Avsnittene 4.4, 4.5 og 4.6 fra læreboken

Disse avsnittene inneholder eksempler på bruk av modulo- og kongruensregning. Her tar vi opp kun litt av dette.

Avsnitt 4.4 Kongruensligninger

La a, b og m være hele tall der $m > 0$.
Da får vi flg. generelle kongruensligning:

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

Vi skal finne en x der $0 \leq x < m$ slik at kongruensen er sann.

Eksempel La $a = 3$, $b = 5$ og $m = 7$.

Løs kongruensligningen $3x \equiv 5 \pmod{7}$.

Vi kan bruke "prøving og feiling".

x	$3x \equiv 5 \pmod{7}$	sann/usann
0	$0 \equiv 5 \pmod{7}$	usann
1	$3 \equiv 5 \pmod{7}$	usann
2	$6 \equiv 5 \pmod{7}$	usann
3	$9 \equiv 5 \pmod{7}$	usann
4	$12 \equiv 5 \pmod{7}$	sann

$x = 4$ er løsningen av $3x \equiv 5 \pmod{7}$.

Avsnitt 4.5 Anvendelser av kongruensregning

Hashing Dette er en viktig teknikk i databehandling. Det er en effektiv måte å lagre verdier slik at det er lett å finne dem igjen.

Det engelske ordet **hash** betyr å hakke opp i småbiter. Hvis vi f.eks. skal lagre tegnstrenger (ord), kan vi "hakke dem opp i småbiter" ved å se på de enkelte tegnene som småbitene. Så bruker vi "småbitene" til å konstruere et **stort** heltall.

Java har en ferdig metode for dette. Den heter **hashCode()**.

```
String s = "Kari";
int hash = s.hashCode();
System.out.println(hash);  //utskrift: 2331181

int hash2 = 'i' + 'r'*31 + 'a'*31*31 + 'K'*31*31*31;
System.out.println(hash2); //utskrift: 2331181
```

Metoden hashCode() er kodet slik at hvis "Kari" er tegnstrengen, blir hashverdien lik

$$K \cdot 31^3 + a \cdot 31^2 + r \cdot 31 + i = 2331181$$

der tallverdien (ascii-verdien) til bokstavene inngår.

Vi skal nå lagre tegnstrengen "Kari" i en tabell med lengde n.

Da legger vi den på indeks lik hash % n.

Hvis f.eks. n = 97 (et primtall), blir "Kari" lagt inn på indeks 2331181 % 97 = 77.

Hvis en tegnstreng får en indeks som er opptatt, kalles det en kollisjon. Da finnes det ulike teknikker for å legge strengen et annet sted, men slik at det likevel er lett å finne den. Dette er tema i faget Algoritmer og datastrukturer.

Kontrollsiffer

Kontrollsiffer brukes for å unngå at feilkoder som brukes til identifikasjon, skrives feil. Eksempler på dette er fødselsnummer (11 siffer), ISBN-nummer for bøker (10 eller 13 siffer), KID-nummer for regninger og mange andre.

I Oblig 2 tas ISBN-13 opp. Her skal vi se kort på ISBN-10. Den bestemmer bokforlaget de 9 første sifrene. Siffer nr. 10 er et kontrollsiffer.

La de 9 første sifrene være $s_1, s_2, s_3, \ldots, s_9$.
Siffer nr. 10, dvs. s_{10}, bestemmes på denne måten:

$$s_{10} = \left(\sum_{i=1}^{9} i \cdot s_i\right) \bmod 11 = (1 \cdot s_1 + 2 \cdot s_2 + 3 \cdot s_3 + \cdots + 9 \cdot s_9) \bmod 11.$$

Når vi bruker mod 11, kan resten bli fra 0 til 10.
Hvis resten blir 10, bruker vi isteden bokstaven X (romertall 10).

Eksempel Forrige utgave av vår lærebok i Diskret matematikk hadde flg. ISBN-10 kode:

$$007-124474-3$$

Her får vi $1\cdot 0+2\cdot 0+3\cdot 7+4\cdot 1+5\cdot 2+6\cdot 4+7\cdot 4+8\cdot 7+9\cdot 4$ $= 179$. $179 \bmod 11 = 3$. Det stemmer derfor at 3 er kontrollsiffer.

Avsnitt 4.6 Kryptering

En mye brukt krypteringsteknikk kalles RSA (Rivest, Shamir, Adleman).

Her bruker man store primtall (flere hundre siffer) og kongruensregning til å konstruere krypteringsnøkler. Du finner mye informasjon om RSA-teknikken på internett. F.eks. på

http://no.wikipedia.org/wiki/RSA