

Tentamen (del 2) (4 högskolepoäng) i Programkonstruktion och datastrukturer (1DL201)

Lars-Henrik Eriksson

Onsdag 20 mars 2013, kl. 08:00 – 11:00, i Bergsbrunnagatans skrivsal 1

Hjälpmedel: Inga. ***Inte*** heller elektronisk utrustning.

Hjälp: Läraren kommer att besöka skrivsalen c:a klockan 09:00.

Anvisningar: Markera i tabellen nedan ***inte mer än ett*** svar per fråga genom att ***kryssa över*** bokstaven för det svarsalternativ som du väljer. ***Lämna bara in denna sida.*** Det är inte meningen att du skall lämna kommentarer till dina svar. Om du tycker att någon fråga är oklar eller felaktig, markera fråganumret med en ★ på ***den här*** sidan, och förklara ***på baksidan av detta blad*** vad du menar att problemet är och vilka antaganden du gjort för att kunna svara på frågan.

	Fråga	Svar					Fråga	Svar				
betyg 3 U	1	A	B	C	D	E	2	A	B	C	D	E
	3	A	B	C	D	E	4	A	B	C	D	E
	5	A	B	C	D	E	6	A	B	C	D	E
	7	A	B	C	D	E	8	A	B	C	D	E
	9	A	B	C	D	E	10	A	B	C	D	E
betyg 4	11	A	B	C	D	E	12	A	B	C	D	E
	13	A	B	C	D	E	14	A	B	C	D	E
	15	A	B	C	D	E						
betyg 5							16	A	B	C	D	E
	17	A	B	C	D	E	18	A	B	C	D	E
	19	A	B	C	D	E	20	A	B	C	D	E

Identitet: Din tentakod (eller namn och personnummer om du saknar kod):

.....

Frågor för betyg 3

Om du ger rätt svar på 7 av de 10 frågorna i detta avsnitt så blir du godkänd med minst betyg **3**, annars blir du underkänd (**U**). Du kan inte kompensera ett dåligt resultat i detta avsnitt med poäng från frågorna för betyg **4** eller **5**.

1. Titta på detta programfragment för en abstrakt datatyp som hanterar vektorer:

```
abstype vector2 = V of real*real with
  fun scalarproduct(V(x1,y1),V(x2,y2)) = x1*x2+y1*y2;
  fun toCartesian(V(x,y)) = (x,y)
  ...
end;
```

Man vill nu skriva ett uttryck som beräknar längden av vektorn `v`. Följande tre uttryck skulle alla fungera och ge rätt resultat ifall `vector2` var en vanlig `datatype` och inte en `abstype` (du behöver alltså inte fundera över matematiken; den är rätt):

- (a) `Math.sqrt(scalarproduct(v,v))`
- (b) `let val (x,y) = toCartesian v in Math.sqrt(x*x+y*y) end`
- (c) `let val V(x,y) = v in Math.sqrt(x*x+y*y) end`

Vilka av dessa uttryck fungerar när nu `vector2` är en `abstype`?

- (A) (a) (B) (a) & (b) (C) Alla tre (D) (a) & (c) (E) (b)

Motivering: Konstruktorn `V` får inte användas utanför den abstrakta datatypen.

2. Titta på denna funktionsdefinition:

```
fun fullmetal(_,elric,[]) = [elric]
  | fullmetal(alchemist,elric,mustang::bradley) =
    if alchemist(elric,mustang) then
      elric::mustang::bradley
    else
      mustang::fullmetal(alchemist,elric,bradley)
```

Vad är värdet av uttrycket `fullmetal(op >,3,[5,4])`?

- (A) `[3,5,4]` (B) `[5,3,4]` (C) `[5,4,3]` (D) `[5,4]` (E) `[3]`

Motivering: `fullmetal` lägger in 3:an in i listan `[5,4]` före det första element som 3 är större (`op >`) än. Eftersom inget sådant element finns läggs 3:a istället sist.

