האוניברסיטה הפתוחה

מבני נתונים ומבוא לאלגוריתמים
מפגש הנחיה מס' 1

מדעי המחשב, קורס מס' 20407

סמסטר 2016ב

מנחה: ג'ון מרברג

האוניברסיטה הפתוחה

הקדמה

- מנחה הקבוצה: ד"ר ג'ון מרברג
  - ייעוץ טלפוני: יום ג' 8:00-9:00 054-4439990
  - דוא"ל: john.marberg@openu.ac.il
- נושאי המפגשים לפי לוח התכנון של הקבוצה
- הגשת ממ"נים
  - רגילים: ממ"נים 13-11, 17-15 (יש להגיש לפחות שלשה)
  - תכנותיים: ממ"ן 18 חובה, 14 רשות (אפשר להגיש בזוגות)
  - תאריכי ההגשה לפי לוח התכנון של הקבוצה
    - בקשות לדחיה יש להגיש למנחה מראש, לפני מועד ההגשה שנקבע
    - מותר לחשוב בחברותא, הכתיבה תהיה עצמאית
      - ניסוחים זהים יסומנו כמועתקים, ולא חשוב מהו המקור
- פורום קבוצתי באתר הקורס
  - מכיל הודעות, מצגות, חומר נלווה
  - מיועד רק להתייחסות ספציפית של מנחה הקבוצה
  - כל הפרסומים בפורום מחייבים – יש לבקר באופן שוטף

האוניברסיטה הפתוחה

מפגש ראשון

- נושאי השיעור
  - פרק 1 בספר – מבוא כללי לאלגוריתמים
  - פרק 2 בספר – תכנון אלגוריתמים, ניתוח אלגוריתמים, הוכחת נכונות של אלגוריתמים
- תוכן העניינים
  - אלגוריתמים – מושגים בסיסיים
  - בעיית המיון
  - בעיית החיפוש

מבוסס על מצגת של ברוך חייקין ואיציק בייז

האוניברסיטה הפתוחה

נושאי הקורס

- **מבוא לאלגוריתמים** – בעיות, סוגי אלגוריתמים, הוכחת נכונות, ניתוח סיבוכיות, הוכחת חסמים, ועוד

- **מבני נתונים** – ארגון של הנתונים לתמיכה באלגוריתמים באופן המוגדר ע"י הפעולות הנדרשות. נדבר על ערמות בינאריות, עצים בינאריים, טבלאות HASH (גיבוב), ועוד.

האוניברסיטה הפתוחה

אלגוריתמים

- **בעיה -** קלט נתון, ופלט מבוקש
- **אלגוריתם** – שיטת חישוב מוגדרת היטב לפתרון בעיה
- **הפעולות הבסיסיות** – פעולות חשבון, השוואה, השמה
- **המודל החישובי** – מודל ה-RAM עם מעבד יחיד
- **תיאור** – הקלט, הפלט, כתיבה בפסידו-קוד
- **נכונות** – עצירה על כל קלט עם הפלט הדרוש
- **סיבוכיות זמן ומקום** – השימוש במשאבי המחשב (זכרון, זמן ביצוע) כתלות בגודל הקלט
- **תכונות "טובות" נוספות של אלגוריתם** – פשוט, קל לניתוח, קל למימוש, דורש מעט זכרון (מעבר לגודל הקלט)

האוניברסיטה הפתוחה

סיבוכיות

- משאבי המחשב
  - **זמן מעבד** – מספר הפעולות הבסיסיות – השוואה, השמה פעולות אלגבריות פשוטות, ...
  - **מקום בזיכרון** – כמות "יחידות הזיכרון" בנוסף לקלט – משתנים, מצביעים, מחסנית הריצה, מבני עזר, ...
- גודל הבעיה
  - גודל **הקלט** – כמות הזיכרון הדרושה לאחסון הקלט (מספר איברים בקבוצה, מספר סיביות במספר, מספר צמתים וקשתות בגרף...)

