Tentamen

Programspråksteori (5p)

14 januari 2000

Skrivtid: 16.00-22.00. Inga hjälpmedel
Totalpoäng: 40 (Godkänt 20)

Uppgift 1 (3p)

Skulle det vara meningsfullt att tillämpa abstraktionsprincipen på
a) literaler?
b) typer?
Motivera ditt svar!

a) Nej, eftersom de inte innehåller några detaljer att abstrahera bort
b) Ja, en typ utgörs av värden och operationer på värdena, alltså ingår någon sorts beräkning, och abstraktionsprincipen säger att abstraktion är möjlig när den syntaktiska klassen omfattas av någon sorts beräkning.

Uppgift 2 (3p)

Sammansatta datatyper kan härledas till i princip fem olika kategorier. Namnge och beskriv tre av dessa.

Kartesisk produkt – en tupel, dvs består av två eller flera element som kan vara av olika typ. Record i Pascal, struct i C.
disjunkt union –värden väljs ut från en av två möjliga och olika typer. Värdet ”taggas” för att markera från vilken av typerna det valts. Records med variantdel ibland annat i Pascal.
avbildning – den avbildar ett värde x från mängden S till ett värde y i mängden T.
powerset – mängden av alla delmängder av S, ”set of T” i Pascal
rekursiva atyper – Typer som definieras i termer av sig själva

Uppgift 3 (3p)

Härled typerna för följande ML-funktioner givet att typen för `not` är Bool → Bool, och att typen för ∘ är $(\beta \rightarrow \gamma) \times (\alpha \rightarrow \beta) \rightarrow (\alpha \rightarrow \gamma)$.

```
fun negation (p) = not o p;
fun cond (b, f, g) =
        fn x =>
        if b (x) then f (x) else g (x)
```

negation: ('a -> Bool) -> ('a -> Bool)
cond: ('a -> Bool) x ('a -> 'b) x ('a -> 'b) -> ('a -> 'b)

Uppgift 4 (3p)

Redogör för två olika sätt att testa typekvivalens.

Strukturekvivalens: innebär att två typer är ekvivalenta om de innehåller exakt lika mängd värden (Algol-68)
Namnekvvivalens: innebär att endast typer som deklarerats på samma ställe anses vara lika (Pascal)

Uppgift 5 (5p)

Välj en av följande två varianter (a eller b) av den här uppgiften:
Jämför följande två programspråk utifrån fördelar, nackdelar, likheter och skillnader, och knyt detta till hur språken kan passa som nybörjarspråk.
a) Ada och C++
eller
b) Prolog och Haskell

Här har ni svarat väldigt olika av naturliga skäl, men också kommit fram till skilda slutsatser för vilket språk som passat bäst som nybörjarspråk. Har man bara motiverat sitt svar och fått med alla delar som efterfrågas i uppgiften har man fått full poäng.

Uppgift 6 (3p)

Vilket är det huvudsakliga syftet med ett kommando i ett programspråk? Vilken programmeringsparadigm kopplas vanligtvis ihop med kommandon?

Det huvudsakliga syftet för kommandon är att uppdatera variabler. Det är främst i språk som tillhör den imperativa språkparadigmen som kommandon hör hemma.

Uppgift 7 (4p)

Beskriv vad som menas med statisk bindning respektive dynamisk bindning.
Hur ordnar man så att en metod får statisk respektive dynamisk bindning i Java?

Vid statisk bindning evalueras t ex en funktionskropp i den omgivning där den definierats, bidning sker vid kompileringen.
Vid dynamisk bidning evalueras funktionskroppen i den omgivning där anropet görs, bindning sker vid exekvering.
I Java är dynamisk bindning default, för att åstadkomma statisk bindning används modifieraren "final".

Uppgift 8 (4p)

Abstraktion kan kort beskrivas som att man bortser ifrån detaljinformation.
Diskutera varför detta är så viktigt i programmeringssammanhang, och ge några exempel på abstraktioner.

Genom att ett språk stöder abstraktion främjar det bl a modularisering, vilket underlättar utvecklingen av stora projekt.
Abstraktion kan också medverka till att kod blir lättare att läsa genom att uppdelning görs i exempelvis funktioner och procedurer.
Även återanvädning av kod stöds genom abstraktion.
Exempel på abstraktioner (generellt) är:
Funktionsabstaktion - abstraktion över uttryck
Procedurabstraktion - abstraktion över kommandon
Generisk abstraktion - abstraktion över deklaration

Uppgift 9 (3p)
Beskriv utförligt vad som menas med inkapsling.

Inkapsling innebär att data kapslas in tillsammans med operationer/metoder som är relevanta för dessa data. Detta görs för att skydda data, åtkomst ska endast kunna ske via operationerna. Den som använder det som inkapslats behöver endast bry sig om *vad* som görs och inte hur, och vilket interface som ger access.

Uppgift 10 (3p)
Vad står följande begrepp för:
- oberoende (independent) process
- tävlande (competing) process
- kommunicerande process

Oberoende process delar inte resurser med någon annan process och därmed spelar det ingen roll i vilken ordning den utförs relativt andra processer, resultatet blir ändå detsamma.
Två tävlande processer vill båda komma åt samma resurs och den som först kommer åt den har exklusiv rätt till den tills den använt den klart. Här beror resultatet på vilken av processerna som först kommer åt den gemensamm resursen.
En process kallas kommunicerande om den är beroende av data från någon annan process för att kunna fortsätta.

Uppgift 11 (3p)
Beskriv vad som menas med lat evaluering och redogör för fördelar och nackdelar med detta.

Lat evaluering innebär att ett argument bara evalueras när det behövs. Första gången det evalueras sparas det sedan undan för eventuell åtkomst senare.
Detta möjliggör separation av kontroll och beräkning. Det gör också att oändliga listor blir möjliga.
Tyvärr är lat evaluering svårt att implementera effektivt. Det är inte heller lämpligt tillsammans med sidoeffekter som ställer till problem.

Uppgift 12 (3p)
Vad är kännetecknande för logikprogrammering?

Hög abstraktionsnivå
Ingen modularisering
Satser uppbyggda av relationer, löses med resolution
Deklarativt
Iteration genom rekursion