

## תכנון מבני נתונים

במהלך הקורס למדנו על מבני נתונים שונים (מערך, רשימה מקושרת, תור, מחסנית, עץ חיפוש בינארי, עץ AVL, עץ 2-3, ערמה).

הדרישות ממבנה נתונים מוצגות תמיד כרשימת פעולות בהן הוא אמור לתמוך וזמן הריצה של כל פעולה.

### תרגיל (דוגמה)

הציעו מבנה נתונים שתומך בפעולות הבאות (מניחים שמספר האיברים הנוכחי הוא  $n$ ):

- הכנסת איבר חדש בזמן  $O(1)$ ;
- חיפוש איבר בזמן  $O(n)$ ;
- הוצאת איבר בזמן  $O(1)$  (כשמקבלים מצביע לאיבר שרוצים להוציא);

### פתרון

רשימה מקושרת דו-כיוונית. (חשוב שהרשימה תהיה דו-כיוונית בכדי שנוכל להוציא איבר מבלי לקבל מראש מצביע לאיבר הקודם.)

### תרגיל (דוגמה)

הציעו מבנה נתונים שתומך בפעולות הבאות (מניחים שמספר האיברים הנוכחי הוא  $n$ ):

- הכנסת איבר חדש בזמן  $O(\log n)$ ;
- חיפוש איבר בזמן  $O(\log n)$ ;
- הוצאת איבר בזמן  $O(\log n)$  (כשמקבלים מצביע לאיבר שרוצים להוציא);

### פתרון

עץ AVL או עץ 2-3.

### תרגיל (דוגמה)

הציעו מבנה נתונים שתומך בפעולות הבאות (מניחים שמספר האיברים הנוכחי הוא  $n$ ):

- הכנסת איבר חדש בזמן  $O(\log n)$ ;
- מציאת האיבר המינימלי בזמן  $O(1)$ ;
- הוצאת איבר בזמן  $O(\log n)$  (כשמקבלים מצביע לאיבר שרוצים להוציא);

### פתרון

ערמה.

כעת ננסה לתכנן מבני נתונים חדשים (מן הסתם יתבססו על מבני הנתונים שלמדנו).

### תרגיל

הציעו מבנה נתונים שתומך בפעולות הבאות (מניחים שמספר האיברים הנוכחי הוא  $n$ ):

- הכנסת איבר חדש בזמן  $O(\log n)$ ;
- חיפוש איבר בזמן  $O(\log n)$ ;
- הוצאת איבר בזמן  $O(\log n)$  (כשמקבלים מצביע לאיבר שרוצים להוציא);
- חישוב מספר האיברים בעץ בזמן  $O(1)$ .

### פתרון

נשתמש בעץ AVL סטנדרטי  $T$  ובנוסף נממש את השדה  $size(T)$  שיחזיק בכל רגע את מספר האיברים הנוכחי בעץ. ביצירת מבנה נתונים חדש נאתחל את  $size(T)$  להיות 0 ( $O(1)$  עבודה נוספת). בכל פעם שנכניס איבר ל  $T$  נגדיל את  $size(T)$  ב 1 ( $O(1)$  עבודה נוספת). בכל פעם שנוציא איבר מ  $T$  נקטין את  $size(T)$  ב 1 ( $O(1)$  עבודה נוספת). פעולת חישוב מספר האיברים בעץ פשוט תחזיר את המשתנה  $size(T)$  ( $O(1)$  עבודה בסך הכל).

### תרגיל

הציעו מבנה נתונים שתומך בפעולות הבאות (מניחים שמספר האיברים הנוכחי הוא  $n$ ):

- הכנסת איבר חדש בזמן  $O(\log n)$ ;
- חיפוש איבר בזמן  $O(\log n)$ ;
- הוצאת איבר בזמן  $O(\log n)$  (כשמקבלים מצביע לאיבר שרוצים להוציא);
- מציאת האיבר בעל המפתח  $k$  הכי קטן בזמן  $O(\log n)$  (כאשר  $k$  ניתן כפרמטר);

### פתרון

נשתמש בעץ 2-3 סטנדרטי  $T$  ובנוסף לכל קודקוד  $x$  נממש את השדה  $size(x)$  שיחזיק את מספר העלים בתת העץ  $T_x$ .

