הרצאת חזרה

תוכנית הקורס

1. מבני נתונים בסיסיים וסימונים אסימפטוטיים
2. מערכים ורשימות מקושרות
3. עצים ועצי חיפוש
4. עצי AVL
5. עצי 2-3
   עצי דרגות
6. רשימות דילוגים
   סיבוכיות משוערכת
7. טבלאות ערבול
8. אחזקת קבוצות זרות
9. מיון
10. מיון
11. טיפול במחרוזות
12. גרפים
13. איסוף אשפה
14. הרצאת חזרה

תוכנית השיעור

<u>חלק ראשון - חזרה על נושאי הקורס</u>

- ❑ מבני נתונים ומימושים שלהם.
- ❑ שמורות מבנה.
  - מבנים מאוזנים.
  - שמירת מידע נוסף בצמתים.
- ❑ בעיות בסיסיות בהן נתקלנו.
- ❑ סיבוכיות.

<u>חלק שני – תרגילים לדוגמה</u>

נראה שאלות ממבחני עבר.

- ❑ נפתור אותן.
- ❑ נדגיש עקרונות להפיק מהפתרונות.

חלק ראשון
חזרה על נושאי הקורס

מבני נתונים ומימושים שלהם

מילון:

מערכים	Remove(x)	Find(x)	Insert(x)
רשימות	$O(n)$	$O(n)$	$O(n)$
עצי חיפוש בינאריים	$O(n)$	$O(n)$	$O(1)$
עצי AVL	$O(h)$		
עצי B+ בפרט עצי 2-3	$O(\log n)$		
רשימות דילוגים			
טבלאות ערבול	$O(1)$*		
Trie	$O(\lvert x\rvert)$**		

*עבור טבלאות ערבול כל החסמים הם בממוצע משוערך.

**עבור Trie מניחים כי המפתחות הן מחרוזות מעל א"ב מגודל קבוע, ו- $|x|$ אורך המחרוזת x.

מבני נתונים ומימושים שלהם

Union-Find:

Makeset(i)	Union(p,q)	Find(i)	
$O(1)$*	$O(n)$	$O(1)$	מערכים
	$O(1)$	$O(n)$	רשימות
	$O(\log n)$ משוערך	$O(1)$	רשימות + מערכים, עם איחוד לפי גודל
	$O(1)$	$O(h)$	עצים הפוכים
	$O(1)$	$O(\log n)$	עצים הפוכים עם איחוד לפי גודל
	$O(\log^* n)$ משוערך		עצים הפוכים עם איחוד לפי גודל וכיווץ מסלולים

* בכל המימושים המצריכים מערך איברים, פעולת Makeset לכלל n האיברים ניתנת לביצוע בזמן $O(n)$.

מבני נתונים ומימושים שלהם

ערימת מינימום (תור עדיפויות):

Init($x_1, \ldots, x_n$), FindMin( ) , Insert(x), DelMin( ), DecKey(p, x).

מימושים:

עץ כמעט שלם המקיים את כלל הערימה, מיוצג באמצעות:

מערך
עץ

ב-2 המימושים הנ"ל סיבוכיות הפעולות היא:

Init($x_1, \ldots, x_n$)	$O(n)$
FindMin( )	$O(1)$
Insert(x)	$O(\log n)$
DelMin( )	
DecKey(p, x)	

מבני נתונים ומימושים שלהם

עץ סיומות\עץ סיומות מוכלל: Init($s_1, \ldots, s_n$).

למבנה זה לא הגדרנו פעולות נוספות, אך ציינו אלגוריתם "קופסא שחורה" שבונה את המבנה בזמן $O(\sum_{i=1}^{n} |s_i|)$ והצגנו שימושים רבים למבנה זה, רובן נסובו סביב בעיות שקשורות לתתי מחרוזות:

- ❑ מציאת תת מחרוזת.
- ❑ מציאת מספר מופעי תת מחרוזת.
- ❑ מציאת תת מחרוזת משותפת ארוכה ביותר.

