



# מבני נתונים ב'07

תרגול 5

30/5/2007 ~~11/4/2007~~

## ערמות



ליאור שפירא

# תזכורת: Heaps

□ עץ בינארי מלא

□ החוק הבסיסי

■ אם צומת B צאצא של צומת A אזי  $\text{Key}(A) \leq \text{Key}(B)$

□ הפעולות הנתמכות

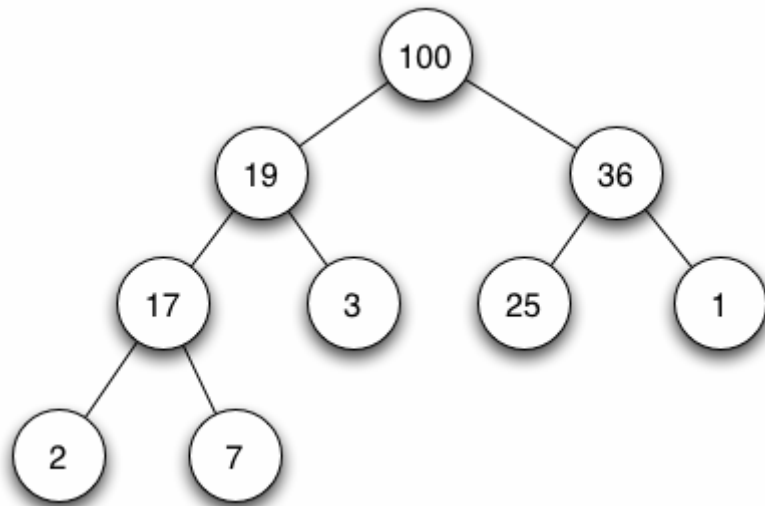
■ Find-min

■ Delete-min

■ Decrease-key

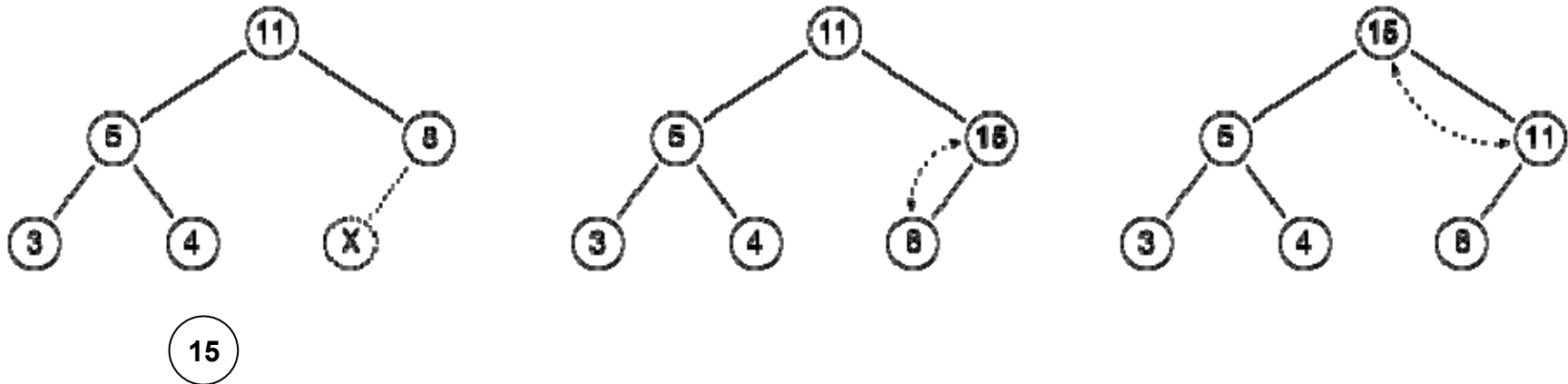
■ Insert

■ Merge



# תזכורת: Heaps

הוספת צומת (עבור max-heap)



