האוניברסיטה הפתוחה

מבני נתונים ומבוא לאלגוריתמים
מפגש הנחיה מס' 6

מדעי המחשב, קורס מס' 20407

סמסטר 2016ב

מנחה: ג'ון מרברג

מה ראינו בפעם הקודמת?

- מיון-מהיר
- שגרת החלוקה: הגרסה שבספר, הגרסה של Hoare
- ניתוח זמן הריצה
  - *חלוקה $\Theta(n)$*
  - מיון מהיר במקרה הטוב $\Theta(n\log n)$
  - מיון מהיר במקרה הגרוע $\Theta(n^2)$
- מימוש אקראי
  - תוחלת זמן הריצה $\Theta(n\log n)$

מפגש שישי

- נושא השיעור
  - פרק 9 בספר – חציונים וערכי מיקום
    - חציונים וערכי מיקום
    - בעיית הבחירה
    - מציאת מינימום ומקסימום
    - מציאת האיבר ה-i בגודלו – פתרון אקראי
    - מציאת האיבר ה-i בגודלו – פתרון דטרמיניסטי
  - תרגילים בבחירה ומיון

מבוסס על מצגת של ברוך חייקין ואיציק בייז

מיון מהיר עם החלוקה של Lomuto

QuickSort (*A*, *p*, *r*)

if $p < r$
    then $q \leftarrow$ Partition (A, p, r)
        QuickSort $(A, p, q-1)$
        QuickSort $(A, q+1, r)$

Partition (*A*, *p*, *r*)

$x \leftarrow A[r]$
$i \leftarrow p - 1$
for $j \leftarrow p$ **to** $r - 1$
    do if $A[j] \leq x$
        then $i \leftarrow i + 1$
            exchange $A[i] \leftrightarrow A[j]$
exchange $A[i + 1] \leftrightarrow A[r]$
return $i + 1$

ערכי מיקום

- הגדרות
  - ***ערך מיקום*** (order statistic): האיבר ה-i הקטן ביותר בקבוצה S בת n איברים
    - הסימון: $x = S_{(i)}$, או $\mathrm{rank}_S(x) = i$
  - ***מינימום*** (minimum): האיבר הקטן ביותר $S_{(1)}$
  - ***מקסימום*** (maximum): האיבר הגדול ביותר $S_{(n)}$
  - ***חציון*** (median): האיבר "האמצעי" $S_{(\lfloor (n+1)/2 \rfloor)}$
    - כאשר n זוגי, זהו החציון <u>התחתון</u>
- אבחנות
  - ערך המיקום של החציון הוא: $\lfloor (n+1)/2 \rfloor$ או בסימון אלטרנטיבי: $\lceil n/2 \rceil$
  - יש בקבוצה <u>לפחות</u> i איברים הקטנים או שווים ל- $S_{(i)}$
    - איברים אלה הם $S_{(1)}, S_{(2)}, \ldots, S_{(i)}$
    - אם איברי הקבוצה שונים זה מזה, יש <u>בדיוק</u> i איברים כאלה
  - כאשר הקבוצה *ממוינת* בסדר לא יורד $S[1..n]$, האיבר ה-i *הקטן ביותר* נמצא במקום ה-i *בסדר הממוין, כלומר* $S[i] = S_{(i)}$

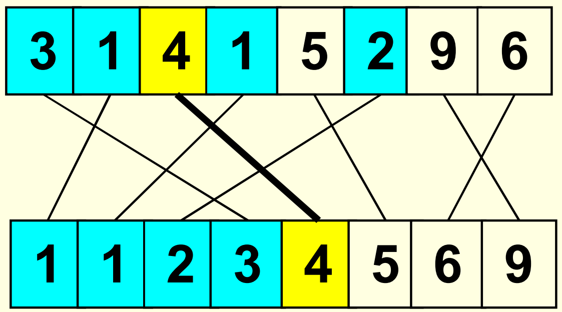
האוניברסיטה הפתוחה

בעיית הבחירה

- בעיית הבחירה (selection):
  - *קלט:* קבוצה A של n מספרים (שונים זה מזה), ומיקום $1 \le i \le n$
  - *פלט:* איבר x ב-A המקיים $x = A_{(i)}$ (או: $\text{rank}_A(x) = i$)
- פתרון נאיבי: מיון
  - האלגוריתם: מיין את A והחזר את $A[i]$
  - זמן ריצה: $O(n\lg n)$ (במיון אופטימלי כגון מיון-מיזוג)
- דוגמה:

מערך לא ממוין *A*

אותו מערך ממוין *A’*



$$\mathbf{4=}A_{(5)} \leftrightarrow \mathbf{rank}_A\mathbf{(4)=5}$$

בעית הבחירה – מקרה פרטי

- בעיית הבחירה כאשר $i = 1$ (מציאת המינימום)
  - באופן דומה: בעיית הבחירה כאשר $i = n$ (מציאת המקסימום)
- אלגוריתם למציאת המינימום

Minimum(*A*)
1. *min* ← *A*[1]
2. **for** *i* ← 2 **to** length[*A*]
3.   **do if** *A*[*i*] < *min*
4.   **then** *min* ← *A*[*i*]
5. **return** *min*

- זמן הריצה נקבע לפי מספר ההשוואות
  - מתבצעות בדיוק $n–1$ השוואות <u>בין האיברים</u> (אופטימלי - מדוע?)

  - כמה פעמים תתבצע שורה 4?
    - במקרה הגרוע (הקלט ממוין הפוך) שורה 4 מתבצעת $n–1$ פעמים
    - ב"מקרה הממוצע" שורה 4 מתבצעת $\Theta(\lg n)$ פעמים

הנימוק: נגדיר מאורע שהאיבר $A[k]$ הוא המינימום של $A[1..k]$

ההסתברות של מאורע זה היא $1/k$

מתקבלת סדרה הרמונית של תוחלות של מאורעות עבור k=1,2, …,n

מציאת מינימום ומקסימום ביחד

- אלגוריתם איטרטיבי נאיבי

SimpleMinMax(A)
1. **return** (Minimum(A), Maximum(A))

- מספר ההשוואות המדויק: $2(n-1) = 2n - 2$

האוניברסיטה הפתוחה

מציאת מינימום ומקסימום ביחד

- אלגוריתם רקורסיבי

RecursiveMinMax(A, p, r)

```
if p < r
   then return (NIL, NIL)
if p = r
   then return (A[p], A[p])
if p = r – 1
   then minval = min(A[p], A[r])
        return (minval, A[p] + A[r] – minval)
q ← ⌊(p+r)/2⌋
(leftmin, leftmax) ← RecursiveMinMax(A, p, q)
(rightmin, rightmax) ← RecursiveMinMax(A, q+1, r)
return (min(leftmin, rightmin), max(leftmax, rightmax))
```

- מספר ההשוואות <u>המדויק</u>: $C(n)=C(\lfloor (n+1)/2 \rfloor) + C(\lceil (n-1)/2 \rceil) + 2$

$$C(1) = 0,\ C(2) = 1$$

כאשר n חזקה של 2: $C(n) = 3n/2 - 2$

אחרת: $C(n) > \lceil 3n/2 \rceil - 2$

האוניברסיטה הפתוחה

מציאת מינימום ומקסימום ביחד (המשך)

- אלגוריתם איטרטיבי
  - הרעיון: לבדוק זוג איברים בכל פעם

```
PairwiseMinMax(A)
n ← length[A]
if n = 1
   then return (A[1], A[1])
if A[1] < A[2]
   then (min, max) = (A[1], A[2])
   else (min, max) = (A[2], A[1])
for i ← 3 to n–1 step 2
     do if A[i] < A[i+1]
           then (min, max) ← (min(min, A[i]), max(max, A[i+1]))
           else (min, max) ← (min(min, A[i+1]), max(max, A[i]))
if n mod 2 = 1
   then (min, max) ← (min(min, A[n]), max(max, A[n]))
return (min, max)
```

מציאת מינימום ומקסימום ביחד (המשך)

- חישוב מספר ההשוואות המדויק באלגוריתם PairwiseMinMax
  - עבור הזוג הראשון, מתבצעת השוואה אחת בלבד
  - עבור כל אחד משאר $\lfloor(n-2)/2\rfloor$ הזוגות מתבצעות 3 השוואות
  - אם n אינו זוגי, מתבצעות עוד 2 השוואות עבור הערך האחרון

