ממ"ן 11

שאלה 1

נשתמש בתבנית של עץ ערכי דרגות: נוסיף

לצומת תכונות כך שבמקום שמירת את

size[x] (גודל תת העץ) נשמור רק את succ-d[x]

שהוא העוקב במקום ה-d.

כל צומת במבנה הנתונים x יכיל:

1. right[x]
2. left[x]
3. p[x]
4. key[x]
5. colour[x]
6. succ[x]
7. d-succ[x]
8. pred d[x]

BUILD (S)

sort S from small to large
by quick sort and save in S1

for i ← 0 to size[S1]
succ-d [S1[i]] ← NULL
if i+d <= size[S1]
succ-d[S1[i]] = S1[i+d]
pred[S1[i]] ← S1[i-1]
succ[S1[i]] ← S1[i+1]
RBINSERT(S1[i])

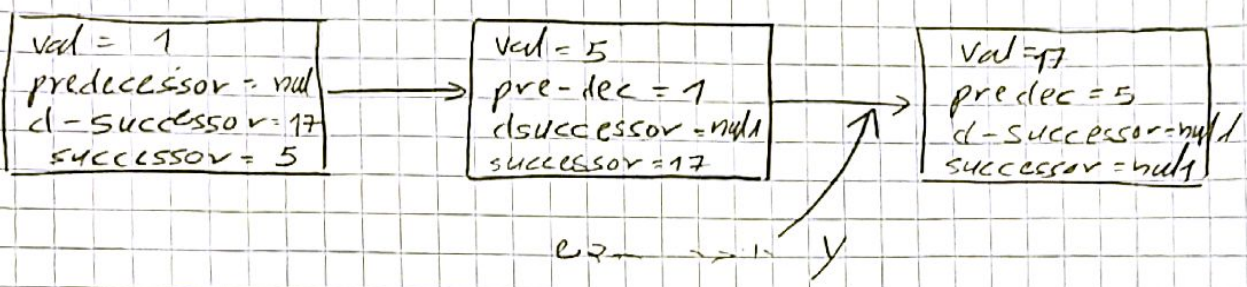
O(n lg n)

O(n · lg n)

O(lg n)

O(n lg n)

INSERT



d-9

1. y.predecessor
2. y successor
3. y.dsuccessor
4. d successor

```
INSERT (S, k)
    y ← NIL[S]
    x ← root[S]
    while   x ≠ NIL[S]
        do   y ← x
            if   key[k] < key[x]
                then   x ← left[x]
                else   x ← right[x]
    p[k] ← y
    if   y = NIL[S]
        than   root[S] ← k
    else   if   key[k] < key[y]
        than   left[y] ← k
               successor[k] ← y
               predessor[k] ← predessor[y]
               successor[predessor[y]] ← k
               predessor[y] ← k
               UPDADE-D-SUCCESSOR(k)
        else   right[y] ← k
               predessor[k] ← y
               successor[k] ← successor[y]
               successor[predessor[y]] ← k
               successor[y] ← k
               UPDATE-D-SUCCESSOR(k)
    left[k] ← NILL[T]
    right[k] ← NILL[T]
    color[k] ← RED
    RB-INVERT-FIX(S, k)
```

UPDATE - D - SUCCESSOR (k)

```
x = k;
for i = 1 to d
    d-successor[x] = d-successor[predecesor[x]]
    x = predecessor[x]
d-successor[x] ← k
```

DELETE

INSERT

successor[presuccessor[x]] = successor[x]

presuccessor[successor[x]] = presuccessor[x]

DELETE (S, k)

RB-DELETE (S, k)

after line 13 in RB-DELETE

successor[presuccessor[k]] = successor[k]