מחיקת השורש



# תרגיל 1

---

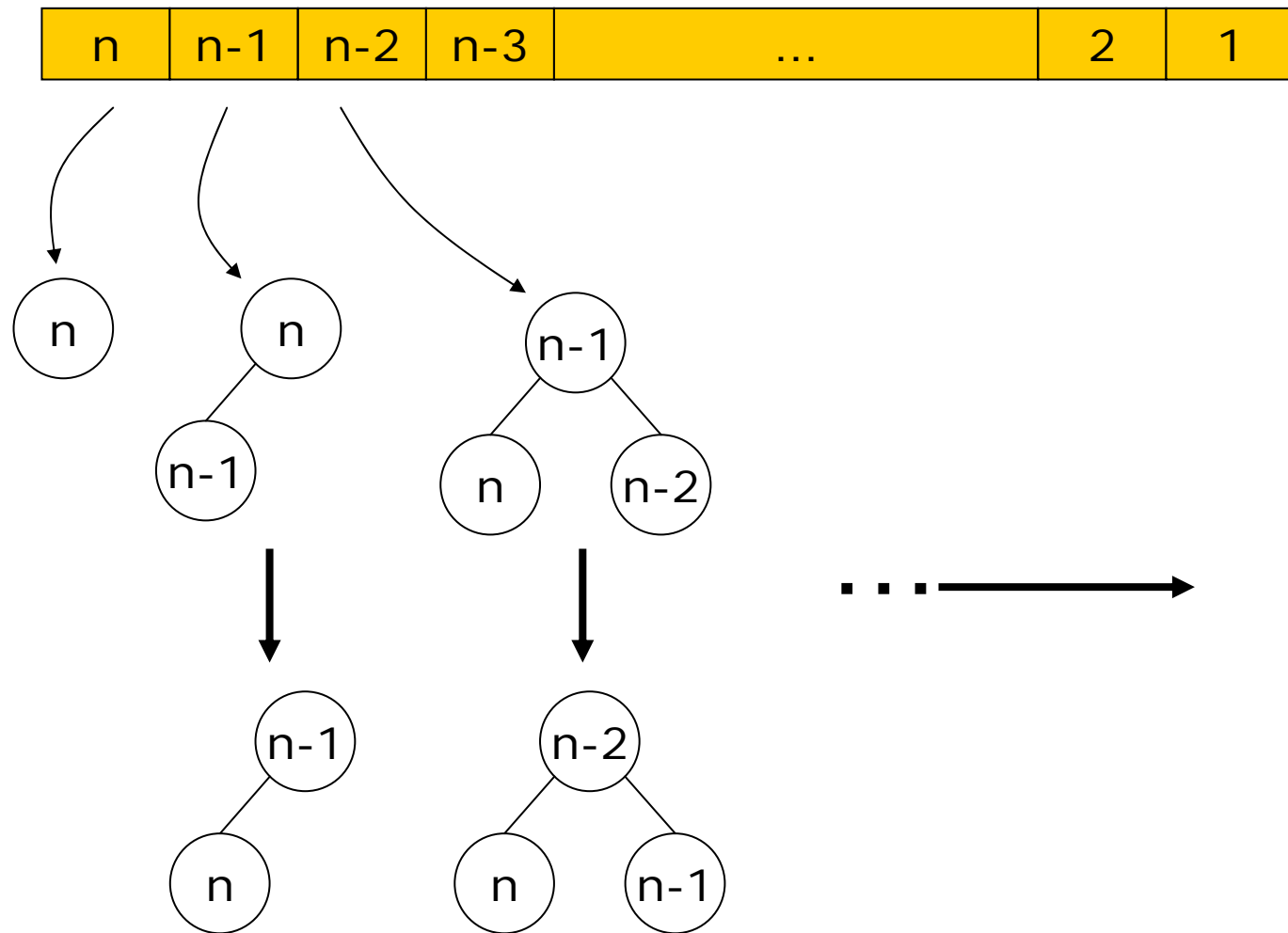
□ בהינתן מערך באורך  $n$ , נרצה ליצור heap ע"י הכנסה סדרתית של ערכי המערך. הראו סדרת הפעולות לוקחת  $\Omega(n \log n)$  במקרה הכי גרוע (worst case)

□ פתרון

■ נצטרך להראות דוגמה של סדרת ההכנסות שלוקחת  $\Omega(n \log n)$  פעולות

■ נחפש סדרה ש"תקשה" כמה שיותר על ה-heap

# תרגיל 1



# תרגיל 1

- כל ערך שנוסיף צריך לבעבע לראש העץ
- $n/2$  ההכנסות האחרונות לוקחות לפחות  $\log(n/2)$  כל אחת

$$\frac{n}{2} \log\left(\frac{n}{2}\right) = \frac{1}{2} n (\log n - \log 2) =$$

$$\frac{1}{2} n \log n - \frac{1}{2} n \log 2 = \theta(n \log n)$$

- מסקנה: W.C =  $\Omega(n \log n)$

## תרגיל 2

---

□ בהינתן heap שתומך בפעולות **extract-min** ו-**insert** בזמן **amortized  $f(n)$** , הראו שניתן למיין מערך מגודל  $n$  בזמן  $O(nf(n))$