זוגי n: $C(n) = 1 + 3(n-2)/2 = 3n/2 - 2$

אי-זוגי n: $C(n) = 1 + 3(n-3)/2 + 2 = 3n/2 - 1.5$

$\Downarrow$

לכל n: $C(n) = \lceil 3n/2 \rceil - 2$

נשים לב כי עבור n שאינו חזקה של 2, האלגוריתם PairwiseMinMax יעיל יותר מאשר RecursiveMinMax

תרגיל 9.1-2: חסם תחתון על מספר ההשוואות

האוניברסיטה הפתוחה

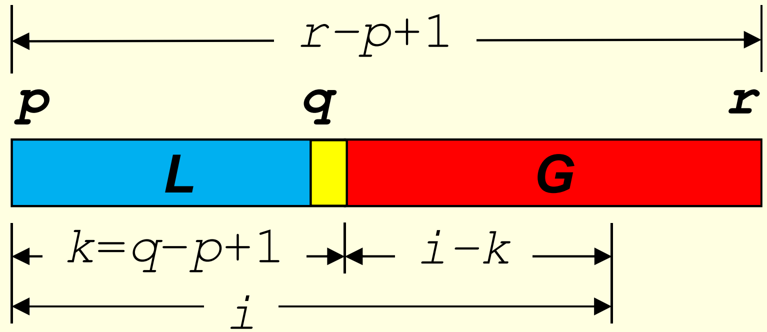
הראו שכל אלגוריתם למציאת המינימום והמקסימום ביחד יבצע במקרה הגרוע לפחות $\lceil 3n/2 \rceil - 2$ השוואות, כלומר האלגוריתם PairwiseMinMax הוא אופטימלי

- נגדיר את הקבוצות הבאות, הקיימות בכל נקודה בכל אלגוריתם:
  A = האיברים שטרם הושוו, B = האיברים שתמיד ניצחו (היו הגדולים בכל השוואה בה השתתפו), C = האיברים שתמיד הפסידו, D = האיברים שגם ניצחו וגם הפסידו
- נגדיר את הסכום $x = |B|+|C|+3|D|$
  - בהתחלת האלגוריתם: $x = 0$
  - בסיום האלגוריתם: A=0, B=1, C=1, $D=n-2$, ולכן $x = 1 + 1 + 3(n-2) = 3n-4$
- הסכום x גדל <u>לכל היותר</u> ב-2 בכל השוואה במקרה <u>הגרוע</u> (הגידול האיטי ביותר)
  - למשל, אם משווים שני איברים מ- A, אחד עובר ל-B והאחר ל-C, ולכן x גדל ב-2.
  - למשל, אם איבר מ-A מושווה עם איבר מ-B
    - במקרה הגרוע: האיבר מ-B מנצח ונשאר ב-B, והאיבר מ-A עובר ל-C, ולכן x גדל ב-1
    - במקרה הטוב: האיבר מ-B מפסיד ועובר ל-D ואילו האיבר מ-A עובר ל-B, ולכן x גדל ב-3
- המסקנה: האלגוריתם מבצע לפחות $(3n-4)/2$ השוואות.
  - מכיוון שמספר ההשוואות תמיד שלם, יש לפחות $\lceil 3n/2 \rceil - 2$ השוואות

מציאת האיבר ה-*i* בגודלו
גרסה אקראית

- באופן כללי, אנו מחפשים את האיבר ה-*i* בגודלו ב- $A[p..r]$, כאשר $1 \le i \le r-p+1$
- הפתרון על-ידי אלגוריתם אקראי רקורסיבי (עמ' 154 בספר), המשתמש בשגרת החלוקה האקראית המבוססת על שיטת Lomuto

```
RandomizedSelect (A, p, r, i)
  if p = r
      then return A[p]
  q ← RandomizedPartition (A, p, r)
  k ← q–p+1
 if i = k
     then return A[q]
     elseif i < k
         then return RandomizedSelect (A, p, q–1, i)
         else return RandomizedSelect (A, q+1, r, i–k)
```

