

Tentamen (del 2) (4 högskolepoäng) i Programkonstruktion och datastrukturer (1DL201) + Del omtenta i PK/PK1/PK2/PM1/AD1

Lars-Henrik Eriksson och Pierre Flener
Onsdag 16 mars 2011, kl 08:00 – 11:00, i Polacksbacken

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

Hjälp: En av huvudlärarna kommer att besöka skrivsalen kl 09:00 – 10:00 (senast).

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.

Gammal kurs: Om du skriver omtentan i PK (1DL200), PK1 (1IT021) eller PM1 (2AD524) svara endast på frågorna märkt “PK fråga” och kryssa här: ☐

Om du skriver omtentan i AD1 (1DL210) eller PK2 (1IT022) svara endast på frågorna märkt “AD1 fråga” och kryssa här: ☐

	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:

(PK
fråga)

```
abstype vector2 = V of real*real with
  fun addvectors(V(x1,y1),V(x2,y2)) = V(x1+x2,y1+y2);
  ?1?
  ...
end;
?2?;
```

Det finns tre alternativ för kod på platserna ?1? och ?2?:

- (a) `fun scalarproduct(V(x1,y1),V(x2,y2)) = x1*x2+y1*y2;` vid ?1?
och `fun vectorlength v = Math.sqrt(scalarproduct(v,v))` vid ?2?
- (b) Inget vid ?1?
och `fun vectorlength (V(x,y)) = Math.sqrt(x*x+y*y)` vid ?2?
- (c) `fun scalarproduct(V(x1,y1),V(x2,y2)) = x1*x2+y1*y2;` vid ?1?
och `fun vectorlength (V(x,y)) = Math.sqrt(x*x+y*y)` vid ?2?

Vilka alternativ är möjliga för att `vectorlength` skall bli korrekt definierad? Självva den numeriska uträkningen är korrekt i samtliga fall, frågan gäller struktureringen av programmet.

- (A) (a) (B) (a) & (b) (C) (b) (D) (b) & (c) (E) Något annat

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

2. Titta på denna funktionsdefinitionen:

(PK
fråga)

```
fun dagon(yogsothoth, 0, shubniggurath) = shubniggurath
  | dagon(yogsothoth, nyarlathotep, shubniggurath) =
    dagon(yogsothoth, nyarlathotep-1, yogsothoth(nyarlathotep, shubniggurath));
```

Vad är värdet av uttrycket `dagon(op -, 3, 0)`?

- (A) ~ 6 (B) ~ 4 (C) ~ 2 (D) 0 (E) 2

Justification: Anropen av `dagon` ger uttrycket `op -(1, op -(2, op -(3, 0)))`, som beräknas till $1 - (2 - (3 - 0)) = 2$.

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

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

Vad blir värdet av det sista uttrycket `!x`?

- (A) `[1,2,3,4]` (B) `[2,3,4]` (C) `[3,4]` (D) `[11,12,13,14]` (E) `[12,13,14]`

Justification: Definitionen av `pop` påverkar inte `x`. `x` sätts till `[11,12,13,14]`, sedan anropas `pop` som sätter `x` till `[12,13,14]`.

4. What is a tight asymptotic bound on the runtime of the `v` function below? Let n be the number of nodes of its binary tree argument. Assume the tree is *balanced*. (AD1 fråga)

```
datatype d = E | D of d * d  
fun v(E) = 0  
  | v(D(L,R)) = 1 + Int.max(v(L),v(R))
```

- (A) $\Theta(n)$ (B) $\Theta(n \cdot \lg n)$ (C) $\Theta(n^2)$ (D) $\Theta(n^2 \cdot \lg n)$ (E) $\Theta(2^n)$

Justification: By Theorem 2 (case 1), as $f(n) = \Theta(1)$ is dominated by $n^{\log_b a} = n$, for $a = 2 = b$.

