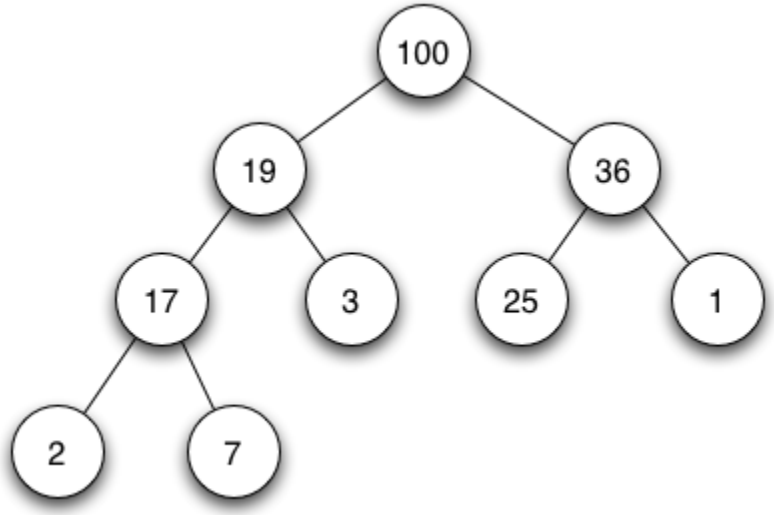
מבני נתונים 2010

ערמות בינומיות ופיבונצ'י

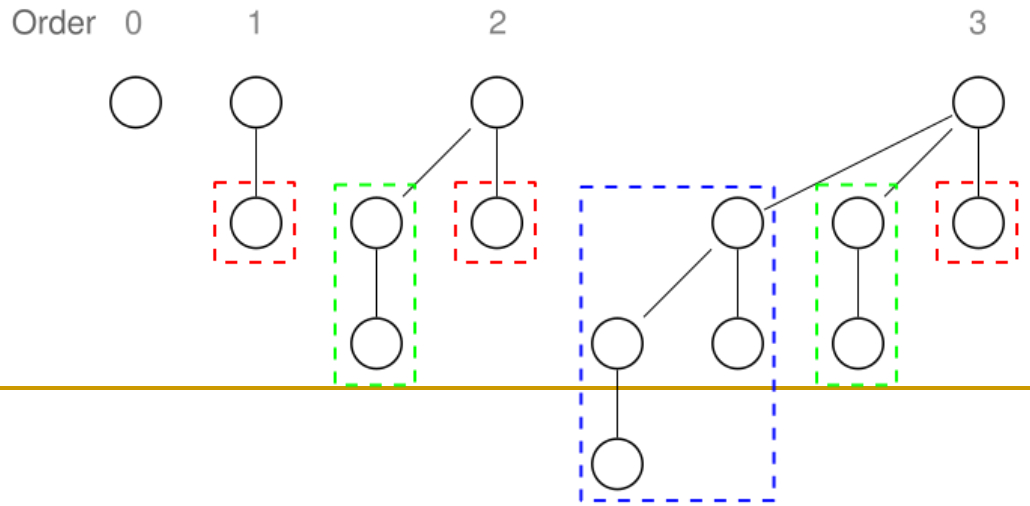
תזכורת: Heaps

- עץ בינארי מלא
- החוק הבסיסי
  - אם צומת B צאצא של צומת A אזי Key(A)≤Key(B)
- הפעולות הנתמכות
  - Find-min
  - Delete-min
  - Decrease-key
  - Insert
  - Merge



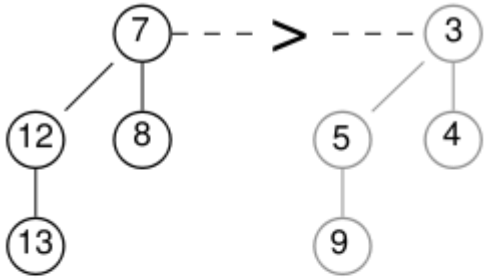
Binomial Heaps :תזכורת

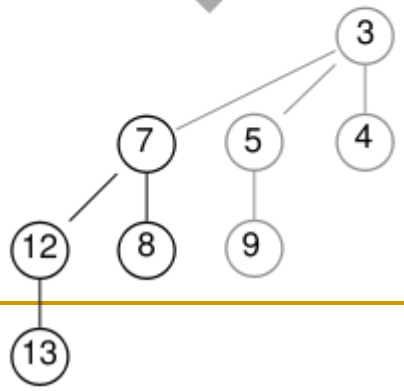
- תחילה נגדיר Binomial Tree:
  - עץ בינומי מדרגה 0 מכיל צומת יחידה
  - עץ בינומי מדרגה k הוא
    - בעל שורש מדרגה k
    - ילדיו הינם עצים בינומיים מדרגות k-1,k-2,…,0 (בסדר זה)
    - עץ בינומי מדרגה k מכיל 2^k צמתים והינו בגובה k



