TENTAMEN I DVA 201 FUNKTIONELL PROGRAMMERING MED F#

Torsdagen den 20 augusti 2015, kl 8:10 – 12:30

Kurslitteratur är inte tillåten, och inte eller andra hjälpmedel som på något sätt kan ersätta kurslitteraturen (t.ex. egna anteckningar, andra böcker i ämnet, kopior av OH-bilder, datorer eller räknare med dito lagrad information). Endast generella hjälpmedel är tillåtna, som räknare utan lagrad information av betydelse för kursen, ordbok, allmän formelsamling och liknande. För godkänt krävs 15 poäng, max är 30 poäng. Resultatet offentliggörs senast torsdagen den 10 september 2015.

Vänligen observera följande:

- Motivera alltid dina svar. Bristande motivering kan ge poängavdrag. Omvänt kan även ett felaktigt svar ge poäng, om det framgår av motiveringen att tankegången ändå är riktig.
- Skriv tydligt!
- Varje blad skall vara försedd med uppgiftsnummer och bladnummer.
- Endast en uppgift på ett och samma blad.
- Skriv enbart på ena sidan av ett blad.
- Uppgifterna är inte nödvändigtvis sorterade i svårighetsgrad. Om du kör fast kan det löna sig att gå vidare till nästa uppgift.
- Lösningsförslag kommer att finnas på kursens hemsida efter att tentan är slut.

Frågor: Björn Lisper på 021-151709.

UPPGIFT 1 (7 POÄNG)

Deklarera en funktion `sumpos`, som tar en lista av heltal (`int`) som argument och returnerar summan av alla positiva tal i listan!

a) Ge en deklaration som använder direkt rekursion. *(4p)*

b) Ge en deklaration som använder F#:s inbyggda högre ordningens funktioner på listor på ett icke-trivialt sätt. *(3p)*

UPPGIFT 2 (2 POÄNG)

Betrakta följande F#-deklarationer:

```
let f x = x + x
let g = fun x -> x + x
```

Blir det någon skillnad på funktionerna `f` och `g` och i så fall vilken?

UPPGIFT 3 (4 POÄNG)

`List`-modulen innehåller en funktion `List.partition : ('a -> bool) -> 'a list -> ('a list * 'a list)`. Denna funktion tar ett predikat `p` och en lista `l` som argument: den returnerar ett par av listor, där den första listan innehåller alla element i `l` för vilka predikatet `p` är sant och den andra listan innehåller de element i `l` för vilka `p` är falskt. T.ex. gäller att

```
List.partition (fun x -> x > 0) [1;-1;2;3;-2] = ([1;2;3],[-1;-2])
```

Ge din egen deklaration av `List.partition`!

UPPGIFT 4 (2 POÄNG)

Betrakta följande F#-deklarationer:

```
let rec f x = if x > 0 then f (x+1) else 0
let g x y = if x = 0 then 1 else f 1
```

a) Vad blir resultatet av anropet `g 0 (f 1)` om vi använder ivrig evaluering (call-by-value)? Motivera ditt svar. *(1p)*

b) Vad blir resultatet av anropet `g 0 (f 1)` om vi använder lat evaluering (call-by-need)? Motivera ditt svar. *(1p)*

UPPGIFT 5 (4 POÄNG)

Deklarera en funktion `addarray` som tar en array av flyttal (`float`) som argument och returnerar summan av dess element. Din funktion ska utifrån sett inte ha några sidoeffekter, men ska internt använda en lokalt deklarerad muterbar referenscell där summan successivt ackumuleras. Funktionen ska använda rekursion – imperativa F#-loopar är inte tillåtna. Du får heller inte använda någon av de inbyggda funktionerna i F# som kan summera arrayer direkt, som `Array.sum`.

UPPGIFT 6 (6 POÄNG)

a) Deklarera en polymorf datatyp för binära träd, där noder kan ha 0, 1 eller 2 söner och data lagras i löven (och enbart där). *(2p)*

b) Deklarera en funktion som tar ett träd av typen som deklarerats i *a)*, som innehåller flyttal (`float`), och som summerar talen som är lagrade i trädet. *4p)*

UPPGIFT 7 (5 POÄNG)

Funktionen `list_and` definieras av

```
let rec list_and a =
    if a = [] then true else List.head a && list_and (List.tail a)
```

a) Vad beräknar funktionen `list_and`? *(1p)*

b) Härled en typ för `list_and`! För full poäng ska den härledda typen vara den mest generella. Ordentlig motivering krävs. *(4p)*

Ledning: det gäller att `List.head : 'a list -> 'a` och `List.tail : 'a list -> 'a list`.

Lycka till! *Björn*