האוניברסיטה הפתוחה

ניתוח סיבוכיות

- זמן הריצה – פונקציה אי-שלילית של גודל הקלט T: N → N.
  - לדוגמא: $T(n) = 3n^2 + 4n + 10$
- נשתמש בסימונים מיוחדים שיוגדרו בהמשך
  - לדוגמא: $T(n) = \Theta(n^2)$ $T(n)=\Omega(\log_2 n)$ $T(n)=O(n\log_2 n)$
  - בינתיים נאמר כי הסימונים מבטאים סדרי גודל של זמן הריצה
    לדוגמא: סדר גודל של n^2, או $\log_2 n$, או $n\log_2 n$
  - נתעלם מסדרי גודל נמוכים וממכפלות בקבוע.
    לדוגמא: $10 + 4n + n^2$ יקרא סד"ג של n^2 (מדוע?)
  - הסימון מאפשר לנתח אלגוריתמים ביתר פשטות ולהשוות בין ביצועיהם
- מקרים לניתוח
  - **המקרה הטוב (best case)**: הקלט שעבורו $T(n)$ מינימלי
  - **המקרה הגרוע (worst case)**: הקלט שעבורו $T(n)$ מקסימלי – הקלט של "יריב מרושע" (adversary) שמכיר את האלגוריתם
  - **המקרה הממוצע (average case)**: התוחלת (ממוצע משוקלל) של $T(n)$ על כל הקלטים האפשריים בגודל n

האוניברסיטה הפתוחה

בעיית המיון

- **קלט** – אוסף של ערכים + יחס סדר מלא ביניהם
- **פלט** – תמורה מסודרת של הערכים

- דוגמאות
  - מערך של מספרים ממשיים עם יחס סדר ≤
  - יישובים במפה לפי מרחק בקו אווירי מהבית

- תכונות טובות: מהירות, יציבות, מיון "במקום" (in place)

- **שיטות מיון** (על מערך ממשיים עם היחס ≤)
  - **מבוסס השוואות** (ללא הנחות על ערכי הקלט)
    - מותר לבצע רק השוואות בין איברים
    - דוגמאות: מיון-בועות, מיון-הכנסה, מיון-בחירה, מיון-מיזוג, מיון-ערמה, מיון-מהיר, ...
  - **לא מבוסס השוואות** (עם הנחות על ערכי הקלט):
    - מותר לבצע פעולות מתמטיות מגוונות על ערכי הקלט
    - דוגמאות: מיון-מנייה (counting-sort), מיון-בסיס (radix-sort), מיון-דלי (bucket-sort)