5. After performing the insertions (using the algorithm seen in the course) of 9, 5, 1, 3, 0, 4, in that order, into the initially empty AVL tree, what is the balance factor of the tree rooted at the node with key 1 in the resulting AVL tree? (AD1 fråga)

- (A) left-unbalanced (B) left-heavy (C) stable (D) right-heavy (E) right-unbalanced

Justification: The resulting AVL tree is `3(1(0, Void), 5(4, 9))`.

6. After performing the insertions (using the algorithm seen in the course) of the keys 9, 5, 1, 3, 0, 4, in that order, into the initially empty binomial *max*-heap, what is the *rank* of the binomial tree containing the node with key 1 in the resulting heap? (AD1 fråga)

- (A) 0 (B) 1 (C) 2 (D) 3 (E) ≥ 4

Justification: The resulting heap is `[4(0), 9(3(1), 5)]`.

7. Which of the following statements on the *average*-case performance of hashing under *chaining* are true? (AD1 fråga)

- (a) The maximum chain length grows in proportion to the load factor.
(b) The average chain length grows in proportion to the load factor.
(c) A hash table of m cells can store strictly more than m elements.

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

Justification: The maximum chain length grows *slower* than the load factor.

8. Consider the hash table below of $m = 7$ cells, where \perp denotes the element at a cell that was never used and Δ denotes a deleted element: (AD1 fråga)

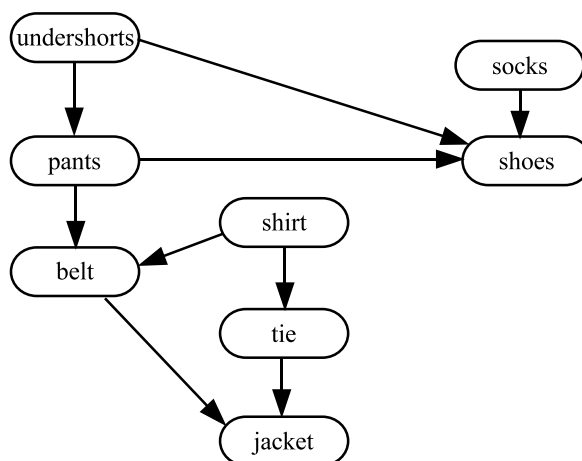
0	1	2	3	4	5	6
\perp	11	Δ	\perp	\perp	21	\perp

Consider the ordinary hash function $hash'(key) = \text{“the **rightmost** digit of key ”}$ under **open addressing** with the **quadratic** probing function $f(i) = i^2$. Perform **no rehashing**. Assume that duplicate keys are **not** allowed. What is the number of probes made when inserting 41?

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

Justification: Key 41 goes into cell 2 upon 4 probes, at indices $1+0^2 = 1$, $1+1^2 = 2$, $1+2^2 = 5$, $(1+3^2) \bmod 7 = 3$ for $i = 0, 1, 2, 3$, because under the given assumption we **must** probe beyond the Δ in order to make sure no 41 exists yet in the table.

9. Alfons needs to be careful when getting dressed in the mornings. Each edge from vertex u to vertex v in the acyclic directed graph below means that item u must be put on before item v : (AD1 fråga)



Assume the graph is represented as an array of adjacency lists. Assume that array is indexed by **increasing** order on the vertex names. Assume each adjacency list is sorted by **increasing** order on the vertex names. What is the topological sorting order (using the DFS-based algorithm seen in the course) of the vertices?

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

Justification: By the DFS-based topological sort algorithm.

10. Reconsider the graph and its representation assumptions of Question 9. What is the *breadth*-first search order (using the algorithm seen in the course) of this graph when starting from the undershorts and not making any restarts? (AD1 fråga)

- (A) undershorts, shoes, pants, belt, jacket
- (B) undershorts, pants, shoes, belt, shoes, jacket
- (C) undershorts, pants, belt, jacket, shoes
- (D) undershorts, pants, shoes, belt, jacket
- (E) undershorts, pants, belt, shoes, jacket

Justification: By the BFS algorithm.