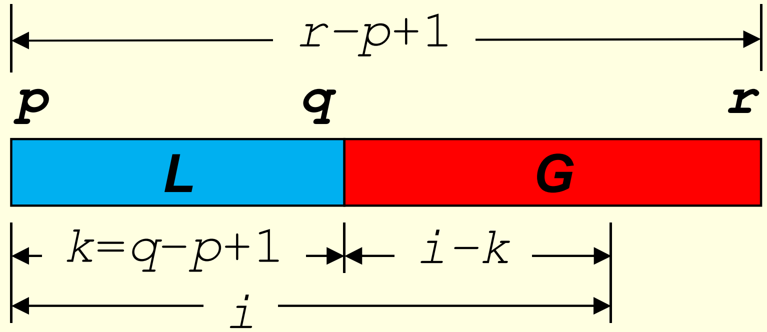


- הערות
  - אם $i = k$ אז סיימנו;
    אחרת, אם $i < k$ אז ממשיכים לחפש את האיבר ה-*i* בגודלו באזור *L*;
    אחרת, ממשיכים לחפש את האיבר ה-$(i-k)$ בגודלו באזור *G*
  - יעילות החיפוש תלויה ביחס בין *L* ו-*G*, לפי החלוקה האקראית

מציאת האיבר ה-i בגודלו

- פתרון אקראי דומה משתמש בשיטת החלוקה של Hoare:

```
RandomizedSelect1(A, p, r, i)
if p = r
   then return A[p]
q ← RandHoarePartition(A, p, r)
k ← q–p+1
if i ≤ k
   then return RandomizedSelect1(A, p, q, i)
   else return RandomizedSelect1(A, q+1, r, i–k)
```



- הערות
  - אם $i \le k$ אז ממשיכים לחפש את האיבר ה-i בגודלו באזור L;
    אחרת, ממשיכים לחפש את האיבר ה-$(i–k)$ בגודלו באזור G
  - יעילות החיפוש תלויה ביחס בין L ו-G, לפי החלוקה האקראית

האוניברסיטה הפתוחה

פתרון אקראי – דוגמה

מחפשים מינימום ($i = 1$) במערך

3	2	9	0	7	5	4	8	6	1

המקרה הטוב:

חלוקה לשני חלקים שווים (בכל שלב איבר החלוקה הוא החציון) – מתבצעות בסך הכל $O(n)$ השוואות

3	2	9	0	7	5	4	8	6	1

3	2	0	1	4	5	9	8	6	7

0	1	3	2	4	5	9	8	6	7

המקרה הגרוע:

בכל שלב נבחר כאיבר ציר האיבר הגדול ביותר – מתבצעות $\Theta(n^2)$ השוואות

3	2	9	0	7	5	4	8	6	1

3	2	1	0	7	5	4	8	6	9

3	2	1	0	7	5	4	6	8	9

3	2	1	0	6	5	4	7	8	9

• • •

0	1	2	3	4	5	6	7	8	9

פתרון אקראי – זמן ריצה

- זמן הריצה (מספר ההשוואות) ***במקרה הגרוע***
  - מחפשים את *המינימום*
  - בכל שלב שגרת החלוקה האקראית בוחרת את האיבר *הגדול ביותר* כאיבר ציר
  - לפיכך החלוקה האקראית מבצעת בכל שלב $r–p$ השוואות, ומחלקת את התת-מערך $A[p..r]$ לשני תת-מערכים: $G = A[r..r]$ וְ- $L = A[p..r–1]$
  - הריצה ממשיכה בתת-מערך השמאלי L
  - סה"כ מתבצעות $\Sigma_{k=1}^{n-1}(n–k) = \Theta(n^2)$ השוואות
- תוחלת זמן הריצה
  - התוחלת היא לינארית $E(T(n))=O(n)$
  - הוכחה פורמאלית נמצאת בספר (עמ' 155)
- מסקנה: *ניתן למצוא כל ערך מיקום בתוחלת זמן לינארית*

פתרון אקראי – זמן ריצה

- *תוחלת* זמן הריצה נקבעת על-ידי מספר ההשוואות
  - לכל $1 \le k \le n$ נגדיר את המאורע A_k:
    A_k = {בתת-מערך $A[p..q]$ יש בדיוק k איברים}
  - עבור כל k מניחים שהאיבר המבוקש יימצא באזור הגדול יותר (כי רוצים לקבל חסם עליון על התוחלת)
  - עושים ממוצע משוקלל:

$$\mathrm{E}[T(n)] \le \Sigma_{k=1}^{n} \Pr\{A_k\}\cdot[T(\max(k-1, n-k)) + O(n)]$$

