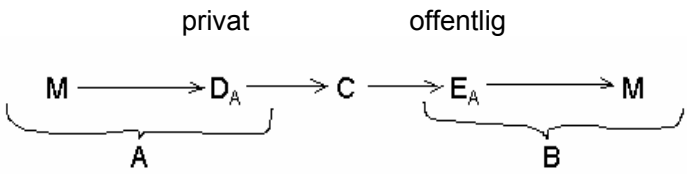
För ett system med hemlig nyckel sammanfaller sekretess och autenticitet, eftersom E_k och D_k använder samma nyckel, eller lätt nycklar som är lätt bestämbara från varandra.

En del system med offentlig nyckel kan användas om man vill uppnå bara det ena kravet. Om A är avsändare och B mottagare med E_A, D_A, E_B och D_B respektive individs chiffrerings- och dechiffreringsavbildningar, så kan sekretess respektive autenticitet åstadkommas enligt nedanstående diagram.

Sekretess

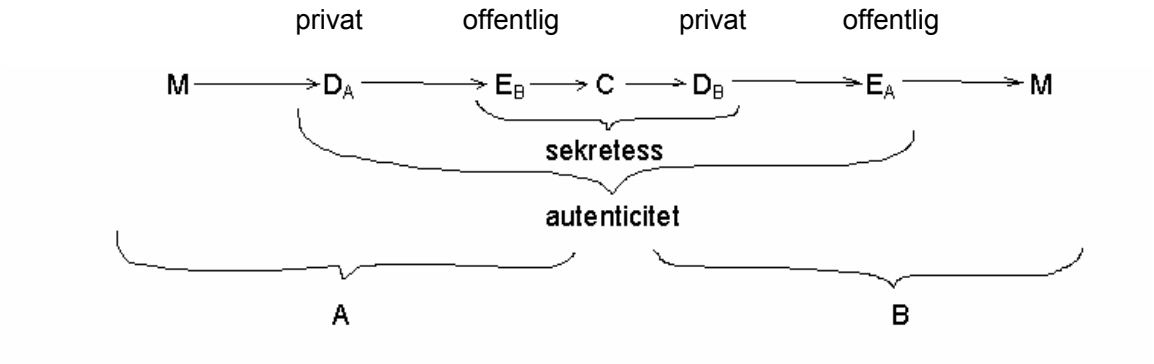


Autenticitet



Eftersom endast A känner till D_A, kan det bara vara A, som kan ha chiffrerat meddelande M till C, som dechiffreras med den offentliga avbildningen E_A. Om man vill åstadkomma både sekretess och autenticitet kan man göra enligt följande.

sekretess och autenticitet



A chiffrerar M enligt $C = E_B(D_A(M))$ och B dechiffrerar enligt $E_A(D_B(C)) = E_A(D_B(E_B(D_A(M)))) = E_A(D_A(M)) = M$.

Observera emellertid att inte alla kryptosystem med privat nyckel kan användas för både sekretess och autenticitet. RSA-systemet, som presenteras nedan, möjliggör emellertid båda.

7.2 Digitala signaturer

En *digital signatur* är en egenskap som är privat för en användare eller en process och som används för att underteckna ett meddelande. Låt B vara mottagare av ett meddelande M signerat av A. A:s signatur skall då uppfylla följande krav.

(i) B måste kunna verifiera att det är A som signerat M.
(ii) Ingen skall kunna förfalska A:s signatur.
(iii) Om A förnekar att han signerat ett meddelande M, skall det vara möjligt för en domare (tredje part) att lösa tvisten mellan A och B.

En digital signatur fastställer alltså *avsändaridentitet*. Den är analog med ett vanligt undertecknande. RSA-systemet möjliggör på ett enkelt sätt digital signering, enligt följande schema.

A signerar M genom att räkna ut $C = D_A(M)$.
B bekräftar A:s signatur genom att kontrollera att $E_A(C) = M$.
En domare kan lösa en eventuell tvist genom att kontrollera att $E_A(C) = M$.

Konventionella system som DES möjliggör dataautenticitet, men inte avsändarautenticitet, eftersom både avsändare och mottagare förfogar över samma nyckel. Mottagaren skulle ju då kunna förfalska A:s signatur, och ingen tredje part skulle kunna lösa tvisten.

7.3 Informationsteori

För att kunna bedöma säkerheten i ett chiffer krävs insikter i en vetenskapsgren som heter *informationsteori.* Denna vetenskapsgrenkan grundades av Claude Shannon 1948. Informationsteori ger en teoretisk grund för bland annat kryptografi. Man kan mäta den teoretiska säkerheten hos ett chiffer med osäkerheten beträffande klartexten givet en chiffertext. Om man ingenting kan veta om klartexten oavsett hur mycket chiffertext man har tillgång till, har chiffret *perfekt sekretess.* Med ett undantag (se nedan) lämnar alla praktiska chiffer någon information om klartexten. När längden av tillgänglig chiffertext ökar, så avtar

normalt osäkerheten om klartexten mot noll. Man har då tillgång till tillräckligt mycket information för att entydigt bestämma klartexten.

De flesta chiffer är teoretiskt brytbara med bara ett par hundra bitars klartext. Detta betyder emellertid inte att dessa chiffer är osäkra, eftersom det kan vara beräkningsmässigt svårt att bestämma klartexten. Den viktiga frågan är som tidigare sagts inte huruvida ett chiffer är obetingat säkert, utan huruvida det är beräkningsmässigt säkert.

Informationsteori mäter *informationsmängd* i ett meddelande genom medelvärdet av antal bitar som går åt att koda alla möjliga meddelanden i en optimal kodning. Fältet KÖN i en databas innehåller t ex endast en informationsbit, eftersom det kan kodas med en bit ("man" med "0" och "kvinna" med "1"), medan fältet LÖN kräver fler bitar då många fler möjligheter måste täckas upp. Formellt mäts informationsmängden i ett meddelande med meddelandets *entropi*

$$H(X) = -\sum_{i=1}^{n} p(X_i)\log_2 p(X_i)\,.$$

Här är $X_1, \ldots X_n$ n möjliga meddelanden som förekommer med sannolikheterna $p(X_1), \ldots p(X_n)$ där $\sum_{i=1}^{n} p(X_i) = 1$. Det viktade medelvärdet $H(X)$ ger det förväntade antalet bitar i ett i ett optimalt kodat meddelande.