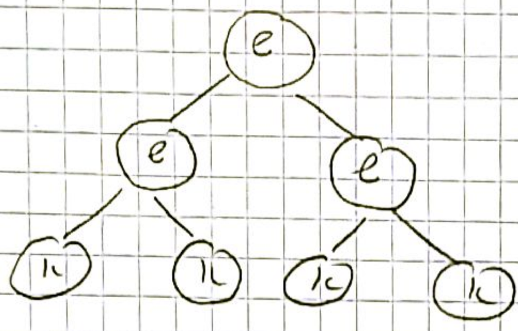
presuccessor[successor[k]] = presuccessor[k]

UPDATE - D - SUCCESSOR(k)

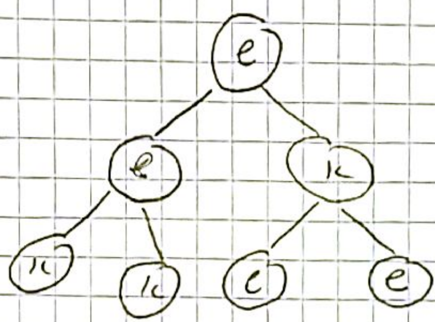
continue with RB-DELETE

שאלה 2

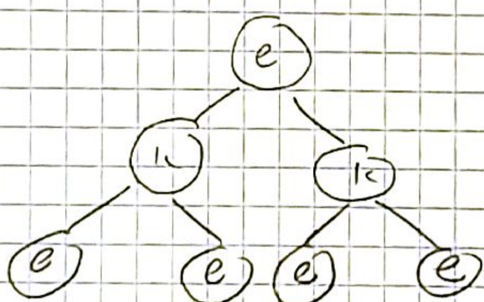
א. עץ כזה יכול להתקיים לדוגמא:



ב. כן עץ כזה יכול להתקיים לדוגמא:



ג. כן קיים עץ כזה:



ד. לא ניתן להרכיב עץ כזה

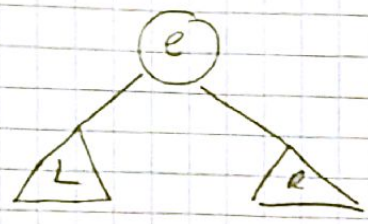
כי ב הגובה הוא שחור ולכן

בנתיב לכל 5 שחורים ו-2 אדומים

להרכיב 2 תתי עצים המקיימים את

תנאי 4 ו-5. ישנם כמה אופציות

לסיכום העץ:



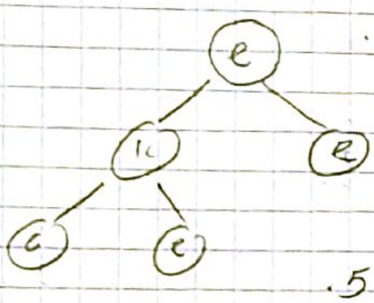
1. הבנים של השורש אחד אדום והשני שחור-

ולכן נבנה עץ בהתאם:

שבהם אין 2 שחורים

רק אדום שנים לא

נקבל בגובה לתחתונה 5.



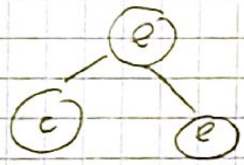
2. הבנים של השורש שניהם שחורים:

במקרה זה נותר

לנו 3 שחורים ו-1

אדומים. אם במקרה זה

שמתחת לכל מקרה

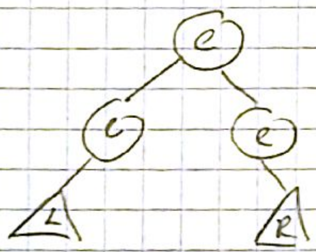


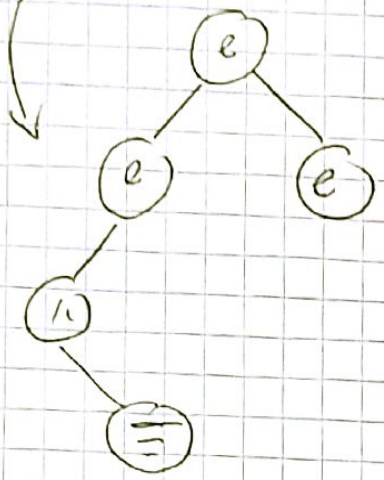
ו. תת עץ ששונה היא

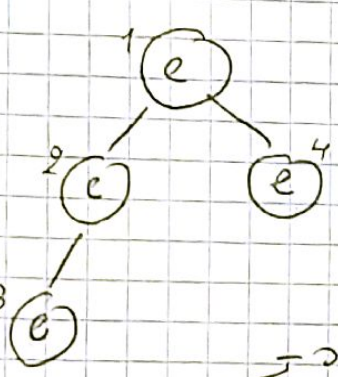
ומכאן תנאי 3 לא

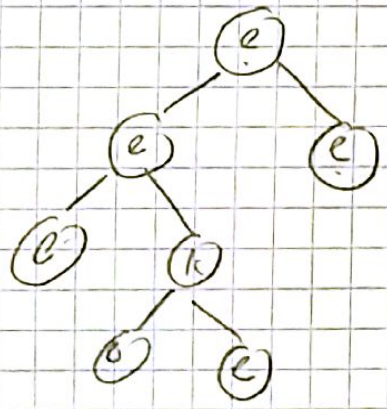
כל המקרים כולן שנוספים

שחורים לתת עץ ימני.









Scanned by CamScanner

שאלה 3

נשתמש בעץ אדום שחור מורחב
[illegible] לכל צומת x את הסכום של כל
ערכי העץ אשר הוא שורשו.

1. פעולת INSERT תהיה דומה לעדכון
של [illegible] שחור [illegible]
[illegible] sum
במקום size.

[illegible]:

א. [illegible] את [illegible] y
[illegible] INSERT עם y
[illegible]:

ב. [illegible] x [illegible]

sum[x] = sum[x] + y

ג. [illegible]

RB-INSERT-FIX [illegible] LEFT-ROTATE

[illegible]

[illegible] LEFT-ROTATE [illegible]

sum[x] ← sum[left[x]] + sum[right[x]] + left[x] + right[x]

sum[y] ← sum[left[y]] + sum[right[y]] + left[y] + right[y]

Scanned by CamScanner

LEFT-Rotate

sum[x] ← sum[left[x]] - sum[right[x]] + left[x] - right[x]

sum[y] ← sum[left[y]] + sum[left[y]] - left[y] - left[y]

RIGHT-Rotate

PAIR-DIFF

PAIR-DIFF (S, d)

▷ create arr with sort element
using inorder tree walk on S

j ← 1

J ← 9

for i ← 1 to length[arr]

if (arr[J] - arr[i] = d)

