Antal blad / *Number of* sheets: Ø6

TENTAMEN / EXAMINATION

Anvisningar: *Skriv din anonymitetskod på varje blad.*
Endast en uppgift får lösas på varje blad.
Var vänlig skriv tydligt!

Instructions: *Write your anonymous code on each sheet.*
Answer only one question on each sheet.
Please write clearly!

Vänligen texta anonymitetskoden i textboxen enligt exempel nedan!
Please write the Anonymous Code clearly in the textbox like example below!

Bokstäver/Letters:
A-B-C-D-E-F-G-H-I-J-K-L-M-N-O
P-Q-R-S-T-U-V-W-X-Y-Z-Å-Ä-Ö

Siffror/Numbers:
Ø-1-2-3-4-5-6-~~7~~-8-9

Exempel: | A | B | C | 1 | ~~7~~ | Ø | - | Ø | 1 | ~~7~~ |

DVGBØ3 Datastrukturer och algoritmer
Kurskod + Kurs / *Course Code + Course:*

Delkurs / *Part course:*

Anonymitetskod / *Anonymous code* = *Kurskod + kodnr / course code + code number*
D V G B Ø 3 - Ø 3 7

✓

Tentamensdatum / *Examination date:*
15-01-2018

Behandlade uppgifter / *Solved problems*

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
X	X													
16	17	18	19	20	21	22	23	24	25	26	27	28	29	30

Ifylles av lärare / *To be completed by the examiner*

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
9,5	10,5													
16	17	18	19	20	21	22	23	24	25	26	27	28	29	30

Poäng / Marks gained: 20

Betyg / Grade: 3

Max poäng / Total marks gained: 40

För Gk poäng / Marks gainedto be passed: 20

Examin. lärare / Kursansvarig signatur / Signature of the examiner

Kerstin Andersson
Namnförtydligande / Clarification of the signature

3500000682

Häftområde

Skriv ej i detta område
Leave this area blank

Ange anonymitetskod / *Write your anonymity code*
(Vid icke anonym tentamen ange kurskod + namn + personnummer)
(For non-anonymous exams write the course code + name + civic registration number)

DVGB03-037

Löpande sidnr
Consecutive no:
1

Uppgift nr /
Question no:
1

Poäng / *Points awarded:* 2,5

Lärarens anteckning
Examiner's remarks:

53, 32, 19, 15, 29, 33, 12, 87, 10, 61

1

a)

0	55
1	
2	15
3	29
4	
5	
6	32
7	19
8	33
9	87
10	10
11	61
12	12

Linear probing

55 mod 13 = 0 insert
32 mod 13 = 6 insert
19 mod 13 = 6 collision
19 → 6+1 mod 13 = 7 insert
15 mod 13 = 2 insert
29 mod 13 = 3 insert
33 mod 13 = 7 collision
33 → 7+1 mod 13 = 8 insert
12 mod 13 = 12 insert
87 mod 13 = 9 insert
10 mod 13 = 10 insert
61 mod 13 = 9 collision
61 → 9+1 mod 13 = 10 collision
61 → 9+2 mod 13 = 11 insert

= 3

Double hashing

0	55
1	
2	15
3	29
4	61
5	
6	32
7	33
8	19
9	87
10	10
11	
12	12

55 mod 13 = 0 insert
32 mod 13 = 6 insert
19 mod 13 = 6 collision
19 → 6+7 mod 13 = 0 collision
19 → 6+2 mod 13 = 8 insert
15 mod 13 = 2 insert
29 mod 13 = 3 insert
33 mod 13 = 7 insert
12 mod 13 = 12 insert
87 mod 13 = 9 insert
10 mod 13 = 10 insert
61 mod 13 = 9 collision
61 → 9+10 mod 13 = 6 collision
61 → 9+8 mod 13 = 4 insert

2

2

Double hashing fungerar bättre än linear probing när man använder sig av större hashtabeller och flet värden. kollisionerna skulle minska drastiskt.
På exemplet ovan så är det samma antal kollitioner men skulle man utöka n och tabellstorleken så blir skillnaden klar.

PS hoppas att jag inte har blandat ihop open och closed-addressing...