Exempel 7.4: Om vi antar att alla meddelanden är lika sannolika, dvs $p(X_i) = 1/n$ för i = 1, ... , n, så

$$H(X) = -\,n \cdot \frac{1}{n} \cdot \log_2 \frac{1}{n} = \log_2 n\,.$$

Sålunda krävs $\log_2 n$ bitar för att koda ett meddelande. Om $n = 2^k$ så $H(X) = k$ och k bitar behövs för att koda varje möjligt meddelande.

Givet n, så är $H(X)$ maximal om $p(X_1) = \ldots = p(X_n) = 1/n$, dvs om alla meddelanden är lika sannolika. $H(X)$ avtar när sannolikhetsfördelningen blir skevare, och når ett minimum, $H(X) = 0$, då $p(X_i) = 1$ för något meddelande X_i.

Exempel 7.5: Antag att X representerar en 32-bitars heltalsvariabel. Då kan X högst ha 32 informationsbitar. Om små värden på X är vanligare än stora (som det ofta är i t ex datorprogram) så $H(X) < 32$, och om det exakta värdet på X är känt, så $H(X) = 0$.

Entropin hos ett meddelande mäter dess osäkerhet i det att entropin ger antalet informationsbitar som måste läras när meddelandet t ex har blivit gömt i en chiffertext. Om exempelvis kryptoanalytikern vet att chiffertextblocket Z§3P7K svarar mot någon av klartexterna MAN eller KVINNA, så är osäkerheten bara en bit. Kryptoanalytikern behöver bara bestämma en symbol (den första) i klartexten för att kunna avgöra vilken klartext det är. Om man vet att blocket svarar mot en lön, så är osäkerheten mer än en bit, men inte längre än $\log_2 n$ bitar, där n är antalet möjliga löner.

För att försvåra statistiska analyser för kryptoanalytikern kan man eliminera överflödiga bitar innan chiffrering. Detta kan åstadkommas i datorsystem genom att packa meddelanden med hjälp av t ex Huffmankoder innan chiffrering.

Shannon har också föreslagit två chiffreringstekniker för att hindra eller försvåra attacker baserade på statistiska analyser - *confusion* och *diffusion*. Confusion innebär att sambandet

mellan nyckel och klartext skall vara så komplex som möjligt. Diffusion innebär att de statistiska egenskaperna hos klartexten fördelas över hela chiffertexten.

Många moderna chiffer, som t ex DES och RSA, åstadkommer confusion och diffusion genom komplexa chiffreringstransformationer över stora datablock. Dessa chiffertextblock kan dessutom göras funktionellt beroende av varandra vilket fördelar hela klartexten över hela chiffertexten. Meddelandet M delas upp i n lika stora block $M_1, M_2, \ldots, M_n$. I DES innehåller t ex varje block 64 bitar. I sin enklaste variant krypterar man varje 64-bitsblock för sig. Meddelandet $M = M_1M_2 \ldots M_n$ krypteras då $C = C_1C_2 \ldots C_n$. Tekniken kallas *Electronic Code Book* (ECB). Den mest uppenbara svagheten är att två identiska meddelandeblock krypteras lika och därmed ger man den fientlige kryptoanalytikern en gratismöjlighet att hitta en infallsvinkel när han skall försöka forcera chiffret. Det finns flera alternativ till ECB som chiffrerar lika meddelandeblock olika. En svårighet är att hitta tekniker som inte förstör hela meddelandet om några bitar överförs fel i en kommunikationskedja.

Med följande teknik, *Cipher Block Chaining* (CBC), förstörs högst två block om någon bit transmitteras fel. För att illustrera tekniken låter vi f var en chiffreringsavbildning. Vi antar dessutom att klartexten och chiffertexten är bitsträngar. Beräkna för $i = 1, 2, ..., n$ chiffertextblocket C_i enligt $C_i = f(M_i \oplus C_{i-1})$, där $\oplus$ är bitvis addition modulo två. Givet chiffertexten, kännedom om den inversa avbildningen f^{-1} och "fröet" C_0, sker dechiffrering med avbildningen $M_i = C_{i-1} \oplus f^{-1}(C_i)$. Formlerna visar varför eventuella fel inte fördelas ut över hela meddelandet och gör det oläsbart; klartextblock M_i beror bara på chifferblock C_i och C_{i-1}. I databastillämpningar är tyvärr ofta ECB nödvändig om data skall vara enkelt tillgängliga för läsning eller skrivning.

8 Några exempel på chiffer

I följande avsnitt beskrivs några viktiga typer av chiffer samt ges exempel på dylika.

8.1 Transpositionschiffer

8.1.1 Kolumntranspositionschiffer

har redan behandlats (exempel 7.1).

8.1.2 Periodiska permutationschiffer.

Ofta delas klartexten in i block med en fix längd (period) d. Inom varje period permuteras sedan klartextsymbolerna med en permutation f: {1, 2, …, d} → {1, 2, …, d}. Nyckeln blir k = (d, f). Successiva perioder chiffreras genom att permutera symbolerna i blocket enligt f. Ett klartextmeddelande

$$M = m_1 m_2 \ldots m_d m_{d+1} \ldots m_{2d}$$

chiffreras

$$C = m_{f(1)} m_{f(2)} \ldots m_{f(d)} m_{d+f(1)} \ldots m_{d+f(d)},$$

där $m_1 m_2 \ldots m_d$, $m_{d+1} \ldots m_{2d}$ och så vidare är block av längd d.

Exempel 8.1: Antag d = 4 och att f definieras av

i	1	2	3	4
f(i)	2	4	1	3.

Klartexten M = MATE MATI KENS LYCK ORUS chiffreras då till $E_K(M)$ = C = AEMT AIMT ESKN YKLC RSOU, dvs inom varje block flyttas den andra klartextsymbolen till position ett, den fjärde till position två etc. Vid dechiffrering används den inversa permutationen.

i	1	2	3	4
$f^{-1}(i)$	3	1	4	2.

Periodiska permutationschiffer kan, likt transpositionschiffer, uppfattas som transpositioner av kolonner i en matris i vilken meddelandet skrivits in radvis enligt följande.

1	2	3	4
M	A	T	E
M	A	T	I
K	E	N	S
L	Y	C	K
O	R	U	S.

Fördelen med periodiska permutationschiffer är att permutationerna sker rad för rad, vilket gör chiffret smidigare att hantera i datortillämpningar.

8.2 Substitutionschiffer

8.2.1 Enkla substitutionschiffer

8.2.1.1 Affina chiffer

Caesars chiffer (exempel 7.2) definieras av avbildningen $f(x) = (x + 3) \bmod m$, och är ett exempel på ett *affint chiffer*. En *affin avbildning* är en avbildning på formen $f(x) = (ax + b) \bmod m$. Denna avbildning gör sambandet mellan klartextsymboler och chiffertextsymboler något mer komplicerad än Caesars chiffer.