מבני נתונים ומימושים שלהם

גרפים:

	Neighbors(i)	Exists(i,j)
מטריצת סמיכויות	$O(n)$	$O(1)$
רשימת סמיכויות	$O(\deg(i))$	$O(\deg(i))$

בנוסף, ראינו כיצד להשתמש בייצוגים אלו לגרפים ובמבני נתונים אחרים בכדי לפתור בעיות בסיסיות בגרפים, כגון:

- ❑ מיון טופולוגי.
- ❑ חישוב עץ פורש מינימום.
- ❑ חישוב מסלול ארוך ביותר בגרף מכוון ללא מעגלים (בתרגול).

שמורות מבנה – איזון מבנים

במהלך הקורס התרכזנו כמעט בלעדית במימושי מבנים באמצעות עצים.

במימושים אלו פעולות בסיסיות לרוב דרשו:

- סיור מהשורש לצומת (או מצומת לשורש).
- זמן $O(1)$ לכל רמה במסלול.
- סה"כ $O(h)$ זמן, כאשר h הוא גובה העץ.

בהמשך, ראינו כיצד להבטיח כי גובה העץ יהיה חסום ע"י $h = O(\log n)$, מבלי לפגוע בזמן הריצה בכל רמה של הפעולות הבסיסיות.

דוגמאות לאיזון מבנים:

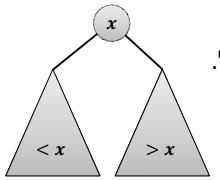
1. במימושי מילונים למיניהם באמצעות עצים (עצי AVL, B+, רשימות דילוגים).
2. ב-Union Find ממומש באמצעות עצים הפוכים.

שמורות מבנה – Invariants

במהלך הקורס ראינו כיצד הבטחת קיום תכונות מסוימות של מבנים בין פעולה לפעולה (שמורות) הועילו לנו במימוש פעולות המבנים, ואף פעולות נוספות.

אחת הדוגמאות הראשונות של שמורה שראינו היא שמורת עץ החיפוש הבינארי:

בעץ חיפוש בינארי לכל צומת יש לכל היותר 2 בנים, ולכל צומת עם מפתח x כל הצמתים בתת העץ השמאלי של הצומת קטנים מ-x ואלו בימני גדולים מ-x.



שמורה זו אפשרה חיפוש יעיל.

למעשה, כל התכונות שדרשנו מהמבנים שלנו היו שמורות של המבנים. בעת תיאור שמורה, היה עלינו להסביר איך משתמשים בשמורה וכיצד מתחזקים אותה. בשקפים הבאים נזכיר כמה דוגמאות בולטות שחזרו על עצמן במהלך הקורס.

שמורות מבנה – מידע נוסף בצמתים

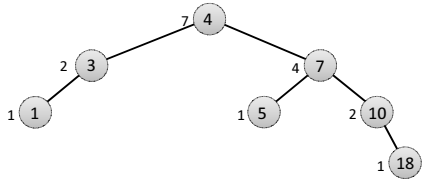
ברבים מהמבנים ראינו כיצד הוספת מידע נוסף בצמתים עוזרת לפתור ביעילות בעיות נוספות אותן המבנה הבסיסי לא ידע לפתור.

בשקפים הבאים נציג כמה דוגמאות למידע נוסף כזה שראינו במהלך הסמסטר.

מידע נוסף בצמתים – דוגמאות

במילונים מבוססי עצים:

1. שמירת מספר הצמתים\עלים בתת\ העץ בכדי:
   - ❑ למצוא את האיבר ה-k בגודלו במילון Select(k).
   - ❑ למצוא את מספרו הסידורי של איבר x במילון: Rank(x).



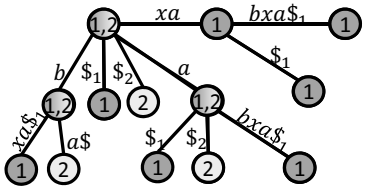
2. סכום האיברים בתת עץ – כדי לחשב סכום האיברים עד לאיבר מסוים.

3. סכום האיברים הזוגיים בתת העץ כדי לחשב סכומים אלו בכל העץ (בתרגול)

מידע נוסף בצמתים (המשך) – דוגמאות

בעצי סיומות (גם מוכללים)

1. בעץ סיומות מוכלל של שתי מחרוזות s_1, s_2, סימון לכל צומת פנימי האם הוא מייצג תת מחרוזת של s_1, של s_2, או של שתיהן, בכדי:
   - ❑ מציאת תת מחרוזת משותפת ארוכה ביותר.