Binomial Heaps :תזכורת

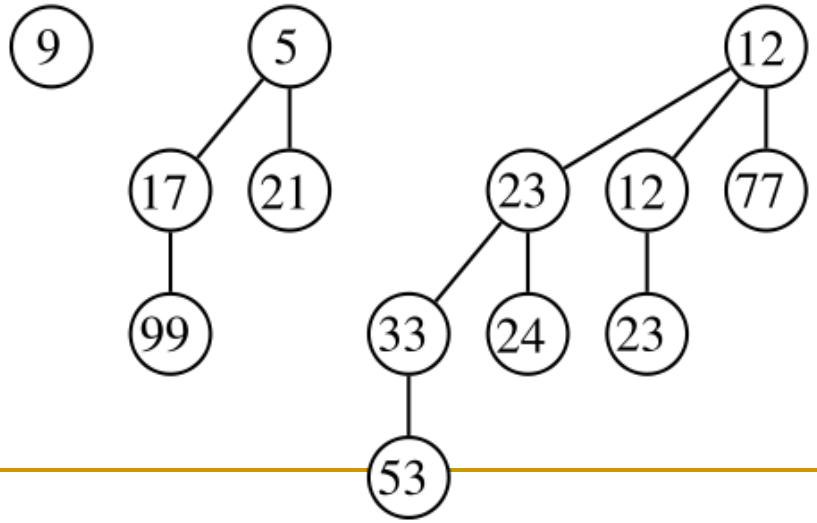
- איחוד שני עצים בינומיים מסדר k-1
  - ניצור עץ בינומי מסדר k ע"י תליית אחד העצים כבן השמאלי ביותר של העץ השני





Binomial Heaps :תזכורת

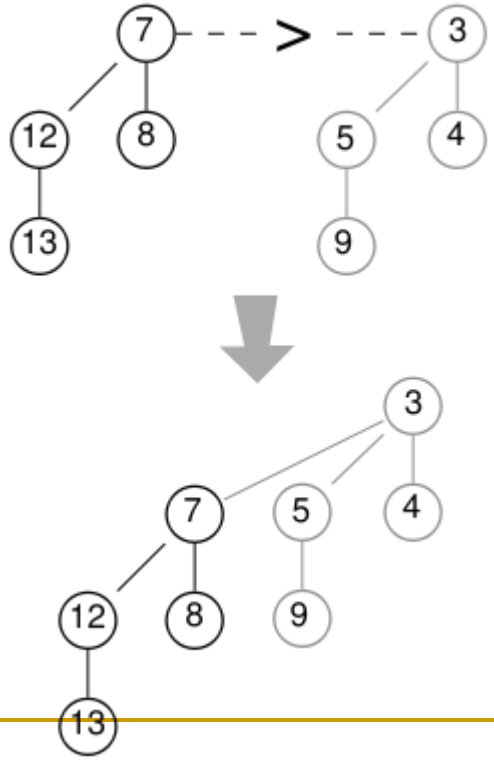
- הגדרת Binomial Heap
  - סט עצים בינומיים המקיימים את התכונות הבאות
    - כל עץ מקיים את תכונת minimum-heap (כל צאצא גדול מההורה שלו)
    - עבור כל סדר k של עץ בינומי, יש 0 או 1 עצים כאלו ב-heap



Binomial Heaps :תזכורת

- פעולת Merge של שני עצים מדרגה k

```
function mergeTree(p, q)
   if p.root <= q.root
      return p.addSubTree(q)
   else
      return q.addSubTree(p)
end
```