Exempel 8.2: Låt $f(x) = (17x + 5) \bmod 29$. Observera att $\gcd(17, 29) = 1$, varför f är inverterbar.
Klartexten M = HEJ chiffreras enligt

$f(H) = f(7) = (17 \cdot 7 + 5) \bmod 29 = 8 = I$
$f(E) = f(4) = (17 \cdot 4 + 5) \bmod 29 = 15 = P$
$f(J) = f(9) = (17 \cdot 9 + 5) \bmod 29 = 13 = N,$

dvs C = IPN.
Inversen, dechiffreringsavbildningen, kan beräknas till
$f^{-1}(x) = (12x + 27) \bmod 29$. (Gör det!)

Chiffer som ovanstående är utomordentligt enkla att bryta.

Exempel 8.3: I ett affint chiffer antas klartextsymbolen E (= 4) svara mot chiffertext-symbolen F (= 5) och klartextsymbolen H (= 7) mot W (= 22). Forcera chiffret.

Villkoren ger $\begin{cases} f(4) = 5 \bmod 29 \\ f(7) = 22 \bmod 29 \end{cases}$

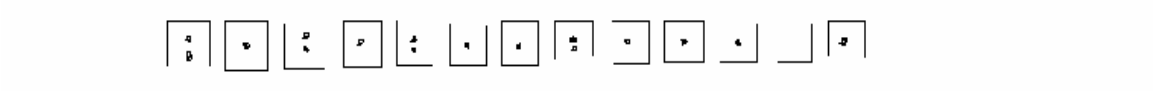
Detta tillsammans med chiffreringsalgoritmen $f(x) = (ax + b) \bmod 29$ ger följande ekvationssystem.

$$\begin{cases} 4a + b \equiv 5 \pmod{29} \\ 7a + b \equiv 22 \pmod{29} \end{cases}$$

Systemet har lösningen a = 25 och b = 21. Genomför själv detaljerna. Observera att ekvationssystemet är entydigt lösbart i modulen! Varför? Det är inte alltid så. Huruvida systemet är entydigt lösbart i modulen eller ej beror på modul och sifferuppgifter.

8.2.1.2 Kyrkogårdschiffer

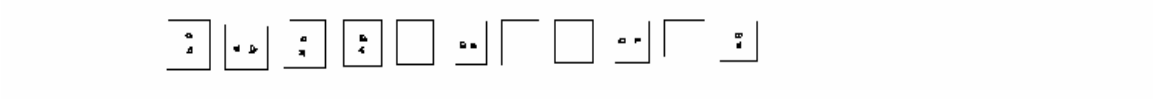
Exempel 8.4: Följande "chiffertext" hittades på en gravsten på Trinity Churchyard i New York 1794.



Därav namnet kyrkogårdschiffer.
Chiffertexten ovan har naturligtvis en engelsk klartext, men det tog lång tid innan någon kunde forcera chiffret. Ett liknande chiffer på svenska skulle kunna beskrivas med nyckeln

A :	B :	C :		J	K	L		S ·	T ·	U ·		Ä··	Ö··	
D :	E :	F :		M	N	O		V ·	W·	X ·				
G :	H :	I :		P	Q	R		Y ·	Z ·	Å·				

och meddelandet M = DÖDENÄRNÄRA chiffreras



Ovanstående typer av chiffer är triviala. De är normalt entydigt dekrypterbara om man har tillgång till en chiffertext med ca 30 symboler.

8.2.2 Homofona substitutionschiffer

Enkla substitutionschiffer bryts enkelt genom att undersöka de olika symbolernas frekvens. Ett sätt att försvåra frekvensanalyser är att låta varje klartextsymbol motsvaras av flera chiffertextsymboler. I ett homofont substitutionschiffer avbildas sålunda varje symbol a i klartextalfabetet A ini en mängd f(a) av chiffersymboler kallade *homofoner*. Avbildningen f från klartextalfabet till chiffertextalfabet är alltså på formen

$$f: A \rightarrow P(C),$$

där P(C) är potensmängden till C.

Ett klartextmeddelande $M = m_1m_2 \ldots m_n$ chiffreras $C = c_1c_2 \ldots c_n$, där varje c_i väljs slumpmässigt från mängden av homofoner $f(m_i)$ till m_i. Ju fler homofoner det finns till varje symbol i klartextalfabetet, desto starkare chiffer kan konstrueras. I gränsfallet där varje bokstav i klartextmeddelandet chiffreras till en ny chiffertextsymbol, kan chiffret vara omöjligt att bryta.

8.2.2.1 Bealechiffer

I över ett århundrade har amatörkryptoanalytiker försökt forcera ett chiffer, som man tror beskriver var en skatt finns. Denna skatt skall ha grävts ner någonstans i Virginia ca 1820 av ett gäng äventyrare ledda av Thomas Jefferson Beale. Chiffret, som antas beskriva var skatten finns, är det första av tre chiffer som Beale lämnade efter sig. Det andra löstes ca 1880 och beskriver den påstådda skatten, samt säger att det första beskriver var den finns. Det tredje chiffret antas vara en lista över äventyrarnas anförvanter.

Det andra chiffret är ett exempel på ett homofont substitutionschiffer. Nyckeln är USA:s Declaration of Independence vars ord numreras från ett och framåt i den ordning de förekommer i deklarationen. Beale chiffrerade varje klartextsymbol genom att ange numret på ett ord i vilket symbolen var första bokstav.

Exempel 8.5: Låt avsnitt tre i detta kompendium vara nyckel, där rubriken inte räknas så "med" är ord nr ett. Vi tar dessutom bara hänsyn till löpande text varför exempel och tabeller hoppas över. Klartexten M = MATEMATIK kan då chiffreras

C = 22 25 33 9 1 13 54 20 88.

Exempel 8.6: Låt nyckeln vara given av följande tabell över homofoner

Bokstav	Homofoner
A	17 15 38 59
E	11 03 18 28 71
I	13 00 51 07
K	05 16 37
M	14 61 12 93
T	08 31 19 27 39 45

M = MATEMATIK kan chiffreras
C = 141527036117310716.

Chiffer av Beales typ är ohanterliga (varför?). Chiffertypen i exempel 8.6 är mer hanterbar, men kräver för att vara starkt många homofoner, att antalet homofoner för respektive bokstav är valt så att bokstavfrekvenser effektivt döljs samt täta byten av nyckel.