  - אבל $\Pr\{A_k\}=1/n$ לכל $1 \le k \le n$ (כי איבר הציר נבחר בצורה אקראית), ולכן, בהנחה שכל האיברים שונים זה מזה:

$$\mathrm{E}[T(n)] \le \Sigma_{k=1}^{n} (1/n)\cdot[T(\max(k-1, n-k)) + O(n)]$$

  - מקבלים פתרון בשיטת ההצבה (עמ' 156 בספר):

$$\mathrm{E}[T(n)] = O(n)$$

תרגיל

- פתרו את בעיית הבחירה על-ידי אלגוריתם אקראי איטרטיבי. (תרגיל 9.2-3)

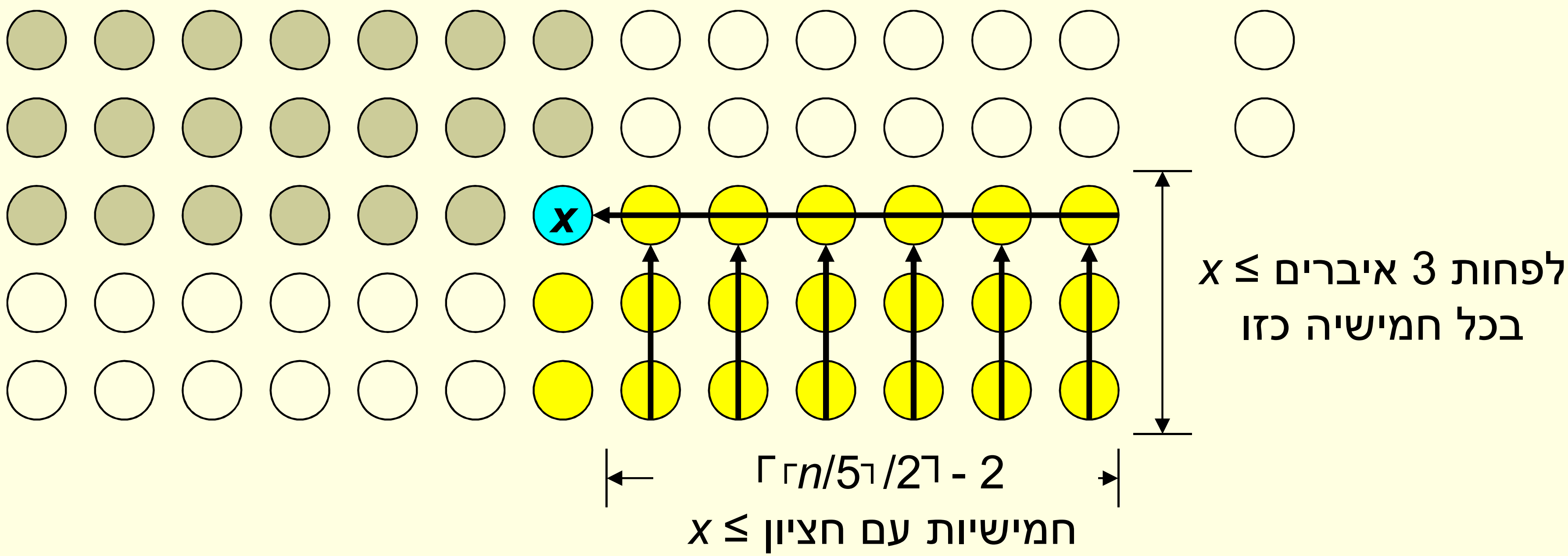
- נשתמש באלגוריתם הבא

```
IterativeRandomizedSelect(A, p, r, i)
while p < r
      do q ← RandomizedPartition(A, p, r)
         k ← q – p + 1
         if i = k
            then return A[q]
         elseif i < k
            then r ← q – 1
            else  p ← q + 1
                  i ← i – k
return A[p]
```

האוניברסיטה הפתוחה

בחירה - פתרון דטרמיניסטי
הרעיון

- נרצה להבטיח שהחלוקה תיתן תמיד יחס "טוב" בין הצדדים
  - כלומר יחס קבוע שלא תלוי בגודל הקבוצה
  - למשל 30% - 70%
- השיטה:
  - נחלק את האיברים לקבוצות בגודל 5 (יכולה להיות קבוצה אחת חלקית)
  - נמצא חציון של כל חמישיה, ואח"כ חציון של כל החציונים (מסומן x)