3. Man matar i tur och ordning in följande deklarationer och uttryck till ett ML-system:

```
val x = ref [1,2,3,4];
fun pop(x) = let
    val top = hd (!x)
in
    (x := tl (!x); top)
end;
x := [11,12,13,14];
[pop x, pop x, pop x];
```

Vad blir värdet av det sista uttrycket [pop x, pop x, pop x]?

(A) [1,2,3] (B) [1,1,1,1] (C) [11,12,13] (D) [12,13,14] (E) [11,11,11]

Motivering: Definitionen av **pop** påverkar inte **x**. **x** sätts till [11,12,13,14]. Sedan anropas **pop** tre gånger. Varje anrop ger tillbaka första elementet i listan som **x** refererar till och plockar bort detta element.

4. På en från början tom kö utför man i tur och ordning följande operationer:

- enqueue 1
- enqueue 2
- enqueue 3
- dequeue
- enqueue 4
- dequeue

Vilket är nästa tal att tas ut ur kön?

(A) 1 (B) 2 (C) 3 (D) 4 (E) Inget – kön är tom

Motivering: Man lägger in (enqueue) de fyra talen 1, 2, 3 och 4 i kön och tar bort (dequeue) de två första. Nästa tal att tas ut är alltså 3.

5. Efter att i tur och ordning ha satt in värdena 5, 9, 1, 3, 0, 4, i ett från början tomt binärt sökträd *utan balansering* (med hjälp av algoritmen som vi gått igenom i kursen), vilken är höjden på trädet? (Höjden på ett träd är det största avståndet från roten till ett löv.)

(A) 1 (B) 2 (C) 3 (D) 4 (E) 5

Motivering: Med programexemplen i kursen så får man det binära trädet **Bst ((5, 5), Bst ((1, 1), Bst ((0, 0), Void, Void), Bst ((3, 3), Void, Bst ((4, 4), Void, Void))), Bst ((9, 9), Void, Void))**. (Programmet kräver "tabellvärden" i noderna – nycklarna har använts även som värden.)

6. Ett syfte med datastrukturen binomialheap ("binomial heap") är att på ett effektivt sätt...
- (A) gå igenom noderna i en graf
 - (B) söka i en datamängd
 - (C) implementera en stack
 - (D) implementera en prioritetskö
 - (E) balansera ett träd

Motivering: Med en binomialheap kan man effektivt plocka ut det minsta eller största av en samling värden. Om värdena används som prioriteter så får man en prioritetskö.

7. Program som läser från filer gör i allmänhet fyra olika saker som berör filen:

- (a) Öppnar filen
- (b) Testar för filslut
- (c) Läser från filen
- (d) Stänger filen

Titta på funktionen `scan` nedan:

```
scan(file, text)
```

TYPE: `string*string->int`

PRE: `filename` är namnet på en fil som går att läsa.

POST: Numret på den första raden i filen `filename` som är lika med `text` eller `~1` om en sådan rad inte finns.

```
fun scan(file, line) =  
  let  
    fun scan'(stream, line, n) =  
      if TextIO.endOfStream stream then  
        (TextIO.closeIn stream; ~1)  
      else if line =  
        String.substring(valOf(TextIO.inputLine stream), 0,  
                          size(valOf(TextIO.inputLine stream))-1) then  
        n  
      else  
        scan'(stream, line, n+1)  
    in  
      scan'(TextIO.openIn file, line, 1)  
  end;
```

(Syftet med anropet av `String.substring` är att ta bort det radbrytningstecken som finns i slutet av varje rad som läses med `TextIO.inputLine`.)

`scan` är felaktig. Felen berör två av de fyra punkterna ovan. Vilka?

- (A) Öppning och läsning
- (B) Öppning och stängning
- (C) Test av filslut och läsning
- (D) Test av filslut och stängning
- (E) Läsning och stängning