עץ סיומות מוכלל של המחרוזות ba ו- $xabxa$ עם המידע הנ"ל

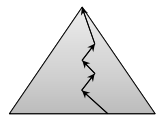
2. שמירת c_v - המיקום הראשון של תת המחרוזת המיוצגת ע"י המסלול מהשורש ועד v, למטרת:
   - ❑ מציאת מופע ראשון של מחרוזת בטקסט.
   - ❑ דחיסת אינפורמציה.

מידע נוסף בצמתים – עדכון\חישוב

נשים לב שלרוב תיארנו מידע נוסף כך שלכל צומת v המידע של v ניתן היה לחישוב לפי המידע שבבניו, ובזמן שתלוי רק במספר בניו.

עובדה זו אפשרה לנו:

1. לחשב את המידע הנוסף לכל הצמתים ע"י סיור Postorder.
2. לעדכן את המידע הנוסף ביעילות לאחר עדכוני המבנה (הכנסה\הוצאה וכו').
   - הצמתים עבורם השתנה המידע הנוסף נמצאים על מסלול החיפוש.



   - עדכון המידע הנוסף מתבצע מתחתית מסלול העדכון כלפי מעלה. דבר זה מבטיח שבזמן עדכון כל צומת במסלול התיקון, המידע הנוסף בכל אחד מבניו תקין.

בעיות בסיסיות

במהלך הקורס דננו במספר בעיות בסיסיות, ביניהן:

בעית המיון:

1. QuickSort.
2. HeapSort.
3. חסמים תחתונים למיון.
4. אלגוריתמים יעילים למקרים בהם ידוע מידע נוסף על הקלט.

מציאת האיבר ה-k בגודלו במערך:

1. בזמן לינארי בממוצע הסתברותי.
2. בזמן לינארי במקרה הגרוע, באמצעות אלגוריתם Select (חציון החציונים).

סיבוכיות

בתחילת הקורס חזרנו על מושג הסיבוכיות האסימפטוטית, בכדי לפרמל את מושג היעילות (בזמן ובמקום) של פתרונות לבעיות.

<u>תזכורת להגדרה הנפוצה בקורס:</u>

תהיינה $f(n), g(n)$ פונקציות חיוביות. נאמר כי
$f(n) = O(g(n))$
אם קיימים קבועים $0 < c$ ו- $0 < n_0$ כך שלכל $n_0 \leq n$ מתקיים $f(n) \leq c \cdot g(n)$.

למה הגדרות כאלו מעניינות אותנו?

<u>תשובה:</u> פתרונות נאיביים יכולים לעבוד על קלט קטן, אבל עבור קלט גדול המחשב שלכם "ייתקע". דיון בסיבוכיות אסימפטוטית נותן מושג של סדרי גודל של קלטים עבורם פתרון לבעיה הוא בר-שימוש.

סיבוכיות (המשך)

בהמשך הקורס דיברנו על סיבוכיות מעבר לסיבוכיות המקרה הגרוע. דיברנו על:

1. סיבוכיות בממוצע על הקלט – ממוצע על פני כל הקלטים האפשריים.
   - **גובה עצי חיפוש בינאריים.**
   - **טבלאות ערבול (פיזור אחיד).**
2. סיבוכיות בממוצע הסתברותי - ממוצע על פני הרנדומיות של האלגוריתם.
   - **רשימות דילוגים.**
   - **טבלאות ערבול (ערבול אוניברסלי).**
   - **אלגוריתמי מיון\מציאת חציון רנדומיים.**
3. סיבוכיות משוערכת – סיבוכיות המקרה הגרוע של <u>סדרת</u> פעולות.
   - **מערכים דינמיים וטבלאות ערבול.**
   - **מימושי Union Find.**

סיבוכיות משוערכת – שיטות הוכחה שראינו

1. <u>שיטת הצבירה</u>

   הוכחת חסם באופן ישיר על סך זמן הריצה.