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. Vilket av dessa påståenden stämmer *inte* in på abstrakta datatyper:

(PK
fråga)

- (A) Funktionerna i den abstrakta datatypens gränsyta är de enda som kan utnyttja hur datatypen är representerad.
- (B) Riktig användning av abstrakta datatyper minskar risken för fel i programmen.
- (C) Ett skäl att använda abstrakta datatyper är att lätt kunna ändra datarepresentationen.
- (D) Riktig användning av abstrakta datatyper gör programmet mer effektivt.
- (E) Att göra en datatyp abstrakt kan begränsa vilka beräkningar man kan göra med den.

Justification: Det går bara att komma åt representationen av värden ur den abstrakta datatypen genom funktionerna i gränsytan, vilket inte behöver vara det mest effektiva i varje situation.

12. Detta program (med numrerade rader) är tänkt att skapa en lista av alla rader i en fil – med det avslutande radbrytningstecknet i slutet på varje rad borttaget:

(PK
fråga)

```
1 fun readLines f =
2   let
3     fun rl'(is) =
4       if TextIO.endOfStream is then
5         []
6       else
7         (String.substring(valOf(TextIO.inputLine is),
8                           0, size(valOf(TextIO.inputLine is))-1) ::
9         rl' is)
10  in
11    rl'(TextIO.openIn f)
12 end;
```

Det finns felaktigheter eller något som saknas på en eller flera rader. Vilken/vilka?

- (A) 5 & 7 & 8 (B) 5 (C) 4 & 5 (D) 7 & 8 (E) 9 & 11

Justification: Strömmen stängs inte. Två anrop till `inputLine` ger två olika rader.

13. What is the length, measured in symbols, of the longest *unevaluated* arithmetic expression built by the `len` function below? Let n be the number of elements of its list argument. Assume each digit, arithmetic operator, and parenthesis is a symbol. (AD1 fråga)

```
fun len [] = 0
  | len (x::xs) = 1 + ( len xs )
```

- (A) 1 (B) n (C) $4 \cdot n + 1$ (D) $n \cdot \lg n$ (E) n^2

Justification: By induction on n .

14. Which of the following statements on walks of a right-rotatable *balanced* binary search tree T of $n \geq 2$ *distinct* elements are true? (AD1 fråga)

- (a) The inorder walk of T takes time logarithmic in n .
 (b) The inorder walk of T lists the elements of T by strictly increasing keys.
 (c) The preorder walk of T is unchanged after a right rotation at the root of T .

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

Justification: Because the inorder walk takes time *linear* in n , and because it is the *inorder* walk that is preserved by rotations. Red herring: The balancedness assumption, as inorder and preorder walks are defined for *all* binary trees.

15. What is a tight asymptotic bound on the runtime of the `v` function below? Let n and m be the numbers of elements of its first and second list arguments, respectively. Note that `v` returns a list whose length is the *sum* of the lengths of its arguments. Assume `d(L)` returns two sub-lists of list L of about *half* the length $|L|$ of L , *always* in $\Theta(|L|/2)$ time. Assume `w(L)` returns a list of the length $|L|$ of list L , *always* in $\Theta(|L|^2)$ time. (AD1 fråga)

```
fun v([ ],C) = C
  | v([x],C) = x::C
  | v(xxs,C) = let val (A,B) = d(xxs) in v(w(A),v(B,C)) end
```

- (A) $\Theta(m \cdot \lg n)$ (B) $\Theta(n \cdot \lg m)$ (C) $\Theta(n \cdot \lg n)$ (D) $\Theta(m \cdot n)$ (E) $\Theta(n^2)$

Justification: By Theorem 2 (case 3), as $f(n) = \Theta(n^2 + n)$ dominates $n^{\log_b a} = n$, for $a = 2 = b$. Red herrings: $|C| = m$ and $|v(L,C)| = |L| + |C|$, as the second argument has no impact on the runtime.

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. Betrakta följande program:

(PK
fråga)

```
fun Morgenstern(_,[]) = []  
  | Morgenstern(Amerika,keine::Lust) =  
    case Amerika keine of  
      NONE => Morgenstern(Amerika,Lust)  
    | SOME Reise => Reise::Morgenstern(Amerika,Lust);
```

Vilken typ har Morgenstern?