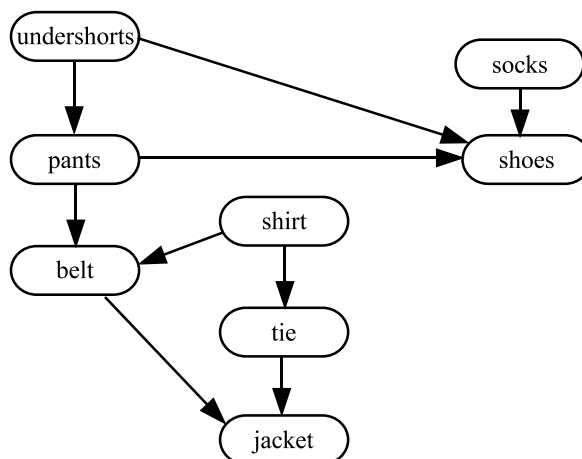
Motivering: Första och tredje argumentet till `String.substring` skall använda samma sträng, men eftersom det är två anrop av `inputLine` så läses två olika rader. Filen stängs inte om man hittar en rad som är lika med `text`.

8. Givet en tom hashtabell med 5 platser, numrerade 0 – 4 och en hashfunktion $hash(key) = key \bmod 5$. Öppen adressering med linjär probing används. Man sätter först in nyckeln 17, sedan nyckeln 32. På vilken plats hamnar nyckeln 32?

- (A) 0
- (B) 1
- (C) 2
- (D) 3
- (E) 4

Motivering: 17 hashar till plats 2 som är ledig så 17 sätts in där. 32 hashar också till 2 som nu är upptagen. Eftersom vi har linjär probing så kontrolleras nästa plats (3). Den är ledig, så 32 sätts in där.

9. Titta på följande graf:

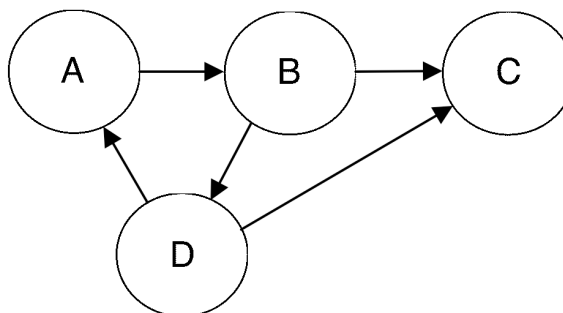


Vilket av följande alternativ är en korrekt topologisk sortering av noderna i grafen?

- (A) shirt, socks, tie, undershorts, pants, belt, jacket, shoes
- (B) undershorts, pants, belt, shirt, tie, jacket, socks, shoes
- (C) belt, jacket, pants, shirt, shoes, socks, tie, undershorts
- (D) tie, belt, pants, shirt, shoes, socks, jacket, undershorts
- (E) Något annat

Motivering: (A) är en uppräknings av noderna sådan att om det finns en väg från nod X till nod Y så ligger nod X före nod Y i uppräkningsen.

10. Titta på följande graf:



Vi skall representera denna graf som en tabell av "adjacency lists". Både själva tabellen och varje "adjacency list" skall ordnas i bokstavsordning.

Vilken nod finns i andra elementet i den "adjacency list" som hör till tredje raden i tabellen?

- (A) A
- (B) B
- (C) C
- (D) D
- (E) Ingen – denna "adjacency list" har färre än två element.

Motivering: C är den tredje noden i bokstavsordning, så tredje raden i tabellen är för noden C. Eftersom det inte finns några utgående kanter från C så är dess "adjacency list" tom.

Frågor för betyg 4

Om du fått minst betyg **3** genom dina svar på de föregående frågorna och dessutom svarar rätt på minst 3 av de 5 frågorna i detta avsnitt så blir du godkänd med minst betyget **4**. Du kan inte kompensera ett dåligt resultat i detta avsnitt med poäng från frågorna för betyg **3** eller **5**.

11. Efter att i tur och ordning ha satt in värdena 5, 9, 1, 3, 0, 4, i ett från början tomt rödsvart träd (med hjälp av algoritmen som vi gått igenom i kursen), vilken är höjden på trädet? (Höjden på ett träd är det största avståndet från roten till ett löv.)

(A) 1 (B) 2 (C) 3 (D) 4 (E) 5

Motivering: Det rödsvarta träd man får med kursens programexempel är RBT (Black, RBT (Black, RBT (Red, Void, 0, Void), 1, Void), 3, RBT (Black, RBT (Red, Void, 4, Void), 5, RBT (Red, Void, 9, Void))).