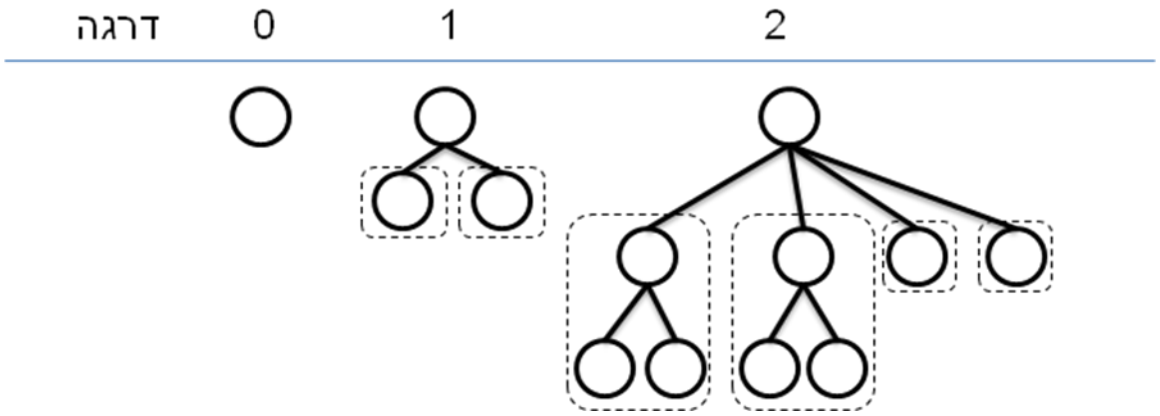
Binomial Heaps :תזכורת

- פעולת Insert
  - ניצור heap חדש המכיל את האבר החדש, ונבצע merge בין שני ה-heaps
- פעולת minimum
  - עלינו לחפש את הערך המינימלי מבין שורשי העצים בheap
- פעולת delete-min
  - מצא את האבר ומחק אותו
  - הפוך את בניו ל-binomial heap ומזג את שני ה-heaps
- פעולת Decrease-key
  - בדומה לפעולות ב-heap רגיל
- פעולת Delete
  - שנה את הערך ל-∞ ובצע delete-min

הדגמה באנימציה של Binomial Heaps: http://www.cse.yorku.ca/~aaw/Sotirios/BinomialHeap.html

נגדיר עצים בינומיים "שמנים" בצורה הבאה:

- עץ בינומי "שמן" מדרגה 0 מכיל צומת אחד בלבד.
- עץ בינומי "שמן" מדרגה k ניתן לבנות משלושה עצים בינומיים "שמנים" מדרגה k-1 כאשר נתלה שניים מהם על העץ השלישי.



תארו מבנה נתונים אנלוגי לערמה בינומית (binomial heap) המשתמש בעצים "שמנים".

ערמה בינומית "שמנה" מוגדרת בצורה הבאה:
בערמה יש n עצים בינומיים "שמנים" כאשר יש לכל היותר 2 עצים מדרגה k, לכל k. שורשי העצים מחוברים ביניהם ברשימה מקושרת. הפעולות מוגדרות בדומה לערמה בינומית רגילה וכאשר יש צורך לעשות merge בין העצים (יש יותר משני עצים מדרגה k) בונים משלושה עצים מדרגה k עץ יחיד מדרגה k+1.

תארו פעולת meld של שתי ערמות בינומיות "שמנות" שהגדרתם בסעיף הקודם.

פעולת meld הינה מיזוג בין שתי ערמות בינומיות "שמנות". הדבר מקביל לחיבור שני מספרים בבסיס 3 (כמו שבערמות בינומיות הדבר היה מקביל לחיבור מס' בבסיס 2). נתחיל מדרגה 0 ונוסיף את העצים בערמה אחת לשנייה. אם יש סה"כ 3 או יותר עצים מדרגה 0 בערמה החדשה נמזג שלושה ליצירת עץ יחיד מדרגה 1. כעת נעבור לדרגה הבאה (1) ונמשיך כך...

ערמות פיבונאצ'י

Operation	Linked List	Binary Heap	Binomial Heap	Fibonacci Heap †	Relaxed Heap
make-heap	1	1	1	1	1
is-empty	1	1	1	1	1
insert	1	log *n*	log *n*	1	1
delete-min	*n*	log *n*	log *n*	log *n*	log *n*
decrease-key	*n*	log *n*	log *n*	1	1
delete	*n*	log *n*	log *n*	log *n*	log *n*
union	1	*n*	log *n*	1	1
find-min	*n*	1	log *n*	1	1