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

Justification: (inget)

17. Här följer några påståenden om programmering med sidoeffekter:

(PK
fråga)

- (a) Det är viktigt i vilken ordning olika delar av programmet beräknas.
- (b) Man kan byta ut uttryck med samma värde mot varandra.
- (c) Ändringar av en datastruktur (t.ex. ref-cell i ML) kan aldrig påverka en annan datastruktur.
- (d) Värdet av en funktion blir ointressant om man använder sidoeffekter.
- (e) Utan sidoeffekter är loopar som `while` meningslösa.
- (f) Man kan inte vinna i effektivitet genom att använda sidoeffekter.

Hur många av dessa påståenden är rätt?

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

Justification: Bara (a) och (e) är korrekta.

18. Which of the following statements on trees are true?

(AD1
fråga)

- (a) A non-empty binary tree where all nodes have 0 or 2 children has one more leaf node than non-leaf nodes.
- (b) Every binomial tree is also a search tree, under some suitable generalisation of binary search trees to k -ary search trees, where $k \geq 2$ is a constant.
- (c) Every binomial tree is also a balanced tree, under some suitable generalisation of the AVL balancing property to k -ary trees, where $k \geq 2$ is a constant.

(A) None (B) (a) only (C) (a) & (b) (D) All (E) Another answer

Justification:

(a): True. Let $L(n)$ be the number of leaf nodes of such a tree with n non-leaf nodes:

$$L(n) = \begin{cases} 2 & \text{if } n = 1 \\ L(n-1) + 1 & \text{if } n > 1 \end{cases} = n + 1, \text{ to be proved by induction on } n.$$

(b): False, because the elements of a binomial tree do not need to satisfy any invariant (such as a heap invariant), hence no such generalisation can exist.

(c): False, because a binomial tree of rank k has k subtrees of heights $k-1, k-2, \dots, 0$ respectively, hence for $k \geq 3$ there will always be two subtrees whose heights differ by more than one.

19. Consider the hash table below of $m = 7$ cells, where \perp denotes the element at a cell that was never used:

(AD1
fråga)

0	1	2	3	4	5	6
\perp	11	22	33	\perp	55	\perp

Consider the ordinary hash function $hash'(key) = \text{“the *leftmost* digit of *key*”}$ under **open addressing**. Perform **no rehashing** and assume that duplicate keys **are** allowed. What is the difference in the numbers of probes made when trying to insert the key 14 under quadratic probing (with the probing function $f(i) = i^2$) compared to linear probing (with $f(i) = i$)?

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

Justification: Under linear probing, key 14 goes into cell 4 upon 4 probes (at indices 1, 2, 3, 4 for $i = 0, 1, 2, 3$); under quadratic probing, the insertion fails upon 7 probes (at indices 1, 2, 5, 3, 3, 5, 2 for $i = 0, 1, 2, 3, 4, 5, 6$). Red herring: The fact that duplicate keys are allowed is irrelevant here, as there is no Δ value yet.

20. What is a tight asymptotic bound on the runtime of the k function below? Let n be the number of elements of its list argument. Assume that $d(L)$ returns four sub-lists of list L of about **half** the length $|L|$ of L , **always** in $\Theta(|L|)$ time.

(AD1
fråga)

`fun k([]) = []`

`| k(x::xs) = let val (P,Q,R,S) = d(xs) in k(P) @ k(Q) @ x::k(R) @ k(S) end`

(A) $\Theta(n)$ (B) $\Theta(n \cdot \lg n)$ (C) $\Theta(n^2)$ (D) $\Theta(n^2 \cdot \lg n)$ (E) Another bound

Justification: By Theorem 2 (case 2), as $|k(L)| = \Theta(|L|^2)$ (by case 1 for $a = 4 \wedge b = 2 \wedge f(|L|) = \Theta(1)$), so that $f(n) = \Theta\left(3 \cdot \left(\frac{n}{2}\right)^2 + n + 1\right) = \Theta(n^{\log_b a}) = \Theta(n^2)$, for $a = 4 \wedge b = 2$.