האוניברסיטה הפתוחה

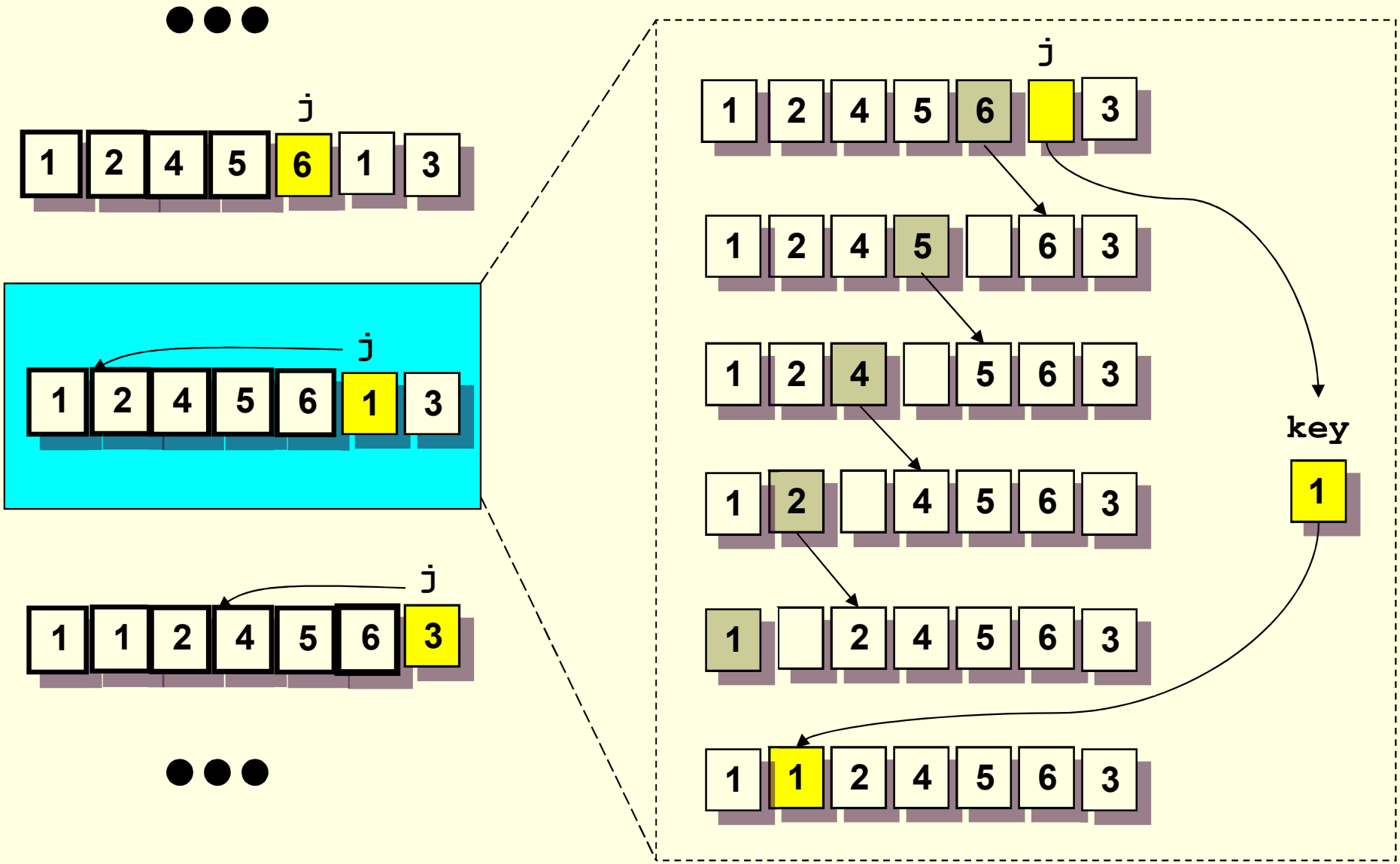
מיון אינקרמנטלי

- הרעיון:
  - בונים תת סדרה ממוינת הולכת וגדלה. בכל שלב מוסיפים עוד איבר לתת הסדרה ע"י ביצוע *n* צעדים לכל היותר
  - תת סדרה התחלתית בגודל 1 היא ממוינת
  - בסוף השלב ה-*j* מתקבלת תת-סדרה ממוינת בגודל *j*

- דוגמאות:
  - **מיון-בועות**: העבר את האיבר ה-*j* בגודלו למקומו הנכון ע"י ביצוע החלפות בין איברים סמוכים (בעבוע).
  - **מיון-בחירה**: מצא את האיבר ה-*j* בגודלו והחלף אותו עם האיבר הנמצא במקום ה-*j*.
  - **מיון-הכנסה**: הכנס את האיבר שבמקום ה-*j* בקלט למקומו הנכון בתת הסדרה הממוינת 1..j-1 ע"י הזזת איברים בתת הסדרה

האוניברסיטה הפתוחה

מיון-הכנסה – דוגמה



האוניברסיטה הפתוחה

מיון הכנסה – האלגוריתם

```
INSERTION-SORT(A)
1  for j ← 2 to length[A]
2       do key ← A[j]
3          ▷ Insert A[j] into the sorted sequence A[1 . . j − 1].
4          i ← j − 1
5          while i > 0 and A[i] > key
6               do A[i + 1] ← A[i]
7                  i ← i − 1
8          A[i + 1] ← key
```