2. <u>שיטת התשלומים</u>
   - בפעולה ה-i נקצה a_i ₪.
   - ההפרש בין המחיר המעשי t_i ומה שאנו משלמים a_i נצבר בבנק.
   - אם חשבון הבנק לא נכנס לחוב, הרי שמה ששילמנו הוא חסם על עלות הפעולות.
3. <u>שיטת הפוטנציאל</u>
   - הגדרת ערך מספרי של מבנה הנתונים בשלב ה-i, הפוטנציאל ϕ_i.
   - הגדרת המחיר המשוערך להיות $a_i = t_i + \phi_i - \phi_{i-1}$.
   - אם סוכמים על פני m פעולות מקבלים $\sum_{i=1}^{m} a_i = \sum_{i=1}^{m} t_i + \phi_m - \phi_0$.

הרעיון: נרצה להתייחס לפוטנציאל שקטן הרבה במהלך פעולות יקרות ולא גדל יותר מדי במהלך פעולות זולות.

חלק שני
תרגילים

חיפוש כל האיברים ב- Union-Find

מועד א חורף 2009-2010

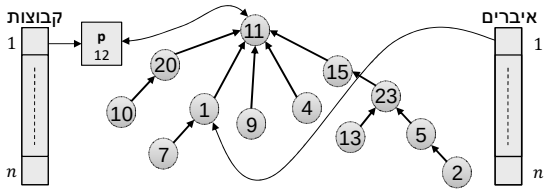
נתון מבנה Union-Find (UF) אשר ממומש ע"י עצים הפוכים. איברי המבנה ממוספרים: $1,2,\ldots,n$. איברים אלה מחולקים לקבוצות זרות כלשהן, אשר התקבלו ע"י רצף כלשהו לא ידוע של פעולות Union.

בשלב זה מריצים את האלגוריתם הבא:

```
for(int i = 1; i ≤ n; i++) {
        Find(i)
}
```

כאשר פעולת Find מבצעת כיווץ מסלולים.

הוכיחו כי האלגוריתם הנ"ל רץ בזמן $O(n)$ במקרה הגרוע.



חיפוש כל האיברים ב- Union-Find (המשך)

```
for(int i = 1; i ≤ n; i++) {
        Find(i)
}
```

<u>אבחנות:</u> אם בנוסף לכיווץ המסלולים היה נתון כי מתבצע איחוד לפי גודל:

1. מרחק כל איבר משורש קבוצתו הוא $O(\log n)$. לכן ניתן להראות חסם של $O(n\log n)$ על זמן ריצת האלגוריתם.
2. הסיבוכיות המשוערכת של פעולות UF במימוש זה היא $O(\log^* n)$.

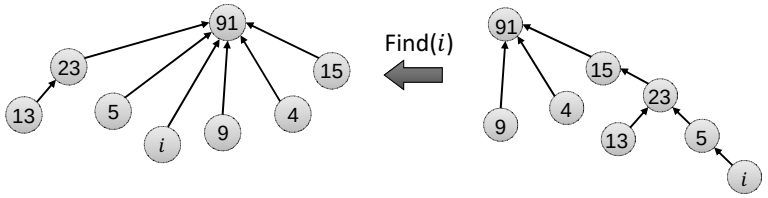
כיוון שניתן לבצע לכל היותר $n-1$ פעולות Union שמשנות את המבנה, ופעולות Find רק מקלות על פעולות Find עוקבות, ניתן להניח שהתבצעו פחות מ-$2n$ פעולות Union-Find מאתחול המבנה.

לכן זמן הריצה הכולל של כל פעולות המבנה מאתחולו הן $O(n\log^* n)$, ובפרט זמן הריצה של האלגוריתם הנ"ל היא $O(n\log^* n)$.

אנחנו מתבקשים להראות חסם הדוק יותר של $O(n)$ עבור האלגוריתם הנ"ל, וזאת ללא תלות באופן האיחודים.

חיפוש כל האיברים ב- Union-Find (המשך)

לפני שנוכיח את החסם, ניזכר בפעולת חיפוש עם איחודים בעץ הפוך, וננסה לקבל אינטואיציה למה חסם זה נכון.