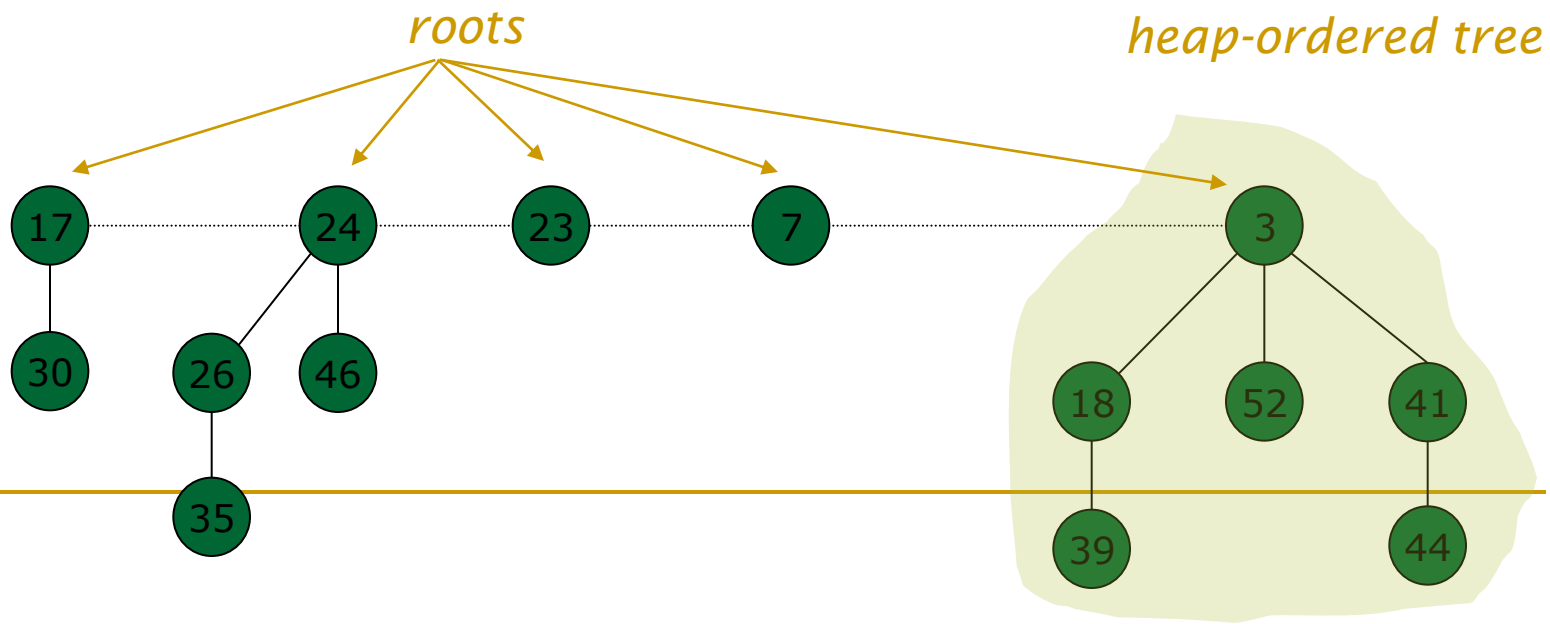
n = number of elements in priority queue

† amortized

ערימות פיבונאצ'י - מבנה

- Fibonacci heap.
  - Set of heap-ordered trees.
  - Maintain pointer to minimum element.
  - Set of marked nodes.