12. Titta igen på grafen i uppgift 9. Antag att den representeras med en tabell av “adjacency lists” där tabellen är ordnad i *omvänd bokstavsordning* medan listorna är ordnade i *bokstavsordning*. Vi använder bredden först-algoritmen (BFS) som vi gått igenom i kursen. Algoritmen börjar med första raden i tabellen och gör omstarter vid behov. I vilken ordning besöker algoritmen noderna i grafen?

(A) undershorts, pants, shoes, belt, jacket, tie, socks, shirt

(B) undershorts, pants, shoes, belt, jacket, shirt, socks, tie

(C) undershorts, shoes, pants, belt, jacket, tie, socks, shirt

(D) undershorts, shoes, pants, belt, jacket, shirt, socks, tie

(E) Någon annan ordning

Motivering: Enligt BFS-algoritmen.

13. Vi har ett *balanserad* binärt sökträd T med minst två (olika) element. Trädet kan roteras åt höger vid roten. Vilket/vilka av följande påståenden är sanna?

(a) Tiden för en preorder-traversering av T är linjärt proportionell mot trädets höjd. (Höjden på ett träd är det största avståndet från roten till ett löv.)

(b) En preorder-traversering av T går (alltid) igenom elementen i T i stigande nyckelordning.

(c) En preorder-traversering av T är oförändrad efter en högerrotation av T vid roten.

(A) a (B) b (C) c (D) Alla (E) Inga

Motivering: Tiden är proportionell mot antalet noder eftersom alla noder måste gås igenom. Eftersom preordertraverseringen går igenom roten först så kommer den dels inte att gå igenom noderna i nyckelordning, dels kommer den att ändras vid en rotation eftersom rotnoden ändras då.

14. Titta på hashtabellen nedan med $m = 7$ platser. \perp betecknar en plats som aldrig använts, medan Δ betecknar en plats som nu är tom men varit använd tidigare.

0	1	2	3	4	5	6
\perp	11	Δ	Δ	\perp	25	36

Låt hashfunktionen vara $hash(key) = key \bmod 10$. Vi använder öppen adressering med den kvadratiske test(probing)funktionen $f(i) = i^2$. Antag att dublettnycklar *inte* är tillåtna. Hur många platser måste man testa (probe) när man skall sätta in 42 i tabellen?

- (A) ≤ 1 (B) 2 (C) 3 (D) 4 (E) ≥ 5

Motivering: Nyckeln 42 hamnar på plats 2 efter 4 tester, på platserna $2 + 0^2 = 2$, $2 + 1^2 = 3$, $2 + 2^2 = 6$, $(2 + 3^2) \bmod 7 = 4$ ($i = 0, 1, 2$ respektive 3). Eftersom dublettnycklar inte är tillåtna måste vi fortsätta testa förbi ett Δ tills vi hittar en \perp för att vara säkra på att inte 42 redan finns i tabellen.

15. Utgå ifrån en tom binomial(min)heap och sätt in elementen 9, 5, 1, 3, 0, 4, i tur och ordning med hjälp av algoritmen som givits i kursen. Vilken är sedan höjden av det binomialträd som innehåller noden med nyckeln 3? (Höjden på ett träd är det största avståndet från roten till ett löv.)

- (A) 0 (B) 1 (C) 2 (D) 3 (E) ≥ 4

Motivering: Den binomialheap man får med kursens programexempel är `Node(1, 0, [Node(0, 4, [])])`, `Node(2, 1, [Node(1, 5, [Node(0, 9, [])])])`, `Node(0, 3, [])`.

Frågor för betyg 5

Om du fått minst betyg 4 genom dina svar på de föregående frågorna och dessutom svarar rätt på minst 3 av de 5 frågorna i detta avsnitt så blir du godkänd med betyg 5. Du kan inte kompensera ett dåligt resultat i detta avsnitt med poäng från frågorna för betyg 3 eller 4.

16. Titta igen på funktionen från uppgift 2:

```
fun fullmetal(_,elric,[]) = [elric]
  | fullmetal(alchemist,elric,mustang::bradley) =
    if alchemist(elric,mustang) then
      elric::mustang::bradley
    else
      mustang::fullmetal(alchemist,elric,bradley)
```

Vilken typ har fullmetal?