נגדיר שני מושגים שיעזרו לנו בהוכחה:

<u>קשת שורש</u>: קשת בין בן של שורש לשורש.

<u>קשת פנימית</u>: קשת שאינה קשת שורש.

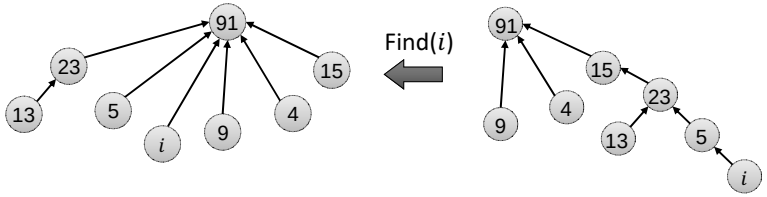
חיפוש כל האיברים ב- Union-Find (המשך)

<u>הוכחה בשיטת הצבירה:</u>

הפעולה Find(i) מורכבת מפעולות שעלותן לינארית באורך המסלול מ-i לשורש קבוצת i. כלומר העלות הכוללת לינארית במספר הקשתות בהן נבקר בסיורים.

1. לכל צומת i במהלך Find(i) נבקר בקשת שורש אחת.
2. קשת פנימית מופיעה כחלק ממסלול חיפוש <u>בדיוק</u> פעם אחת.

לכן נבקר בלכל היותר $2n = O(n)$ קשתות במהלך האלגוריתם וזמן ריצתו $O(n)$.



חיפוש כל האיברים ב- Union-Find (המשך)

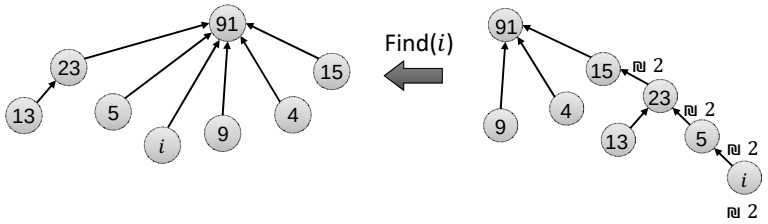
הוכחה באמצעות שיטת החיובים:

בתוכנית התשלומים הבאה נשלם מראש (בפעולה הראשונה) על הכל:

- נקצה לכל צומת וקשת בעצים ההפוכים 2 ₪ .
- כיוון שמדובר ביער, מספר הקשתות קטן מ-n.
- מכאן שבפעולה הראשונה נשלם לכל היותר $4n$ ₪ .

נראה שתשלום זה ישלם על כלל הפעולות מבלי להיכנס לחוב.

- מכאן זמן הריצה הכולל הוא $O(n)$.



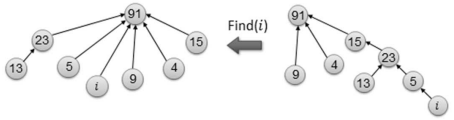
חיפוש כל האיברים ב- Union-Find (המשך)

הוכחה בשיטת הפוטנציאל:

הפוטנציאל יהיה מספר הקשתות הפנימיות. נבחין:

- $\phi_0 \leq n$.
- $\phi_n = 0$.

Find(i): הפעולה ה- i



עבור מסלול חיפוש באורך k קשתות:

- מחיר הפעולה המעשי הוא $t_i = k$.
- $k-1$ קשתות פנימיות מוחלפות בקשתות שורש והפוטנציאל יורד ב-$(k-1)$.
- מכאן נקבל $a_i = t_i + \phi_i - \phi_{i-1} = k - (k-1) = 1$.

נסכום על פני n הפעולות ונקבל:

$n \geq \sum_{i=1}^{n} a_i = \sum_{i=1}^{n} t_i + \phi_n - \phi_0$.

נעביר אגפים ונקבל $2n \geq n + \phi_0 \geq \sum_{i=1}^{n} t_i$.

חיפוש כל האיברים ב- Union-Find (המשך)

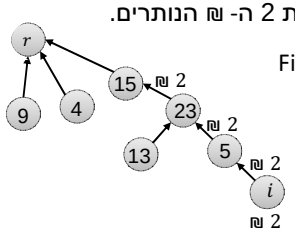
במהלך פעולת Find(i) הגוררת עליה במעלה מסלול באורך k קשתות:

- המחיר המעשי הוא $2k$ ₪.