□ פתרון

■ נבצע  $n$  פעולות הכנסה בזמן  $O(nf(n))$

■ מבצע  $n$  פעולות הוצאת מינימום בזמן  $O(nf(n))$

■ סה"כ  $O(nf(n))$

□ אלגוריתם מיון זה נקרא **heap-sort**

□ בשיעור תלמדו כי מיון כל איבר הוא  $\Omega(\log n)$

# תרגיל 3 – Median Heap

---

□ ממשו מבנה נתונים התומך בפעולות

■ insert בזמן  $O(\log n)$

■ extract-median בזמן  $O(\log n)$

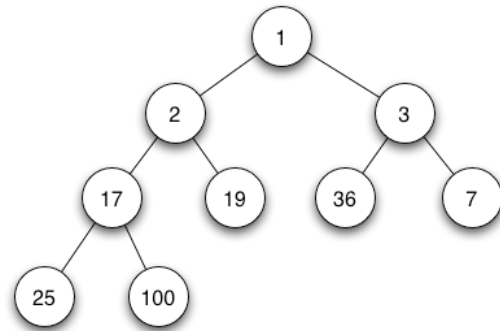
■ find-median בזמן  $O(1)$

2	4	5	7	8	12	14	15	20
---	---	---	---	---	----	----	----	----



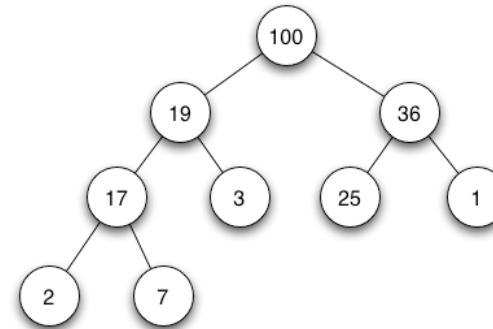
# תרגיל 3 - פתרון

Min-heap



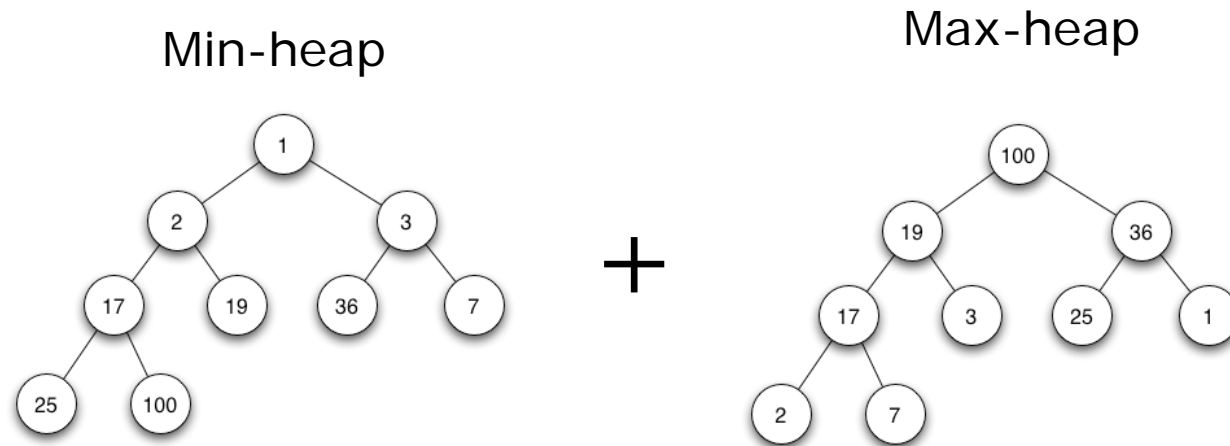
האברים הגדולים  
(מהחציון)

Max-heap

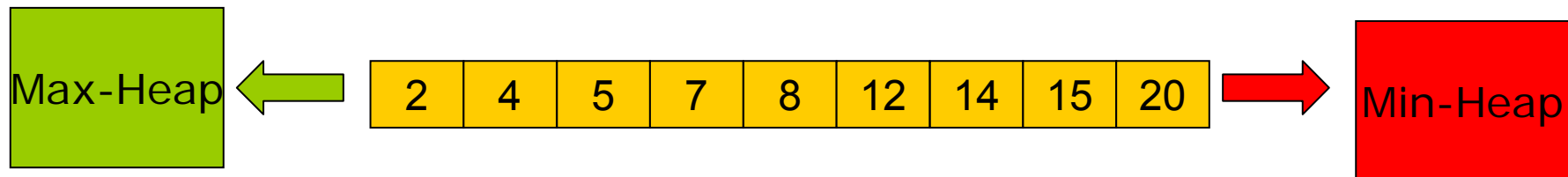


האברים הקטנים  
(עד החציון)