- (A) ('a * 'a -> bool) * 'a * 'a list -> 'a list
- (B) (int * int -> bool) * int * int list -> int list
- (C) (string * string -> bool) * string * string list -> string list
- (D) bool * string * string list -> string list
- (E) ('a * 'b -> bool) * 'a * 'b list -> 'b list

Motivering: Av första klausulen till fullmetal så kan man se att värdet av fullmetal måste vara en lista med element av samma typ som andra argumentet. Alltså är alternativ (E) fel. Av testet i andra klausulens if-uttryck ser man att första argumentet alchemist är en funktion. Alltså är alternativ (D) fel. Koden fungerar med samtliga övriga tre alternativ, men eftersom det inte finns något som begränsar typen av listorna etc. till int eller string så är funktionen polymorf och bara alternativ (A) kan vara rätt.

17. Vilket (om något) av följande påståenden om funktioner i ML är *felaktigt*?

- (A) En funktion kan beräknas av uttryck på liknande sätt som andra data.
- (B) Uttrycket length [1,2,3] är *inte* en funktion.
- (C) Man kan se på typen av ett uttryck om det är en funktion eller inte.
- (D) Argumenten till en funktion beräknas innan funktionen anropas.
- (E) Alla påståenden ovan är korrekta

Motivering: Så är det!

18. Vilket (om något) av följande påståenden om binomialheapar är **felaktigt**?
- (A) En heap innehåller aldrig mer än ett binomialträd av varje rang.
 - (B) En heap kan användas för att sortera en datamängd i $O(n \log n)$ tid.
 - (C) Man kan alltid lägga in ett element i en heap i $O(n)$ tid, men inte alltid i $O(\log n)$ tid.
 - (D) Man kan alltid ta fram minsta elementet i en min-heap i $O(\log n)$ tid, men inte alltid i $O(1)$ tid.
 - (E) Alla påståenden ovan är korrekta

Motivering: Man kan både lägga in och ta bort element ur en heap i $O(\log n)$ tid.

19. Data (utanför datorn) representeras i datorn av värden som hör till någon datastruktur (datatype). Vilket av följande påståenden om datarepresentation och datastrukturer stämmer **inte**?
- (A) Datastrukturer som deklarerats med **datatype** måste ha datastrukturinvariant, men det behöver inte datastrukturer som deklarerats med **abstype**.
 - (B) Det finns i allmänhet ingen representation av ett visst slags data som alltid är bäst, utan vilken datarepresentation som är lämpligast beror bland annat på vilka beräkningar man skall göra.
 - (C) Ett syfte med abstrakta datatyper är att man förhållandevis enkelt skall kunna byta datarepresentation i programmet.
 - (D) Datastrukturinvarianter syftar bl.a. till att utesluta "orimliga" värden i en viss datarepresentation.
 - (E) Datastrukturer som deklarerats med **abstype** måste ha en uppsättning funktioner (och/eller värden) som utgör gränssyta, men det behöver inte datastrukturer som deklarerats med **datatype**.

Motivering: Behovet av datastrukturinvariant har inget att göra med om **abstype** eller **datatype** används.

20. Författaren till *Da Vinci-koden*, Dan Brown, skriver i kapitel 4 och 5 av sin bok *Gåtornas palats* att den påhittade datorn TRANSLTR rutinmässigt hittar 64 bitars lösenord för krypterade texter på ungefär 10 minuter. TRANSLTR har tre miljoner processorer som arbetar parallellt. Den hittade ett visst lösenord med en miljon bitar på tre timmar. Vad är den **genomsnittliga** tiden som krävs för att hitta 70-bitars lösenord?

- (A) ≈ 11 min. (B) ≈ 12 min. (C) ≈ 13 min. (D) ≈ 11 timmar (E) Något annat

Motivering: Att gissa n bitar tar tid $\Theta(2^n)$ i värsta fall, alltså kommer ytterligare 6 bitar att öka den genomsnittliga körtiden med en faktor $2^6 = 64$.