8.2.3 Polyalfabetiska substitutionschiffer

Eftersom enkla substitutionschiffer använder en enda avbildning från klartext- till chiffertext-symboler, så bevaras bokstävernas frekvensfördelning i chiffertexten. Ett sätt att dölja fördelningen är att använda homofona substitutionschiffer, ett annat är att använda sig av flera olika avbildningar som i polyalfabetiska substitutionschiffer. De flesta sådana är periodiska med en period d. Man har då d st chiffertextalfabet $C_1, C_2, \ldots C_d$.

Låt $f_i : A \rightarrow C_i$, med $1 \leq i \leq d$, vara avbildningar från klartextalfabetet A till respektive chiffertextalfabet. En klartext

$M = m_1m_2 \ldots m_dm_{d+1}$ chiffreras då

$C = f_1(m_1)f_2(m_2) \ldots f_d(m_d)f_1(m_{d+1}) \ldots.$

8.2.3.1 Vigenérechiffer

Detta är ett periodiskt substitutionschiffer baserat på skiftade alfabet. Låt nyckeln vara $k = k_1k_2 \ldots k_d$, där k_i för $i = 1, 2, \ldots, d$ ger förskjutningens storlek i det i:te alfabetet. Chiffreringsalgoritmen är $f_i(x) = (x + k_i) \bmod n$.

Exempel 8.7: Låt k = KRYPTO, modulen n = 29 och M = MATEMATIKÄRROLIGT.
Vi får $f_1(M) = (M + K) \bmod n = (12 + 10) \bmod 29 = 22 = X$
$f_2(A) = (0 + 17) \bmod 29 = 17 = R$, osv, men detta arrangeras mer överskådligt på följande sätt.

M = MATE MATI KÄRR OLIG T
k = KRYP TOKR YPTO KRYP T
C = XROT COAZ FNHC YTDV J

Mellanslagen finns bara med av läsbarhetsskäl.

De inversa substitutionerna blir $f_i^{-1}(x) = (x - k_i) \bmod n$ (Varför?). Det finns flera olika varianter av ovanstående konstruktion.

Man skulle kunna misstänka att ett sätt att konstruera ett säkert chiffer skulle vara att låta nyckeln vara lika lång som meddelandet. Detta är nämligen ett nödvändigt villkor för att ett chiffer skall vara säkert. Dock måste dessutom varje symbol i nyckeln vara slumpmässigt vald, och så är inte fallet om vi t ex låter nyckeln vara detta dokument där vi börjar avläsa bokstäver i och med rubriken till avsnitt två. Följande exempel illustrerar detta.

Exempel 8.8: M = MATE MATI KÄRR OLIG T
k = FUNK TION SBEG REPP E
C = RUDD CIEV ÖÖVX DQXV X

I ovanstående korta exempel med ett ovanligt ord som nyckel framgår förstås inte problemen. Men vissa bokstavskombinationer är mycket vanligare än andra oavsett vilket språk det gäller, och sådan information kan en fientlig kryptoanalytiker utnyttja. Chiffret ovan är dock obetingat säkert om varje symbol i nyckeln är slumpmässigt genererad, nyckeln är lika lång som klartexten samt används endast en gång.

8.2.3.2 One-Time Pads

De ovan avslutningsvis skisserade villkoren ger villkoren för ett obetingat säkert chiffer. Chiffer av denna typ kallas one-time pads. Att dylika chiffer inte kan brytas illustreras med följande exempel.

Exempel 8.9: Antag att vi har snappat upp följande chiffertext på ett eller annat sätt, samt att vi vet att det är ett chiffertexten till ett one-time pad-chiffer.

C = ... XVYAAC .....

Eftersom varje nyckel är lika sannolik skulle det kunna vara så att nyckeln är

k_1 = ... QRPLAS ....., med följande dekrypterade meddelande
M = ... HEJSAN .....

Men det är precis lika sannolikt att nyckeln är

k_2 = ... NELLTÄ ....., med klartexten
M = ...KANSKE ....

(Kontrollera att respektive dechiffrering är korrekt utförd).

Det finns alltså ingen information kvar i chiffertexten som kan användas för att avslöja klartexten.

8.2.3.3 Vernamchiffer

Dessa konstruerades 1917 av Gilbert Vernam, och är one-time pads i vilka klartext och nyckel är lika långa bitsträngar. Chiffreringsalgoritmen är

$c_i = f(m_i) = m_i \oplus k_i$, där $\oplus$ är bitvis addition modulo två.

Dechiffrering sker enligt

$m_i = f(c_i) = c_i \oplus k_i$.

Observera att $k_i \oplus k_i = 0$ varför $f(c_i) = c_i \oplus k_i = (m_i \oplus k_i) \oplus k_i = m_i$.

Exempel 8.10: Låt M = 1101 0110 och k = 0110 1000. Då blir
C = 1011 1110.

Observera alltså att ovanstående chiffer är obetingat säkra under förutsättning att nyckeln är lika lång som klartextmeddelandet, symbolerna i nyckeln är slumpmässigt valda och nyckeln används endast en gång. Kan man ha någon nytta av dylika chiffer?

För *Data encryption standard (DES)* hänvisas till Denning (se litteraturlista)

Övningsuppgifter

8.1 Kontrollera att $f^{-1}(x) = (12x + 27) \bmod 29$ i exempel 8.2.

8.2 Lös ekvationssystemet i exempel 8.3.

8.3 Dekryptera kyrkogårdschiffret i exempel 8.4.

8.4 Konstruera ett eget homofont substitutionschiffer.

8.5 Visa att $f_i^{-1}(x) = (x - k_i) \bmod n$ i exempel 8.7.

8.6 Kontrollera dechiffreringen i exempel 8.9.

8.7 Använd dig av nyckeln BAD och chiffrera klartexten MINERA med Vigenéres metod. Använd som vanligt A = 00, B = 01, osv.

8.8 I ett affint substitutionschiffer $f(x) = (ax + b) \bmod 29$ antas P ha chiffrerats till Q och W till D.

a/ Bestäm f genom att beräkna a och b.
b/ Chiffrera klartexten KOLONN.
c/ Bestäm inverstransformationen.
d/ Dechiffrera GYX.

8.9 Ett hemligt meddelande M chiffreras med ett periodiskt permutationschiffer med blocklängd 8 och nyckel 73614852.

a/ Chiffrera

M = BEGÄRANAVSLÅSINLEDOMEDELBARTOPERAT IONENENLIGTORDER.

b/ Dechiffrera

C = ALLTLGSILTPMUAPOIOCGCNHANDÅLSDMEDFRA JEÄGENLJILKUVA.

