

## חידה לחימום

כדי להיכנס לבניין, לילך צריכה להקיש קוד המורכב מספרות, שכל אחת מהן היא 1,2,3 או 4, ולבסוף להקיש סולמית.

ניתן להקיש ספרה יותר מפעם אחת.

לילך שכחה את הקוד, אך זכרה שסכום הספרות שלו הוא 10.

מהו מספר האפשרויות השונות שתצטרך לילך לבדוק, במקרה הגרוע ביותר, עד שתגיע לקוד הנכון, **אם מספר הספרות להקשה הוא בדיוק 4?**

למשל, אם הסכום היה 4 (במקום 10), אז התשובה הייתה 1, כי הדרך היחידה להגיע לסכום 4 עם 4 ספרות היא 1,1,1,1.

שימו לב שסדר הקשת הספרות הוא חשוב.

# מבני נתונים ויעילות אלגוריתמים

(31.12.2014)

## עצי חיפוש בינאריים

עץ חיפוש בינארי (BST - Binary Search Tree) הוא עץ בינארי שהערכים מאורגנים בו באופן שמתקיימת התכונה הבאה (הנקראת תכונת עץ החיפוש הבינארי):

ערכו של כל צומת בעץ גדול יותר מערכי כל הצמתים בתת-העץ השמאלי שלו, וקטן או שווה לערכי כל הצמתים בתת-העץ הימני שלו.

תכונה זו של עץ החיפוש הבינארי מאפשרת לנו להדפיס רשימה ממוינת של כל הערכים שבעץ, על-ידי סריקה תוכית (inorder traversal) של העץ.

עבור קבוצה נתונה של ערכים, ייתכן יותר מעץ חיפוש בינארי אחד המייצג אותה, כפי שממחיש התרגיל הבא.

### שאלה

שרטטו עצי חיפוש בינאריים בגבהים 2,3,4,5 ו-6, עבור קבוצת הערכים {1,4,5,10,16,17,21}.

### 1. חיפוש

נכתוב כעת אלגוריתם המקבל כפרמטר עץ חיפוש בינארי  $T$  וערך  $x$ , ומחזיר את מיקומו של  $x$  בעץ  $T$ . אם  $x$  לא קיים בעץ  $T$ , אז יוחזר הערך המסמל עץ ריק.

$(T, x)$  מציג-חיפוש

1. אם  $x = (T)$  (אחזר-עלה) אז  $x = (T)$ , אλλι:  $x < (T)$

1.1 החזר  $T$

2. אחזר:

2.1 אם  $x \geq (T)$  (אחזר-עלה) אז  $x = (T)$ , אλλι:  $x > (T)$

2.1.1 החזר מציג-חיפוש  $(T, x)$  (תת-עץ-שמאלי)

2.2 אחזר:

2.2.1 החזר מציג-חיפוש  $(T, x)$  (תת-עץ-שמאלי)

ניתן לראות דוגמא לפעולת האלגוריתם בקישור הבא:

<http://www.cs.jhu.edu/~goodrich/dsa/trees/btree.html>

ישנו דמיון ברור בין אופן פעולת האלגוריתם, לבין האלגוריתם לחיפוש בינארי (בתוך מערך ממוין). אפשר לכתוב את האלגוריתם הזה גם כאלגוריתם איטרטיבי (המשתמש בלולאות), ולא בצורה רקורסיבית:

$(T, x)$  קצת-חיפוש

1. כף צודק?  $f$  אף-ריק?  $(T)$   $and$  (אחזר-עור)  $(x \neq T)$ ,  $g$  צ:

1.1 אט אחזר-עור  $(T)$ ,  $x \geq$ , אף:

1.1.1 תת-צף-ימני  $T \leftarrow (T)$

1.2 אחרת:

1.2.1 תת-צף-אמפי  $T \leftarrow (T)$

2. החזר  $T$

מהי סיבוכיות זמן הריצה של האלגוריתם? סיבוכיות זמן הריצה של האלגוריתם, עבור עץ שגובהו  $h$ , היא  $\Theta(h)$ . במקרה הגרוע ביותר, עבור עצי חיפוש בינארי בלתי מאוזנים, הנראים כמו רשימה לינארית מקושרת, סיבוכיות זמן הריצה תהיה  $\Theta(n)$ . במקרה הטוב ביותר, העץ יהיה מאוזן, ואז סיבוכיות זמן הריצה תהיה  $\Theta(\log n)$ . אפשר להראות שבמקרה הממוצע, הסיבוכיות עדיין  $\Theta(\log n)$ .

