Forelesning 2

Flere pseudokoder. Representasjoner av tall.

Dag Normann - 16. januar 2008

KONTROLLSTRUKTURER

Mandag innførte vi pseudokoder og kontrollstrukturer.
Vi hadde tre typer grunn-instruksjoner:

- Input *variabel*
- *variabel* $\leftarrow$ *term*
- Output *variabel*

Kontrollstrukturer

Vi hadde fem kontrollstrukturer

- **If** $\cdots$ **then**
- **If** $\cdots$ **then** $\cdots$ **else**
- **While** $\cdots$ **do**
- **Repeat** $\cdots$ **until**
- **For** $\cdots$ **to** $\cdots$ **do**

Vi skal se på noen flere eksempler på pseudokoder.

Kontrollstrukturer

Det er ingen som vet om algoritmen som er beskrevet i den neste pseudokoden vil terminere for alle input.
Det betyr at den muligens **ikke** er en algoritme i bokas forstand.
Den forutsetter at vi kan skille mellom partall og oddetall.

Eksempel (Ubegrenset while-løkke)

1 Input x [$x \geq 1$ heltall.]
2 **While** $x > 1$ **do**
  2.1 **if** x partall **then**
    2.1.1 $x \leftarrow \frac{x}{2}$
    else
    2.1.2 $x \leftarrow 3x + 1$
3 Output x

Kontrollstrukturer

Vi kan finne en enkel pseudokode for å finne ledd nr. n i følgen

$$1, 1, 2, 3, 5, 8, 13, 21, 34, 55, \cdots$$

av Fibonacci-tall:

Eksempel (Fibonacci)

1 Input n [$n \geq 1$ heltall]
2 $x \leftarrow 1$
3 $y \leftarrow 1$
4 **For** $i = 2$ **to** n **do**
  4.1 $z \leftarrow x$
  4.2 $x \leftarrow x + y$
  4.3 $y \leftarrow z$
5 Output x

Kontrollstrukturer

Parenteser i forskjellige former brukes mye i matematikk, informatikk og spesielt i programmer.

Eksempelvis, under utarbeidelsen av denne beamer- presentasjonen forekommer "parenteser" som

$$\backslash begin\{center\} \cdots \backslash end\{center\}$$

og

$$\backslash begin\{eksempel\} \cdots \backslash end\{eksempel\}$$

en rekke steder, og det er viktig at de står riktig i forhold til hverandre.

Dette kontrolleres når råmanuskriptet kompileres.

Det neste eksemplet på en pseudokode er en parentes-sjekker.

Kontrollstrukturer

Vi kan kontrollere om en liste venstre og høyreparenteser

$$(\;) \; (\; (\;) \;) \;) \; (\;) \; (\; (\;) \;) \; (\;)$$

er lovlig eller ikke, ved å telle opp eller ned - opp ved (og ned ved) - fra venstre mot høyre.

Hvis vi til slutt ser at vi har like mange parenteser av hvert slag, og aldri underveis har flere) enn (, er uttrykket i orden.
I den neste eksemplet gir vi to input, lengden av uttrykket og sekvensen av parenteser.
Vi sjekker uttrykket fra venstre mot høyre.
Vi bruker variabelen val til å holde orden på om sekvensen s å langt er i orden.
Vi bruker variabelen y til å telle overskuddet av (.

Kontrollstrukturer

1 Input n [Lengden av uttrykket, antall parenteser totalt]
2 Input $x_1 \cdots x_n$ [En liste av venstre og høyreparenteser]
3 $y \leftarrow 0$
4 $val \leftarrow$ JA
5 **For** $i = 1$ **to** n **do**
  5.1 **if** $x_i =$ (**then**
    5.1.1 $y \leftarrow y + 1$
    else
    5.1.2 **if** $y = 0$ **then**
      5.1.2.1 $val \leftarrow$ NEI
      else
      5.1.2.2 $y \leftarrow y - 1$
6 **if** $y > 0$ **then**
  6.1 $val \leftarrow$ NEI
7 Output val

OVER TIL KAPITTEL 2

TALLMENGDER

Hvilke tall vi betrakter er avhengig av hva vi ønsker å bruke dem til.

I MAT1030 vil vi stort sett betrakte følgende typer tall:

- Naturlige tall $\mathbb{N}$

  $$1, 2, 3, \cdots$$

- Hele tall $\mathbb{Z}$

  $$\cdots, -3, -2, -1, 0, 1, 2, \cdots$$

- Rasjonale tall $\mathbb{Q}$

  Tall som kan skrives som en brøk $\frac{p}{q}$

- Reelle tall $\mathbb{R}$

  "alle tallene"

Tallmengder

Mange mener at tall er punkter på tall-linja, og at det ikke spiller noen rolle om vi betrakter 2 som et naturlig tall, et heltall, et rasjonalt tall eller et reelt tall.

I programmeringsammenheng kan det spille en stor rolle hva slags verdier en variabel kan få lov til å ta, og representasjonen av et tall som et dataobjekt kan variere med hva slags type tall vi betrakter.
Dette skal vi se nærmere på siden.

Tallmengder

Det finnes andre tallmengder som også er av interesse i matematikk og informatikk,eksempelvis

- Komplekse tall
- Algebraiske tall
- Kvaternioner

REPRESENTASJON AV TALL

Så langt tilbake vi har informasjon om, har mennesker og kulturer hatt muntlig og skriftlig språk for tall.

Romertallet

MCMXXVIII

er en alternativ måte å skrive

1928

på.

Hvis vi blir bedt om å skrive et program for addisjon av to tall, betyr det mye om vi bruker den romerske eller dagens måte å skrive tall på.