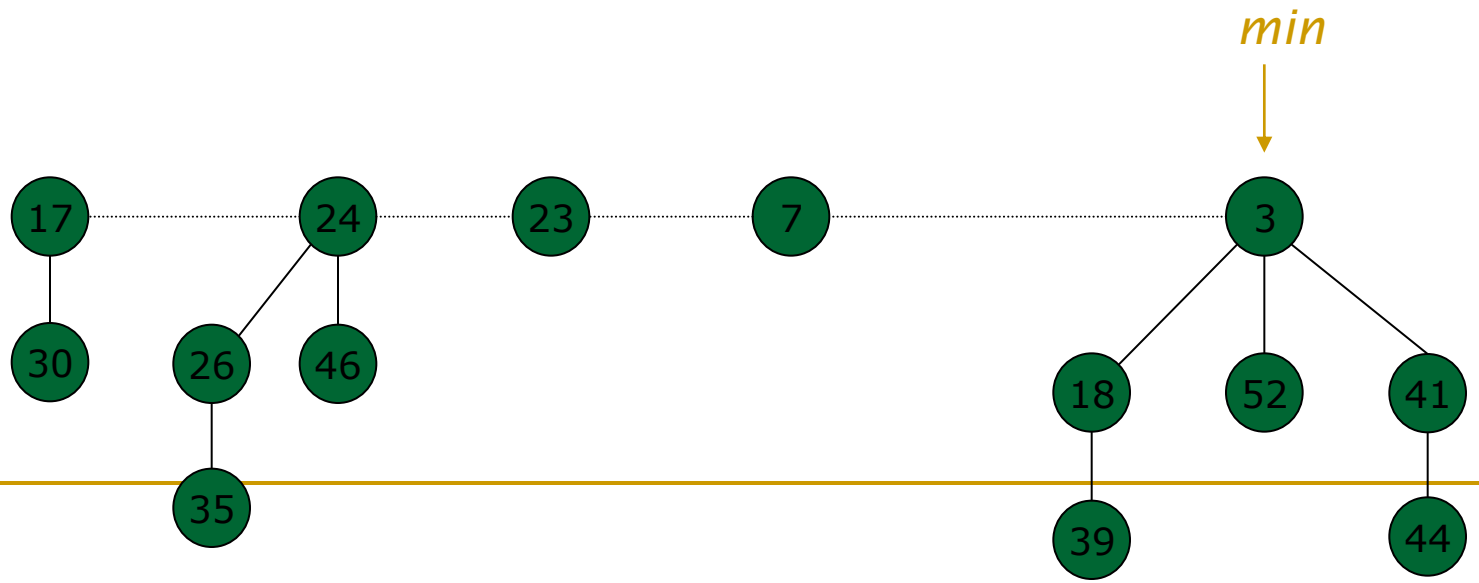
each parent larger than its children



ערימות פיבונאצ'י - מבנה

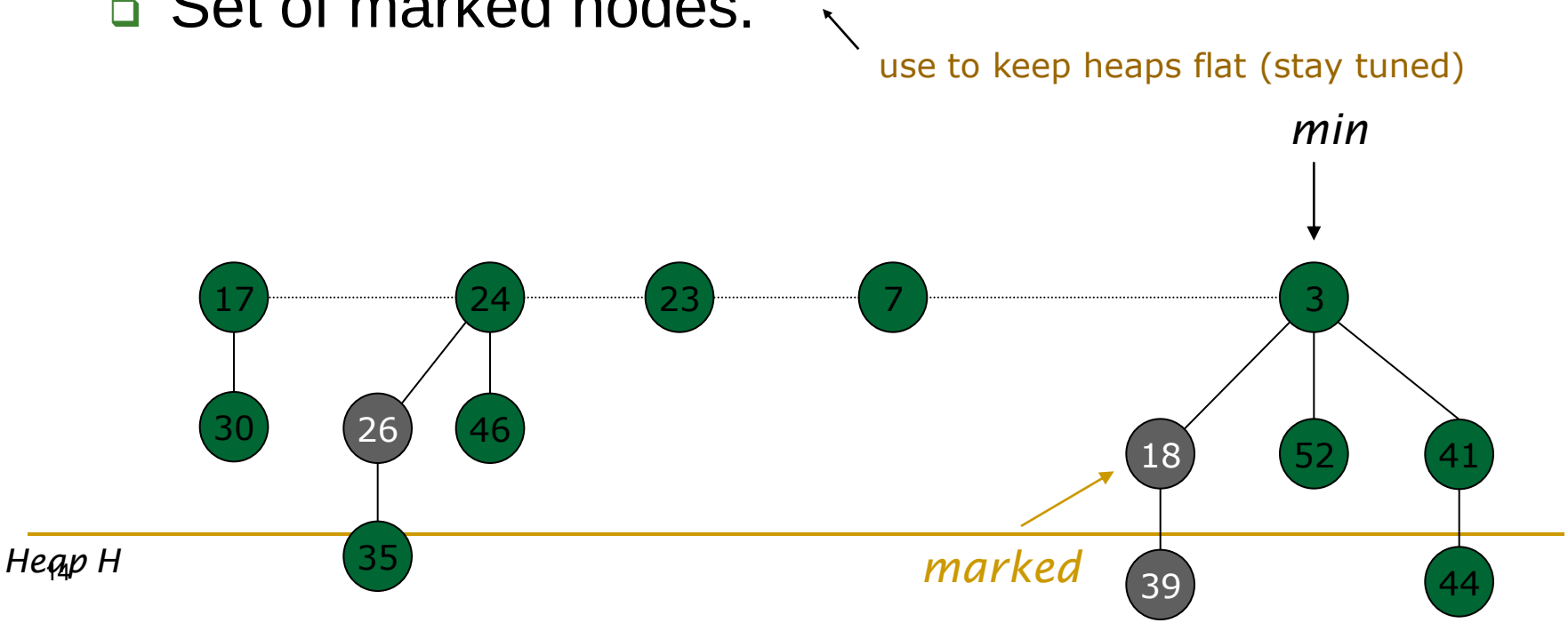
- Fibonacci heap.
  - Set of heap-ordered trees.
  - Maintain pointer to minimum element.
  - Set of marked nodes.