האוניברסיטה הפתוחה

מיון הכנסה

- **נכונות:**

א. <u>תנאי עצירה</u>: בלולאה הפנימית (5-7), i מוקטן בכל איטרציה ולכן בלולאה הראשית (1-8) כל איטרציה היא סופית, ויש n איטרציות

ב. <u>שמורת הלולאה</u>: לפני כל איטרציה של הלולאה הראשית, התת-מערך $A[1..j-1]$ מורכב מהאיברים שהיו בו בהתחלה בסדר ממוין.

- <u>אתחול</u>: הטענה נכונה עבור j = 2 (התת-מערך A[1] ממוין)
- <u>תחזוקה</u>: אם הטענה נכונה לפני איטרציה כלשהי, אז היא תהיה נכונה גם לפני האיטרציה הבאה. (מדוע?)
- <u>סיום</u>: כאשר $j = n + 1$ המערך A ממוין כולו, כנדרש.

- **סיבוכיות:**
  - <u>זמן ריצה</u>: $\Theta(n^2)$ במקרה הגרוע (הקלט ממוין הפוך)
    כי האלגוריתם יבצע $j - 1$ הזזות בכל איטרציה של הלולאה הראשית
  - <u>מקום</u>: $\Theta(1)$ - מיון במקום (in place)