### שאלה

נניח שבעץ חיפוש בינארי מאוחסנים מספרים בין 1 ו-1000, וברצוננו לחפש את המספר 363. אילו מבין הסדרות הבאות אינן יכולות להיות סדרות צמתים שנבחנו במהלך החיפוש (קראו מימין לשמאל)?

א. 363, 297, 344, 330, 398, 401, 252, 2

ב. 363, 263, 258, 898, 244, 911, 220, 924

## 2. מינימום ומקסימום

לפי תכונת עץ החיפוש, כל איברי תת-העץ השמאלי קטנים משורש העץ. לכן, על מנת למצוא את האיבר הקטן ביותר בעץ יש לעבור מצומת לתת-העץ השמאלי שלו (ואף פעם לא לתת-העץ הימני שלו). הצומת הראשון שאין לו בן שמאלי יהיה המינימום בעץ. האלגוריתם פועל בהנחה שעץ החיפוש הבינארי  $T$  שהוא מקבל – אינו ריק.

$(T)$  מינימום-קצת-חיפוש

1. אט צף-ריק? (תת-צף-אמפי)  $((T))$ , אף:

1.1 החזר  $T$

2. אחרת:

2.1 החזר מינימום-קצת-חיפוש (תת-צף-אמפי)  $((T))$

### שאלה

א. כתבו אלגוריתם מילולי מינימום-קצת-חיפוש, המקבל עץ חיפוש בינארי לא ריק  $T$  ומחזיר את

המקום של הצומת בעל הערך המקסימלי בעץ.

ב. כתבו גרסאות איטרטיביות (לא רקורסיביות) עבור שני האלגוריתמים מינימום-קצת-חיפוש

ו-מינימום-קצת-חיפוש.

זמן הריצה של שני האלגוריתמים האלה על עץ בגובה  $h$  הוא  $\Theta(h)$ , שכן שניהם סורקים מסלולים במורד העץ. אם כך, עבור עץ חיפוש בינארי מאוזן בעל  $n$  צמתים, זמן הריצה הוא  $\Theta(\log n)$ .

### 3. עוקב וקודם

בהינתן צומת בעץ חיפוש בינארי, לעיתים יש חשיבות לכך שנוכל למצוא את העוקב לצומת זה בסדר הממוין הנקבע על-ידי סריקה תוכית של העץ. אם כל הערכים שונים זה מזה, העוקב לצומת  $x$  הוא הצומת בעל הערך הקטן ביותר הגדול מהערך של  $x$ . מבנהו של עץ חיפוש בינארי מאפשר לנו לקבוע את העוקב לצומת בלי להשוות ערכים.

האלגוריתם **עוקב-בינארי** מקבל עץ בינארי  $T$  לא ריק וצומת  $T1$  בעץ  $T$ , ויחזיר את העוקב לצומת  $T1$  בעץ חיפוש בינארי (אם קיים כזה). אם לצומת  $T1$  אין עוקב, יוחזר עץ ריק.

#### עוקב-בינארי $(T, T1)$

1. אם  $א$  עץ-ריק? (תת-עץ-ימני  $(T1)$ ), אזי:

1.1 מינימום-עוקב-בינארי (תת-עץ-ימני  $(T1)$ )  $T2 \leftarrow$

2. אחרת:

2.1 הורה-עוקב-בינארי  $(T, T1)$   $T2 \leftarrow$

2.2 כף עוקב  $(א$  עץ-ריק?  $(T2)$  וחס (תת-עץ-ימני  $(T2) = T1$ ), כף ע:

2.2.1  $T1 \leftarrow T2$

2.2.2 הורה-עוקב-בינארי  $(T, T1)$   $T2 \leftarrow$

3. החזר  $T2$

הקוד של האלגוריתם מחולק לשני מקרים: אם תת-העץ הימני של צומת  $T1$  אינו ריק, אזי העוקב ל- $T1$  הוא פשוט הצומת השמאלי ביותר בתת-העץ הימני. מוצאים צומת זה על-ידי זימון, בשורה 1.1, של האלגוריתם **מינימום-עוקב-בינארי**.