בחירה – פתרון דטרמיניסטי

- פתרון על-ידי אלגוריתם דטרמיניסטי Select
  - הרעיון: להבטיח חלוקה "טובה" של המערך בכל מקרה

1. אם המערך A בגודל 1, עצור והחזר את האיבר היחיד
2. אחרת:

a) חלק את *A* לקבוצות בגודל 5 (כולל קבוצה חלקית של האיברים שנשארו בסוף)
b) מצא את החציון בכל קבוצה ואחסן את החציונים במערך עזר *B*
c) קרא רקורסיבית ל-Select על מערך העזר *B* כדי למצוא את החציון של החציונים; זה יהיה איבר הציר *x* במערך *A*
d) חלק את *A* סביב איבר הציר *x* באמצעות השגרה PartitionWithPivot
e) קרא רקורסיבית ל-Select על תת-המערך המתאים, כרגיל

האוניברסיטה הפתוחה

האלגוריתם לפתרון דטרמיניסטי

Select (A, p, r, i)

1. $n \leftarrow r - p + 1$
2. **if** $n = 1$
3.   **then return** $A[p]$
4. $m \leftarrow \lceil n/5 \rceil$
5. **for** $j \leftarrow 1$ to m
6.   **do** $B[j] \leftarrow \text{Median5}\ (A, p+5(j-1), \min(p+5j-1, r))$
7. $x \leftarrow \text{Select}\ (B, 1, m, \lceil m/2 \rceil)$
8. $q \leftarrow \text{PartitionWithPivot}\ (A, p, r, x)$
9. $k \leftarrow q - p + 1$
10. **if** $i \leq k$
11.   **then return** $\text{Select}(A, p, q, i)$
12.   **else return** $\text{Select}(A, q+1, r, i-k)$

- האלגוריתם PartitionWithPivot מקבל את איבר הציר כפרמטר x ומבצע חלוקה של המערך סביבו בשיטת Hoare

האלגוריתם לפתרון דטרמיניסטי (המשך)

- אלגוריתם עזר למציאת חציון של תת מערך בגודל עד 5 איברים

Median5(A, p, r)
Input: Sub-array $A[p..r]$, where $1 \le r-p+1 \le 5$
Output: The median of $A[p..r]$
1. InsertionSort(A, p, r)
2. **return** $A[\lceil (r+p-1)/2 \rceil]$

- האלגוריתם Median5 רץ בזמן $O(1)$

- האינדקס בשורה 2 הוא של איבר החציון בתת המערך הממוין $A[p..r]$

$$(p-1) + \lceil (r-p+1)/2 \rceil = \lceil (r+p-1)/2 \rceil$$

האוניברסיטה הפתוחה

האלגוריתם לפתרון דטרמיניסטי (המשך)

- אלגוריתם לחלוקה עם איבר ציר נתון x
  - זהה לאלגוריתם HoarePartition, למעט בחירת איבר הציר

PartitionWithPivot(A, p, r, x)

$i \leftarrow p - 1$

$j \leftarrow r + 1$

while true **do**

    repeat $j \leftarrow j - 1$ **until** $A[j] \leq x$

    repeat $i \leftarrow i + 1$ **until** $A[i] \geq x$

    if $i < j$

        then exchange $A[i] \leftrightarrow A[j]$

        else return j

זמן הריצה: $O(n)$

האוניברסיטה הפתוחה

פיתרון דטרמיניסטי – דוגמה

מחפשים חציון במערך A[1..11] הבא (n=11)
הקריאה ההתחלתית: Select(A, 1, 11, 6)

3	2	9	0	7	5	10	4	8	6	1

מפצלים את A[1..11] ל-3= $m=\lceil n/5 \rceil$ חמישיות

3 2 9 0 7 | 5 10 4 8 6 | 1

מוצאים את 3 חציוני החמישיות ע"י Median5

0 2 **3** 7 9 | 4 5 **6** 8 10 | **1**

ממלאים את B[1..3] עם חציוני החמישיות

3 6 1

מפעילים את Select רקורסיבית על B ומוצאים את החציון x=3

1 **3** 6

מסדרים את A[1..11] באמצעות PartitionWithPivot עם איבר הציר 3
וממשיכים עם האזור הימני וערך מיקום 6-3=3

