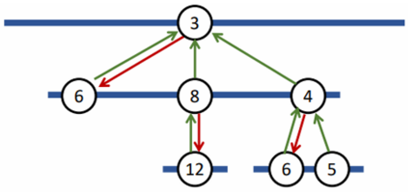
# ערמת פיבונאצ'י

ערמת מינימום. רשימה מקושרת דו-כיוונית של שורשים, עם מצביע min למינימלי מביניהם. כל קודקוד מחזיק: m – דגל marked, left, right - אחים, π – אבא, c – אחד הילדים, degree – מספר ילדים, k – ערך שמור.




לכל קודקוד יש את המצביעים הבאים:
מצביע לאבא.
מצביע לאח ימין ולאח שמאל (כאשר האחים הם גם ברשימה מקושרת דו-כיוונית מעגלית).
מצביע לאחד מהילדים שלו (כדי להגיע לבן אחר, ניתן לעבור דרך המצביעים של האחים).

**הוספת קודקוד:** מוסיפים בתור שורש חדש, מעדכנים את min. O(1)

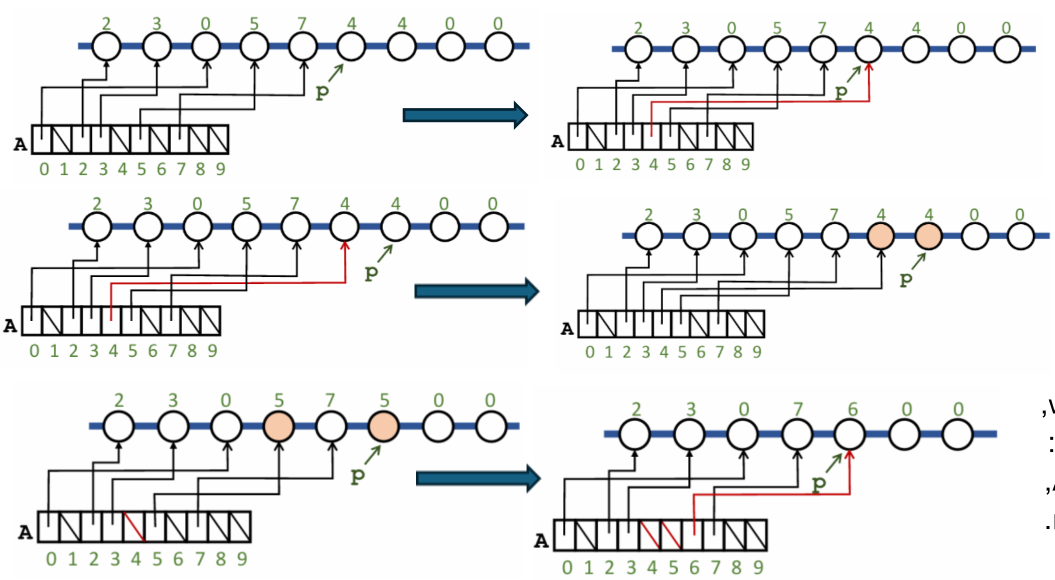
**הוצאת המינימלי:** כל אחד מהילדים שלו נהייה שורש חדש, ומבצעים consolidate – זה מוודא שאין שני שורשים מאותה דרגה:
נשמור מערך A בגודל D. D הוא חסם עליון לדרגה המקסימלית – נקבע אותו בהמשך.
A(i) מצביע על השורש שיש לו דרגה i. עוברים על המערך עם מצביע p. ←
אינווריאנטה של הלולאה: בשורשים עד p, יש לכל היותר שורש אחד מכל דרגה. ואם יש אז A מצביע עליו.

```
for(p ∈ roots)
  while(A[deg(p)] points to root q)
    A[deg(p)] ← null
    merge q, p
  A[deg(p)] ← p
```

דוגמה:
A לא מצביע על שורש באותה דרגה כמו p, אז לא נכנסים לwhile.
מעדכנים את A שיצביע על p:

ומקדמים את p, בלולאה הראשית: עכשיו A מצביע על שורש באותה דרגה כמו p:

נאחד אותם ונמחק את המצביע בA. אנחנו עדיין בwhile, ו-A מצביע לשורש בדרגה כמו p:
נאחד ונמחק. עכשיו אין כפיל בA, אז ה while מסתיים ו-p מתקדם.