בכדי לשלם מחיר זה:

- כל $k-1$ הקשתות הפנימיות ישלמו את 2 ה- ₪ שקיבלו בפעולה הראשונה.
- הצומת i ישלם את 2 ה- ₪ הנותרים.

דוגמא: Find(v_0)



הוכחה שהבנק לא נכנס לחוב:

- כל קשת פנימית מופיעה פעם אחת בחיפוש (ואז נעלמת).
- לכל צומת i נבצע Find(i) בדיוק פעם אחת.
- לכן בכל פעולה יהיה לנו מספיק כסף בבנק לשלם על הפעולה.

נקודות לקחת מהוכחה זו:

כפי שציינו קודם, דרך מבטיחה להגדיר פוטנציאל שיעזור להוכיח חסמים על הסיבוכיות המשוערכת של פעולה היא ע"י בחירת פוטנציאל להיות ערך מספרי של מבנה הנתונים שלא גדל יותר מדי בפעולות הקלות וצונח בפעולות היקרות.

במקרה הזה לקחנו את מספר הקשתות הפנימיות.

במהלך הקורס ראינו מספר דוגמאות נוספות:

- בMultiPop – מספר האיברים במחסנית.
- במונה הבינארי – מספר ה-1-ים במונה.
- במערכים דינאמיים (בתרגול) – מרחק מספר האיברים ממחצית גודל המערך.

מילון נוסף למספרים

מועד א אביב 2010 + תוספת

סעיף א

דרוש מבנה נתונים, אשר מחזיק מספרים שלמים חיוביים הנתונים בבסיס בינארי, ותומך בפעולות הבאות:

Init() אתחול מבנה ריק. $O(1)$.

Insert(x) הכנס את המספר x למבנה.

Find(x) החזר האם x נמצא במבנה.

Delete(x) מחק את המספר x מהמבנה.

$O(\log x)$

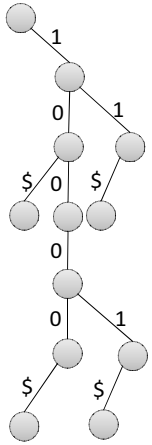
רמז:

שימו לב שהסיבוכיות הנדרשת לכל פעולה אינה תלויה במספר האיברים במבנה.

מילון נוסף למספרים

פתרון סעיף א

כדי לייצג מספר שלם x, די להשתמש ב-$\lceil \log_2(x+1) \rceil = O(\log x)$ ביטים.



- במלים אחרות, די ב-$O(\log x)$ אותיות מעל הא"ב $\Sigma = \{0,1\}$.
- נשתמש ב-Trie של המספרים מיוצגים כמחרוזות בינאריות מה-MSB ל-LSB.
- פעולות הכנסה, הוצאה וחיפוש של מחרוזת s בTrie לוקחות $O(|s|)$ זמן. בפתרוננו מספר x מיוצג כמחרוזת בינארית באורך $O(\log x)$. מכאן שפתרון זה עומד בדרישות התרגיל.

נקודות לקחת מפתרון זה:

ייצוג הנתונים יכול להיות חלק גדול מפתרון בעיה.

במקרה זה, ברגע שחשבנו לייצג כל מספר ע"י מחרוזת שהיא הייצוג הבינארי שלו, התרגיל נפתר כמעט מיידית.

מילון נוסף למספרים

סעיף נוסף

הוסיפו למבנה מסעיף א' את הפונקציונליות הבאה:

Max() מחזיר את המספר הגדול ביותר במבנה, x_{max}.

סיבוכיות זמן: $O(\log x_{max})$ במקרה הגרוע.

תארו אילו שינויים יש לעשות למימוש שהצעתם בסעיף א כדי לתמוך בפעולה.

ניסיון ראשון:

בתרגול ראינו כיצד למצוא את המחרוזת הגדולה ביותר לקסיקוגרפית בTrie בזמן שתלוי רק באורכה – בכל צומת יש לפנות בקשת שמייצגת אות גדולה ביותר.