מצד שני, אם התת-עץ הימני של  $T1$  הוא ריק ול- $T1$  יש עוקב  $T2$ , אזי  $T2$  הוא האב-הקדמון הנמוך ביותר של  $T1$  אשר הבן השמאלי שלו גם הוא אב קדמון של  $T1$ . כדי למצוא את  $T2$ , נשתמש ב-**הורה-עוקב-בינארי**, שהוא אלגוריתם עזר שעלינו לכתוב.

### שאלה

א. כתבו אלגוריתם **הורה-עוקב-בינארי** המקבל עץ חיפוש בינארי  $T$  לא ריק, וצומת  $Tc$  בעץ.

האלגוריתם יחזיר את מיקום ההורה של  $Tc$  בעץ  $T$ . אם אין ל- $Tc$  הורה בעץ, נחזיר עץ ריק.

ב. מהי סיבוכיות זמן הריצה של האלגוריתם **הורה-עוקב-בינארי**?

ג. על סמך תשובתכם בסעיף ב', קבעו מהי סיבוכיות זמן הריצה של האלגוריתם **עוקב-בינארי**.

## שאלה

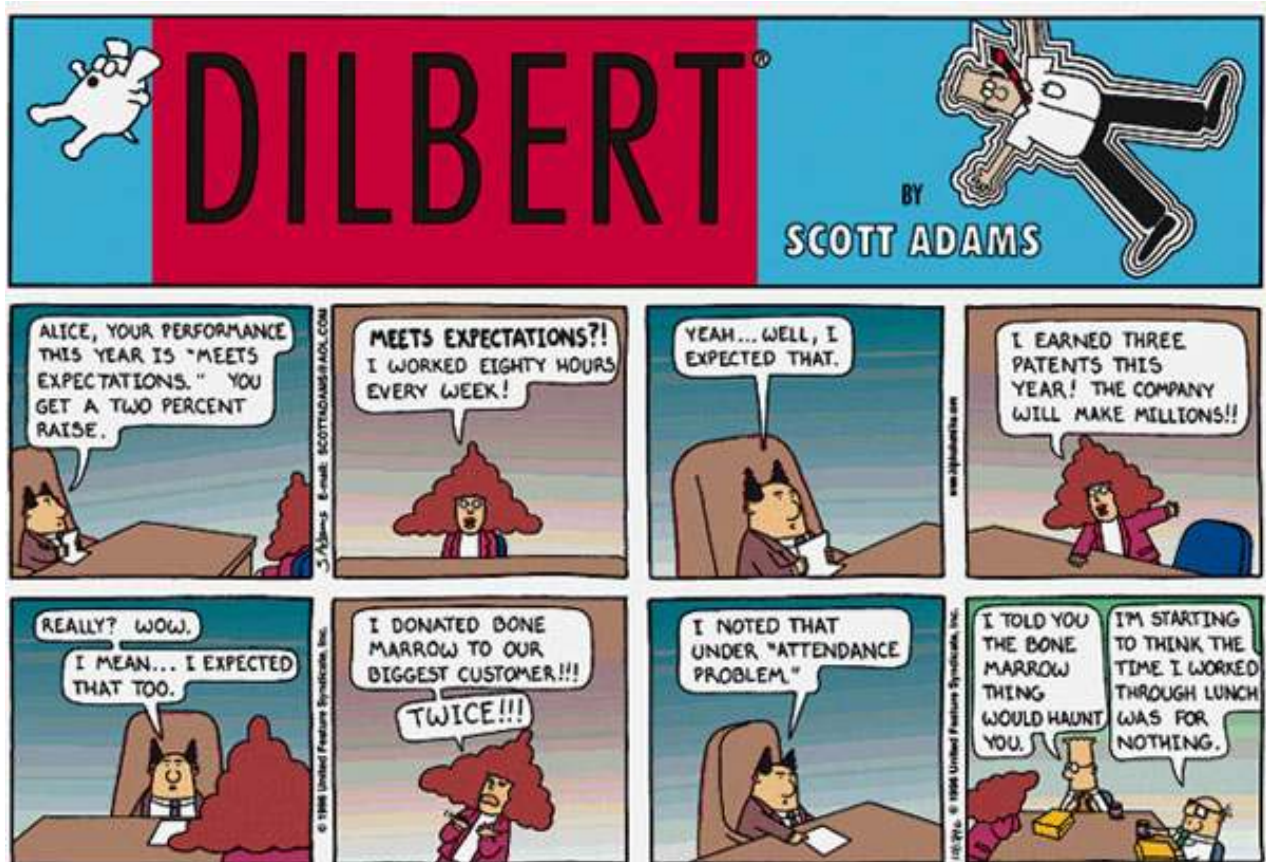
כתבו אלגוריתם **קודם-בצע-חיפוש** אשר מקבל עץ חיפוש בינארי T לא ריק וצומת T1 בעץ, ומחזיר את המקום של הצומת שערכו קודם לערכו של T1. אם אין ל-T1 קודם אז האלגוריתם יחזיר עץ ריק. הנכם רשאים להשתמש באלגוריתם העזר **הוכה-בצע-חיפוש**.