find-min takes O(1) time



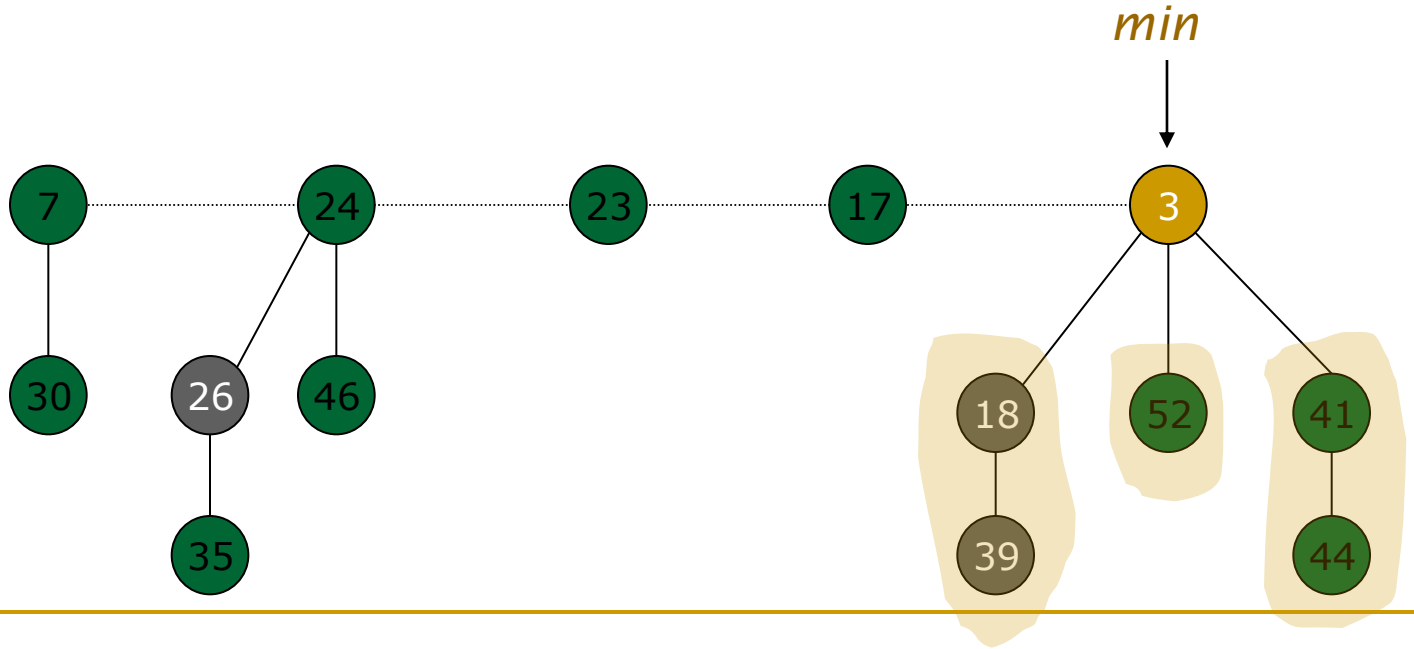
ערימות פיבונאצ'י - מבנה

- Fibonacci heap.
  - Set of heap-ordered trees.
  - Maintain pointer to minimum element.
  - Set of marked nodes.



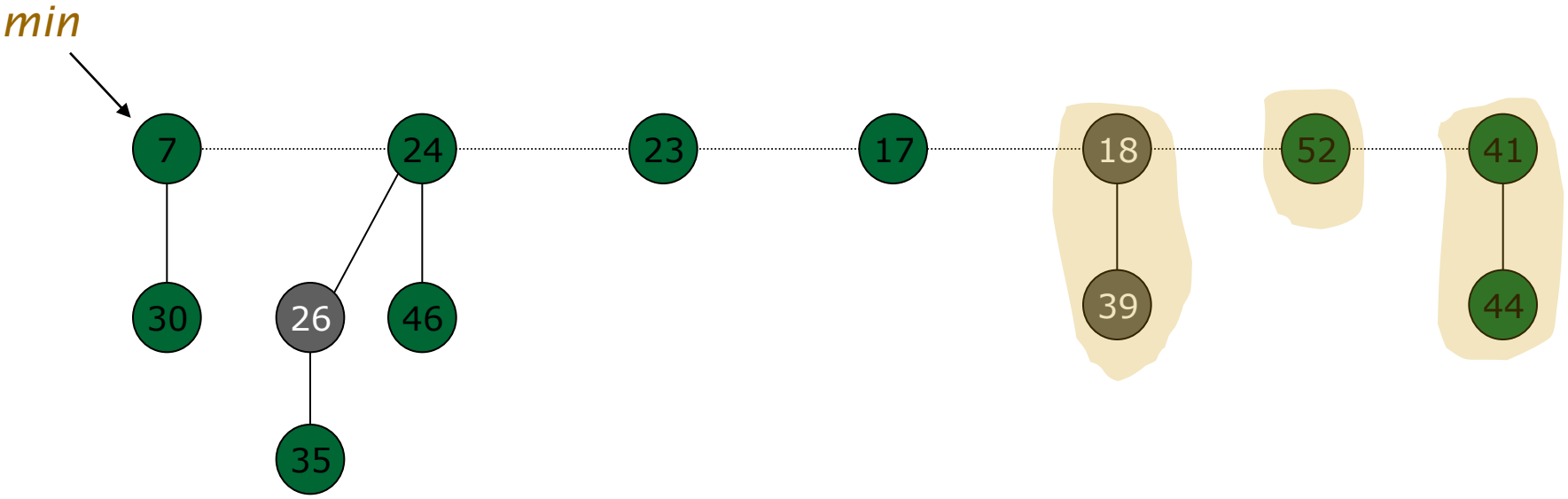
Cascading cuts & Successive linking

- פעולת delete-min
  - יודעים מי המינימום בערימה
  - נתלה את הבנים שלו כעצים בערימה
  - נבצע successive linking – מכל דרגה עץ יחיד!