לצערנו, לא את מחרוזת זו אנו מבקשים. למשל, המחרוזת 11 גדולה לקסיקוגרפית מהמחרוזת 10000, אבל זה אינו הסדר בין המספרים שמחרוזות אלו מייצגים:

$11_2 = 3 < 16 = 10000_2$.

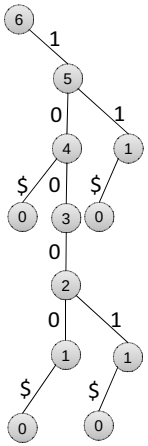
מילון נוסף למספרים

ניסיון שני – פתרון לסעיף הנוסף:

אבחנה: בין שני מספרים, מספר "ארוך יותר" (בעל יותר ספרות) הוא גדול יותר. בין מספרים מאותו אורך, המספר הגדול יותר לקסיקוגרפית הוא הגדול יותר.

רעיון פתרון:

נשמור בכל צומת את גובהו.



שימוש במידע הנוסף:

נתחיל מהשורש ועד שנגיע לעלה:

- נפנה לבן הגבוה יותר.
- במקרה של שוויון בגבהי הבנים, נמשיך בקשת שמייצגת 1.

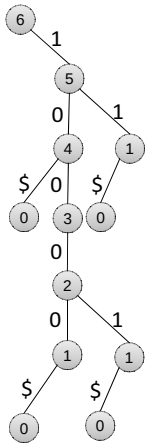
זמן ריצה: $O(\log x_{max})$.

מילון נוסף למספרים

עדכון המידע הנוסף בהכנסות\הוצאות:

בעת הכנסה או הוצאה של מספר מהמבנה, ייתכן וגבהי צמתים השתנו.

- צמתים אלו נמצאים במסלול החיפוש.
- גובה צומת הוא המקסימלי מבין גבהי בניו, ועוד 1.
- מכאן שגבהי הצמתים ניתנים לעדכון ב-$O(1)$ בכל רמה.
- סה"כ $O(\log x)$ אחרי הכנסת\הוצאת x.



נקודות לקחת מפתרון זה:

עוד דוגמה לתועלת הוספת מידע נוסף בצמתים.

בעת הגדרת מידע נוסף, יש:

- לתאר מפורשות את המידע הנוסף בכל צומת.
- להסביר כיצד להשתמש במידע הנוסף.
- לתאר כיצד לעדכן את המידע הנוסף.

מילון נוסף למספרים

פתרון לסעיף ב (המשך)

עדכון המידע הנוסף בהכנסות\הוצאות:

זהה לאופן בו עדכננו את גבהי הצמתים בסעיף הקודם, ומכאן זמן $O(\log x)$.

ההבדל היחיד הוא בנוסחה לפיה מעדכנים את המידע הנוסף בצמתים על מסלול החיפוש:

- **מספר העלים בתת העץ של צומת הוא סכום מספר העלים בתתי העצים של בניו.**

נקודות לקחת מפתרון זה:

- שוב, ייצוג נכון ושימוש במידע נוסף בצמתים מאפשר פתרון שאלה נוספת.
- הוספת מבנה נוסף למבנה בו השתמשתם עד כה עשויה להקל על הפתרון.

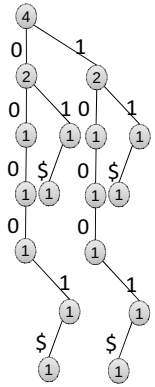
מילון נוסף למספרים

סעיף ב

הוסיפו למבנה מהסעיף הקודם את הפעולה הבאה:

CountMod(m, d) החזר את מספר האיברים x המקיימים $x \equiv m \ (mod\ 2^d)$.

m, d מספרים שלמים. סיבוכיות זמן: $O(d)$ במקרה הגרוע.



פתרון לסעיף ב

- נחזיק Trie נוסף של המספרים, מיוצגים מה-LSB ל-MSB.
- בכל צומת נשמור מידע נוסף – מספר העלים בתת העץ.

שימוש במידע הנוסף:

בכדי לענות על CountMod(m, d):

- נבדוק כמה מחרוזות מתחילות ב-d הביטים ה-LSB של m.

זמן: $O(d)$.