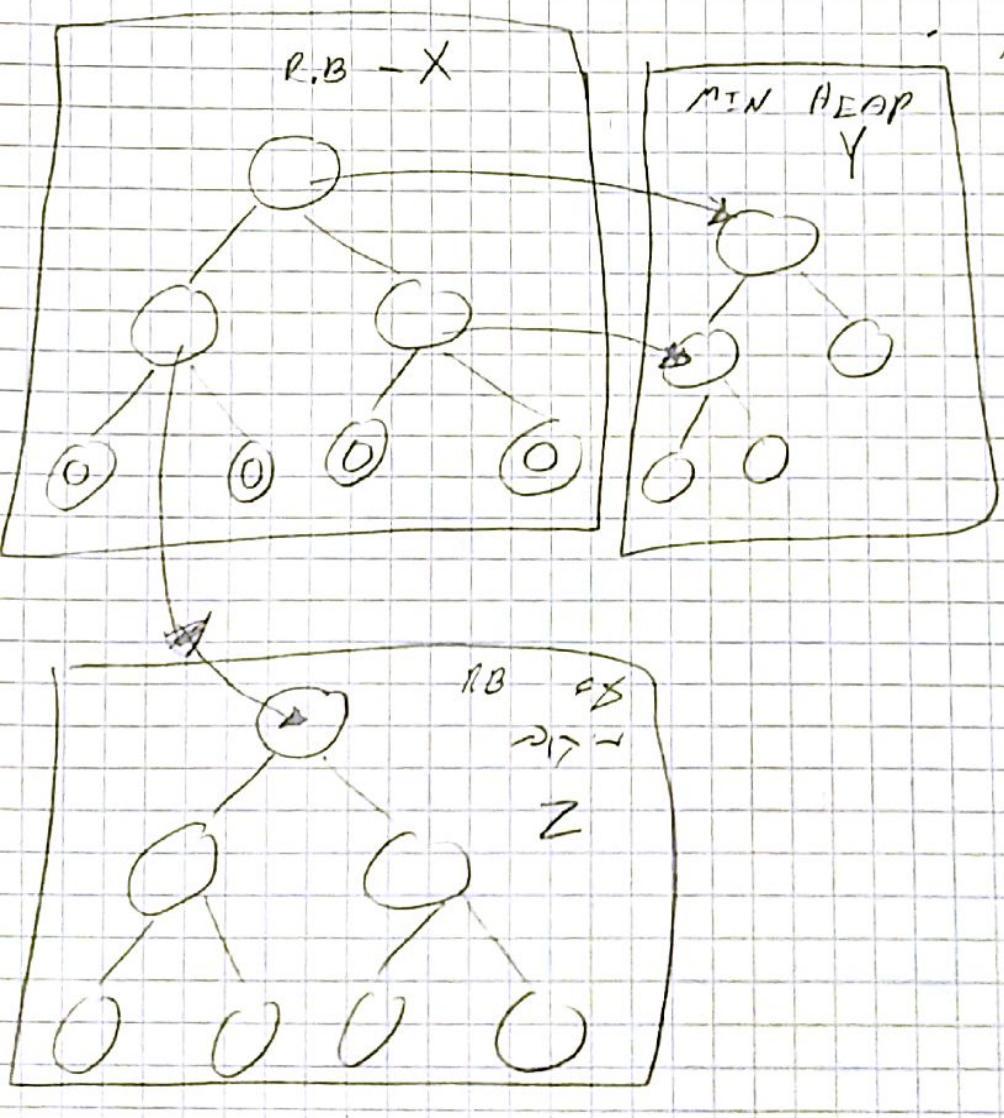
```
i ← i+1
else if (arr[j] - arr[i] < d)
    j ← j+1
else if (arr[j] - arr[i] = d
    print (i, j);
```

sum(s, l_1, l_2)
    res1 ← sum(s, l_1)
    res2 ← sum(s, l_2)
    return res1 - res2

```
SUM (S, k)
    sum ← 0
    x ← root[S]
    while  x ≠ nil
        if  left[x] > k
            sum ← sum + sum[left[x]] + x
            x ← left[x]
        if  x ≤ k
            x ← right[x]

    return  sum
```

שאלה 4



1. [illegible] Search [illegible]
[illegible] x [illegible]
[illegible]
[illegible] $O(\lg n)$

2. [illegible] INSERT - [illegible]
א. [illegible] RB [illegible] y [illegible]
$O(\lg n)$

ב. [illegible] z [illegible]
[illegible] 1. [illegible]
[illegible]
INSERT [illegible]
[illegible] $O(\lg n)$

ג. [illegible]
$O(\lg n)$

3. DELETE(S,1) [illegible]
[illegible]
[illegible] Y [illegible]
[illegible]
RB-REMOVE

[illegible]
[illegible]
[illegible] RB-REMOVE [illegible]
[illegible]
[illegible]

Scanned by CamScanner

O(3lgn) ...

4. ... MODE(S) ... MODE(S) ...
... Y ...
root[y] = ...

5. ... z ...
O(1) ...