## שאלה

- א. נניח ש-T הוא עץ חיפוש בינארי בעל ערכים שונים זה מזה, x הוא עלה ב-T, ו-y הוא אביו. הסבירו מדוע הערך שב-y הוא הערך הקטן ביותר ב-T הגדול מערכו של x, או שהוא הערך הגדול ביותר ב-T הקטן מערכו של x.
- ב. הראו שאם לצומת בעץ חיפוש בינארי יש שני בנים, אזי לעוקב לו אין בן שמאלי ולקודם לו אין בן ימני.

## שאלה

ניתן לממש סריקה תוכית על עץ חיפוש בינארי בעל n צמתים על-ידי מציאת האיבר המינימאלי בעץ בעזרת **א'ני'אום-בצע-חיפוש**, ואז ביצוע n-1 קריאות ל-**צוק-בצע-חיפוש**. כתבו אלגוריתם זה באופן מפורש, ונתחו את סיבוכיות זמן הריצה שלו.



## 4. הכנסה

כדי להכניס ערך חדש לעץ חיפוש בינארי T, נשתמש באלגוריתם הוסף-צף-חיפוש המקבל עץ חיפוש בינארי T וערך x, ומוסיף את x לעץ, תוך שמירה על תכונת עץ החיפוש הבינארי.

הוסף-צף-חיפוש (T,x)

1. אם צף-ריק? (T), אזי:

1.1 בנה-צף (x, אתחל-צף, אתחל-צף)  $T \leftarrow$

2. אחרת:

2.1  $\text{found} \rightarrow \text{סקר}$

2.2 כל עוד לא found, בצע:

2.2.1 אם אחזר-שורש (T)  $x <$ , אזי:

2.2.1.1 אם צף-ריק (תת-צף-שמאלי (T)), אזי:

2.2.1.1.1  $\text{found} \rightarrow \text{אמת}$

2.2.1.2 אחרת:

2.2.1.2.1  $T \leftarrow$  (T) תת-צף-שמאלי (T)

2.2.2 אחרת:

2.2.2.1 אם צף-ריק (תת-צף-ימני (T)), אזי:

2.2.2.1.1  $\text{found} \rightarrow \text{אמת}$

2.2.2.2 אחרת:

2.2.2.2.1  $T \leftarrow$  (T) תת-צף-ימני (T)

2.3 בנה-צף (x, אתחל-צף, אתחל-צף)  $\text{temp} \leftarrow$

2.4 אם אחזר-שורש (T)  $x <$ , אזי:

2.4.1 החלף-תת-צף-שמאלי (T, temp)

2.5 אחרת:

2.5.1 החלף-תת-צף-ימני (T, temp)

## שאלה

כתבו גירסה רקורסיבית של האלגוריתם הוסף-צף-חיפוש.