Cascading cuts & Successive linking

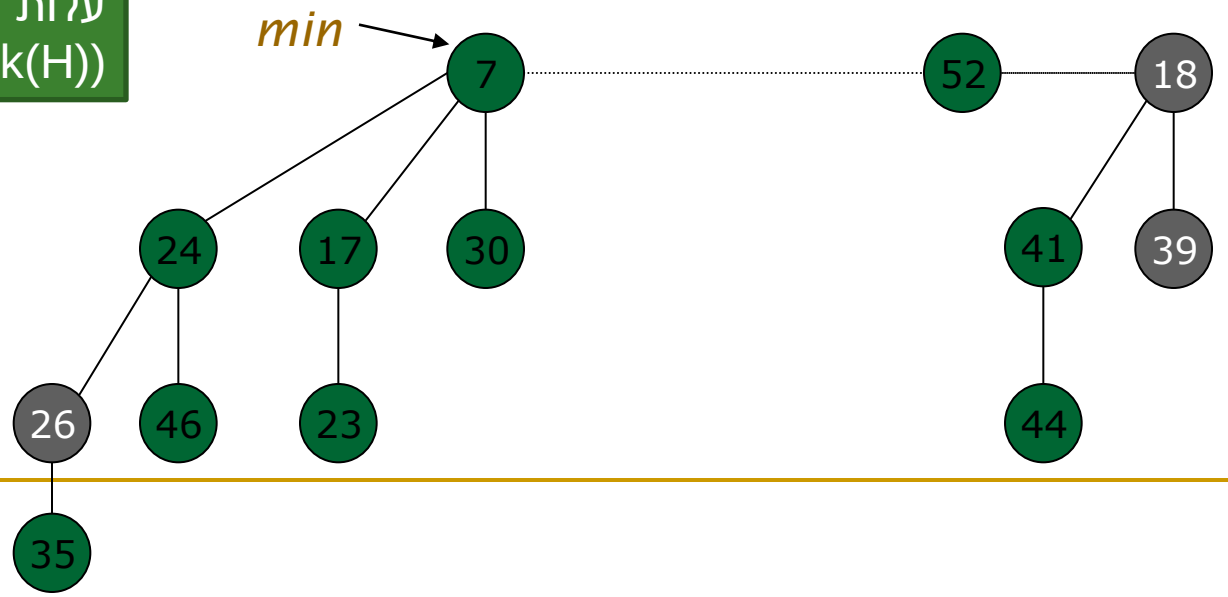
- פעולת delete-min
  - יודעים מי המינימום בערימה
  - נתלה את הבנים שלו כעצים בערימה
  - נבצע successive linking – מכל דרגה עץ יחיד!



Cascading cuts & Successive linking

- פעולת delete-min
  - יודעים מי המינימום בערימה
  - נתלה את הבנים שלו כעצים בערימה
  - נבצע successive linking – מכל דרגה עץ יחיד!

עלות הפעולה:
Amortized O(rank(H))



ערמות פיבונצ'י

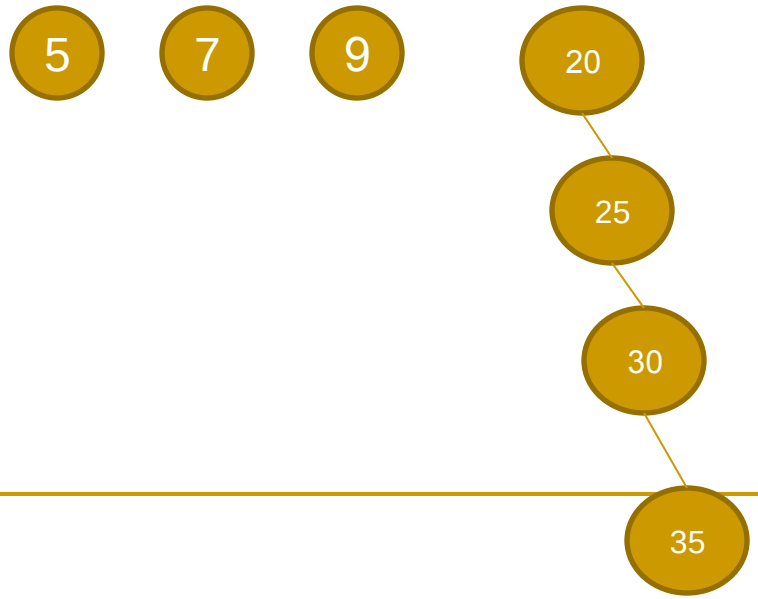
- כמו שראיתם בכיתה:

$$Rank(H)<\log_{\Phi}(n)$$

תרגיל 1 – ערמות פיבונאצ'י

- האם ניתן לבנות ערמת פיבונאצ'י ובה עץ אחד מעומק n?
- פתרון
  - נניח שאנו יודעים לפתור עבור n=k
  - נפתור עבור n=k+1
    - נגדיר x',y',z' כך ש-
    - נכניס אותם לתוך הערמה
    - נפעיל extract-min

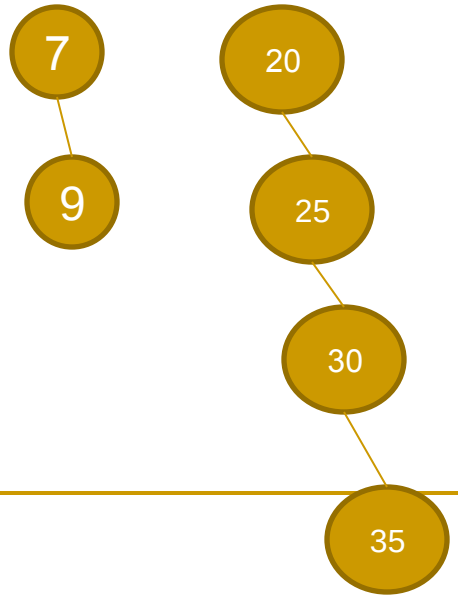
$$key[z'] < \mathrm{k}\, ey[y'] < key[x'] < \min[H]$$



תרגיל 1 – ערמות פיבונאצ'י

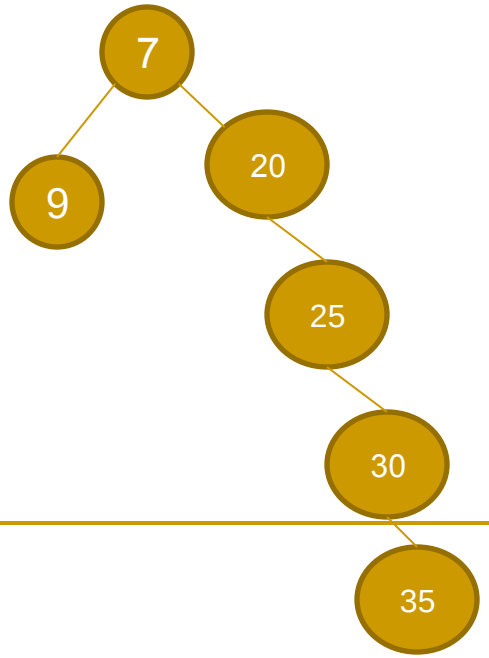
- האם ניתן לבנות ערמת פיבונאצ'י ובה עץ אחד מעומק n?
- פתרון
  - נניח שאנו יודעים לפתור עבור n=k
  - נפתור עבור n=k+1
    - נגדיר x',y',z' כך ש-
    - נכניס אותם לתוך הערמה
    - נפעיל extract-min

$$key[z'] < \mathrm{k}\, ey[y'] < key[x'] < \min[H]$$



תרגיל 1 – ערמות פיבונאצ'י

- האם ניתן לבנות ערמת פיבונאצ'י ובה עץ אחד מעומק n?
- פתרון
  - נניח שאנו יודעים לפתור עבור n=k
  - נפתור עבור n=k+1
    - נגדיר x',y',z' כך ש-
    - $key[z'] < key[y'] < key[x'] < \min[H]$
    - נכניס אותם לתוך הערמה
    - נפעיל extract-min
    - נמחק את x' (9)



תרגיל 2

- בערמות פיבונצ'י, אנחנו מבצעים cascading cuts בצומת v אם הוא איבד צומת בן מאז הפעם האחרונה שהוא נתלה על צומת אחרת. נניח שנבצע CC רק אם v איבד שני בנים מאז, כיצד משתנה הלמה:
  - x צומת בערמת פיב', $y_1,\ldots,y_n$ בנים של x, מסודרים לפי הסדר בו נתלו על x (הכי ישן ועד הכי חדש). אזי $rank(y_i) \geq i-2$ לכל i.

תרגיל 2

- Answer
  - $rank(y_i) \geq i-3$
  - Since y_i had the same rank as x when it became a child of x
  - x must have had at least i-1 children at that time, so y_i had at least i-1 rank.
  - It could have lost at most two children since then, therefore rank at least i-3