8.10 a/ Chiffertexten

BSSBYHOVW CO AYÖAWYURNHLJSBCYÄYHLRÖW YKSBHBHYÖKSLJZCBYQDY CYLJQQYLBGG YUJHYLBCHSAPU G CYZDCWBN CYWCYOEVNJCÄJCW YG IAVBAÄYLSJPL BCY SÖB

har framställts med ett enkelt substitutionschiffer. Bestäm klartexten med hjälp av en frekvensanalys. De vanligaste bokstäverna i svenska språket är i ordning A, E, N, T, R, S, L, I, O, D. Det är även möjligt att blanktecken blivit

chiffrerat som en bokstav.

b/ Undersök om chiffret är affint. Eftersom blanktecken kan förekomma kan det vara bra att testa med modulen 30.

9 RSA-systemet

RSA-systemet är ett system med offentlig nyckel (se ovan avsn 7). Dylika system föreslogs först av Diffie och Hellman 1975. Det viktigaste av de hitintills föreslagna systemen med offentlig nyckel är RSA-systemet som föreslogs av Rivest, Shamir och Adlemann 1977/78. Den matematiska grunden för systemet är, att det är lätt att hitta stora primtal, men beräkningsmässigt svårt att faktorisera produkter av stora primtal. Vilken storleksordning de primtal som bör användas skall ha är fortfarande en diskussionsfråga. I litteraturen förekommer uppgifter från ca 100 siffror till ca 200 siffror. I texten nedan används uppgiften att primtalen bör vara av storleksordning 10^{150}. Det kan också vara värt att notera att ingen har bevisat att det inte kan finnas snabba algoritmer för att faktorisera stora tal. Upptäcks dylika är RSA-systemet ointressant.

För att konstruera ett chiffer enligt RSA-systemet går vi till väga på följande sätt.

1. Välj två primtal p och q av storleksordning 10^{150}, och beräkna $n = pq$. Då är $n \approx 10^{300}$.

2. Välj e så att $\gcd(e, \varphi(n)) = 1$. För e skall dessutom gälla att $e > \max(p, q)$ och $e > \log_2 n$. Då är chiffreringsnyckeln $\langle e, n \rangle$ klar. Chiffreringsalgoritmen är $E : \mathbf{Z}_n \to \mathbf{Z}_n$, $C = E(M) = M^e \bmod n$. Här skall $M < n$. Om meddelandet är längre än n delas det upp i block med längd mindre än n.

3. För dechiffrering krävs att vi hittar ett tal d sådant att $ed \equiv 1 \pmod{\varphi(n)}$. Dechiffreringsnyckeln blir då $\langle d, n \rangle$, och dechiffreringsalgoritmen blir $D : \mathbf{Z}_n \to \mathbf{Z}_n$, $M = D(C) = C^d \bmod n$.

4. Förstör slutligen p, q, $\varphi(n)$ och publicera e och n tillsammans med chiffreringsanvisningar.

Sats 9.1: Det gäller att $D(E(M)) = M$ med beteckningar enligt ovan.

Bevis: $D(E(M)) = D(M^e \bmod n) = (M^e \bmod n)^d \bmod n = M^{ed} \bmod n = M^{\varphi(n)\cdot k+1} \bmod n =$
$= ((M^{\varphi(n)})^k \cdot M) \bmod n = M$.
Observera att $M^{\varphi(n)} \equiv 1 \pmod n$ (Eulers sats), och $ed \equiv 1 \pmod{\varphi(n)}$ (punkt 3 ovan) ger att $ed = \varphi(n)\cdot k + 1$ för något heltal k.
Alltså gäller $D(E(M)) = M$.

Exempel 9.1: Vi skall konstruera ett "låtsasexempel" för att illustrera idéerna ovan. I detta exempel behöver man inte kraftfullare räknetekniska hjälpmedel än en vanlig funktionsräknedosa.

1. Välj $p = 59$ och $q = 53$.
   Då är $n = pq = 3127$, och $\varphi(n) = \varphi(3127) = (59 - 1)(53 - 1) =$
   $= 3016$.

2. Välj $e = 543$. Kontrollera att $\gcd(543, 3016) = 1$ (Gör det!).
   Chiffreringsnyckel $\langle e, n \rangle = \langle 543, 3127 \rangle$.
   Chiffreringsalgoritm $C = M^{543} \bmod 3127$.

3. Bestäm d så att $ed \equiv 1 \pmod{3016}$, dvs hitta en lämplig lösning till $543d + 3016y = 1$.

Euklides´ algoritm ger

$$3016 = 5\cdot 543 + 301$$
$$543 = 1\cdot 301 + 242$$
$$301 = 1\cdot 242 + 59$$
$$242 = 4\cdot 59 + 6$$
$$59 = 9\cdot 6 + 5$$
$$6 = 1\cdot 5 + 1$$
$$5 = 5\cdot 1$$

Algoritmen baklänges ger $1 = 6 - 1\cdot 5 = 6 - (59 - 9\cdot 6) = 10\cdot 6 - 59 =$
$= -59 + 10\cdot(242 - 4\cdot 59) = 10\cdot 242 - 41\cdot 59 =$
$= 10\cdot 242 - 41\cdot(301 - 1\cdot 242) = -41\cdot 301 + 51\cdot 242 =$
$= -41\cdot 301 + 51\cdot(543 - 1\cdot 301) = 51\cdot 543 - 92\cdot 301 =$
$= 51\cdot 543 - 92\cdot(3016 - 5\cdot 543) = 511\cdot 543 - 92\cdot 3016,$

dvs $d = 511$.

Dechiffreringsnyckel $\langle d, n\rangle = \langle 511, 3127\rangle$.
Dechiffreringsalgoritm $M = C^{511} \bmod 3127$.

4. Det gäller att $M < 3127$, varför vi måste dela upp ett meddelande i block så att detta uppfylls. Låt A = 00, B = 01, C = 02, ... Vi kan då tillåta blocklängden två bokstäver. Om vi vill chiffrera meddelandet M = MATEMATIKX gör vi enligt följande.

MA	1200
TE	1904
MA	1200
TI	1908
KX	1022,

dvs M = 1200 1904 1200 1908 1022.
Blocklängden blir sålunda fyra decimala siffror valda med hjälp av alfabetet ovan.

För chiffrering av MA beräknas $1200^{543} \bmod 3127$, dvs
$E(MA) = E(1200) = 1200^{543} \bmod 3127$. Detta beräknas med hjälp av upprepad kvadrering enligt tidigare modell.