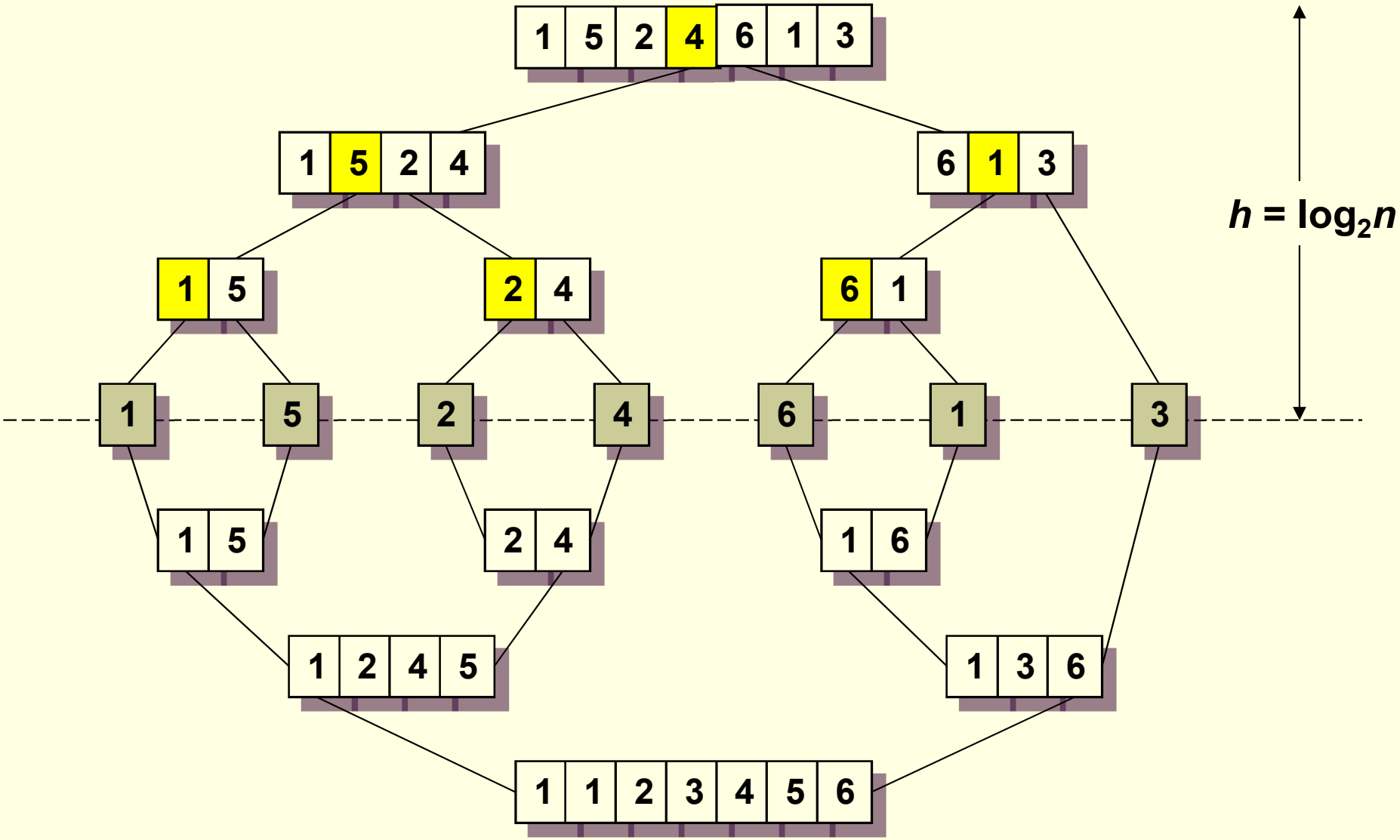
האוניברסיטה הפתוחה

מיון בשיטת הפרד ומשול

- הרעיון:
  - *"הפרד"*: *חלק* את הסדרה הלא-ממוינת לשתי תת-סדרות שוות בגודלן
  - *"משול"*: *מיין* כל תת-סדרה באופן רקורסיבי
  - *"צרף"*: *חבר* את התת-סדרות הממוינות לסדרה ממוינת אחת
- דוגמאות:
  - **מיון-מיזוג**: חלק את הסדרה לשתי תת-סדרות, *מיין* את שני החלקים באמצעות קריאות רקורסיביות, *ומזג* את התוצאות (באמצעות מערך-עזר)
  - **מיון-מהיר**: בחר איבר "ציר", *חלק* את הסדרה לאיברים הקטנים מאיבר הציר ולאיברים הגדולים ממנו, *ומיין* את שני החלקים באמצעות קריאות רקורסיביות. (אין צורך במיזוג!)
- יתרונות וחסרונות של רקורסיה לעומת פיתרון איטרטיבי:
  - בדרך כלל הרקורסיה פתרון יותר טבעי לבעיה ויותר פשוט לתאור
  - לקריאות הרקורסיביות עלות נוספת בזיכרון (מחסנית הקריאות) ובזמן הריצה

האוניברסיטה הפתוחה

מיון-מיזוג - דוגמה

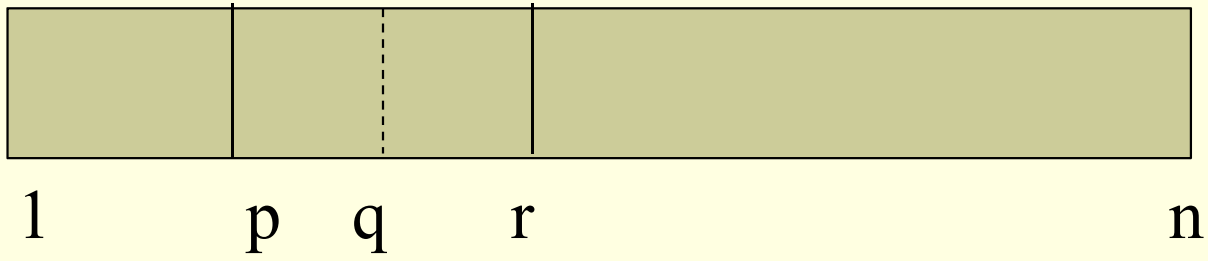


האוניברסיטה הפתוחה

מיון מיזוג - האלגוריתם

```
MERGE-SORT(A, p, r)
1  if p < r
2     then q ← ⌊(p + r)/2⌋
3          MERGE-SORT(A, p, q)
4          MERGE-SORT(A, q + 1, r)
5          MERGE(A, p, q, r)
```