Representasjon av tall

De tallene vi bruker til daglig kalles desimaltall, eller tall i 10-tallsystemet.

Dette er et plass-siffersystem med basis 10.
Det betyr igjen at hvert siffer angir et antall 10'er potenser, og sifferets posisjon forteller oss hvor stor potensen er.

Representasjon av tall

Eksempel

a) 258 står for

$$2 \cdot 10^2 + 5 \cdot 10^1 + 8 \cdot 10^0.$$

b) $3,14$ står for

$$3 \cdot 10^0 + 1 \cdot 10^{-1} + 4 \cdot 10^{-2}$$

Tverrsumtesten

Tverrsummen til et desimaltall er summen av alle sifrene.

Eksempel

- Tverrsummen til 234 er $2+3+4=9$
- Tverrsummen til 15987 er $1+5+9+8+7=30$
- Tverrsummen til 2825 er $2+8+2+5=17$

Legg merke til at resten vi får når vi deler tallet på 9 er det samme som vi får når vi deler tverrsummen på 9.
Kan dette forklares matematisk?

Tverrsumtesten

Påstand (Tverrsumtesten)
Hvis vi skriver et tall n på desimalform og lar $T(n)$ være tverrsummen til n, så får vi samme rest når vi deler n på 9 som når vi deler $T(n)$ på 9.

Tverrsumtesten

Bevis
La

$$a_k \dots a_0$$

være desimalformen til n. Da er

$$n = a_k 10^k + \cdots + a_1 10 + a_0$$

Hvis $1 \leq i \leq k$ kan $10^i - 1$ deles på 9, siden sifrene består av bare 9-tall.
Vi har at

$$n = T(n) + a_k(10^k - 1) + \cdots + a_1(10 - 1)$$

(trenger litt ettertanke), og påstanden følger (trenger litt ettertanke til).

BINÆRE TALL

Det er kulturelt betinget at vi bruke 10 som basis i tallsystemet vårt.

Alle tall > 1 kan i prinsippet brukes.

I informatikksammenheng er det like naturlig å bruke 2, 8 og 16 som basistall.

Bruker vi 2 som basis, sier vi at tallet er på binær form.

Binære tall

Eksempel
Vi tolker en binær form (hvor alle sifrene er 0 eller 1) omtrent som om det var et desimaltall, bortsett fra at vi erstatter 10 med 2:

- $1010_2 = 1 \cdot 2^3 + 0 \cdot 2^2 + 1 \cdot 2^1 + 0 \cdot 2^0 = 8 + 2 = 10_{10}$
- $11011_2 = 16 + 8 + 2 + 1 = 27$
- $100111001_2 = 256 + 32 + 16 + 8 + 1 = 313$

Binære tall

Binære tall er viktige i informatikk fordi digitalisering innebærer at all informasjon skal lagres som 0'er og 1'ere, i bits.

Vi skal komme tilbake til hvordan tall representeres i datamaskiner, men bruk av binær representasjon er helt sentral der.

Binære tall

Binær representasjon kan selvfølgelig også brukes til desimaltall.

- $0,100101_2 = \frac{1}{2} + \frac{1}{16} + \frac{1}{64} = \frac{32+4+1}{64} = \frac{100101_2}{64}$
- $0,01101_2 = \frac{1}{4} + \frac{1}{8} + \frac{1}{32} = \frac{8+4+1}{32} = \frac{01101_2}{32}$

Binære tall

Det finnes en enkel prosedyre for å regne ut verdien av et binært tall:

1 Input n [Antall sifre]
2 Input $x_1 \ldots x_n$ [en sekvens av 0'er og 1'ere]
3 $y \leftarrow 0$ [Skal bli verdien på sekvensen tolket som et binært tall]
4 **For** $i = 1$ **to** n **do**
  4.1 $y \leftarrow 2y$
  4.2 **If** $x_i = 1$ **then**
    4.2.1 $y \leftarrow y + 1$
5 Output y

Vi gir et regneeksempel, med input $n = 4$ og $x_1x_2x_3x_4 = 1101$, på tavla.

Binære tall

Det finnes også prosedyrer for å regne ut verdien av binære tall på desimalform.

Vi skal se på binære tall på formen $0,100110111000101$ og liknende, og hvilken algoritme vi kan bruke for å finne verdien av $0,100110111000101_2$.

1 Input n
2 $e \leftarrow 1$
3 **For** $i = 1$ **to** n **do**
  3.1 $e \leftarrow \frac{e}{2}$
4 Input $x_1 \dots x_n$ [Det binære tallet som står bak komma]
5 $y \leftarrow 0$
6 **For** $i = 1$ **to** n **do**
  6.1 $y \leftarrow 2y$
  6.2 **If** $x_i = 1$ **then**
    6.2.1 $y \leftarrow y + 1$
7 $y \leftarrow y \cdot e$
8 Output y

Binære tall

Det er selvfølgelig viktig å kunne finne verdien av et tall skrevet på binær form.

Det er også viktig å kunne finne binærformen til et tall.

Vi skal se på to pseudokoder, en som finner binærformen til et heltall, og en som finner binærformen til et tall mellom 0 og 1.

I begge tilfeller vil vi finne siffer nr. k fra komma.

Binære tall

Binærformen til et heltall: Skriver ut siffer nr. k i binærformen til n.

1 Input n
2 Input k
3 **For** $i = 1$ **to** k **do**
  3.1 **If** n like tall **then**
    3.1.1 $n \leftarrow \frac{n}{2}$
    3.1.2 $z \leftarrow 0$
    else
    3.1.3 $n \leftarrow \frac{n-1}{2}$
    3.1.4 $z \leftarrow 1$
4 Output z