הרעיון המרכזי בתחזוקת שדות ה  $size(\cdot)$  הוא שאחרי כל שינוי בצומת פנימי  $y$ , נעדכן את השדה  $size(y)$  להיות סכום שדות ה  $size(\cdot)$  של בניו של  $y$  בעץ ולאחר מכן נעבור לטפל באבא של  $y$  בעץ. מכיוון שגם בתהליך ההכנסה וגם בתהליך ההוצאה מספר הקודקודים שמשתנים בכל רמה הוא קבוע (בפרט כולם בניים לאותו אבא), ומכיוון שמספר הרמות בעץ 2-3 הוא  $O(\log n)$ , סך כל תוספת העבודה הנדרשת היא  $O(\log n)$  (גם בהכנסה וגם בהוצאה).

נסתכל על תהליך הכנסת איבר חדש ל  $T$  ונניח שהוספנו את העלה החדש  $x$  לעץ. ראשית נאתחל את  $size(x)$  להיות 1. כעת נטפס מ  $x$  אל השורש ולכל קודקוד פנימי  $y$  שמהווה אב קדמון של  $x$ , נעדכן את  $size(y)$  באופן הבא. אם אין צורך לפצל את  $y$  (לכל היותר 3 בניים), אז  $size(y)$  מקבל את סכום משתני ה  $size(\cdot)$  של בניו של  $y$  וניתן להמשיך לאבא של  $y$ . אחרת, נניח ש  $y$  הוא קודקוד בעל 4 בניים  $z_1, z_2, z_3, z_4$  ואנו מעוניינים לפצל את  $y$  ל  $y_1$  (עם בניים  $z_1, z_2$ ) ו  $y_2$  (עם בניים  $z_3, z_4$ ). בזמן ביצוע הפיצול נאתחל  $size(y_1) = size(z_1) + size(z_2)$  ו  $size(y_2) = size(z_3) + size(z_4)$  ולאחר מכן ניתן להמשיך לאבא של  $y$  בעץ.

נסתכל על תהליך הוצאת איבר מ  $T$  ונניח שהוצאנו את העלה  $x$  מהעץ. נטפס מ  $x$  אל השורש ולכל קודקוד פנימי  $y$  שמהווה אב קדמון של  $x$ , נעדכן את  $size(y)$  באופן הבא. אם אין צורך להשלים את  $y$  בבניים נוספים (לפחות 2 בניים), אז  $size(y)$  מקבל את סכום משתני ה  $size(\cdot)$  של בניו של  $y$  וניתן להמשיך לאבא של  $y$ . אחרת, נניח ש  $y$  הוא קודקוד בעל בן יחיד. אם ניתן להשלים את  $y$  בבן נוסף מאחיו  $y'$ , אז נבצע את ההשלמה ואחריה נעדכן את  $size(y)$  ואת  $size(y')$  להיות סכום משתני ה  $size(\cdot)$  של בנייהם בעץ (החדש). אחרת, נבטל את  $y$ , נצרף את בנו היחיד של  $y$  לאח  $y'$  של  $y$  ואז נעדכן את  $size(y')$  להיות סכום משתני ה  $size(\cdot)$  של בניו בעץ (החדש). לאחר מכן ניתן להמשיך לאבא של  $y$  בעץ.

כיצד נמצא את האיבר בעל המפתח ה  $k$  הכי קטן? נרד במורד העץ מהשורש כשבכל רמה אנחנו מחליטים לאיזה בן להמשיך באמצעות שדות ה  $size(\cdot)$  וערך ה  $k$  הנוכחי (שמתעדכן כל הזמן).

פסאודו-קוד:

**rank**( $T, k$ )

1.  $x = root(T)$
2. if ( $size(x) < k$ ) then return( NULL ) // there is no such element
3. while ( $is\_leaf(x) == false$ )
  - 3.1. do
  - 3.2. if ( $k \leq size(first(x))$ )
    - 3.2.1. do // the desired element is in the first subtree
    - 3.2.2.  $x = first(x)$
    - 3.2.3. continue
    - 3.2.4. done
  - 3.3.  $k = k - size(first(x))$  // accounting for the elements we pass
  - 3.4. if ( $k \leq size(second(x))$ )
    - 3.4.1. do // the desired element is in the second subtree
    - 3.4.2.  $x = second(x)$
    - 3.4.3. continue
    - 3.4.4. done
  - 3.5.  $k = k - size(second(x))$  // accounting for the elements we pass
  - 3.6.  $x = third(x)$  // the desired element is in the third subtree
  - 3.7. done
4. return( $x$ )