$543 = (1000011111)_2$

Sätt $r = 1$

$n_0 = 1$ $r = 1200$
$a_1 = 1200^2 \bmod 3127 = 1580$

$n_1 = 1$ $r = (1200 \cdot 1580) \bmod 3127 = 1038$
$a_2 = 1580^2 \bmod 3127 = 1054$

$n_2 = 1$ $r = (1038 \cdot 1054) \bmod 3127 = 2729$
$a_3 = 1054^2 \bmod 3127 = 831$

$n_3 = 1$ $r = (2729 \cdot 831) \bmod 3127 = 724$

$a_4 = 831^2 \bmod 3127 = 2621$

$n_4 = 1$ $\quad r = (724 \cdot 2621) \bmod 3127 = 2642$
$a_5 = 2621^2 \bmod 3127 = 2749$

$n_5 = 0$ $\quad r = 2642$
$a_6 = 2749^2 \bmod 3127 = 2169$

$n_6 = 0$ $\quad r = 2642$
$a_7 = 2169^2 \bmod 3127 = 1553$

$n_7 = 0$ $\quad r = 2642$
$a_8 = 1553^2 \bmod 3127 = 892$

$n_8 = 0$ $\quad r = 2642$
$a_9 = 892^2 \bmod 3127 = 1406$

$n_9 = 1$ $\quad r = (2642 \cdot 1406) \bmod 3127 = 2903$

Alltså $\quad E(1200) = 2903$

För dechiffrering beräknas $D(2903) = 2903^{511} \bmod 3127$. Genomför dessa beräkningar och kontrollera att $D(2903) = 1200$. Efter lite räknande får man

C = 2903 1084 2903 3021 2042.

Vi ser här nackdelen med att chiffrera varje block för sig. Blocket MA förekommer två gånger och chiffreras båda gångerna till 2903. Detta kan dock åtgärdas med tidigare nämnd teknik (se avsnitt 7).

9.1 Om primtal och RSA-systemet

För att kunna beräkna d, så krävs kunskap om värdet på $\varphi(n) = \varphi(pq)$. För att kunna beräkna $\varphi(n)$ krävs att vi kan faktorisera n. Det finns för tillfället inga kända algoritmer för detta som är tillräckligt beräkningsmässigt effektiva.

Om p och q inte är väl valda, så kan n vara lätt att faktoruppdela.

9.1.1 Hur välja primtal?

Det rekommenderas att de valda primtalen skall vara så kallade "säkra" primtal. Med detta avses primtal på formen $p = 2p' + 1$, där även p' är ett primtal. En del författare rekommenderar att även p' bör vara på denna form, medan andra menar att det inte spelar någon roll om man väljer "säkra" primtal eller ej. Dessutom bör inte p och q ha lika många siffror. Ett annat krav är att både $p - 1$ och $q - 1$, där p och q är de för konstruktionen av ett chiffer enligt RSA-systemet valda primtalen, bör innehålla stora primfaktorer.

Generering av primtalen kan göras så att siffrorna i talen genereras slumpmässigt, varefter man använder sannolikhetstester för att avgöra huruvida talen är primtal eller ej.

9.1.2 Primtalstester

Det gäller att, om gcd(a, p) = 1 och p är ett primtal, så $a^{p-1} \equiv 1 \pmod{p}$, vilket ger att $a^p \equiv a \pmod{p}$ (Fermats sats, se ovan). Observera att omvändningen av satsen inte gäller!

Exempel 9.2: Att p i Fermats sats inte måste vara ett primtal inses av följande motexempel

$n = 341 = 11 \cdot 31, \qquad 2^{341} \equiv 2 \pmod{341}$ (Kontrollera!)

Den kontrapositiva formuleringen av Fermats sats är

Om a^n inte är kongruent med a modulo n, så är n inte ett primtal.

Detta kan användas för att visa att ett tal inte är ett primtal.

Exempel 9.3: 63 är inte ett primtal, ty

$2^{63} = 2^{60} \cdot 2^3 = (2^6)^{10} \cdot 2^3 = 64^{10} \cdot 8 \equiv 1^{10} \cdot 8 = 8 \pmod{63}$,

och 8 är definitivt inte kongruent med 2 modulo 63.

Definition 9.1: Talet n kallas ett *pseudoprimtal till basen* b om och endast om b > 0, n är sammansatt och $b^n \equiv b \pmod{n}$.

Exempel 9.4: Så är t ex 341 ett pseudoprimtal till basen 2 enligt exempel 9.2.

Det gäller att pseudoprimtalen är glesare fördelade än primtalen.

9.1.3 Millers test för basen b

Låt n > 0, och låt $n - 1 = 2^s \cdot t$, där s > 0 och t är ett udda positivt tal. Vi säger att ett tal n klarar Millers test för basen b, om $b^t \equiv 1 \pmod{n}$ eller om $b^{2^j \cdot t} \equiv -1 \pmod{n}$ för något heltal j, sådant att $0 \le j \le s - 1$.

Sats 9.2: Om n är ett primtal och b är ett positivt heltal där n inte delar b, så klarar n Millers test för basen b.

Om n inte är ett primtal kan man utnyttja följande sats.

Sats 9.3 (Rabins probabilistiska primtalstest) Låt n vara ett positivt heltal. Välj k styck olika positiva heltal som alla är mindre än n, och genomför Millers test på n för var och en av dessa k baser. Om n är sammansatt, så är sannolikheten att n klarar alla k testerna mindre än $(1/4)^k$.

Exempel 9.5: k = 100 ger $(1/4)^{100} \approx 10^{-60}$, som är en rätt liten sannolikhet.

Definition 9.2: Definiera funktionen π: $\mathbf{Z}_+ \to \mathbf{N}$, $\pi(x)$ = antal primtal som är mindre än eller lika med x.

Exempel 9.6: Det gäller att $\pi(3) = 2$, $\pi(8) = 4$ osv.

Följande sats är ett mycket avancerat resultat i matematik och bevisades först 1896 av Hadamard och de la Vallée-Poussain oberoende av varandra. Gauss hade dock redan 1793 uppställt som en hypotes att sambandet gäller. Satsen ger en uppskattning av funktionen π

för stora x med hjälp av elementära funktioner.

Sats 9.4: Primtalssatsen $\dfrac{\pi(x)}{x/\ln x} \to 1$, då $x \to \infty$.