- הקריאה הראשית: Merge-Sort(*A*,1,length[*A*])



מיון-מיזוג

- **נכונות**:

יש להוכיח שהאלגוריתם עוצר ומחזיר פלט נכון עבור כל מערך בגודל ***n***

- האלגוריתם עוצר כי בכל קריאה רקורסיבית המערך קטן יותר
- הפלט נכון - הוכחה באינדוקציה על גודל תת-הסדרה
  - *בסיס*: האלגוריתם ממין נכון סדרה בגודל 1
  - *הנחת האינדוקציה*: האלגוריתם ממיין בצורה נכונה כל תת-סדרה בגודל $k < n$
  - *צעד*: נוכיח כי האלגוריתם ממיין בצורה נכונה סדרה בגודל n בהסתמך על הנחת האינדוקציה ונכונות האלגוריתם merge

- **סיבוכיות**:

- זמן: $\Theta(n\lg n)$ – יש $\Theta(\lg n)$ רמות בעץ הקריאות (בכל מקרה), ובכל רמה מתבצע מיזוג בין $\Theta(n)$ איברים בזמן לינארי
- מקום: $\Theta(n)$ – האלגוריתם משתמש במערך עזר בגודל n *לצורך המיזוג* (מספיק מערך אחד לכל רמות הרקורסיה)