?

varför

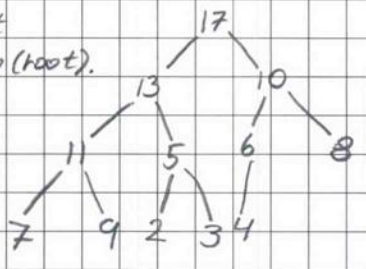
Universitetstryckeriet

Behandla endast en uppgift per sida / *Answer only one question per page*

1)

b) $A = \langle 17, 13, 10, 11, 5, 6, 8, 7, 9, 2, 3, 4 \rangle$

A är en max-heap för att det största talet placeras i toppen (root).
En heap kan antingen vara en max eller min-heap där kraven då är följande.



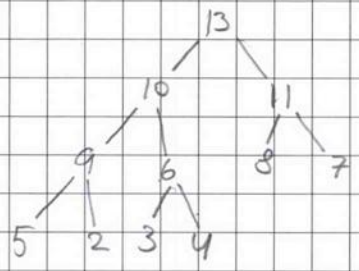
Max-heap := $LC < N > RC$
Min-heap := $LC > N < RC$
Detta gäller då för alla noder i trädet och inte bara toppen.

R

Heap-Extract-Max plockar ut det första värdet i A, alltså det första. Kör sedan heapify för att kolla att villkoret Max-heap := $LC < N > RC$ stämmer för alla noder i heapen.

A kommer se ut som följande efter att funktionen har exekverat

$A = \langle 13, 10, 11, 9, 6, 8, 7, 5, 2, 3, 4 \rangle$



Ej korrekt.

Häftområde

Skriv ej i detta område
Leave this area blank

Ange anonymitetskod / *Write your anonymity code*
(Vid icke anonym tentamen ange kurskod + namn + personnummer)
(For non-anonymous exams write the course code + name + civic registration number)

DVGBØ3-Ø37

Löpande sidnr
Consecutive no:
3

Uppgift nr /
Question no:
1

Poäng / *Points awarded:* 5

Lärarens anteckning
Examiner's remarks:

Mer ingående

1)

c) Ett AVL träd är ett självbalanserande binärt sökträd (BSt) där höjdskillnaden i trädet får inte vara större än 1.

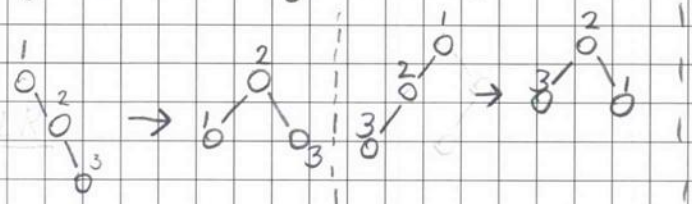
$|height(Lc(t)) - height(Rc(t))| < 2$

AVL = BSt
BSt := LC < N < RC
LC = BSt RC = BSt

Varje nod får ha 0 eller 2 barn.

SLR, SRR, DLR och DRR är functioner för att balansera ett AVL träd vid insättning eller borttagning av noder så att trädet fortfarande ska uppfylla sina krav.

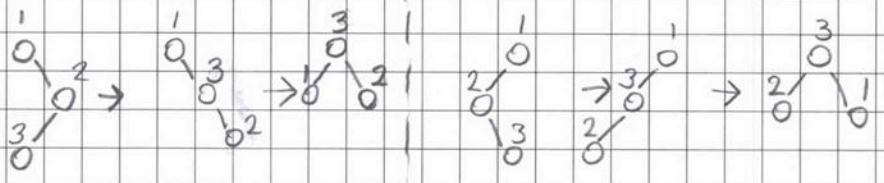
$|height(Lc(t)) - height(Rc(t))| < 2$



DLR = SRR + SLR
DRR = SLR + SRR

trädet ovan uppfyller inte kraven för ett AVL träd. vi kör en SLR på nod index 2

SRR är en spegelvänd SLR



DRR

ovan är en DLR

SLR = Single left rotation
SRR = Single right rotation
DLR = Double left rotation
DRR = Double right rotation

Universitetstryckeriet

Behandla endast en uppgift per sida / *Answer only one question per page*

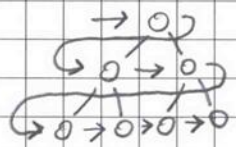
Uppgift nr / *Question no:* 2-a

Poäng / *Points awarded:* 1

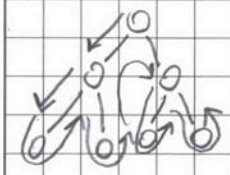
Lärarens anteckning
Examiner's remarks:

2/ a) Bredd-först och djup-först sökalgoritmer är nog lättast att förklara via ett träd.

