INSTITUT FOR MATEMATIK OG DATALOGI

SYDDANSK UNIVERSITET

Skriftlig eksamen

D M 2 2

Fredag den 11. juni 2004 kl. 9.00 – 13.00

Opgavesættet består af 4 sider inklusive denne forside, og indeholder 4 opgaver. Opgaverne vægtes på følgende måde:

Opgave 1:	25 %
Opgave 2:	25 %
Opgave 3:	25 %
Opgave 4:	25 %

Det bemærkes at spørgsmålene indenfor de enkelte opgaver ikke nødvendigvis vægtes lige meget.

Alle skriftlige hjælpemidler samt brug af lommeregner er tilladt. Der må gerne henvises til resultater fra lærebøgerne, men henvisninger til andre bøger er ikke acceptable som led i besvarelsen.

Opgave 1 (25 % ∼ 60 minutter)

Vi betragter følgende Haskell program:

```
mystery xs  = snd (ms (0,0) xs)
ms (x:xs) (c,s)
        = ms (cc,ss) xs
        where cc = max 0 (c + x)
              ss = max s cc
ms [] (c,s) = (c,s)
```

NB: Fejl: Rækkefølgen af parametre til `ms` er ikke konsekvent den samme.

1.a Angiv typesignaturerne for `mystery` og `ms`.

1.b Hvad er resultaterne af kaldene `mystery [1,2,-3,4,5]` og `mystery []`?

1.c Hvad er tidskompleksiteten af funktionen `mystery`? Resultatet skal begrundes.

1.d Vis ved induktion, at givet argumentlisten $[a_0, \cdots, a_{n-1}]$, da beregner funktionen `mystery` værdien af $M_n = \max_{0 \leq j \leq k \leq n}(S_{jk})$ hvor $S_{jk} = \sum_j^{k-1} a_i$.

Opgave 2 (25 % ∼ 60 minutter)

I forbindelse med den nuværende politiske diskussion om strukturrevision ser man på fjernsynet danmarkskort hvor amter eller kommuner er farvelagte. Det er et klassisk problem (nu løst) om det kan lade sig gøre at farvelægge et sådant landkort med kun fire farver, således at to tilstødende lande ikke har samme farve.

Skriv et Prolog program der givet en liste af lande, hver med en liste af tilstødende lande ("naboer"), foretager en sådan farvelægning af landkortet. Databasen tænkes at bestå af "facts" på formen `naboer(holland,[belgien,tyskland]).` Resultatet af programmet skal være en liste `[holland/gul,belgien/blaa,...]`.

NB: `holland/gul` og `Land/Farve` er lovlige repræsentationer af tupler i Prolog. Du skal selvfølgelig ikke definere hele databasen for Europa, et par stykker eller tre til illustration er nok.

Opgavesættet fortsættes

Opgave 3 (25 % $\sim$ 60 minutter)

Haskell lærebogen har flere eksempler på mere eller mindre effektive programmer der beregner Fibonacci tallene. Disse er som bekendt rekursivt definerede ved $f_n = f_{n-1}+f_{n-2}$, med fx. initialværdier $f_0 = 1$ og $f_{-1} = 0$. Det hurtigste af bogens programmer genererer alle Fibonacci tallene op til f_n i lineær tid, $\mathcal{O}(n)$. Vi skal her lave et program der bestemmer det n'te Fibonacci tal i logaritmisk tid, $\mathcal{O}(\log n)$. Ideen hertil er baseret på at rekursionen kan skrives på matrix-form:

$$\left\{ \begin{array}{c} f_n \\ f_{n-1} \end{array} \right\} = \left\{ \begin{array}{cc} 1 & 1 \\ 1 & 0 \end{array} \right\} \left\{ \begin{array}{c} f_{n-1} \\ f_{n-2} \end{array} \right\} = \cdots = \left\{ \begin{array}{cc} 1 & 1 \\ 1 & 0 \end{array} \right\}^{n-1} \left\{ \begin{array}{c} 1 \\ 0 \end{array} \right\}.$$

Det er nu simpelt at vise at for passende værdier af a og b gælder der:

$$A = \left\{ \begin{array}{cc} 1 & 1 \\ 1 & 0 \end{array} \right\} \quad \Rightarrow \quad A^i = \left\{ \begin{array}{cc} 1 & 1 \\ 1 & 0 \end{array} \right\}^i = \left\{ \begin{array}{cc} a+b & b \\ b & a \end{array} \right\}.$$

Man kan derfor benytte "del-og-hersk" metoden til at finde det n'te Fibonacci tal, idet der så også gælder at:

$$A^{2i} = A^i A^i = \left\{ \begin{array}{cc} a+b & b \\ b & a \end{array} \right\} \left\{ \begin{array}{cc} a+b & b \\ b & a \end{array} \right\} = \left\{ \begin{array}{cc} p+q & q \\ q & p \end{array} \right\}$$

I en tidligere eksamensopgave indgik et Haskell program til potensopløsning af flydende tal, her er det i lettere omskrevet form:

```
pow :: Float -> Int-> Float
pow x n
    | n <  0           = 1/pow x (-n)
    | n == 0           = 1
    | n `mod` 2 == 0   = pow (x*x) (n `div` 2)
    | otherwise        = x * pow (x*x) ((n-1) `div` 2)
```

Dette program er netop baseret på "del-og-hersk" metoden, og har logaritmisk køretid.

3.a Udtryk værdierne af (p, q) ved (a, b). Bemærk at denne beregning svarer til kvadreringen i programmet `pow`, samt at matricerne er entydigt bestemt ved deres højre søjle.

3.b Hvilken beregning skal nu være den analoge til beregningen af `x * ...` i otherwise tilfældet i `pow`?

3.c Skriv et Haskell program der udnytter ovenstående til beregning af det n'te Fibonacci tal, således at det har logaritmisk tidskompleksitet.

Opgavesættet fortsættes

Opgave 4 (25 % $\sim$ 60 minutter)

4.a Definer et prædikat `palin(A)` som undersøger om listen `A` er et palindrom, dvs. at den læses ens forfra og bagfra. Vis dernæst sekvensen af delmål under en succesfuld udførelse af målet (goal): `palin([1,2,1]).`

4.b Find en "most general unifier" for hvert af følgende par af prædikater, såfremt en sådan eksisterer:

1: $f(X, g(Y))$ og $f(g(Z), Z)$,

2: $f(g(X), X)$ og $f(Y, h(Y))$,

3: $f(X, g(X), b)$ og $f(a, g(Z), Z)$,

4: `append(W,W,[1,2,1,2])` og `append([X|A],B,.(X,.(C,[])))`

Såfremt ingen løsning kan findes skal der argumenteres herfor.

4.c Reducer følgende logiske udtryk i sædvanlig prædikatnotation:

$$\forall X(\exists Y(c(X,Y) \Rightarrow (\neg b(X) \wedge \exists Z(a(Y,Z) \wedge d(X)))))$$

til konjunktiv normalform og videre til clausal normalform, idet alle skridt dokumenteres.

Opgavesættet slutter her