1	2	0	9	7	5	10	4	8	6	3

האוניברסיטה הפתוחה

פיתרון דטרמיניסטי – דוגמה (המשך)

ממשיכים בתת-מערך $A[4..11]$ (הפעם n=8)
הקריאה הרקורסיבית: Select(A, 4, 11, 3)

1	2	0	9	7	5	10	4	8	6	3

מפצלים את A[4..11] ל- $m=\lceil n/5 \rceil=2$ חמישיות

9	7	5	10	4

8	6	3

מוצאים את 2 חציוני החמישיות ע"י Median5

4	5	7	9	10

3	6	8

ממלאים את $B[1..2]$ עם חציוני החמישיות

7	6

מפעילים את Select רקורסיבית על B ומוצאים את החציון x=6

6	7

מסדרים את $A[4..11]$ באמצעות
PartitionWithPivot עם איבר הציר 6
וממשיכים עם האזור השמאלי ואותו ערך מיקום 3

1	2	0	3	4	5	10	7	8	9	6

האוניברסיטה הפתוחה

פיתרון דטרמיניסטי – דוגמה (המשך)

ממשיכים בתת-מערך A[4..6] (הפעם n=3)
עם הקריאה הרקורסיבית: Select(A, 4, 6, 3)

1	2	0	**3**	**4**	**5**	10	7	8	9	6

מפצלים את A[4..6] ל-1= $m=\lceil n/5 \rceil$ חמישיות

3	**4**	**5**

מוצאים את חציון החמישיה ע"י Median5

3	**4**	**5**

ממלאים את B[1] עם חציוני החמישיות

4

מפעילים את Select רקורסיבית על B ומוצאים את החציון x=4

4

מסדרים את A[4..6] באמצעות
PartitionWithPivot עם איבר הציר 4
וממשיכים עם האזור הימני עם ערך מיקום 3-2=1

1	2	0	**3**	**4**	**5**	10	7	8	9	6

האוניברסיטה הפתוחה

פיתרון דטרמיניסטי – דוגמה (סוף)

ממשיכים בתת-מערך *A*[6..6] (הפעם *n*=1)
עם הקריאה הרקורסיבית: Select(*A*, 6, 6, 1)

1	2	0	3	4	**5**	10	7	8	9	6

הגענו לתנאי העצירה – מערך בן איבר אחד
לכן עוצרים ומחזירים את האיבר היחיד בו: 5
שהוא החציון המבוקש

5

סיכום:

במהלך הריצה Select מבצע *מיון חלקי* של המערך, כך שלבסוף:

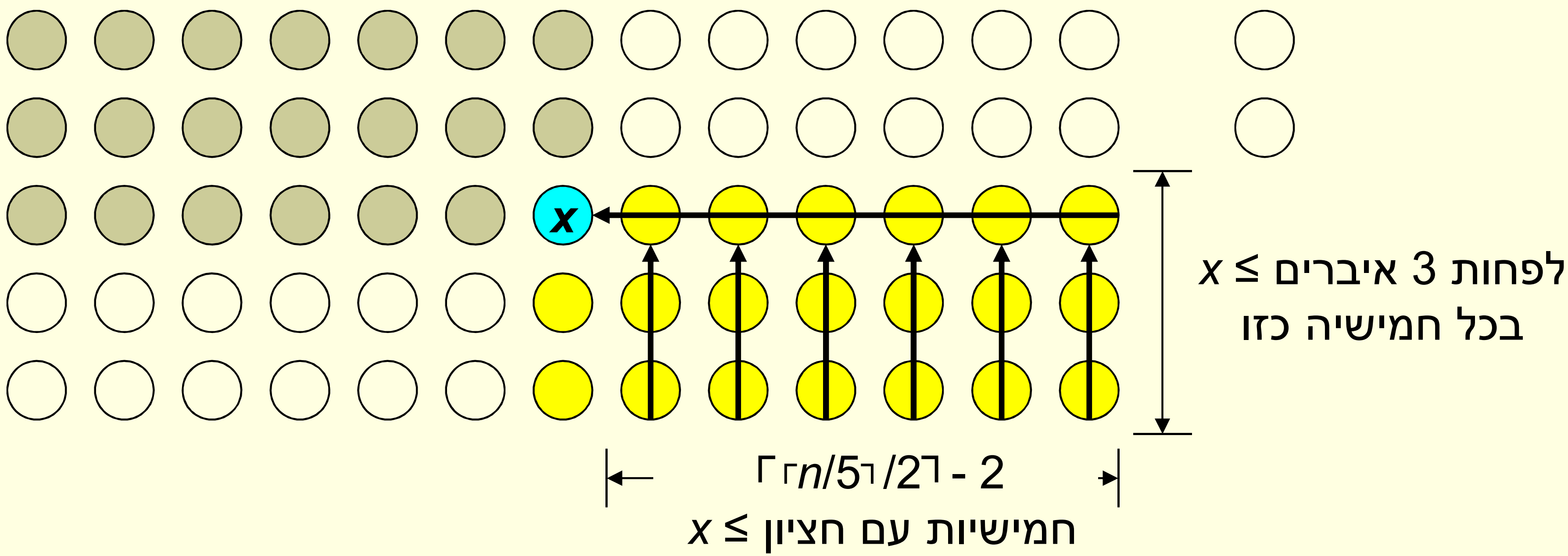
1. האיבר ה-*i* בגודלו נמצא ב-*A*[*i*] – זהו האיבר המבוקש
2. האיברים ה-1,2,...,i–1 בגודלם נמצאים בתת-מערך השמאלי A[1..i–1], לאו דוקא לפי סדר
3. האיברים ה-i+1,i+2,...,n בגודלם נמצאים בתת-מערך הימני A[i+1..n], לאו דוקא לפי סדר

לא מתבצע מיון מלא – זמן הריצה הוא $O(n)$ ולא $O(n\lg n)$

האוניברסיטה הפתוחה

פתרון דטרמיניסטי – זמן ריצה

- זמן הריצה (מספר ההשוואות) ***במקרה הגרוע***
  - בחמישיות שהחציון שלהן גדול מהציר x יש לפחות 3 איברים גדולים מ-x
  - x הוא החציון של חציוני החמישיות, לכן מדובר בחצי מהחמישיות, לא כולל הקבוצה האחרונה (אם ישנה) והחמישיה הכוללת את x עצמו
  - מכאן שיש ב-A לפחות $3(\lceil\lceil n/5\rceil/2\rceil - 2) \geq 3n/10 - 6$ איברים *גדולים* מ-x; באופן דומה, יש ב-A לפחות $3n/10 - 6$ איברים *קטנים* מ-x



פיתרון דטרמיניסטי – זמן ריצה

- כלומר, החלוקה *הכי גרועה* של *A* שיכולה לקרות היא:
  - קבוצה קטנה בת $3n/10 - 6$ איברים
  - קבוצה גדולה בת $7n/10 + 6$ איברים
- במקרה הגרוע *i* שייך לקבוצה *הגדולה* יותר
- לכן נוסחת הנסיגה היא

$$T(n) \le T(\lceil n/5 \rceil) + T(7n/10+6) + O(n)$$

- פתרון (מתקבל בשיטת ההצבה – ראה עמ' 159 בספר):

$$T(n) = O(n)$$

תרגיל 9.3-8

- נתונים שני מערכים *ממוינים* A ו- B, *כל אחד* בגודל n.
- מצאו אלגוריתם המחזיר את החציון המשותף של כל $2n$ האיברים בזמן לוגריתמי.
- נתאר את האלגוריתם בצורה לא פורמלית:
  1. אם $n=1$, עצור והחזר את המינימום של שני האיברים
  2. השווה את החציון a של המערך A עם החציון b של המערך B
  3. אם $a=b$, עצור – זהו החציון המבוקש
  4. אם $a<b$, בצע קריאה רקורסיבית עם החצי הימני של A והחצי השמאלי של B, כולל a ו- b
  5. אחרת ($a>b$), בצע קריאה רקורסיבית עם החצי השמאלי של A והחצי הימני של B, כולל a ו- b
- נוסחת הנסיגה:

$$T(2n) = T(n) + \Theta(1) = \Theta(\lg n)$$