# תרגיל 3 - פתרון



- נשתמש ב-max-heap ו-min-heap
- $n/2$  הערכים הגדולים ביותר יישמרו ב-max-heap
- השאר יישמרו ב-min-heap
- החציון תמיד נמצא בשורש של אחד מהם



# תרגיל 3 - פתרון

## □ Find-median

- If ( $\text{size}(\text{minheap}) > \text{size}(\text{maxheap})$ )
  - return  $\text{getmin}(\text{minheap})$
- Else
  - return  $\text{getmax}(\text{maxheap})$

$O(1)$

## □ Insert(x)

- If ( $x < \text{getmin}(\text{minheap})$ )
  - $\text{Insert}(\text{maxheap}, x)$
- Else
  - $\text{Insert}(\text{minheap}, x)$
- If ( $\text{abs}(\text{size}(\text{minheap}) - \text{size}(\text{maxheap})) > 1$ )
  - Balance heaps (move root from bigger heap to smaller heap)

$O(\log n)$

## □ Extract-Median

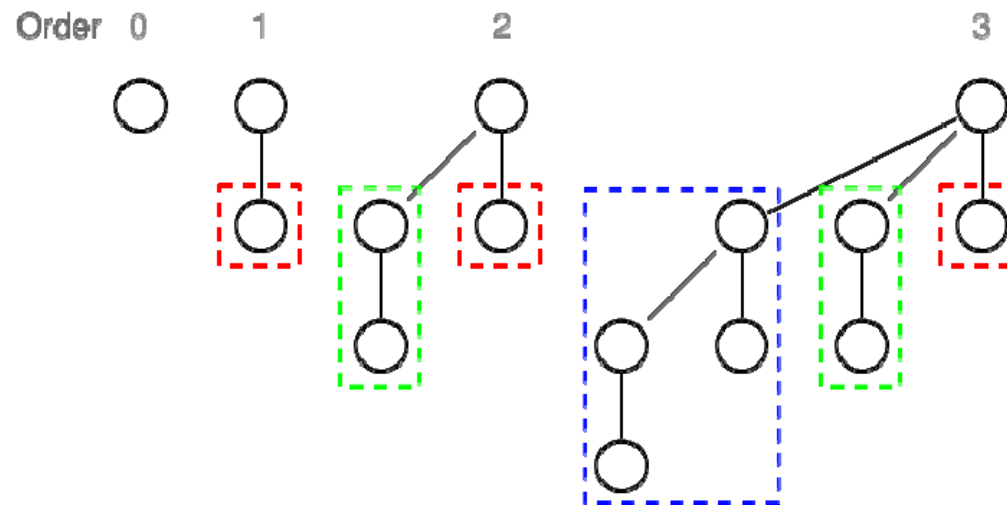
- Extract median from the max-heap or min-heap...

$O(\log n)$

# תזכורת: Binomial Heaps

## □ תחילה נגדיר Binomial Tree:

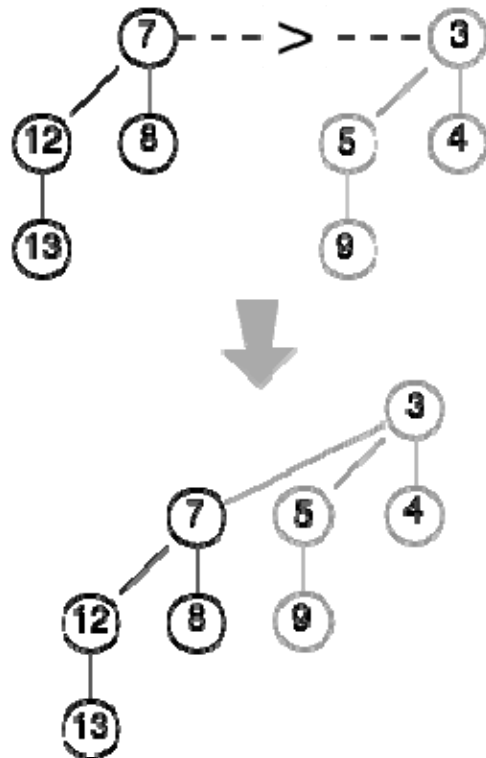
- עץ בינומי מדרגה 0 מכיל צומת יחידה
- עץ בינומי מדרגה  $k$  הוא
  - בעל שורש מדרגה  $k$
  - ילדיו הינם עצים בינומיים מדרגות  $0, 1, \dots, k-1$  (בסדר זה)
  - עץ בינומי מדרגה  $k$  מכיל  $2^k$  צמתים והינו בגובה  $k$