## שאלה

תארו את עצי החיפוש הבינארי המתקבלים מהפעלת האלגוריתם **הוסף-צף-חיפוש** על סדרות האיברים הבאות, משמאל לימין (על אף שמדובר באותם האיברים, בשל הסדר השונה מתקבל עץ שונה):

א. 15,26,6,9,30,5,8,29

ב. 30,29,26,15,9,8,6,5

ג. 15,9,26,5,8,6,29,30

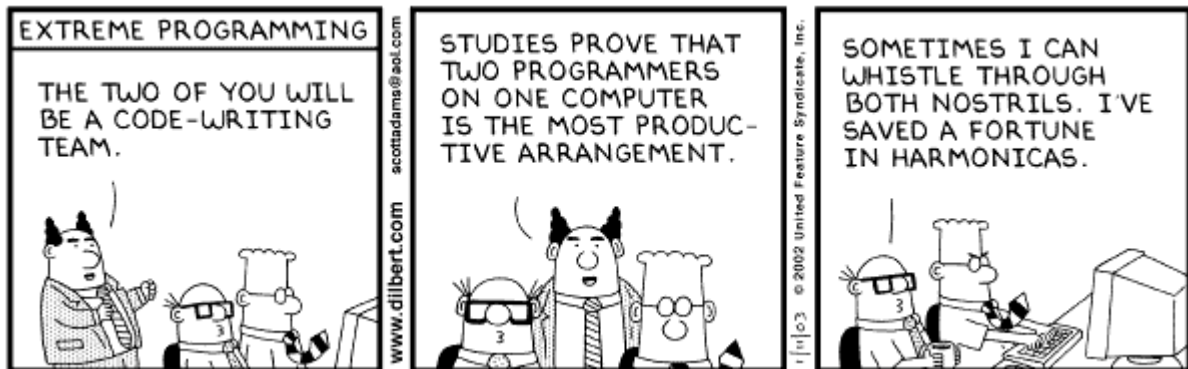
ד. 15,6,5,26,30,9,29,8

ה. 26,30,15,5,29,8,9,6

## שאלה

ניתן למיין קבוצה נתונה בת  $n$  מספרים באופן הבא: תחילה בונים עץ חיפוש בינארי המכיל מספרים אלה (מכניסים את המספרים לעץ בזה אחר זה באמצעות **הוסף-צף-חיפוש**). לאחר מכן מדפיסים את המספרים בסדר המתקבל על-ידי סריקה תוכית של העץ. נכנה שיטת מיין זו בשם **מיין-עץ** (tree-sort).

כתבו אלגוריתם **מיין-צף** (A), המקבל כפרמטר מערך A המכיל מספר שלמים. האלגוריתם יבצע מיין-עץ (יבנה עץ חיפוש בינארי מאיברי A), ויחזיר את A כשהוא ממוין בסדר עולה. מהי סיבוכיות האלגוריתם?



## 5. מחיקה

נניח שנרצה לכתוב אלגוריתם **מחק-צף-חיפוש** המקבל עץ חיפוש בינארי T לא ריק וצומת Tdel אותו יש למחוק מהעץ, תוך שמירה על תכונת עץ החיפוש הבינארי.

דרך אפשרית לעשות זאת, היא להסיר מהעץ T את כל התת-עץ ש-Tdel הוא שורשו, ואז להכניס אחד-אחד, בעזרת **הוסף-צף-חיפוש**, את כל איברי התת-עץ הזה, פרט לשורש Tdel.

## שאלה

א. כתבו את האלגוריתם **מחק-צף-חיפוש**, בהסתמך על הרעיון האלגוריתמי שלעיל.

ב. נתחו את סיבוכיות זמן הריצה במקרה הגרוע ביותר (W.C) של האלגוריתם.

בהמשך הקורס נלמד כיצד לממש את האלגוריתם **מחק-צף-חיפוש** בצורה יעילה יותר.