Detta resultat ger då att sannoliketen att ett udda tal som är mindre än eller lika med x är ett primtal är

$$\frac{\pi(x)}{x/2} \approx \frac{2x}{x \cdot \ln x} = \frac{2}{\ln x}.$$

Låter vi så x vara ett av storleksordning 10^{150} får vi att x är ett primtal med sannolikheten

$$\frac{2}{\ln 10^{150}} = \frac{2}{150 \cdot \ln 10} \approx \frac{1}{173}.$$

Vi kan alltså förvänta oss behöva undersöka cirka 200 heltal för att hitta ett hundrafemtiosiffrigt primtal. Detta klarar en snabb dator med effektiva algoritmer på några sekunder.

Det finns andra tekniker än den ovan skisserade med Millers test, och detta är ett högaktuellt forskningsområde, varför den intresserade hänvisas till speciallitteratur.

Följande resultat används för att testa huruvida mersennetal (se uppgift 9.7) är primtal eller ej.

Sats 9.5: (Lucas-Lehmers test) Låt p vara ett udda primtal, och låt $M_p = 2^p - 1$ vara det p:te mersennetalet.
Definiera

$$r_1 = 4$$
$$r_k = (r_{k-1}^2 - 2) \bmod M_p, \quad k \geq 2$$

Då gäller att M_p primtal om och endast om $r_{p-1} \equiv 0 \pmod{M_p}$.

Exempel 9.6: Låt $p = 5$. Då är $M_5 = 2^5 - 1 = 31$.

Sätt $r_1 = 4$
Då blir $r_2 = (16 - 2) \bmod 31 = 14$
$r_3 = (14^2 - 2) \bmod 31 = 8$
$r_4 = (8^2 - 2) \bmod 31 = 0$

Alltså är 31 ett primtal i kraft av sats 9.5.

Denna algoritm är snabb. Det är möjligt att avgöra huruvida M_p är ett primtal med högst $O(p^3)$ bitoperationer.

Övningsuppgifter

9.1 Bevisa att $E(D(C)) = C$. Jämför med sats 9.1.

9.2 Varför är $e = 521$ ett olämpligt val då $p = 59$ och $q = 53$? Jämför med exempel 9.1.

9.3 Kontrollera och räkna färdigt exempel 9.1.

9.4 Konstruera ett eget RSA-system. Detta innebär att bestämma p och q, bestämma lämpligt e, beräkna d och att ge lämpliga chiffreringsanvisningar.

9.5 Konstruera två system och använd dem för att visa hur autenticitet och sekretess kan garanteras. Du kan ju t ex använda exempel 9.1 och övning 9.4.

9.6 Ett meddelande M har i ett RSA-system chiffrerats till 31 enligt följande ekvation.

$$M^{43} \bmod 77 = 31$$

Bestäm M.

9.7 Om $m \in \mathbf{Z}_+$ så kallas $M_m = 2^m - 1$ det *m:te mersennetalet,* och om det gäller för ett primtal p att $M_p = 2^p - 1$ också är ett primtal så kallas M_p ett *mersenneprimtal.* Undersök om M_7, M_{11}, M_{13} är mersenneprimtal. Pröva gärna med något riktigt stort tal också.

9.8 Konstruera några pseudoprimtal.

10 Blandade uppgifter till kapitel 4 - 9

Uppgifterna 1 - 14 kan användas vid repetition av teori. Ett flertal av nedanstående uppgifter har förekommit på tidigare tentamina.

10.1. Låt p och q vara två primtal, och låt φ vara Eulers funktion. Bevisa att $\varphi(pq) = \varphi(p)\,\varphi(q)$.

10.2. a/ Definiera uttrycken *a ≡ b (mod m)* och *a mod m*.
b/ Visa att $a \equiv b \pmod m$ om och endast om $a \bmod m = b \bmod m$.

10.3. a/ Definiera begreppet *delare*.
b/ Bevisa att om $a|b$ och $a|c$, så $a|xb+yc$ för alla heltal x och y.

10.4. a/ Definiera *Eulers funktion* φ.
b/ Formulera och bevisa Eulers sats. Du behöver inte bevisa de hjälpsatser du använder.

10.5. Låt E och D vara chiffrerings- respektive dechiffreringsfunktion i ett RSA-system med modulen n, dvs $E(M) = M^e \bmod n$ och $D(C) = C^d \bmod n$, där e och d är multiplikativa inverser modulo $\varphi(n)$.
Visa att $D(E(M)) = M$, där M uppfyller villkoret $0 < M < n$.

10.6. a/ Definiera *gcd(a, b)*, dvs den *största gemensamma delaren* till a och b.
b/ Visa att om $a = qb + r$, så är $\gcd(a, b) = \gcd(b, r)$. Här är a, b, q och r heltal som uppfyller villkoren för att respektive största gemensamma delare skall vara definierad.

10.7. a/ Definiera begreppet delare.
b/ Visa att om $a \mid b$ och $b \mid a$, så gäller att $a = \pm b$.

10.8. a/ Förklara hur både sekretess och autenticitet kan åstadkommas inom RSA.
b/ Definiera begreppen *envägsfunktion (one way function)* och *envägsfunktion fallucka (one way trapdoor function)*.
c/ Är det svårt att forcera (”knäcka”) DES?

10.9. a/ Förklara kortfattat vad ett *polyalfabetiskt substitutionschiffer* är.
b/ Vilka villkor måste vara uppfyllda för att ett chiffer skall vara *obetingat säkert*?
c/ Vilka krav ställs på en *digital signering*.

10.10. a/ Varför kan sekretess och autenticitet hållas isär inom RSA men inte i DES?
b/ Diskutera säkerheten i RSA respektive DES.

10.11. a/ Vad är en *envägsfunktion* (one-way function)?
b/ Vad är en *envägsfunktion med fallucka* (trapdoor one-way function)?

10.12. a/ Vad är ett *periodiskt permutationschiffer*? Vad händer med symbolernas relativa frekvenser med dylika chiffer?
b/ Vad är ett *homofont substitutionschiffer*? Vilken är poängen med att införa homofoner?
c/ Vad är ett *one-time pad*? Vilken är poängen med dylika?

10.13. a/ Vad är ett produktchiffer?
b/ Förklara begreppen sekretess och autenticitet.
c/ Vilka krav måste vara uppfyllda för att ett chiffer skall vara obetingat säkert?

10.14. a/ Uppskatta hur lång tid det tar att forcera DES via ”brute force”, dvs genom att kontrollera samtliga nycklar. Antag att 10^9 nycklar kan kontrolleras per sekund. (Det gäller att $\lg 2 \approx 0{,}3$.)
b/ Varför möjliggör RSA-systemet, men inte DES, digital signering?
c/ Vilken är den matematiska grunden för RSA-systemets säkerhet?