# תזכורת: Binomial Heaps

□ איחוד שני עצים בינומיים מסדר  $k-1$

■ ניצור עץ בינומי מסדר  $k$  ע"י תליית אחד העצים כבן השמאלי ביותר של העץ השני



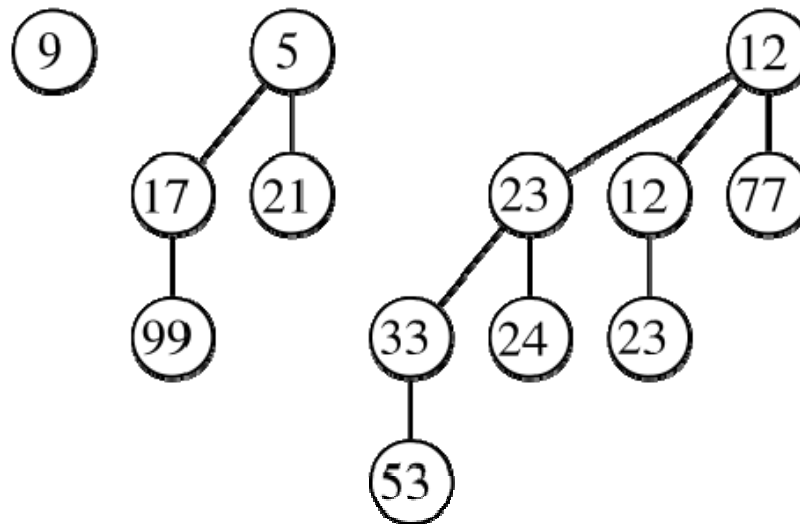
# תזכורת: Binomial Heaps

## □ הגדרת Binomial Heap

■ סט עצים בינומיים המקיימים את התכונות הבאות

□ כל עץ מקיים את תכונת minimum-heap (כל צאצא גדול מהורה שלו)

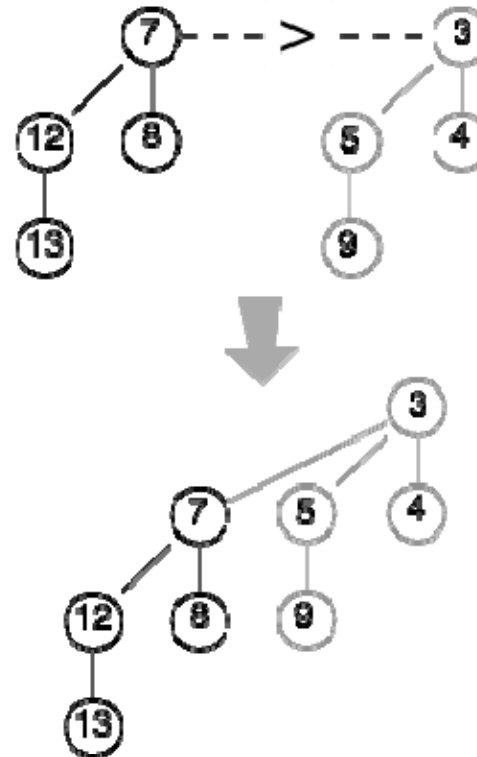
□ עבור כל סדר  $k$  של עץ בינומי, יש 0 או 1 עצים כאלו ב-heap



# תזכורת: Binomial Heaps

□ 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
```



# תזכורת: Binomial Heaps

---

## □ פעולת merge של שני Binomial Heaps

```
function merge(p, q)
  while not ( p.end() and q.end() )
    tree = mergeTree(p.currentTree(), q.currentTree())
    if not heap.currentTree().empty()
      tree = mergeTree(tree, heap.currentTree())
      heap.addTree(tree)
    else
      heap.addTree(tree)
    end if
    heap.next() p.next() q.next()
  end while
end
```



# תזכורת: Binomial Heaps

---

- 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-min
  - בדומה לפעולות ב-heap רגיל
- פעולת Delete
  - שנה את הערך ל- $-\infty$  ובצע delete-min

[הדגמה של Binomial Heap](http://www.cse.yorku.ca/~aaw/Sotirios/BinomialHeap.html)