האוניברסיטה הפתוחה

בעיית החיפוש

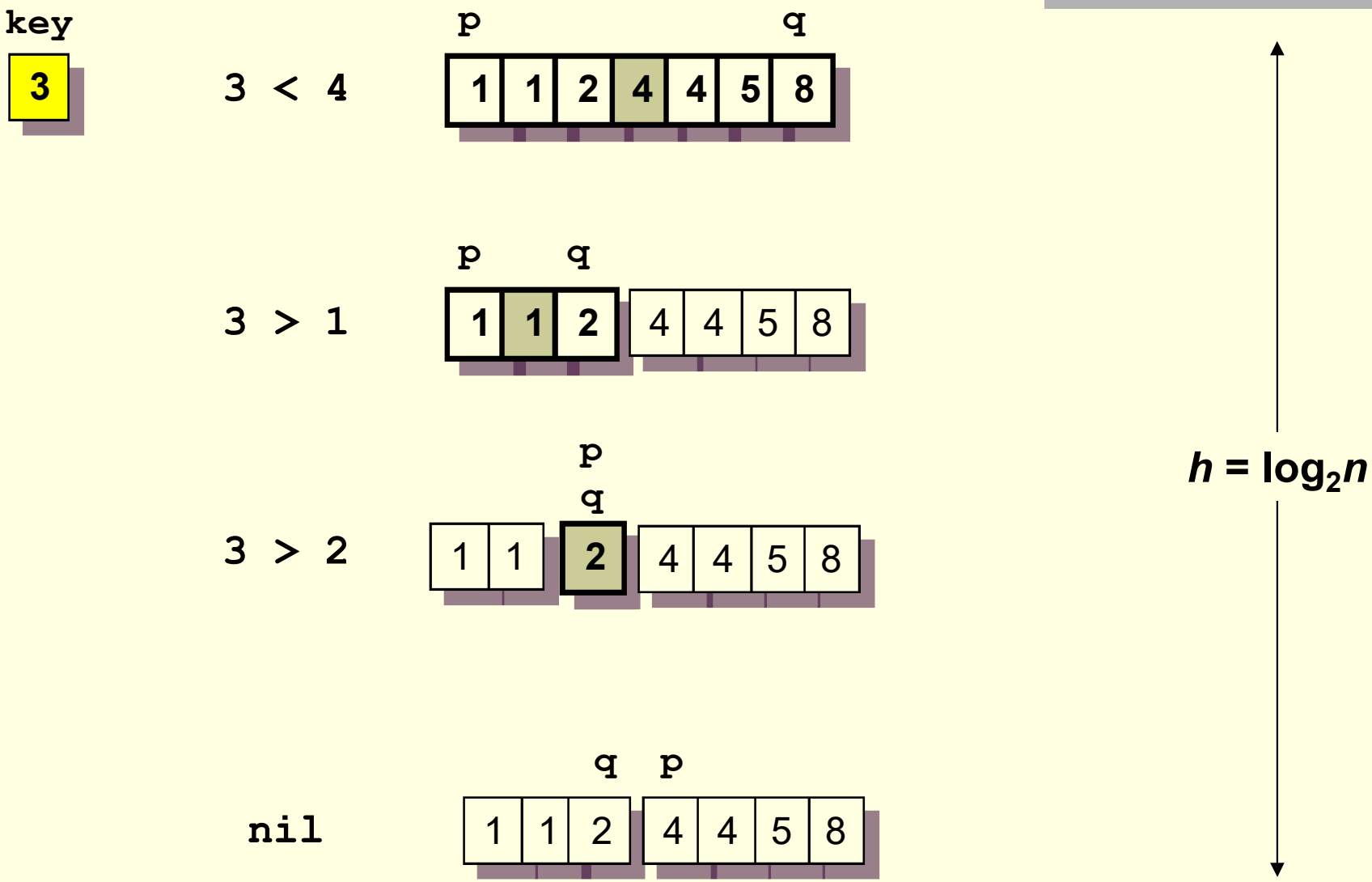
- **קלט** – אוסף של נתונים + קריטריון חיפוש
- **פלט** – הנתונים (אחד או יותר) העומדים בקריטריון החיפוש, אם ישנם
- **דוגמאות לבעיות חיפוש**
  - חיפוש האיבר הראשון בעל מפתח (ערך) X במערך ממוין
  - חיפוש האיבר הגדול ביותר במערך
  - חיפוש החציון של מערך
- **שיטות לחיפוש במערך ממוין**
  - חיפוש קווי (לינארי)
  - חיפוש חציה (בינארי)

חיפוש בינארי

- **שיטה**: הפרד ומשול
- *הקלט:* מערך ממוין בסדר עולה (או לא יורד),
  ומפתח v לחיפוש
- *תנאי עצירה:* v אינו נמצא במערך ריק
- *צעד*: השווה בין v לבין המפתח של האיבר האמצעי במערך
  - אם הם שווים, החזר את אינדקס האיבר האמצעי
  - אחרת, אם v קטן יותר, חפש (רקורסיבית) בתת-מערך השמאלי
  - אחרת, חפש (רקורסיבית) בתת-מערך הימני

האוניברסיטה הפתוחה

חיפוש בינארי - דוגמה



האוניברסיטה הפתוחה

חיפוש בינארי - האלגוריתם

```
BINARY-SEARCH(A, p, r, v)
1  if p > r
2     then return NIL
3  q ← ⌊(p + r)/2⌋
4  if  v < A[q]
5     then return BINARY-SEARCH(A, p, q − 1, v)
6  else if  v > A[q]
7     then return BINARY-SEARCH(A, q + 1, r, v)
8  else return q
```

הקריאה הראשית: Binary-Search(A,1,length[A],v)

סיבוכיות:

זמן: $\Theta(\log_2 n)$ במקרה הגרוע והממוצע

מקום: $\Theta(1)$

חיפוש בינארי

- **נכונות**:

יש להוכיח שהאלגוריתם עוצר ומחזיר פלט נכון עבור כל מערך ממוין בגודל n

  - האלגוריתם עוצר כי בכל קריאה רקורסיבית המערך קטן יותר
  - הפלט נכון - הוכחה באינדוקציה על גודל המערך
    - בסיס: טריויאלי עבור מערך ריק ($n = 0$)
    - הנחה: האלגוריתם נכון לכל מערך ממוין בגודל $k < n$
    - צעד: נוכיח עבור מערך ממוין בגודל n בהסתמך על הנחת האינדוקציה ובדיקת איבר החציון
- **סיבוכיות זמן ריצה**: במקרה הגרוע (המפתח לא נמצא)
  - *נוסחת נסיגה*: אם $n > 0$ אז $T(n) = T(n / 2) + \Theta(1)$
    אחרת $T(n) = \Theta(1)$
  - *פתרון הנוסחה*: $T(n) = \Theta(\log_2 n)$

האוניברסיטה הפתוחה

תרגיל

- יהא A מערך בגודל n. במקומות $A[1 .. m]$ נמצא הערך 0, כאשר **<u>m אינו ידוע</u>** **<u>מראש</u>** (יתכן ש-m קטן מאד ביחס ל-n). בשאר המקומות ב-A נמצא הערך 1.
- כתבו אלגוריתם COUNT0(A) המחזיר את הערך m, כלומר את מספר המקומות במערך A המכילים 0. אם כל המערך מכיל 1, מוחזר הערך 0.
- זמן הריצה של האלגוריתם צריך להיות $\Theta(\lg m)$.
  (כלומר, זמן הריצה תלוי בכמות האפסים ולא בגודל המערך כולו!)
- **רמז**: הערך m הוא אינדקס המקום הגבוה ביותר במערך המכיל 0
- **הנחייה:** השתמש בפתרון דו-שלבי.
  - בשלב ראשון יש להגביל את התחום במערך שבו צריך לבצע חיפוש.
  - בשלב השני יתבצע חיפוש בינארי בתחום <u>המוגבל</u> למציאת האינדקס m.
    הנח כי קיים אלגוריתם BINARY-SEARCH-01(A, p, q) שמחזיר את m.
    באיזה קריטריון חיפוש משתמש אלגוריתם חיפוש זה?

האוניברסיטה הפתוחה

פתרון

- אלגוריתם דו-שלבי
  - בשלב הראשון נקבע את תחום החיפוש (שורות 6-9)
  - בשלב השני נחפש בתחום זה
- קביעת התחום מתבצעת ע"י דגימת $A[1], A[2], A[4], \ldots, A[2^i]$
  - עוצרים את הדגימה כאשר $A[2^i]=1$
  - מספר הדגימות לכל היותר $\lg(m)+1$
  - תחום החיפוש הבינארי: $A[1..2^i]$
  - גודל התחום לכל היותר $2m$
  - אפשר לשכלל ע"י צמצום נוסף של תחום החיפוש: $A[\lceil q/2 \rceil..q]$

כלומר תחום בגודל לכל היותר $m+1$

```
COUNT0(A)
1.   if A[1]=1
2.       then return 0      ► no 0 in array
3.   n ← Length[A]
4.   if A[n]=0
5.       then return n      ► all are 0
6.   k ← 1
7.   while k<n and A[k]=0
8.       do k ← k*2
9.   q ← min(k, n)          ► m ≤ q
10.  return BINARY-SEARCH-01(A, 1, q)
```

האוניברסיטה הפתוחה

פתרון (המשך)

BINARY-SEARCH-01(A, p, q)

1. **if** $p>q$ or $A[p]=1$ or $A[\text{Length}[A]]=0$
2. **then return** nil
3. $mid \leftarrow (\lfloor (p+q)/2 \rfloor)$ ► observe that $mid<\text{Length}[A]$ due to line 1
4. **if** $A[mid]=0$
5. **then if** $A[mid+1]=1$
6. **then return** mid
7. **else return** BINARY-SEARCH-01(A, $mid+1$, q)
8. **else return** BINARY-SEARCH-01(A, p, $mid-1$)