מערך מעגלי

נשמור: $\text{array}, \text{maxlen}, \text{length}, \text{start}$. האיבר ה־$i$ נמצא ב: $\text{array}\left[(L.\text{start}+i) \mod L.\text{maxlen}\right]$. היתרון על סתם מערך זה שאפשר למחוק/להכניס איבר <u>משני הכיוונים</u> ב־$O(1)$ (כדי למחוק/להכניס מההתחלה נוסיף/נחסיר מ־start, מודולו maxlen).

אתחול מערך מגודל n ב־$O(1)$

נשמור שלושה מערכים: A המערך עצמו, Legals מערך אינדקסים חוקיים, Positions מערך אינדקסים ל־Legals, ונשמור גם LegalsSize כמות האיברים ב־Legals. כל אינדקס שמופיע ב־Legals עד ל־LegalsSize הוא אינדקס שאותחל כבר. ועבור כל אינדקס שאותחל נשמור ב־Positions את המיקום של האינדקס ב־Legals.

אפשר לדעת האם האיבר ה־i אותחל לפי Positions: אם האינדקס $\text{Positions}[i]$ בתוך התחום של LegalsSize, וגם האיבר שמופיע ב־Legals במקום הזה הוא האינדקס, אז האיבר ה־i אותחל כי מובטח לנו שכל מה שבתוך התחום של LegalsSize הוא באמת אינדקס מאותחל. בקריאה נבדוק האם איבר מאותחל ובכתיבה אם האיבר לא מאותחל אז נוסיף את האינדקס ל־Legals ונרשום את המיקום ב־Positions.

עץ חיפוש בינארי

תכונה: לכל צומת, המפתח של כל איבר בתת העץ השמאלי קטן, והמפתח של כל איבר בתת העץ הימני גדול מהמפתח של הצומת.

פעולת Successor: אם יש בן ימני, נלך ימינה ואז כל הדרך שמאלה. אם אין בין ימני, נלך למעלה עד לפניה הראשונה ימינה (כלומר ־ שהגענו למעלה דרך הבן השמאלי). אם המפתח היה כבר הכי גדול אז נקבל null.

```
if x.right ≠ null then:
    return Min (x.right)
y ← x.parent
while y ≠ null and x = y.right
do:
    x ← y
    y ← x.parent
return y
```

בנוסף: אם עוברים על כל העץ אז עוברים על כל קשת לכל היותר פעמיים לכן העלות היא $O(n)$.

פעולת Delete: אם אין ילדים ־ קל. אם יש רק בן אחד ־ נחליף בהורה את האיבר בבן שלו. אם יש שני ילדים: נחליף את x ב־successor של x, שנסמנו ב־y. ל־y אין בן שמאלי (כי הולכים ימינה ואז כל הדרך שמאלה). לכן נוציא את y מהעץ ונכניס אותו חזרה במקום x.

מעבר על העץ

Pre-Order

```
if x ≠ null then
    Process (x)
    Pre-Order (x.left)
    Pre-Order (x.right)
```

In-Order

```
if x ≠ null then
    In-Order (x.left)
    Process (x)
    In-Order (x.right)
```

Post-Order

```
if x ≠ null then
    Post-Order (x.left)
    Post-Order (x.right)
    Process (x)
```

עץ AVL

תכונה: נגדיר $\text{BF}(v) = h(v.\text{left}) - h(v.\text{right})$, אז $|\text{BF}(v)| \leq 1$ לכל צומת v. נובע שהגובה הוא $O(\log n)$, הוכחה עם עץ פיבונאצ'י (הבן השמאלי של F_h הוא F_{h-2} והימני F_{h-1}).

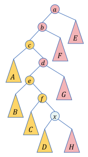
פעולת רוטציה: מתקנת אם $|\text{BF}(v)| = 2$ אבל מה שמתחת בסדר. אם $\text{BF}(v) = 2$ נפצל למקרים לפי ה־BF של הבן השמאלי $(-1, 0/1)$, ואם $\text{BF}(v) = -2$ נפצל למקרים לפי ה־BF של הבן הימני $(-1/0, 1)$.

פעולת הכנסה: נכניס כרגיל, ואז נעלה למעלה בעץ ונחפש AVL criminals (עם $|\text{BF}(v)| = 2$). אם ה־BF חוקי והגובה לא השתנה בפעולת ההכנסה אז אפשר לעצור, אחרת נמשיך לעלות למעלה ואם $|\text{BF}(v)| = 2$ נבצע רוטציה. אחרי הכנסה יש לכל היותר רוטציה אחת לכן אפשר לעצור גם במקרה הזה.

פעולת מחיקה: יכול להיות שיהיו כמה רוטציות בדרך למעלה, ונתחיל מההורה של מי שמחקנו באמת (למשל זה יכול להיות ה־successor).

פעולת Join ששומרת על BF: נניח ש־$T_1 < x < T_2$ (כל איבר ב־T_1, T_2). סתם Join זה קל: ניצור שורש x, את T_1 נשים משמאל ואת T_2 נשים מימין לשורש. כדי לשמור על BF: בה"כ $h(T_1) \leq h(T_2)$. נמצא את b הצומת הראשון בשושלת השמאלית (ללכת כל הזמן שמאלה) של T_2 כך ש־$h(b) \leq h(T_1)$ (ולכן $h(b) \in \{h(T_1) - 1, h(T_1)\}$). נסמן ב־$c$ את ההורה של b, נצרף מתחת ל־c את x כך שהבן השמאלי שלו הוא T_1 והבן הימני שלו הוא תת העץ של b. לאחר מכן נבצע תיקונים בדרך למעלה החל מ־x. סך הכל $O(\log n)$.

פיצול עץ באמצעות Join: בהינתן צומת x בעץ T, נרצה לפצל לשני עצים: $T_1 < x < T_2$. נעשה Join לכל הכתומים ולכל הורודים ונקבל שני עצים.



הסיבוכיות יוצאת $O(\log n)$, כי כל Join עולה כמו ההפרש בין הגבהים.

עצי rank

פעולות חדשות: Select מחזיר את האיבר ה־k בגודלו, Rank מחזיר את המיקום של איבר בסדר הממוין.

נשמור לכל צומת **שדה חדש** שנקרא size. אפשר לעדכן אותו בכל הפעולות ולשמור על אותו זמן הריצה כי זה <u>שדה נוסף שתלוי רק בבנים הישירים של הצומת</u>.

פעולת $\text{Tree-Select}(T, k)$:

```
r ← T.left.size + 1
if k = r: return T
if k < r: return Tree-Select (T.left, k)
if k > r: return Tree-Select (T.right, k − r)
```

פעולת $\text{Rank}(x)$: נאתחל את ה־rank עם כמות הצמתים בתת העץ השמאלי ועוד 1 (x עצמו). נלך למעלה ובכל צומת שבאנו אליו מימין נוסיף גם את כמות הצמתים בתת העץ השמאלי שלו ועוד 1.

finger trees

nO.