לצייר השיעורים ואכזר את ערכי המפתחות, חってきて השיעור אין יותר ביעורים, את ערכי המפתחות: 

הנחת יסוד

עצי אדום-שחור: הגדרה

עצי חיפוש בינאריים: לכל צומת בודד בצבע אדום או שחור, כך ש:
1. העלים הם שחורות.
2. לכל מסלול מהשורש לעלה מכיל את אותו מספר של צומתים שחורים (החוק השחור).
3. לכל צומת אדום, אם יש לו אב, האב הוא שחור (החוק האדום).

דוגמה: נוכיח כי עומקו של עץ אדום-שחור עם n מפתחות הוא O(logn).

תרגיל: נוכחים גם עצים שעב ערכים שונים, אך העבتعاون עם O между על ידי ערך מקסימלי.
למה 1: נסמן ב \( b_d \) את העומק השחור ב-\( d \) ואת העומק ב-\( d \). נראה כי \( d_b = O(\log n) \):

\[
d_b = O(\log n) \quad \text{знакי ייבוע לאב שולח}
\]

למה 2: מציא נקודה בשורש\( b, \) אם קדקודים יש.

\[
d_b \geq 2^{\log(n+1)} - 1 \quad \text{ל縱ק 2 האורח ב-1.}
\]

למה 3: המסרɔ הספרות ב-\( d_b \)

\[
d_b = O(\log n) \quad \text{знакי ייבוע לאב שולח}
\]

למה 4: נессב נקודות כ- \( d \) מין קדקדים בט.

\[
d \leq 2d_b \quad \text{знакי ייבוע לאב שולח}
\]

למה 5: \( n_b \geq 2^{d_b - 1} \quad \text{ל縱ק 2 האורח ב-1.}

\[
\frac{d}{2} \leq d_b \leq \log(n_b + 1) \leq \log(n + 1)
\]
Insert -- definition

Convert a leaf to a red internal node with two leaves. This may create a violation to property 2. To restore it we walk up towards the root applying one of the following cases (each case has a symmetric version).

---

Insert -- non terminal cases

1. (1)
   - z
   - y
   - x
   - A
   - B
   - C
   - D

   ===>

2. (2)
   - z
   - y
   - x
   - A
   - B
   - C
   - D

   ===>

---

Insert -- terminal cases

3. (3)
   - x
   - A
   - B

   ===>

4. (4)
   - z
   - y
   - x
   - w
   - w

   ===>

5. (5)
   - z
   - y
   - x
   - w
   - w

   ===>

---

Insert - analysis

$O(\log n)$ time worst case, since the height is $O(\log n)$.

Suppose you start with an empty tree and do $m$ insertions such that the point of insertion is given to you each time, how much time does it take?

Obviously $O(m \log n)$, but maybe we can prove it cannot be that bad?

---

Delete -- example

Each time we do a color-flip-step the number of red nodes decreases by one.

$\Phi(tree) = \#red$ nodes

$\Delta \Phi(insert) = O(1) - \#color-flips-steps$

$\Rightarrow$ amortized(insert) = $O(1)$

and the sequence actually takes $O(m)$ time.
Delete -- definition

Replace the parent of the external node containing the item with the sibling subtree of the deleted item.

If the parent of the deleted item is black then we create a short node.

To restore the black constraint we go bottom up applying one of the following cases.

Delete -- fixing a short node

(1) \hspace{1cm} \implies \\
(2) \hspace{1cm} \implies 

Delete -- fixing a short node (cont)

(3) \hspace{1cm} \implies \\
(4) \hspace{1cm} \implies \\
And apply one of the previous 3 cases.

Delete + insert - analysis

The previous potential won’t do the trick

\( \Phi(\text{tree}) = \# \text{red nodes} \)

Here are the transformation that we want to release potential:

Delete + insert -- analysis

O(log n) time, since the height is O(log n)

Suppose you start with an empty tree and do m insertions and deletions such that the point of insertion is given to you each time, how much time does it take?

Obviously O(m log n), but maybe we can prove it cannot be that bad?
Delete + insert -- analysis

\[ \#(\text{tree}) = \#() + 2\#() \]

\[ \Rightarrow \text{amortized(delete)} = O(1) \]
\[ \text{amortized(insert)} = O(1) \]

sequence of \( m \) delete and inserts, starting from an empty tree takes \( O(m) \) time