10.15. Bestäm om möjligt en lösning till $368x + 523y = 2$.

10.16. Bestäm om möjligt en lösning till $352x + 421y = 3$. Bestäm också samtliga lösingar.

10.17. Bestäm om möjligt en lösning till $628x + 411y = 2$.

10.18. Bestäm samtliga lösningar till $321x + 508y = 5$ med hjälp av Euklides´ algoritm.

10.19. Konstruera dechiffreringsnyckeln till ett RSA-chiffer med chiffreringsnyckel $e = 23$ och $n = 221$.

10.20. I ett förmodat affint chiffer gissar man att O = 14 chiffreras till F = 5, P = 15 till H = 7 och R = 17 till N = 13. Är detta möjligt vid räkning med modulen 29?

10.21. För ett affint chiffer gäller att $f(x) = (15x + 21) \bmod 34$. Dechiffrera meddelandet 3, dvs beräkna $f^{-1}(3)$.

10.22. I ett ”RSA-system” med $e = 17$ och $n = pq = 77$ skall ”meddelandet” $M = 75$ chiffreras. Vad blir C?

10.23. a/ Beräkna $\varphi(2500)$.
b/ Beräkna $10^{17} \bmod 17$.
c/ Vilka tal satisfierar den kvadratiska kongruensen $x^2 + 1 \equiv 0 \pmod 5$?

10.24. a/ Beräkna $\varphi(200)$.
b/ Beräkna $(-211) \bmod 48$.
c/ Beräkna $35^{321} \bmod 36$

10.25. Bestäm f^{-1} då $f(x) = (33x + 8) \bmod 56$.

10.26. Bestäm multiplikativ invers till 68 modulo 319.

10.27. Lös följande system av kongruenser $\begin{cases} x \equiv 1 \pmod 3 \\ 2x \equiv 3 \pmod 5 \\ 2x \equiv 4 \pmod 8 \end{cases}$.

10.28. Lös följande system av kongruenser $\begin{cases} x + 4y \equiv 10 \pmod{14} \\ 2x + 4y \equiv 12 \pmod{14} \\ 3x + 2y \equiv 10 \pmod{14} \end{cases}$.

10.29. Bestäm, t ex genom att använda kinesiska restsatsen, vilka tal som har rest ett vid division med fyra, rest tre vid division med fem och rest fem vid division med sju.

10.30. Lös $\begin{cases} 3x + 4y \equiv 7 \pmod{14} \\ 5x + 6y \equiv 8 \pmod{14} \end{cases}$

11 Tips och lösningsförslag till vissa övningsuppgifter

Eftersom det är min allvarliga pedagogiska övertygelse att ett "facit" enbart är av ondo, och inte tillför något, så följer här enbart några tips hur vissa uppgifter kan lösas. Uppgifter vars "lösning" lätt kan kontrolleras av läsaren/lösaren själv ser jag det som meningslöst att bifoga ett "svar" till. Ett skäl för ovanstående strategi är att man i ett "facit" förstås inte kontrollerar huruvida ens lösning är korrekt utan kontrollerar om man fått samma svar som författaren. Detta kan ju vara trevligt, men ännu trevligare är ju att veta om man gjort rätt. Ett annat skäl är att man, om det nu finns ett "facit", skaffar sig felaktiga betingade reflexer. Det är bättre att lösa fem uppgifter och att själv kontrollera att man gjort och resonerat korrekt, än att lösa tio uppgifter och kontrollera om man gjort likadant som författaren.

1.3	För att visa att $A \subseteq B$ skall man visa att $\forall x(x \in A \rightarrow x \in B)$. Visa då att $x \in \emptyset \rightarrow x \in A$ är falskt för varje x.
1.4, 5	Svaret i båda fallen är "nej" - Hitta motexempel.
1.7	Ja. Visa att $A \subseteq B$ och $B \subseteq A$, under förutsättningen att $P(A) = P(B)$.
1.8	Om $A \cap B = \emptyset$, så $\lvert A \cup B \rvert = n + m$. Hur blir det om $A \cap B \neq \emptyset$?
2.4	Tag godtyckligt $y \in f(S \cap T)$, let reda på motsvarande $x \in S \cap T$, osv. Konstruera lämplig funktion som inte är injektiv.
2.8	Multiplicera $(f^{-1} \circ g^{-1})$ med $(g \circ f)$ från vänster, utnyttja att funktionssammansättning är associativ (dvs att $f \circ (g \circ h) = (f \circ g) \circ h$) och drag slutsatsen att $(g \circ f)^{-1}$ (inversen till $(g \circ f)$) är just $f^{-1} \circ g^{-1}$.
3.2	Funktionen är begränsad för stora x, dvs för x större än något heltal k.
3.3	Utnyttja definitionen av O.
3.4	Utnyttja att $\sum_{i=1}^{n} i^k = 1^k + 2^k + ... + n^k \leq n \cdot n^k = n^{k+1}$. Vad kan vi få för stortordo-uppskattning av $\sum_{i=0}^{n} k^i$ (Använd formeln för summan av en geometrisk talföljd)?
3.5	$O(n^2)$ - Varför?
4.7h	Leta efter ett par primtalstvillingar.
4.9	Använd definition av delare.
4.10	Om $2^p - 1$ är ett primtal, så är $2^{p-1}(2^p - 1)$ perfekt. Försök visa detta. Varje jämnt perfekt tal kan skrivas på detta sätt. Inget udda perfekt tal är känt. Skriv gärna ett litet program på miniräknare eller dator.
5.4	Observera att det i många fall är det mycket omständigt att beräkna multiplikativ invers, varför det i dessa fall är en mycket olämplig metod.

12 Litteratur

Det finns massor med information om kryptering på internet, varför sökning där rekommenderas. I övrigt har jag framför allt använt nedanstående litteratur.

Becket B	1988	Introduction To Cryptology Blackwell
Bovet D P, m fl	1994	Introduction to the Theory of Complexity Prentice Hall
Brassard G	1988	Modern Cryptology Springer
Denning D E R	1982	Cryptography And Data Security Addison-Wesley
Koblitz N	1987	A Course In Number Theory And Cryptography Springer
Kranakis E	1987	Primality And Cryptography Wiley-Teubner
Rosen K H	1995	Discrete Mathematics and its Applications McGraw-Hill
Rosen K H	1988	Elementary Number Theory And Its Applications Addison-Wesley
Singh S	1999	Kodboken, Norstedts, Stockholm
SIS, TR 312	1985	Kryptering i ADB-system