Bredd-först-sökning börjar i toppen och jobbar sig sekvensiellt neråt nivå efter nivå.



djup-först-sökning startar också i toppen. Jobbar sig sedan ner till botten och går igenom alla noder den kommer i kontakt med.



```
Preorder(treeref t)
    Print(t)
    Preorder(Lc(t))
    Preorder(Rc(t))
```

Dessa algoritmer skriver upp träd eller graf som en lista.

Uppgiften gäller (grafer

Häftområde

Skriv ej i detta område
Leave this area blank

Ange anonymitetskod / *Write your anonymity code*
(Vid icke anonym tentamen ange kurskod + namn + personnummer)
(For non-anonymous exams write the course code + name + civic registration number)

DVGB03 - 037

Löpande sidnr
Consecutive no:
5

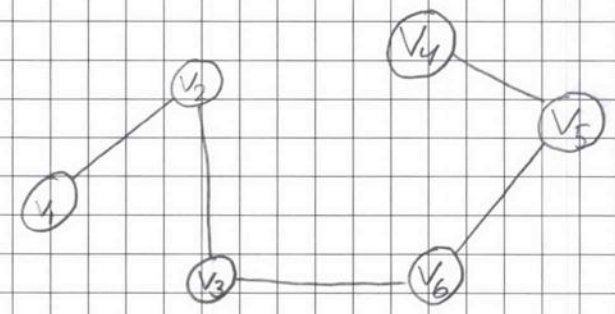
Uppgift nr /
Question no:
2b

Poäng / *Points awarded:*
0,5

Lärarens anteckning
Examiner's remarks:

2/

b/



Kruskals algoritm är girig då den alltid tar den lokala kortaste vägen.

väg??

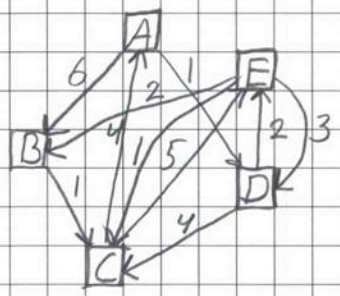
Motivera!

Universitetstryckeriet

Behandla endast en uppgift per sida / *Answer only one question per page*

2)

c)



	A	B	C	D	E
A	-	6	-	1	-
B	-	-	1	-	-
C	4	-	-	-	5
D	-	-	4	-	2
E	-	2	1	3	-

A	B	C	D	E
D	6	∞	1	∞
E	a	a	a	a
L	6	∞	1	∞

$S=\{A\}$ min = D

$S=\{A,D\}$ W = D $Q=\{B,C,E\}$

if V.d > V.d + W(U,V) SWAP

B - 6 > 1 + ∞ false
C - ∞ > 1 + 4 true
E - ∞ > 1 + 2 true

A	B	C	D	E
D	6	5	1	3
E	a	D	a	D
L	6	4	1	2

min = E $S=\{A,D,E\}$ $Q=\{B,C\}$ W = E

B - 6 > 3+2 true
C - 5 > 3+1 true

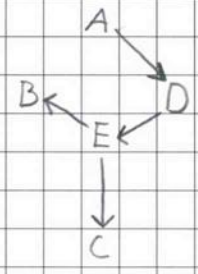
A	B	C	D	E
D	5	4	1	3
E	E	E	a	D
L	2	1	1	2

min = C $S=\{A,D,E,C\}$ $Q=\{B\}$ W = C

B - 5 > 4 + ∞ false

$Q=\emptyset$ loopen är slut.

Spt och viktkostnad blir följande.



	B	C	D	E
A	5	4	1	3

Dijksta beräknar kortaste sträckan, vikterna kan alltså inte vara negativa. vill man räkna med negativa värden så får man använda bellman-ford.

Varför?