**הקטנת ערך – decreaseKey:**
מקטינים את הערך של הקודקוד. אם תכונת הערמה נהרסת, מנתקים את הקודקוד והופכים אותו לשורש חדש ומעדכנים את min. לקודקוד שהוא לא שורש מותר לאבד עד ילד אחד – כשזה קורה, מסמנים אותו marked. אם קודקוד מסומן מאבד ילד – הוא הופך לשורש. (ואם גם אבא שלו היה כבר מסומן, גם הוא הופך לשורש...)

**ניתוח זמן ריצה:**
טענה 1: יהי v קודקוד מדרגה k, ויהיו $w_1 \cdots w_k$ הילדים שלו, לפי הסדר שבו הם נוספו לv.
(יכול להיות שהיו עוד ילדים שנאבדו). אז לכל $i \geq 2$, מתקיים $d(w_i) \geq i-2$.
הוכחה: מוסיפים ילדים רק לשורשים במהלך consolidate. כאשר $w_i$ נוסף לv, הם היו מאותה דרגה: לפחות $i-1$. מאז, $w_i$ איבד לכל היותר ילד אחד.
מסקנה: יהי N(k) המספר המינימלי של קודקודים בשורש מדרגה k. אז $N(k) = F_{k+2}$ (המספר פיבונאצ'י ה- k+2).
הוכחה באינדוקציה על k:

$$N(0) = 1, \quad N(1) = 1 + N(0) = 2, \quad N(2) = 1 + N(0) + N(0), \quad N(3) = 1 + N(0) + N(0) + N(1) \ldots$$

ניח שמתקיים עבור k-1, נוכיח עבור k: יהי שורש r מדרגה k, $v_1 \cdots v_k$ הילדים שלו לפי הסדר שבו הם נוספו. מטענה 1, $d(v_i) \geq i-2$. ומהנ"א, $N(i-2) = F_i$. אז:

$$N(k) \geq 2 + \sum_{i=2}^{k} N(i-2) = 2 + \sum_{i=2}^{k} F_i = 1 + \sum_{i=0}^{k} F_i = F_{k+2}$$

טענה 2: אם נתחיל מערמה ריקה, ונבצע:

$n$ הכנסות, $m \le n$ הוצאות, $k$ הקטנות, זמן הריצה הכולל יהיה:

| no. | basic operation | triggered by | # times |
|---|---|---|---|
| 1 | create new root | `insert()` | $x_1 = n$ |
| 2 | delete root | `deleteMin()` | $x_2 = m$ |
| 3 | move child of deleted root to root list | `deleteMin()` | $x_3 \le m \log n$ |
| 4 | make root child of another root | `deleteMin()` | $x_4 \le x_1 + x_3 + x_8 + x_9$ |
| 5 | set **A** entry to **NULL** | `deleteMin()` | $x_5 = m \log n + x_4$ |
| 6 | set **A** entry to **p** | `deleteMin()` | $x_6 \le x_5$ |
| 7 | set **marked = true** | `decreaseKey()` | $x_7 \le k$ |
| 8 | move decreased node to root list | `decreaseKey()` | $x_8 \le k$ |
| 9 | move marked node to root list | `decreaseKey()` | $x_9 \le x_7$ |

הסברים:

1) ע"פ הגדרה
2) ע"פ הגדרה
3) לשורש מדרגה k יש לפחות $F_{k+2}$ צאצאים, ולכל אחד מהילדים שלו יש דרגה לפחות i-2. אז אם יש n קודקודים, לכל שורש יש לכל היותר logn ילדים. אז אם נמחק m קודקודים, נצטרך להזיז לכל היותר mlogn קודקודים.
4) זה יכול לקרות לכל היותר כמספר השורשים שיש. נסכם את כל הפעולות שמייצרות שורש: $X_1 + X_3 + X_8 + X_9$.
זה יוצא בסה"כ $n + m \cdot \log n + 2k \in O(n + m \log n + k)$.
5) קורה בהתחלה של כל consolidate, האורך של A זה $\log n$. m פעמים. וגם בכל פעם שממזגים בתוך consolidate. שזה קורה עד $X_4$ פעמים.
6) קורה לכל היותר מספר פעמים כמו $X_5$.
7) קורה רק כשקודקוד שהוא לא שורש מאבד ילד – רק בהקטנות. כל קודקוד יכול לקבל סימון רק פעם אחת.
8) כנ"ל, קורה רק בהקטנה.
9) קורה רק אם קודקוד מסומן איבד ילד, אז לכל היותר כמספר הקודקודים המסומנים.

בסה"כ: $O(n + m\log n + k)$.